Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information and Computation 203 (2005) 1–38
Information and Computation
www.elsevier.com/locate/ic

A theory of stochastic systems. Part I: Stochastic automata

Pedro R. D'Argenio [a,b,1], Joost-Pieter Katoen [b,c,*]

[a] *Universidad Nacional de Córdoba, Ciudad Universitaria, 5000 Córdoba, Argentina*
[b] *University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands*
[c] *RWTH Aachen, Ahornstraße 55, D-52074 Aachen, Germany*

Received 28 November 2003; revised 9 February 2005
Available online 16 September 2005

Abstract

This paper presents the theoretical underpinning of a model for symbolically representing probabilistic transition systems, an extension of labelled transition systems for the modelling of general (discrete as well as continuous or singular) probability spaces. These transition systems are particularly suited for modelling softly timed systems, real-time systems in which the time constraints are of random nature. For continuous probability spaces these transition systems are infinite by nature. Stochastic automata represent their behaviour in a finite way. This paper presents the model of stochastic automata, their semantics in terms of probabilistic transition systems, and studies several notions of bisimulation. Furthermore, the relationship of stochastic automata to generalised semi-Markov processes is established.

Keywords: Probabilistic bisimulation; Generalised semi-Markov processes; (continuous) Probabilistic transition systems; Softly timed systems; Stochastic automata

1. Introduction

The design and analysis of systems, like embedded systems, communication protocols or multimedia systems, requires insight into not only the functional, but also into the real-time and performance aspects of applications involved. Research in concurrency theory has recognised the need for

* Corresponding author.
E-mail address: katoen@cs.rwth-aachen.de (J.-P. Katoen).
[1] Partially supported by the NWO visiting Grant B-61-519 and by the ANPCyT project PICT 11-11738.

0890-5401/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.ic.2005.07.001

the additional support of quantitative aspects, and various initiatives have been taken to accomplish such support. A prominent example is the treatment of real-time constraints, where specification formalisms like timed automata [2] have emerged, and where impressive progress has been made in the development of efficient verification algorithms [34,4]. This has resulted in a number of tools (model checkers) that provide interesting experimental platforms for industrial case studies [51,35].

1.1. Stochastic automata

Typical constraints considered in this real-time setting are hard— "the system must always do a certain activity before time t." For many applications, though, real-time constraints are less stringent. Rather than requiring that an activity must always occur before time t, in practice one is usually interested in soft real-time constraints, where a system is required to perform the activity *mostly* before time t. The soft real-time requirements of systems typically have to do with their performance characteristics, and are often also referred to as quality-of-service parameters. They are usually related to stochastic aspects of various forms of time delay, such as, for example, mean and variance of message transfer delay, service waiting times, failure rates, utilisation, etc. In a soft real-time system one typically considers constraints like:

the system should perform an activity before time t in 92% of the cases

This paper proposes a specification model for soft real-time systems. This model, called *stochastic automata*, borrows ideas from timed automata [2] and generalised semi-Markov chains (GSMPs, for short) [19,49]. Stochastic automata extend transitional automata with variables that we call *random clocks* or simply clocks. On entering a state, clocks are initialised by sampling a (continuous, discrete, or singular) probability distribution. Once clocks are initialised they run backwards, all with the same rate. An edge in the automata is labelled with a pair (a, C), where a is an action and C is a set of clocks. Such edge represents a transition that is able to offer action a once all clocks in C have expired, that is, when all clocks in C have taken a non-positive value. Stochastic automata are amenable to composition, thus allowing for a modular and hierarchical construction of models. The compositionality aspects are further studied in an accompanying paper [13].

1.2. Probabilistic transition systems

The semantics of stochastic automata is defined in terms of probabilistic transition systems, which are labelled transition systems that contain two disjoint sets of states: probabilistic and non-deterministic states. This model is inspired by [21,42,43,46]. Paths through a probabilistic transition system are sequences of alternating non-deterministic and probabilistic states. From each probabilistic state, there is exactly one outgoing probabilistic transition. Therefore, probabilistic transitions are defined by a function that maps a probabilistic state onto a probability space whose sample space ranges over the set of non-deterministic states. This sample space will usually be of an uncountable nature since we use probabilistic transitions to give an interpretation to the random clock settings of a stochastic automaton. Non-deterministic transitions are labelled transitions as in ordinary labelled transition systems. To give an interpretation to the stochastic timings in a stochastic automaton, we label these transitions with pairs (a, d) where a is an action name, and d a non-negative real number, indicating the time at which action a happens.

1.3. Bisimulation relations

When studying the behaviour of systems it is important to be able to check whether two systems behave in the same manner. There are many reasons why this is important. For instance, it would help to answer whether the model of a system implementation conforms to its specification. Another reason is that whenever two systems show equivalent behaviour, one of them could be replaced by the other as part of a larger system. In the non-stochastic setting, a well-understood method to deal with these matters that is widely accepted, is to model the specification and the implementation by labelled transition systems and then compare them according to an appropriate notion of equivalence. One of the most prominent notions is *bisimulation* [37]. This paper studies several notions of bisimulation on stochastic automata. These bisimulation relations range from notions that compare solely by inspecting the structure of the two stochastic automata while neglecting the probabilistic information, to more complex equivalences that take the stochastic timing information fully into account. Probabilistic bisimulation—basically a continuous version of Larsen and Skou bisimulation [36]—is defined on probabilistic transition systems and is lifted to stochastic automata in two different ways. Besides, a symbolic probabilistic bisimulation is defined that facilitates the comparison of the probabilistic behaviour of stochastic automata without considering their underlying (infinite) probabilistic transition systems. The relationship between the (four) defined bisimulation relations is studied. In an accompanying paper [13], the congruence properties of these bisimulations are studied.

1.4. Organisation of the paper

Section 2 introduces probabilistic transition systems and probabilistic bisimulation. Stochastic automata, their semantics, and several notions of equivalence are defined in Section 3. Section 4 studies the relation between stochastic automata and GSMPs. Section 5 describes related work, and Section 6 concludes the paper.

2. Probabilistic transition systems

This section introduces probabilistic transition systems and a fundamental equivalence relation: probabilistic bisimulation.

2.1. The model

Probabilistic transition systems generalise labelled transition systems by encoding probabilistic steps in a probabilistic transition relation. A probabilistic transition is a function that maps a state onto a probability space whose sample space ranges over the set of states. We consider a model in which probabilistic and non-deterministic transitions strictly alternate along any possible execution. The set of states is partitioned in two subsets: the *probabilistic states*, to which exactly one possible probabilistic step is associated per state, and the *non-deterministic states* which have zero or more outgoing non-deterministic transitions.

Whereas most probabilistic models in the literature [47,39,36,17,21,42] focus on discrete probability spaces, probabilistic transition systems are aimed to cope with general probability spaces, including discrete, continuous, or singular spaces. This generalisation allows the representation of real-time systems in which time constraints are not necessarily deterministic but depend on some random factor.

For this and subsequent sections, some basic mathematical foundations of probability theory are needed. For a brief introduction into these concepts, the reader is referred to Appendix A.

Definition 1. Let $Prob(H)$ denote the set of probability spaces $(\Omega, \mathcal{F}, P)$ such that $\Omega \subseteq H$. A *probabilistic transition system* (PTS, for short) is a structure $PTS = (\Sigma, \Sigma', \mathcal{L}, T, \longrightarrow)$ where:

(1) Σ is the set of *probabilistic states.*
(2) Σ' is the set of *non-deterministic states* such that $\Sigma \cap \Sigma' = \varnothing$.
(3) $\mathcal{L}$ is a set of *labels.*
(4) $T : \Sigma \to Prob(\Sigma')$ is a (total) function, called *probabilistic transition relation.*
(5) $\longrightarrow \subseteq \Sigma' \times \mathcal{L} \times \Sigma$ is the *labelled (or non-deterministic) transition relation.*

The pair (PTS, σ_0) with initial probabilistic state $\sigma_0 \in \Sigma$ is called a *rooted PTS.*

We use the shorthand notations $\sigma' \xrightarrow{\ell} \sigma$ for $\langle \sigma', \ell, \sigma \rangle \in \longrightarrow$, $\sigma' \xrightarrow{\ell}$ for $(\exists \sigma.\ \sigma' \xrightarrow{\ell} \sigma)$, and $\sigma' \not\xrightarrow{\ell}$ for $\neg(\sigma' \xrightarrow{\ell})$.

Example 2. Tossing a fair coin can be modelled by the following PTS with:

$$\begin{array}{ll} \Sigma = \{\sigma_0\} & T(\sigma_0) = (\Sigma', \wp(\Sigma'), P), \\ \Sigma' = \{\sigma_h, \sigma_t\} & \sigma_h \xrightarrow{head} \sigma_0, \\ \mathcal{L} = \{head, tail\} & \sigma_t \xrightarrow{tail} \sigma_0. \end{array}$$

where P is the unique probability measure defined by $P(\{\sigma_h\}) = P(\{\sigma_t\}) = \frac{1}{2}$. This PTS is depicted in Fig. 1, where states are represented by dots, probabilistic transitions are represented by dotted arrows joined by a dotted arc, and non-deterministic transitions by solid arrows.

As an example that exhibits non-determinism, consider a player who tosses a fair coin but cheats whenever the contender does not pay attention. In such case, he will choose arbitrarily according to his convenience. Fig. 2 depicts the PTS that models this behaviour given that the contender gets distracted with probability $\frac{1}{7}$.

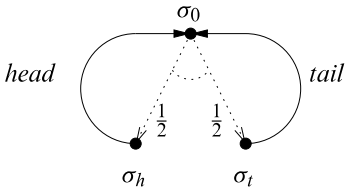

Fig. 1. Tossing a fair coin.

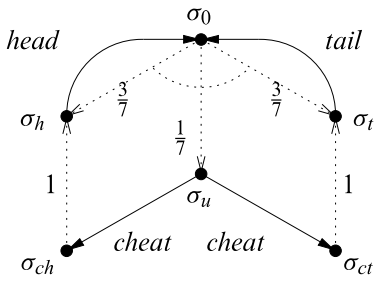

Fig. 2. Tossing a coin and cheating.

In the context of this paper, our interest is to deal with time information. Therefore, we label non-deterministic transitions with *timed actions*, that is, we consider the set of labels $\mathcal{L} = \mathcal{A} \times \mathbb{R}_{\geq 0}$, where $\mathcal{A}$ is a set of action names and $\mathbb{R}_{\geq 0}$ is the set of non-negative real numbers, which are intended to denote the (relative) time at which an action takes place. We denote $a(d)$ whenever $(a, d) \in \mathcal{L}$ and it means "action a occurs right after the system has been idle for d time units".

Example 3. Consider an automatic switch that controls a light as it can be found in a staircase or corridor in a hotel. People can arrive at any time and press the "on" button either to turn on the light or to reset the timer that controls it. The interarrival time is a Poisson process with an arrival every 30 min. Hence, the time difference between two persons turning on the light is a random variable with the (negative) exponential distribution $F_{e,30}(t) = 1 - \mathrm{e}^{-\frac{t}{30}}$. Suppose the light turns itself off after 2 min. This mechanism is thus governed by the deterministic distribution defined by: $D_2(t) = \textbf{if } (t < 2) \textbf{ then } 0 \textbf{ else } 1$.

The behaviour of the switch can be modelled by the rooted PTS $(\mathit{Switch}, \sigma_{\mathit{init}})$, where the components of *Switch* are defined as follows:

$$\Sigma = \{\sigma_{\mathit{init}}, \sigma_{\mathit{on}}\} \cup (\{\sigma_{\mathit{off}}\} \times \mathbb{R}_{\geq 0}) \quad \Sigma' = \mathbb{R}^2_{\geq 0} \cup \mathbb{R}_{\geq 0} \quad \mathcal{L} = \{\mathit{on}, \mathit{off}\} \times \mathbb{R}_{\geq 0} \ ,$$

$$\begin{aligned}
&T(\sigma_{\mathit{init}}) = \mathcal{R}(F_{e,30}) \qquad (d, d') \xrightarrow{\mathit{on}(d)} \sigma_{\mathit{on}} \iff^{\text{def}} 0 \leqslant d \leqslant d',\\
&T(\sigma_{\mathit{on}}) = \mathcal{R}(F_{e,30}, D_2) \quad (d, d') \xrightarrow{\mathit{off}(d')} (\sigma_{\mathit{off}}, d - d') \iff^{\text{def}} 0 \leqslant d' \leqslant d,\\
&T(\sigma_{\mathit{off}}, d) = \mathit{Triv}(d) \qquad d \xrightarrow{\mathit{on}(d)} \sigma_{\mathit{on}} \iff^{\text{def}} 0 \leqslant d.
\end{aligned}$$

Here, $\mathcal{R}(F_{e,30}, D_2)$ is the probability space on the real plane (cf. Appendix A) with the unique probability measure obtained from the distributions $F_{e,30}$ and D_2, and $\mathit{Triv}(d)$ is the trivial probability space on the sample space $\{d\}$.

The rooted PTS can be explained as follows. The system starts in probabilistic state σ_{init} where the light is assumed to be off. From this state, a probabilistic transition is emanating in which the first arrival time d is randomly determined according to $F_{e,30}$. The resulting state d is an element in the sample space $\mathbb{R}_{\geq 0}$ of $T(\sigma_{\mathit{init}}) = \mathcal{R}(F_{e,30})$. In this state, the transition $d \xrightarrow{\mathit{on}(d)} \sigma_{\mathit{on}}$ can be taken, indicating that the light is turned on after d time-units. In the resulting state σ_{on}, a probabilistic transition takes place. It randomly determines the time at which the next arrival will occur and the time at which the light turns itself off. This is registered by a tuple (d, d') where d is the remaining

time to the next arrival, and d' is the remaining time to switch off the light. Notice that $d' = 2$ with probability 1, and d is selected as just above.

Now two situations may occur: either the next arrival will happen soon enough, before the light turns itself off (in which case $d \leqslant d'$), or the light will turn off before somebody arrives ($d' \leqslant d$). In the first case, the "on" button is pressed and both time values need to be set again. This is represented by the transition $(d, d') \xrightarrow{on(d)} \sigma_{on}$. In the other case, the light turns itself off. This is modelled by the transition $(d, d') \xrightarrow{off(d')} (\sigma_{off}, d-d')$; the value $d-d'$ is the remaining time until the next arrival. From state (σ_{off}, d), a probabilistic transition leaves with a trivial probability space containing only element d where the switch waits until it is turned on again.

2.2. Probabilistic bisimulation

Probabilistic bisimulation [36,43] is extended to the general setting of PTSs.

Definition 4. Let $(\Sigma, \Sigma', \mathcal{L}, T, \longrightarrow)$ be a PTS and $\mu : \Sigma \times \wp(\Sigma') \to [0,1]$ be defined by

$$\mu(\sigma, S) \stackrel{\text{def}}{=} \begin{cases} P(S \cap \Omega) & \text{if } S \cap \Omega \in \mathcal{F}, \\ 0 & \text{otherwise,} \end{cases}$$

provided that $T(\sigma) = (\Omega, \mathcal{F}, P)$. Let $R \subseteq (\Sigma \times \Sigma) \cup (\Sigma' \times \Sigma')$ be an equivalence, and Σ'/R be the set of equivalence classes in Σ' induced by R. R is a *probabilistic bisimulation* if for any $\langle \sigma_1, \sigma_2 \rangle \in R$:

(1) for all $S \subseteq \Sigma'/R$, $\mu(\sigma_1, \cup S) = \mu(\sigma_2, \cup S)$, whenever $\sigma_1, \sigma_2 \in \Sigma$; and
(2) for all $\ell \in \mathcal{L}$, $\sigma_1 \xrightarrow{\ell} \sigma_1'$ implies $\sigma_2 \xrightarrow{\ell} \sigma_2'$ and $\langle \sigma_1', \sigma_2' \rangle \in R$, for some $\sigma_2' \in \Sigma$, whenever $\sigma_1, \sigma_2 \in \Sigma'$.

States σ_1 and σ_2 are *probabilistically bisimilar*, notation $\sigma_1 \sim_p \sigma_2$, if there exists a probabilistic bisimulation R with $\langle \sigma_1, \sigma_2 \rangle \in R$. The rooted PTSs (PTS_1, σ_1) and (PTS_2, σ_2) are *probabilistically bisimilar* if $\sigma_1 \sim_p \sigma_2$ in the (disjoint) union of PTS_1 and PTS_2.

Although the definition of probabilistic bisimulation coincides with traditional definitions in the discrete case, e.g., [36,22,43], we remark a necessary difference. In the discrete case, instead of transfer property 1 of Definition 4, it suffices to insist that

$$\mu(\sigma_1, S) = \mu(\sigma_2, S) \text{ for all } S \in \Sigma'/R, \tag{1}$$

i.e., S is an equivalence class instead of a set of equivalence classes. In our case, condition (1) is too weak to deal with, for instance, continuous distribution functions. This is shown in the following example.

Example 5. Let $PTS_i = (\{\sigma_i\}, \mathbb{R} \times \{i\}, \mathbb{R}, T_i, \longrightarrow_i)$, for $i \in \{1,2\}$, where $(d, i) \xrightarrow{d}_i \sigma_i$, and $T_1(\sigma_1)$ and $T_2(\sigma_2)$ are the probability spaces for a uniform distribution on $[0,1]$ and $[1,2]$, respectively.

$(PTS_1, \sigma_1) \not\sim_p (PTS_2, \sigma_2)$ since σ_1 and σ_2 do not agree in their probabilities and hence fail to satisfy the first constraint in Definition 4. However, if the weaker constraint (1) is considered, the relation

$$R = \{\langle \sigma_i, \sigma_j \rangle \mid i, j \in \{1,2\}\} \cup \{\langle (d,i),(d,j) \rangle \mid d \in \mathbb{R} \ \wedge \ i, j \in \{1,2\}\}$$

would be a probabilistic bisimulation since the probability of a point in a continuous probability space is 0.

When proving bisimilarity, the case $\mu(\sigma, \cup S) = 0$ requires special attention. The following proposition states that it suffices to consider only $\mu(\sigma, \cup S) > 0$.

Proposition 6. *R is a probabilistic bisimulation if and only if R is an equivalence relation and for all $\langle \sigma_1, \sigma_2 \rangle \in R$:*

(1) *for all $S \subseteq \Sigma'/R$, $\mu(\sigma_1, \cup S) > 0$ implies $\mu(\sigma_1, \cup S) \leqslant \mu(\sigma_2, \cup S)$; and*
(2) *for all $\ell \in \mathcal{L}$, $\sigma_1 \xrightarrow{\ell} \sigma_1'$ implies $\sigma_2 \xrightarrow{\ell} \sigma_2'$ and $\langle \sigma_1', \sigma_2' \rangle \in R$, for some $\sigma_2' \in \Sigma$.*

Proof. The "only if" part is trivial. For the "if" part it suffices to prove that the first transfer property in Definition 4 is implied by the first condition above. Suppose $\mu(\sigma_1, \cup S) > 0$. Then $\mu(\sigma_1, \cup S) \leqslant \mu(\sigma_2, \cup S)$ and, hence, $\mu(\sigma_2, \cup S) > 0$. By symmetry of R, $\mu(\sigma_2, \cup S) \leqslant \mu(\sigma_1, \cup S)$. Hence, $\mu(\sigma_1, \cup S) = \mu(\sigma_2, \cup S)$. For $\mu(\sigma_1, \cup S) = 0$, the proof follows by contradiction. Assume $\mu(\sigma_1, \cup S) \neq \mu(\sigma_2, \cup S)$. Then $\mu(\sigma_2, \cup S) > 0$. Since R is symmetric, $\mu(\sigma_2, \cup S) \leqslant \mu(\sigma_1, \cup S)$ which contradicts our assumption. □

Some classic results of non-probabilistic bisimulation do not hold in the probabilistic setting. For instance, the union of two probabilistic bisimulations is not always an equivalence relation, and therefore, may not be a probabilistic bisimulation. The transitive closure of the union of two probabilistic bisimulations, however, is a probabilistic bisimulation. This result is essential to prove that $\sim_p$ is an equivalence relation and, hence the largest probabilistic bisimulation.

Theorem 7. *If $(R_i)_{i \in I}$ is a (not necessarily finite) family of probabilistic bisimulations, then $(\bigcup_{i \in I} R_i)^*$ is a probabilistic bisimulation.*

Proof. Since each R_i is an equivalence relation, $(\bigcup_{i \in I} R_i)^*$ is an equivalence relation too. It remains to check the conditions of Definition 4. Let $\langle \sigma, \sigma' \rangle \in (\bigcup_{i \in I} R_i)^*$. Then by definition of transitive closure there are $\sigma_1, \ldots, \sigma_n$ (for $n \geqslant 0$) such that $\sigma R_{i_0} \sigma_1 R_{i_1} \ldots R_{i_{n-1}} \sigma_n R_{i_n} \sigma'$ with $i_j \in I$ for all $j \leqslant n$. Let $S \subseteq \Sigma'/(\bigcup_{i \in I} R_i)^*$. For each $j \leqslant n$, $R_{i_j} \subseteq (\bigcup_{i \in I} R_i)^*$, and hence, there exist $S_{i_j} \subseteq \Sigma'/R_{i_j}$ such that $\cup S = \bigcup S_{i_j}$. Consequently,

$$\mu(\sigma, \cup S) = \mu(\sigma_1, \cup S) = \cdots = \mu(\sigma_n, \cup S) = \mu(\sigma', \cup S),$$

which proves the first transfer property in Definition 4. The second transfer property follows in a straightforward way. □

Corollary 8. *$\sim_p$ is the largest probabilistic bisimulation.*

Proof. Let $\Re$ be the set containing all probabilistic bisimulations. By definition, $\sim_p$ is the smallest set containing all relations in $\Re$. By Theorem 7, we calculate

$$\sim_p = \cup\Re \subseteq (\cup\Re)^* \subseteq \sim_p .$$

As a consequence, $\sim_p = (\cup\Re)^*$ is a probabilistic bisimulation. □

As for classical bisimulation, we can define the notion of probabilistic bisimulation up to $\sim_p$ and show that finding a probabilistic bisimulation up to $\sim_p$ suffices to prove probabilistic bisimilarity.

Definition 9. Let $(\Sigma, \Sigma', \mathcal{L}, T, \longrightarrow)$ be a PTS and $R \subseteq (\Sigma \times \Sigma) \cup (\Sigma' \times \Sigma')$ a symmetric relation. R is a *probabilistic bisimulation up to* $\sim_p$ if for all $\langle \sigma_1, \sigma_2 \rangle \in R$:

(1) for all $S \subseteq \Sigma'/(\sim_p \cup R)^*$, $\mu(\sigma_1, \cup S) > 0$ implies $\mu(\sigma_1, \cup S) \leqslant \mu(\sigma_2, \cup S)$; and
(2) for all $\ell \in \mathcal{L}$, $\sigma_1 \xrightarrow{\ell} \sigma_1'$ implies $\sigma_2 \xrightarrow{\ell} \sigma_2'$ and $\langle \sigma_1', \sigma_2' \rangle \in (\sim_p \cup R)^*$, for some $\sigma_2' \in \Sigma$.

Theorem 10. *Let R be a probabilistic bisimulation up to $\sim_p$. Then*:

$$R \quad \subseteq \quad (\sim_p \cup R)^* \quad \subseteq \quad \sim_p$$

Proof. According to Corollary 8, it suffices to prove that $(\sim_p \cup R)^*$ is a probabilistic bisimulation. It is easy to check that $(\sim_p \cup R)^*$ is an equivalence relation. It remains to check that $(\sim_p \cup R)^*$ satisfies the transfer properties in Proposition 6.

By definition, we have $(\sim_p \cup R)^* = \bigcup_{n \geqslant 0} (\sim_p \cup R)^n$. This facilitates a proof by induction on n. The case for $n = 0$ reduces to check that the identity relation is a probabilistic bisimulation, which is straightforward. For the inductive case, suppose $\langle \sigma_1, \sigma_2 \rangle \in \bigcup_{0 \leqslant h \leqslant n+1} (\sim_p \cup R)^h$. Then there is a $\sigma \in \Sigma \cup \Sigma'$ such that $\langle \sigma_1, \sigma \rangle \in \bigcup_{0 \leqslant h \leqslant n} (\sim_p \cup R)^h$ and $\langle \sigma, \sigma_2 \rangle \in \sim_p \cup R$. For each transfer property we proceed as follows:

1. Let $S \subseteq \Sigma'/(\sim_p \cup R)^*$ and $\mu(\sigma_1, \cup S) > 0$. By the induction hypothesis, $\mu(\sigma_1, \cup S) \leqslant \mu(\sigma, \cup S)$. Besides, $\mu(\sigma, \cup S) \leqslant \mu(\sigma_2, \cup S)$ either because $\langle \sigma, \sigma_2 \rangle \in \sim_p$ and Proposition 6, or because $\langle \sigma, \sigma_2 \rangle \in R$ and Definition 9. Therefore, $\mu(\sigma_1, \cup S) \leqslant \mu(\sigma_2, \cup S)$.
2. Suppose $\sigma_1 \xrightarrow{\ell} \sigma_1'$. By the induction hypothesis, $\sigma \xrightarrow{\ell} \sigma'$ and $\langle \sigma_1', \sigma' \rangle \in (\sim_p \cup R)^*$, for some $\sigma' \in \Sigma$. Besides, $\sigma \xrightarrow{\ell} \sigma'$ implies $\sigma_2 \xrightarrow{\ell} \sigma_2'$ and either $\langle \sigma', \sigma_2' \rangle \in \sim_p$ or $\langle \sigma', \sigma_2' \rangle \in (\sim_p \cup R)^*$, for some $\sigma_2' \in \Sigma$, depending on whether $\langle \sigma, \sigma_2 \rangle \in \sim_p$ or $\langle \sigma, \sigma_2 \rangle \in R$. Thus, $\sigma_2 \xrightarrow{\ell} \sigma_2'$ and $\langle \sigma_1', \sigma_2' \rangle \in (\sim_p \cup R)^*$, for some $\sigma_2' \in \Sigma$. □

3. Stochastic automata

Probabilistic transition systems constitute a framework for the description and comparison of processes with stochastic behaviour. However, they become uncountably large—both in the number of states and in the number of transitions—as soon as systems are modelled with random timing

behaviour, cf. Example 3, the model of the light switch. To overcome this problem, this section introduces *stochastic automata*, a model that allows the representation of such systems in a finite, symbolic way. Stochastic automata are inspired by the so-called *generalised semi-Markov processes* (GSMPs) [49,19,8,44] and *timed automata* [2,26]. They extend automata with *random clocks* which are intended to control the random time of the different activities.

This section is organised as follows. Section 3.1 introduces (random) clock variables. Section 3.2 defines stochastic automata. The concrete semantics of stochastic automata in terms of PTSs is defined in Section 3.3. Finally, Section 3.4 studies notions of equivalence for stochastic automata.

3.1. Random clock variables

Like timed automata, stochastic automata resort to *clock variables* to control and observe the passage of time. Since in our context the time at which events occur is random, clocks are in fact random variables. When a clock is set, it takes a random value whose probability depends on the distribution function of the clock. As time evolves, clocks count down synchronously, i.e., all do so at the same rate. When a clock reaches the value zero, "the clock expires" and this may enable different events.

Let $\mathcal{C}$ be a given set of *(random) clock variables* such that each clock $x \in \mathcal{C}$ has an associated distribution function F_x. Let $\mathbf{V}$ be the set of all *valuations* $v : \mathcal{C} \to \mathbb{R}$. A valuation provides the current value of a clock, e.g., $v(x) = 3.12$ states that the current value of clock x equals 3.12. Note that clocks may take a negative value; this takes place when one has to wait for the expiration of a set of clocks, and some clocks of such set expire before others. Since clocks decrease as time evolves, $v-d$ is used to denote the valuation which is obtained d time-units after the valuation v was observed. Formally, for $d \in \mathbb{R}_{\geqslant 0}$ and $x \in \mathcal{C}$, let

$$(v - d)(x) \stackrel{\text{def}}{=} v(x) - d.$$

As mentioned before, clocks take random values whenever they are set. Several clocks can be set simultaneously and can take different values according to their distribution. Suppose the set $C \subseteq \mathcal{C}$ of clocks needs to be set in the valuation v. For simplicity, assume that C is ordered and let $\vec{C}$ denote this ordered set. If n is the cardinality of C, and $\vec{d} \in \mathbb{R}^n_{\geqslant 0}$ are the (randomly) chosen values for each clock in $\vec{C}$, then $v[\vec{C} \leftarrow\!\!\shortmid \vec{d}]$ denotes the valuation after setting the clocks in C and is defined by

$$v[\vec{C} \leftarrow\!\!\shortmid \vec{d}](x) \stackrel{\text{def}}{=} \begin{cases} \vec{d}(i) & \text{if } x = \vec{C}(i), \quad \text{for some } i \in \{1, \ldots, n\}, \\ v(x) & \text{otherwise,} \end{cases}$$

where $\vec{C}(i)$ and $\vec{d}(i)$ denote the ith element of $\vec{C}$ and $\vec{d}$, respectively.

3.2. The model

Definition 11. A *stochastic automaton* is a structure $SA = (\mathcal{S}, \mathcal{A}, \mathcal{C}, \longrightarrow, \kappa)$ where:

- $\mathcal{S}$ is a set of *locations*.
- $\mathcal{A}$ is a set of *actions*.

- $\mathcal{C}$ is a set of (*random*) *clocks*, each $x \in \mathcal{C}$ has associated a *distribution* F_x.
- $\longrightarrow \subseteq \mathcal{S} \times (\mathcal{A} \times \wp_{\text{fin}}(\mathcal{C})) \times \mathcal{S}$ is the set of *edges*.
- $\kappa : \mathcal{S} \to \wp_{\text{fin}}(\mathcal{C})$ is the *clock setting function*.

Note that the sets S, $\mathcal{A}$, and $\mathcal{C}$ are countable, i.e., these sets are not required to be finite. Sometimes, it will be convenient to distinguish an initial location $s_0 \in \mathcal{S}$. The structure (SA, s_0) is called a *rooted* stochastic automaton.

We write $s \xrightarrow{a,C} s'$ whenever $(s, a, C, s') \in \longrightarrow$ and call C the *trigger set* of the edge $s \xrightarrow{a,C} s'$. If there exists some location s' such that $s \xrightarrow{a,C} s'$, we write $s \xrightarrow{a,C}$.

To each location s a finite set of clocks $\kappa(s)$ is associated. On entering location s, any clock x in $\kappa(s)$ is initialised according to its probability distribution function F_x. Once initialised, the clock variables count down at the same rate of letting time pass. A clock expires if it has reached the value 0. The occurrence of an action is controlled by the expiration of clocks. Thus, whenever $s \xrightarrow{a,C} s'$ and the system is in location s, action a can be offered once *all* clocks in the set C have expired. In this situation, the edge $s \xrightarrow{a,C} s'$ becomes *enabled*. After taking the edge, the system moves to location s'. If, after the expiration of a (possibly empty) set of clocks, more than one edge outgoing from s is enabled, an enabled edge is selected non-deterministically.

Example 12. The light switch of Example 3 can be symbolically described by the stochastic automaton $\mathit{System} \stackrel{\text{def}}{=} (\mathcal{S}, \mathcal{A}, \mathcal{C}, \longrightarrow, \kappa)$ where

$$\mathcal{S} = \{s_0, s_1, s_2\} \qquad \mathcal{A} = \{on, off\} \qquad \mathcal{C} = \{x, y\},$$

$$s_0 \xrightarrow{on,\{x\}} s_1 \qquad \kappa(s_0) = \{x\}$$

$$s_1 \xrightarrow{on,\{x\}} s_1, \quad s_1 \xrightarrow{off,\{y\}} s_2 \qquad \kappa(s_1) = \{x, y\}$$

$$s_2 \xrightarrow{on,\{x\}} s_1 \qquad \kappa(s_2) = \varnothing$$

with $F_x = F_{e,30}$ and $F_y = D_2$. Fig. 3 depicts the stochastic automaton *System*. Circles represent locations, variables in each location are the clocks required to be set by function κ, and edges are represented by the arrows. For convenience, sets are denoted as lists without enclosing braces. The initial location in the rooted stochastic automaton (System, s_0), is represented by a small incoming arrow.

Example 13. Consider a simple queuing system in which jobs arrive and wait until they are executed by a single server. Assume that the queue has infinite capacity. Jobs arrive with an interarrival time that is determined by some distribution F_a, and the execution time of a job by the server is determined by a distribution function F_c. Suppose that both distribution functions are continuous. This system is known as a $G/G/1/\infty$ queue, where the G's stand for general distribution of the arrival and service process, respectively, 1 indicates that there is only one server, and ∞ says that the queue has infinite capacity. (The curious reader is referred to, e.g., [28,23,8] for more details.) The stochastic automaton *Queue* describing the behaviour of the $G/G/1/\infty$ queue is depicted in Fig. 4.

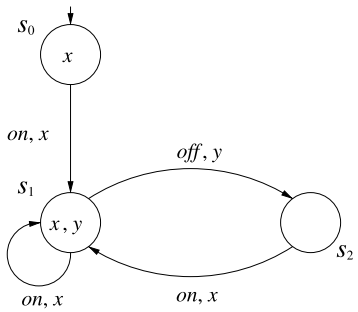

Fig. 3. The switch.

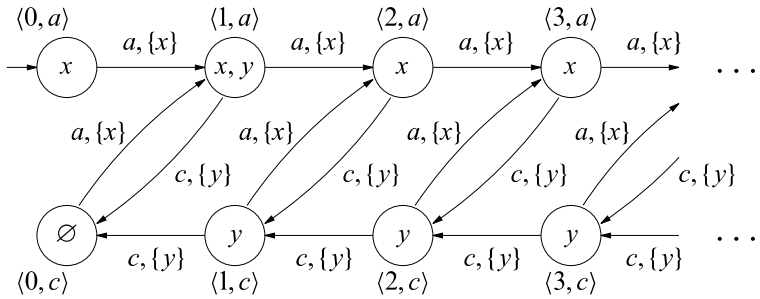

Fig. 4. Stochastic automaton of a $G/G/1/\infty$ queue.

The different components of *Queue* are:

$$\mathcal{S} = \{\langle i, \alpha\rangle \mid i \geqslant 0, \alpha \in \{a, c\}\} \qquad \langle i, \alpha\rangle \xrightarrow{a,\{x\}} \langle i+1, a\rangle,$$

$$\mathcal{A} = \{a, c\} \qquad \langle i+1, \alpha\rangle \xrightarrow{c,\{y\}} \langle i, c\rangle,$$

$$\mathcal{C} = \{x, y\} \qquad F_x = F_a \text{ and } F_y = F_c,$$

$$\kappa(\langle i, a\rangle) = \begin{cases} \{x\} & \textbf{if} \quad i \neq 1, \\ \{x, y\} & \textbf{if} \quad i = 1, \end{cases} \qquad \kappa(\langle i, c\rangle) = \begin{cases} \{y\} & \textbf{if} \quad i \neq 0, \\ \varnothing & \textbf{if} \quad i = 0, \end{cases}$$

where $i \geqslant 0$ and $\alpha \in \{a, c\}$.

Locations in this automaton are pairs where the first component indicates the number of jobs in the system (i.e., queue and server), and the second component indicates whether a job has just arrived (action a) or it has just been completed (action c). Notice that after an arrival (i.e., an occurrence of action a), a location is reached in which clock x is set, and after the completion of a job (i.e., a c-action) a clock y is set with the only exception of location $\langle 0, c\rangle$. Clock x thus controls the job inter-arrival time while y controls the service delay.

Initially, the queue is assumed to be empty and no job is being served. Therefore, this behaviour is modelled by the rooted stochastic automaton $(\mathit{Queue}, \langle 0, a\rangle)$. Observe that at location $\langle 1, a\rangle$ the event that have just arrived is the one to be served. Thus, both the time of the next arrival as well as the completion time are determined. Therefore, both clocks x and y are set in $\langle 1, a\rangle$. At location

$\langle 0, c\rangle$, however, the last job has just been served, and the time for the next job arrival has been already determined. Hence no clock is set in this location.

3.3. Semantics of stochastic automata

The semantics of stochastic automata is defined in terms of PTSs. In fact, we define two different semantics. The first interpretation regards the stochastic automaton as a model of a *closed* system, that is, a system that is complete by itself and requires no external interaction. The second interpretation regards the stochastic automaton as a model of an *open* system, i.e., a system that cooperates with the environment or is intended to be part of a larger system.

3.3.1. The closed system view

When regarding a system as closed, one not only models the components of the system but also the environment with which it interacts. In this way, an action of the whole system can take place as soon as it becomes enabled since there is no external agent that may delay its execution. That is, closed systems possess the *maximal progress* property. We refer to this interpretation as the *closed system behaviour* or *closed behaviour* for short.

Given a stochastic automaton $(\mathcal{S}, \mathcal{A}, \mathcal{C}, \rightarrow, \kappa)$, its closed behaviour is defined by a PTS. We identify its states by the location in which the system is plus the values of the clocks at the current time. Therefore, the set of possible states is given by all the pairs of locations and valuations, i.e., the set $\mathcal{S} \times \mathbf{V}$. Since we need to differentiate between probabilistic and non-deterministic states, we enclose probabilistic states between parenthesis and non-deterministic states between square brackets. Thus, $(\mathcal{S} \times \mathbf{V})$ is the set of probabilistic states with elements ranging over $(s, v), (s_i, v_i), \ldots$, and $[\mathcal{S} \times \mathbf{V}]$ is the set of non-deterministic states with elements ranging over $[s, v], [s_i, v_i], \ldots$.

To define the probabilistic transition relation we need to resort to probability theory. The basic concepts are formally defined in Appendix A. Let $s \in \mathcal{S}$ be a location and $v \in \mathbf{V}$ be a valuation. Suppose $\#\kappa(s) = n$. We define the function $\mathcal{D}_v^s : \mathbb{R}^n \to [\{s\} \times \mathbf{V}]$ by

$$\mathcal{D}_v^s(\vec{d}) \stackrel{\text{def}}{=} \left[s, v[\overrightarrow{\kappa(s)} \leftarrowtail \vec{d}]\right]$$

for all $\vec{d} \in \mathbb{R}^n$. Notice that $\mathcal{D}_v^s$ is injective. Let $\mathcal{R}(F_1, \ldots, F_n)$ be the *unique* probability space induced by the distribution functions $F_1, \ldots, F_n$ in the n-dimensional *Borel space*. Then $\mathcal{D}_v^s(\mathcal{R}(F_1, \ldots, F_n))$ is the *decoration* of $\mathcal{R}(F_1, \ldots, F_n)$ according to $\mathcal{D}_v^s$ (for a definition of decoration, see Appendix A).

For convenience we use the predicate $\mathsf{exp}_d(v, C)$ which is true if and only if all clocks in C have expired in v after d time units, i.e., $\mathsf{exp}_d(v, C)$ is defined as:

$$\forall x \in C.\ \ (v - d)(x) \leqslant 0$$

and the predicate $\mathsf{mpr}_d(s, v)$ which is true if and only if there is no possibility to leave s before d time units, i.e., $\mathsf{mpr}_d(s, v)$ is defined as:

$$\forall d' \in \mathbb{R}_{\geq 0}.\ \ d' < d \Rightarrow \left(\forall b, C.\ \ \left(s \xrightarrow{b,C} \Rightarrow \exists y \in C.\ \ (v - d')(y) > 0\right)\right).$$

Definition 14. Let $SA = (\mathcal{S}, \mathcal{A}, \mathcal{C}, \rightarrow, \kappa)$ be a stochastic automaton. The *closed* (*system*) *behaviour* of SA is defined by

$$PTS_c(SA) \stackrel{\text{def}}{=} (\ (\mathcal{S} \times \mathbf{V}), [\mathcal{S} \times \mathbf{V}], \mathcal{A} \times \mathbb{R}_{\geq 0}, T, \longrightarrow\ )$$

with T and $\longrightarrow$ defined by the following rules:

$$\textbf{Prob}\quad \frac{\overrightarrow{\kappa(s)} = (x_1, \ldots, x_n)}{T(s,v) = \mathcal{D}_v^s(\mathcal{R}(F_{x_1}, \ldots, F_{x_n}))}$$

$$\textbf{Closed}\quad \frac{s \xrightarrow{a,C} s' \qquad \mathsf{exp}_d(v,C) \qquad \mathsf{mpr}_d(s,v)}{[s,v] \xrightarrow{a(d)} \big(s', (v-d)\big)}.$$

The closed behaviour of the rooted stochastic automaton (SA, s_0) in the initial valuation $v_0 \in \mathbf{V}$ is the rooted PTS $(PTS_c(SA), (s_0, v_0))$.

The edge $s \xrightarrow{a,C} s'$ is called *enabled* in the valuation v if it induces a non-deterministic transition outgoing from $[s, v]$. The next edge to be taken is thus one of the enabled edges. Note that $s \xrightarrow{a,\varnothing} s'$ is enabled for any valuation v (even if v is negative for all clocks) as it can occur immediately, i.e., after a delay of 0 time units.

According to Definition 14, for each location s and valuation v there is exactly one probabilistic transition. Moreover, probabilistic and non-deterministic states strictly alternate. So, for any stochastic automaton SA, $PTS_c(SA)$ is indeed a probabilistic transition system.

Let us explain the inference rules. Rule **Prob** considers the setting of the clocks in $\kappa(s)$. Since the values of the clocks are assigned randomly, a probabilistic transition corresponds to this step. Starting from the probabilistic state (s, v), a probabilistic transition is made to the non-deterministic state $[s, v']$ where v' equals v except that all clocks x_i in $\kappa(s)$ are initialised according to their distribution function F_{x_i}. This is established by the decoration $\mathcal{D}_v^s(\mathcal{R}(F_{x_1}, \ldots, F_{x_n}))$. Note that the location does not change in this case. Rule **Closed** can be explained as follows. Assume that the system is at location s with current valuation v. In this state, an edge $s \xrightarrow{a,C} s'$ induces the execution of action a after delaying d time units if at this moment all clocks in C have expired (i.e., $\mathsf{exp}_d(v, C)$ holds), and moreover, no edge (including $s \xrightarrow{a,C} s'$) at the same location became enabled before (i.e., $\mathsf{mpr}_d(s, v)$ is satisfied).

Example 15. Following Definition 14, the closed behaviour of the stochastic automaton *System* of Example 12 is given by:

$$PTS_c(System) = ((\mathcal{S} \times \mathbf{V}), \Big[\mathcal{S} \times \mathbf{V}\ \Big], \mathcal{A} \times \mathbb{R}_{\geq 0}, T, \longrightarrow),$$

where

$$T(s_0, v) = \mathcal{D}_v^{s_0}(\mathcal{R}(F_x)) \quad T(s_1, v) = \mathcal{D}_v^{s_1}(\mathcal{R}(F_x, F_y)) \quad T(s_2, v) = \mathcal{D}_v^{s_2}(\mathcal{R}()),$$

$$\big[s_0, (x := d, y := d')\big] \xrightarrow{on(d)} \big(s_1, (x := 0, y := d' - d)\big) \iff 0 \leqslant d,$$

$$\big[s_1, (x := d, y := d')\big] \xrightarrow{on(d)} \big(s_1, (x := 0, y := d' - d)\big) \iff 0 \leqslant d \leqslant d',$$

$$\big[s_1, (x := d, y := d')\big] \xrightarrow{off(d')} \big(s_2, (x := d - d', y := 0)\big) \iff 0 \leqslant d' \leqslant d,$$

$$\big[s_2, (x := d, y := d')\big] \xrightarrow{on(d)} \big(s_1, (x := 0, y := d' - d)\big) \iff 0 \leqslant d.$$

Choosing v_0 with $v_0(x) = v_0(y) = 0$ as the initial valuation, the rooted PTS $(PTS_c(System), (s_0, v_0))$ defines the closed behaviour of the rooted stochastic automaton $(System, s_0)$.

3.3.2. The open system view

An *open system* is a system that interacts with its environment. The environment can be a user or another system. Basically, an open system is a component of a larger system.

To study reachability properties like freedom from deadlock, it is important to observe how the system behaves in an arbitrary context. That is, the interaction of a system with a certain "well-behaved" component may not induce a deadlock, while a "badly behaved" component could take the system through an undesired path that will end in a deadlock situation. To study these situations, the interpretation of a stochastic automaton as a closed system is not sufficient. Instead, if we interpret a stochastic automaton as an open system, the possibility of interacting with its environment would be considered. In this case an action that is enabled cannot be executed until the environment is also ready to perform it. Therefore, the maximal progress property is dropped in the open semantics.

Definition 16. Let $SA = (\mathcal{S}, \mathcal{A}, \mathcal{C}, \rightarrow, \kappa)$ be a stochastic automaton. The *open* (*system*) *behaviour* of SA is defined by

$$PTS_o(SA) \stackrel{\text{def}}{=} (\,(\mathcal{S} \times \mathbf{V}), [\mathcal{S} \times \mathbf{V}], \mathcal{A} \times \mathbb{R}_{\geq 0}, T, \longrightarrow\,),$$

where T is defined by rule **Prob** as in Definition 14 and $\longrightarrow$ is defined by:

$$\textbf{Open} \quad \frac{s \xrightarrow{a,C} s' \qquad \mathsf{exp}_d(v, C)}{[s, v] \xrightarrow{a(d)} \big(s', (v - d)\big)}.$$

The open behaviour of the rooted stochastic automaton (SA, s_0) in the initial valuation $v_0 \in \mathbf{V}$ is the rooted PTS $(PTS_o(SA), (s_0, v_0))$.

The only difference between the open and closed semantics is that the constraint of maximal progress is present in the inference rule **Closed** but not in **Open**. In the open behaviour, non-deterministic transitions with different time labels may leave the same state, whereas this is impossible in the closed behaviour.

Example 17. According to Definition 16, the open behaviour of stochastic automaton *System* of Example 12, $PTS_o(System)$, consists of the same components as $PTS_c(System)$ in Example 15, except that $\longrightarrow$ is defined by:

$$\begin{aligned}
\big[s_0,(x:=d',y:=d'')\big] &\xrightarrow{on(d)} \big(s_1,(x:=d'-d,y:=d''-d)\big) \iff 0\leqslant d\wedge d'\leqslant d,\\
\big[s_1,(x:=d',y:=d'')\big] &\xrightarrow{on(d)} \big(s_1,(x:=d'-d,y:=d''-d)\big) \iff 0\leqslant d\wedge d'\leqslant d,\\
\big[s_1,(x:=d',y:=d'')\big] &\xrightarrow{off(d)} \big(s_2,(x:=d'-d,y:=d''-d)\big) \iff 0\leqslant d\wedge d''\leqslant d,\\
\big[s_2,(x:=d',y:=d'')\big] &\xrightarrow{on(d)} \big(s_1,(x:=d'-d,y:=d''-d)\big) \iff 0\leqslant d\wedge d'\leqslant d.
\end{aligned}$$

Notice that there is no correlation between the values d' of x and d'' of y. The only requirement is that the time d of occurrence of an action is positive and beyond the time at which the edge becomes enabled (cf. Example 15).

3.4. Equivalences on stochastic automata

By lifting the probabilistic bisimulation of PTS to stochastic automata, we obtain two different notions of equivalences depending on whether the closed behaviour or the open behaviour is considered.

Definition 18. Locations s_1 and s_2 of stochastic automaton SA are *closed p-bisimilar*, denoted $s_1 \sim_c s_2$, if for any valuation $v\in\mathbf{V}$, $(s_1,v)\sim_p(s_2,v)$, where (s_1,v) and (s_2,v) are states in $PTS_c(SA)$. The rooted stochastic automata (SA_1,s_1) and (SA_2,s_2) are *closed p-bisimilar* if

$$(PTS_c(SA_1),(s_1,v_0))\sim_p(PTS_c(SA_2),(s_2,v_0)) \quad \text{for every } v_0\in\mathbf{V}\ .$$

For the open behaviour $PTS_o(SA)$, *open p-bisimilarity*, denoted $\sim_o$, is defined in a similar way. It is immediate that $\sim_c$ and $\sim_o$ are equivalences on the set of locations $\mathcal{S}$. Both open and closed p-bisimulations require to deal with infinite state spaces as well as measure theory. In the following, we define relations that, though strictly finer, allow for symbolic reasoning on stochastic automata, therefore providing a simpler and clearer framework to prove open or closed p-bisimilarity.

The strongest relation that we consider is isomorphism. Two stochastic automata are isomorphic if there exists a bijective function that maps locations from one automaton to locations of the other without disturbing the structure of the stochastic automaton. (The notion of isomorphism could be extended by allowing bijections on the sets of clocks and the actions, but this may not preserve open and closed p-bisimilarity in general.)

Definition 19. $SA=(\mathcal{S},\mathcal{A},\mathcal{C},\rightarrow,\kappa)$ and $SA'=(\mathcal{S}',\mathcal{A},\mathcal{C},\rightarrow',\kappa')$ are *isomorphic*, notation $SA\cong SA'$, if there is a bijection $\mathcal{I}:\mathcal{S}\to\mathcal{S}'$ such that:

(1) $s\xrightarrow{a,C}s' \iff \mathcal{I}(s)\xrightarrow{a,C}'\mathcal{I}(s')$, and
(2) $\kappa(s)=\kappa'(\mathcal{I}(s))$.

Function $\mathcal{I}$ is called an *isomorphism*. The rooted stochastic automata (SA, s_0) and (SA', s_0') are *isomorphic* if $SA \cong SA'$ and $\mathcal{I}(s_0) = s_0'$.

Structural bisimulation is a slightly weaker equivalence than isomorphism. It is a bisimulation on stochastic automata that preserves both actions and sets of clocks.

Definition 20. Let $(\mathcal{S}, \mathcal{A}, \mathcal{C}, \longrightarrow, \kappa)$ be a stochastic automaton. A relation $R \subseteq \mathcal{S} \times \mathcal{S}$ is a *structural bisimulation* if R is symmetric and for all $a \in \mathcal{A}$ and $C \in \mathcal{C}$, whenever $\langle s_1, s_2 \rangle \in R$ the following transfer properties hold:

(1) $s_1 \xrightarrow{a,C} s_1'$, implies $s_2 \xrightarrow{a,C} s_2'$ and $\langle s_1', s_2' \rangle \in R$ for some $s_2' \in \mathcal{S}$; and
(2) $\kappa(s_1) = \kappa(s_2)$.

s_1 and s_2 are *structurally bisimilar*, notation $s_1 \sim_s s_2$, if there exists a structural bisimulation R such that $\langle s_1, s_2 \rangle \in R$. Two rooted stochastic automata (SA_1, s_1) and (SA_2, s_2) are *structurally bisimilar*, if $s_1 \sim_s s_2$ in the (disjoint) union of SA_1 and SA_2.

Structural bisimulation up to $\sim_s$ is useful to prove structural bisimilarity.

Definition 21. Let $(\mathcal{S}, \mathcal{A}, \mathcal{C}, \longrightarrow, \kappa)$ be a stochastic automaton. A relation $R \subseteq \mathcal{S} \times \mathcal{S}$ is a *structural bisimulation up to* $\sim_s$ if R is symmetric and for all $a \in \mathcal{A}$ and $C \in \mathcal{C}$, whenever $\langle s_1, s_2 \rangle \in R$ the following transfer properties hold:

(1) $s_1 \xrightarrow{a,C} s_1'$, implies $s_2 \xrightarrow{a,C} s_2'$ and $\langle s_1', s_2' \rangle \in (\sim_s \cup R)^*$ for some $s_2' \in \mathcal{S}$; and
(2) $\kappa(s_1) = \kappa(s_2)$.

The proof of the following theorem closely resembles the proofs of each of the enumerated facts for strong bisimulation [37] and is therefore omitted.

Theorem 22.

(1) $\sim_s$ *is a structural bisimulation, and*
(2) $\sim_s$ *is an equivalence relation on the set of locations.*
(3) R *is a structural bisimulation up to* $\sim_s$ *implies* $R \subseteq (\sim_s \cup R)^* \subseteq \sim_s$.

Structural bisimulation is defined directly on stochastic automata, but does not consider any stochastic information. Simple modifications to the stochastic automaton like changing the name of a random variable while preserving the probability function, or setting clocks that will never be used, do not lead to any behavioural difference. An example is shown in Fig. 5A, where clock z is never used, while x and y are synonyms. Similarly, a clock that has already expired—and never set again—is irrelevant, as shown in Fig. 5B. Here, in location s_1' clock x has already expired (and not set again), and therefore is not of any importance anymore. Moreover, a unique clock may replace a set of clocks if they are always initialised and triggered together, as in Fig. 5C. (The distribution function of the maximum of a set of independent random variables is defined in Proposition 43.) Open and closed p-bisimilarity do take the probabilistic behaviour into account, but are defined in terms of the underlying, infinite

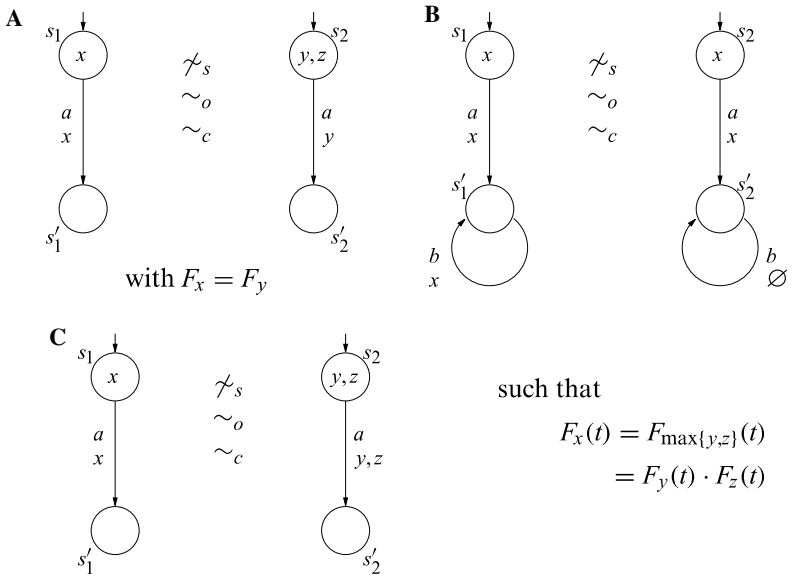

Fig. 5. Examples of non-structurally bisimilar stochastic automata.

PTS. Probabilistic information can be considered at a symbolic level too by defining a symbolic bisimulation on stochastic automata. As a symbolic bisimulation needs to relate clocks, we syntactically characterise those clocks which are important to take into account. Basically, a clock is relevant if it is used somewhere, that is, it is in some trigger set on some edge. For finite path $s_0 \xrightarrow{a_0,C_0} s_1 \xrightarrow{a_1,C_1} \cdots \xrightarrow{a_n,C_n} s_{n+1}$, clock x is relevant in location s_i if $j > i$ is the smallest index such that $x \in C_j$ and x is not set in s_i through s_j. This notion is lifted to locations as follows.

Definition 23. Let $(\mathcal{S}, \mathcal{A}, \mathcal{C}, \longrightarrow, \kappa)$ be a stochastic automaton. The set of *free clock variables* of location $s \in \mathcal{S}$, denoted by $\mathsf{Fv}(s)$, is defined by the smallest set satisfying

$$\mathsf{Fv}(s) = \left\{ x \mid s \xrightarrow{a,C} s' \wedge x \in C \cup \mathsf{Fv}(s') \right\} - \kappa(s).$$

The set of *relevant clock variables* for location $s \in \mathcal{S}$, denoted $\mathsf{Rel}(s)$, is defined by

$$\mathsf{Rel}(s) \stackrel{\text{def}}{=} \left\{ x \mid s \xrightarrow{a,C} s' \wedge x \in C \cup \mathsf{Fv}(s') \right\}.$$

Intuitively, clock x is relevant in location s if for any path starting from s the following holds: if x is in the trigger set of (some edge of) some location along the path, then x is not set until it has triggered some edge along the path. A clock is free in s if, in addition, it is not

set in s. Therefore, $\mathsf{Fv}(s) = \mathsf{Rel}(s) - \kappa(s)$. Note that $\kappa(s) \subseteq \mathsf{Rel}(s)$ may not hold, e.g., clock z is not relevant in location s_2 in the right automaton of Fig. 5A.

A symbolic bisimulation should relate relevant clocks in a particularly organised manner. Roughly speaking, clocks are related when they are set at the same instant and have corresponding distributions. Therefore, each element in a symbolic bisimulation is a triplet $\langle s_1, s_2, SR\rangle$ where s_1 and s_2 are the related locations, and SR is a component (called *synchronisation relation*) explaining how the clocks relate.

Definition 24. Let $\mathcal{C}$ be a set of clocks. For $C_1, C_2, C_1', C_2' \subseteq \mathcal{C}$, let

$$\langle C_1, C_2\rangle \trianglelefteq \langle C_1', C_2'\rangle \iff (C_1 \cap C_1') \cup (C_2 \cap C_2') = \varnothing \vee (C_1 \subseteq C_1' \wedge C_2 \subseteq C_2').$$

$SR \subseteq \wp(\mathcal{C}) \times \wp(\mathcal{C})$ is a *synchronisation relation* if

$$\langle C_1, C_2\rangle, \langle C_1', C_2'\rangle \in SR \Rightarrow \langle C_1, C_2\rangle \trianglelefteq \langle C_1', C_2'\rangle.$$

$\langle C_1, C_2\rangle \trianglelefteq \langle C_1', C_2'\rangle$ can be read as "$\langle C_1, C_2\rangle$ is compatible with $\langle C_1', C_2'\rangle$". To be compatible, C_1 and C_2 should be included in C_1' and C_2', respectively, or should not intersect. Notice that in a synchronisation relation SR all tuples should be compatible with each other, that is, $\langle C_1, C_2\rangle \trianglelefteq \langle C_1', C_2'\rangle$ and $\langle C_1', C_2'\rangle \trianglelefteq \langle C_1, C_2\rangle$, for all pairs of tuples in SR. As a consequence, either $(C_1 \cap C_1') \cup (C_2 \cap C_2') = \varnothing$ or $\langle C_1, C_2\rangle = \langle C_1', C_2'\rangle$.

Definition 25. Let $(\mathcal{S}, \mathcal{A}, \mathcal{C}, \longrightarrow, \kappa)$ be a stochastic automaton. A *symbolic bisimulation* is a relation $R \subseteq \mathcal{S} \times \mathcal{S} \times \wp(\wp(\mathcal{C}) \times \wp(\mathcal{C}))$ that, for all $\langle s_1, s_2, SR\rangle \in R$, satisfies the following properties:

(1) SR is a synchronisation relation such that:
  (a) for $i = 1, 2$, $\bigcup\{C_i \mid \langle C_1, C_2\rangle \in SR\} = \mathsf{Rel}(s_i)$;
  (b) $\langle C_1, C_2\rangle \in SR$ implies $\langle C_1, C_2\rangle \trianglelefteq \langle \kappa(s_1), \kappa(s_2)\rangle$;
  (c) if $\langle C_1, C_2\rangle \in SR$ and $C_i \subseteq \kappa(s_i)$, $i = 1, 2$, then, for all $t \in \mathbb{R}$,

$$\prod_{x \in C_1} F_x(t) = \prod_{y \in C_2} F_y(t)$$

(2) $s_1 \xrightarrow{a, C_1^\star} s_1'$, then there are s_2' and $C_2^\star$ such that $s_2 \xrightarrow{a, C_2^\star} s_2'$, and
  (a) $\langle C_1, C_2\rangle \in SR$ implies $\langle C_1, C_2\rangle \trianglelefteq \langle C_1^\star, C_2^\star\rangle$; and
  (b) $\langle s_1', s_2', SR'\rangle \in R$ for some SR' which is *forward compatible* with SR, i.e.,

$$\begin{aligned}&\left\{\langle C_1, C_2\rangle \in SR \mid (C_1 \cap \mathsf{Fv}(s_1')) \cup (C_2 \cap \mathsf{Fv}(s_2')) \neq \varnothing\right\} - \left(\wp(C_1^\star) \times \wp(C_2^\star)\right)\\ &\quad = \left\{\langle C_1', C_2'\rangle \in SR' \mid (C_1' \cap \mathsf{Fv}(s_1')) \cup (C_2' \cap \mathsf{Fv}(s_2')) \neq \varnothing\right\} - \left(\wp(C_1^\star) \times \wp(C_2^\star)\right).\end{aligned}$$

(3) $s_2 \xrightarrow{a,C_2^\star} s_2'$, then there are s_1' and $C_1^\star$ such that $s_1 \xrightarrow{a,C_1^\star} s_1'$, and
   (a) $\langle C_1, C_2\rangle \in SR$ implies $\langle C_1, C_2\rangle \trianglelefteq \langle C_1^\star, C_2^\star\rangle$; and
   (b) $\langle s_1', s_2', SR'\rangle \in R$ for some SR' which is *forward compatible* with SR.

Locations s_1 and s_2 are *symbolically bisimilar*, denoted $s_1 \sim_\& s_2$, if there exists a symbolic bisimulation R such that $\langle s_1, s_2, SR\rangle \in R$ for some SR satisfying

$$\text{if } \langle C_1, C_2\rangle \in SR \text{ and } x \in (C_1 \cap \mathsf{Fv}(s_1)) \cup (C_2 \cap \mathsf{Fv}(s_2)) \text{ then } C_1 = C_2 = \{x\}. \qquad (*)$$

The rooted stochastic automata (SA_1, s_1) and (SA_2, s_2) are *symbolically bisimilar*, if $s_1 \sim_\& s_2$ in the (disjoint) union of SA_1 and SA_2.

The synchronisation relation SR in $\langle s_1, s_2, SR\rangle$ explains how the relevant clocks in s_1 and s_2 are related. This is mostly explained in item 1. In particular, 1(i) requires that all clocks in SR are relevant, and conversely, that all relevant clocks are related; 1(ii) states that clocks related in the same pair must be set simultaneously; and 1(iii) requires that clocks related in a pair should be set with equal probability in the sense that random variables $\max(C_1)$ and $\max(C_2)$ should have the same distribution. Here, the maximum of a set of clocks is taken as we need to wait until all clocks have expired, i.e., the "slowest" clock determines the overall delay. Recall (cf. Proposition 43) that the distribution function of $\max(C)$ is given by $F_{\max(C)}(t) = \prod_{x\in C} F_x(t)$ for all $t \in \mathbb{R}$ where $F_{\max(\varnothing)}(t)$ equals the constant function 1.

Items 2 and 3 are the transfer properties, and they state how the edges must be simulated. If the left-hand side location has an outgoing arrow, the right one has also an outgoing arrow labelled with the same action name. Moreover, the trigger sets should be compatible with the synchronisation relation, therefore requiring that clocks in $C_1^\star$ are appropriately synchronised with those in $C_2^\star$ (conditions 2(i) and 3(i)). Finally, the target locations must be again related with a new synchronisation relation SR' which should be forward compatible with SR (conditions 2(ii) and 3(ii)). By forward compatible we mean that SR' preserves the old relation imposed by SR for all clocks that are still relevant. Notice that clocks in the previous trigger sets are not required to stay synchronised since they already expired and therefore they do not impose any restriction on further enabling conditions. This is meant to relate stochastic automata like the ones in Fig. 5B.

Two locations may be related by a symbolic bisimulation but they are not necessarily symbolically bisimilar. This has to do with the fact that a triplet $\langle s_1, s_2, SR\rangle$ "remembers" in SR how the clocks were related in the past. For two locations to be symbolically bisimilar, their free clock variables cannot be arbitrarily synchronised. Following the criterion of closed and open p-bisimulation in which two locations are related if they are related in every valuation (see Definition 18), we require that SR satisfies the additional condition $(*)$. This condition requires that a free variable is not related to any other variable at the same side and it is related to the (free) variable of the same name in the other side.

Example 26. The stochastic automata in Figs. 5A–C are symbolic bisimilar according to the respective relations:

$$R_a = \{\ \langle s_1,\ s_2,\ \{\langle\{x\},\{y\}\rangle\}\ \rangle,\ \langle s'_1,\ s'_2,\ \varnothing\rangle\ \},$$
$$R_b = \{\ \langle s_1,\ s_2,\ \{\langle\{x\},\{x\}\rangle\}\ \rangle,\ \langle s'_1,\ s'_2,\ \{\langle\{x\},\varnothing\rangle\}\ \rangle\ \},\ \text{and}$$
$$R_c = \{\ \langle s_1,\ s_2,\ \{\langle\{x\},\{y,z\}\rangle\}\ \rangle,\ \langle s'_1,\ s'_2,\ \varnothing\rangle\ \}$$

Theorem 27. *$\sim_{\&}$ is an equivalence relation on the set $\mathcal{S}$ of locations.*

Proof. The fact that $\sim_{\&}$ is reflexive follows from Theorem 28 below, which states that $\sim_s \subseteq \sim_{\&}$. Symmetry follows from the following: if R is a symbolic bisimulation, it is not difficult to check that $\{\langle s_2, s_1, SR^{-1}\rangle \mid \langle s_1, s_2, SR\rangle \in R\}$ is also a symbolic bisimulation satisfying condition $(*)$ in Definition 25 whenever R satisfies it. The proof of transitivity is more involved and can be found in [11, Appendix E]. □

Theorem 28. *The equivalence relations introduced above are related as follows:*

$$\cong \subset \sim_s \subset \sim_{\&} \subset \sim_o \subset \sim_c.$$

Proof. $\cong \subset \sim_s$. It follows from the fact that the relation $R \stackrel{\text{def}}{=} \{\langle s, \mathcal{I}(s)\rangle \mid s \in \mathcal{S}\}$, where $\mathcal{I}(s)$ denotes the set of states that are isomorphic to s, is a structural bisimulation.

$\sim_s \subset \sim_{\&}$. Let R_s be a structural bisimulation. Notice that for any $\langle s_1, s_2\rangle \in R_s$, the set of free clock variables coincides, and as a consequence, the set of relevant clocks coincides as well, i.e., $\mathsf{Rel}(s_1) = \mathsf{Rel}(s_2)$. We define the relation

$$R_{\&} \stackrel{\text{def}}{=} \left\{\langle s_1, s_2, SR\rangle \mid \langle s_1, s_2\rangle \in R_s \wedge SR = \{\langle\{x\},\{x\}\rangle \mid x \in \mathsf{Rel}(s_1) = \mathsf{Rel}(s_2)\}\right\}.$$

It is not difficult to check that $R_{\&}$ is a symbolic bisimulation. Besides, for any tuple $\langle s_1, s_2, SR\rangle \in R_{\&}$, SR trivially satisfies condition $(*)$ in Definition 25.

$\sim_{\&} \subset \sim_o$. The proof follows from Lemma 44 (see Appendix B). It only remains to notice that if SR satisfies condition $(*)$ in Definition 25, then any $v_1 = v_2$ satisfies the condition (B.1) in Lemma 44.

$\sim_o \subset \sim_c$. Let R be a probabilistic bisimulation relation on the open behaviour of SA. Using Lemma 45 (see Appendix C), it is not difficult to check that R is also a probabilistic bisimulation on the closed behaviour of SA. □

The inclusions in Theorem 28 are strict as shown in Fig. 6. The stochastic automata in Fig. 6A are structural bisimilar, but not isomorphic for evident reasons. The automata in Fig. 6B are not structural bisimilar, since e.g., the initial locations cannot be related (due to different clocks that are set), but are symbolic bisimilar. The automata in Fig. 6C are not symbolic bisimilar as clock x and clocks y and z are not synchronised, but are open p-bisimilar, as both automata can perform an a-action after an equal stochastic delay while evolving to equivalent locations. (The distribution function of the minimum of independent random variables is defined in Proposition 43.) Finally, the stochastic automata in Fig. 6D are closed p-bisimilar as both automata perform an a immediately. They are not open p-bisimilar, because when maximal progress is not considered, action b can take place by waiting long enough in the initial location.

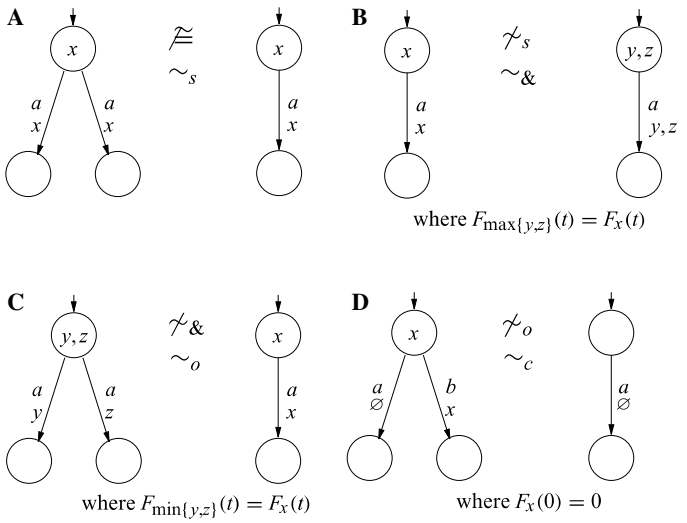

Fig. 6. Comparing equivalences on stochastic automata.

4. Stochastic automata and GSMPs

Generalised semi-Markov processes (GSMPs, for short) [49,19] allow for modelling the behaviour of a wide class of discrete-event systems. They generalise continuous-time Markov chains by allowing, on the one hand, the occurrence time of events to be generally distributed—not only according to memoryless distribution functions—and, on the other hand, that such a timing depends not only on the current state but also on the past states. This section defines GSMPs, defines its semantics in terms of PTSs, and proves that GSMPs are a proper subclass of stochastic automata.

4.1. Generalised semi-Markov processes

A GSMP is defined on top of an automaton sometimes referred to as generalised semi-Markov scheme (GSMS, for short). Transitions in a GSMS are triggered by the occurrence of randomly timed events. A set of active events is associated to each state. These are the events that are possible in that state, i.e., that can cause the execution of a transition. The remaining time until the possible occurrence of an event is determined by an implicit clock; we thus have one clock per event. Clocks are initialised according to a continuous probability distribution function (that only depends on the current state) and run backwards[2] , all with the same pace. We consider discrete-state GSMPs in which transitions are deterministic, i.e., given the current state and the next event, the next state is uniquely determined.

[2] This corresponds to the interpretation of GSMPs with residual lifetimes. An alternative interpretation is to consider spent lifetimes which corresponds to clocks that run forward, as in [5,6].

Definition 29. A *generalised semi-Markov scheme* (GSMS) is a structure $G = (\mathcal{Z}, \mathcal{E}, \mathsf{active}, \mathsf{next}, F)$ where

- $\mathcal{Z}$ is the set of (*output*) *states*;
- $\mathcal{E}$ is a set of *events*;
- $\mathsf{active} : \mathcal{Z} \to \wp(\mathcal{E})$ assigns a set of *active* events to each output state;
- $\mathsf{next} : \mathcal{Z} \times \mathcal{E} \to \mathcal{Z}$ assigns the *next state* according to the current state and the event that is triggered; and
- $F : \mathcal{E} \to (\mathbb{R} \to [0,1])$ assigns to each event a *continuous distribution function* such that $F(e)(0) = 0$; we write F_e instead of $F(e)$.

As initial condition a state $z_0 \in \mathcal{Z}$ is appointed. A *generalised semi-Markov process* (GSMP) is the stochastic process defined by a GSMS.

Example 30. Consider the $G/G/1/\infty$ queue of Example 13. A typical GSMP description of such queuing system is given by the GSMS whose components are defined as follows:

- The set of output states is defined by the non-negative integers $\mathcal{Z} = \mathbb{N}$, where $i \in \mathcal{Z}$ indicates the number of jobs in the system. Initially, the system does not contain any job. We therefore select 0 as the initial state.
- The set of events contains the event a that represents the arrival of a job, and c that represents the completion of a job. Thus, $\mathcal{E} = \{a, c\}$.
- In the initial state 0, there is no job in the system that can be completed. Thus, only an arrival is possible. In the other states either a new job arrives or a job is completed. Hence, $\mathsf{active}(0) = \{a\}$ and $\mathsf{active}(i) = \{a, c\}$ for $i > 0$.
- At any state i, a new job may arrive increasing, as a consequence, the number of jobs in the system to $i{+}1$. At any state $i > 0$ a job can be completed decreasing the number of jobs by one. So, $\mathsf{next}(i, a) = i{+}1$ and $\mathsf{next}(i{+}1, c) = i$.
- The distribution functions associated to the events are the functions F_a and F_c given in Example 13.

This GSMS is depicted in Fig. 7 where a state z is represented by a circle, the set $\mathsf{active}(z)$ is given by the labels of its outgoing arrows, and arrows represent the next state function where $z \xrightarrow{e} z'$ if $\mathsf{next}(z, e) = z'$.

A GSMS behaves as follows. Suppose that the system is in state z. The active events in $\mathsf{active}(z)$ have associated some positive real number. Such number is the time remaining to execute the event. All other events (the inactive ones) have no particular associated value. The

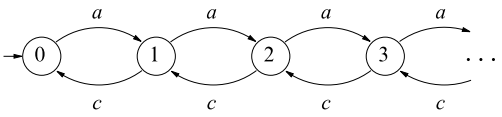

Fig. 7. The GSMS of a $G/G/1/\infty$ queue.

active event with the smallest associated value is selected to be executed. Say $e^\star$ is that event, and $d^\star$ its value. Notice that $e^\star$ is unique with probability 1, since the timing of every event in a GSMS depends on a continuous random variable. Thus, the probability that two active events have the same associated value is zero. The next state is given by $\mathsf{next}(z, e^\star)$. The set of new events $New(z, e^\star)$ is given by

$$New(z, e^\star) \stackrel{\text{def}}{=} \mathsf{active}(\mathsf{next}(z, e^\star)) - (\mathsf{active}(z) - \{e^\star\}),$$

i.e., the events active in the new state which were not active before. The values of these new events are randomly defined according to their distribution function. The set of old events is the set of all active events that remain active:

$$Old(z, e^\star) \stackrel{\text{def}}{=} \mathsf{active}(z) \cap (\mathsf{active}(\mathsf{next}(z, e^\star)) - \{e^\star\}).$$

The value of every old event is decreased by $d^\star$ units of time. This mechanism is known as variable time advance procedure [8,44].

Example 31. In our queuing system example, the sets of new and old events are:

$$New(i, a) = \begin{cases} \{a, c\} & \text{if } i = 0, \\ \{a\} & \text{otherwise,} \end{cases} \quad New(i, c) = \begin{cases} \varnothing & \text{if } i = 1, \\ \{c\} & \text{if } i > 2, \end{cases}$$

$$Old(i, a) = \begin{cases} \varnothing & \text{if } i = 0, \\ \{c\} & \text{otherwise,} \end{cases} \quad Old(i, c) = \{a\} \quad \text{for } i \geqslant 1.$$

We describe the behaviour of a GSMS in terms of probabilistic transition systems.[3]

Let z be a state in $\mathcal{Z}$. Let $e^\star \in \mathsf{active}(z)$ and let $\mathbf{V} \subset \mathcal{E} \to \mathbb{R} \cup \{\perp\}$ be a set of valuations, where $\perp$ represents the undefined value. Let $v \in \mathbf{V}$ and suppose $n = |New(z, e^\star)|$. We define the decoration $\mathcal{D}_v^{z,e^\star} : \mathbb{R}^n \to [\mathcal{Z} \times \mathbf{V}]$ by

$$\mathcal{D}_v^{z,e^\star}(\vec{d}) \stackrel{\text{def}}{=} \left[\mathsf{next}(z, e^\star), v[\overrightarrow{New(z, e^\star)} \leftarrowtail \vec{d}][\mathcal{E} - \mathsf{active}(\mathsf{next}(z, e^\star)) \leftarrowtail \perp]\right]$$

for all $\vec{d} \in \mathbb{R}^n$. Stated in words: if the GSMS is in state z, $e^\star$ is the selected next event, and $\vec{d}$ is the value sampled for the set of new events $New(z, e^\star)$, the next non-deterministic state is determined by the output state $\mathsf{next}(z, e^\star)$ and the valuation where the new events in $New(z, e^\star)$ take the respective sampled value in $\vec{d}$, the inactive events are undefined, while the remaining events (those in $Old(z, e^\star)$) keep their values given by v.

[3] In the literature, the behaviour of a GSMS is defined by giving the distribution function of each transition. This directly corresponds to the probability transition of the probabilistic transition system defined here. For more information the reader is referred to [19,44].

Definition 32. Let $G = (\mathcal{Z}, \mathcal{E}, \mathsf{active}, \mathsf{next}, F)$ be a GSMS. The *behaviour* of G is the probabilistic transition system $PTS(G) \stackrel{\text{def}}{=} (\Sigma, \Sigma', \mathcal{E} \times \mathbb{R}_{\geq 0}, T, \longrightarrow)$ where $\Sigma \subseteq (\mathcal{E} \times (\mathcal{Z} \times \mathbf{V}))$ and $\Sigma' \subseteq [\mathcal{Z} \times \mathbf{V}]$ are defined as follows:

$$\Sigma \stackrel{\text{def}}{=} \{\left(e^{\star}, (z, v)\right) \mid v(e^{\star}) = 0 \wedge (e \in \mathsf{active}(z) \iff v(e) \neq \bot)\},$$
$$\Sigma' \stackrel{\text{def}}{=} \{[z, v] \mid e \in \mathsf{active}(z) \iff v(e) \neq \bot\}$$

and T and $\longrightarrow$ are defined by the following rules:

$$\frac{\overrightarrow{New(z, e^{\star})} = (e_1, \ldots, e_n)}{T\left(e^{\star}, (z, v)\right) = \mathcal{D}_v^{z, e^{\star}}(\mathcal{R}(F_{e_1}, \ldots, F_{e_n}))},$$

$$\frac{e^{\star} \in \mathsf{active}(z) \quad d \in \mathbb{R}_{\geq 0} \quad (v - d)(e^{\star}) = 0 \quad \forall e \in \mathsf{active}(z).\ (v - d)(e) \geqslant 0}{[z, v] \xrightarrow{e^{\star}(d)} \left(e^{\star}, (z, v - d)\right)}.$$

For the special case of the initial state we extend $PTS(G)$ with a new distinguished probabilistic state (z_0) and define

$$\frac{\overrightarrow{\mathsf{active}(z_0)} = (e_1, \ldots, e_n)}{T(z_0) = \mathcal{D}^{z_0}(\mathcal{R}(F_{e_1}, \ldots, F_{e_n}))},$$

where $\mathcal{D}^{z_0}(\vec{d}) \stackrel{\text{def}}{=} \left[z_0, [\overrightarrow{\mathsf{active}(z_0)} \leftarrowtail \vec{d}][\mathcal{E} - \mathsf{active}(z_0) \leftarrowtail \bot]\right]$. Now, the complete behaviour of G is defined by the rooted probabilistic transition system $(PTS(G), (z_0))$.

Notice that T is in fact a function. As a consequence, $PTS(G)$ is a well-defined probabilistic transition system.

4.2. Relating GSMPs to stochastic automata

The relation between stochastic automata and GSMPs is shown by providing a mapping from GSMSs onto stochastic automata. The existence of this mapping indicates that GSMPs are properly included in stochastic automata. We show that the mapping preserves the underlying (closed) behaviour.

Definition 33. Let $G = (\mathcal{Z}, \mathcal{E}, \mathsf{active}, \mathsf{next}, F)$ be a GSMS with initial output state z_0. The translation of G into a stochastic automaton is defined by the rooted stochastic automaton $(SA(G), (z_0, \varnothing))$ where $SA(G) \stackrel{\text{def}}{=} (\mathcal{Z} \times \wp(\mathcal{E}), \mathcal{E}, \mathcal{E}, \rightarrow, \kappa)$ with $\rightarrow$ defined by

$$\frac{e \in \mathsf{active}(z)}{(z,E) \xrightarrow{e,\{e\}} (\mathsf{next}(z,e), \mathsf{active}(z) - \{e\})}$$

and $\kappa((z,E)) \stackrel{\text{def}}{=} \mathsf{active}(z) - E$.

The mapping introduces a location as a pair (z,E) where z is a state of the GSMP and E is the set of events that are already active. The initial location is $(z_0, \varnothing)$. For each active event in the output state z, there is an outgoing edge from any location (z,E). This edge is labelled with event e (i.e., the action) and the set of clocks $\{e\}$. Therefore $\mathsf{active}(z) = \{e \mid (z,E) \xrightarrow{e,\{e\}}\}$. Since in a GSMP exactly one clock is associated to an event, we obtain singleton sets as trigger sets.

There are many locations (z,E) that are unreachable. All reachable locations have the form $(\mathsf{next}(z,e), \mathsf{active}(z) - \{e\})$ for every (reachable) $z \in \mathcal{Z}$ and $e \in E$. Notice that, for these reachable locations, $\kappa((\mathsf{next}(z,e), \mathsf{active}(z) - \{e\})) = \mathsf{active}(\mathsf{next}(z,e)) - (\mathsf{active}(z) - \{e\}) = \mathit{New}(z,e)$.

Example 34. Fig. 8 depicts the reachable part of the stochastic automaton obtained from the GSMS model of the $G/G/1/\infty$ queue as defined in Example 30. Note that this stochastic automaton is isomorphic to the one given in Example 13.

The correctness of the translation is stated by showing that the behaviour of a GSMS G is closed p-bisimilar to $SA(G)$.

Theorem 35. *$(PTS(G), (z_0))$ and $(PTS_c(SA(G)), ((z_0, \varnothing), v))$ are probabilistic bisimilar for any valuation v.*

Proof. Let relation R be defined by the symmetric closure of the set

$$\begin{aligned}
&\left\{\left\langle\left(e^{\star}, (z,v)\right), \left((\mathsf{next}(z,e^{\star}), \mathsf{active}(z) - \{e^{\star}\}), v'\right)\right\rangle \mid \forall e \in \mathsf{active}(z).\ v(e) = v'(e)\right\} \\
&\quad \cup \quad \left\{\langle (z_0), ((z_0, \varnothing), v)\rangle \mid v \in \mathbf{V}\right\} \\
&\quad \cup \quad \left\{\langle [z,v], [(z,C), v']\rangle \mid \forall e \in \mathsf{active}(z).\ v(e) = v'(e)\right\}
\end{aligned}$$

The proof that R is a probabilistic bisimulation up to $\sim_p$ is routine. □

Notice that a translation of stochastic automata into GSMSs is not possible in general as stochastic automata allow non-continuous probability distribution functions and may exhibit non-determinism.

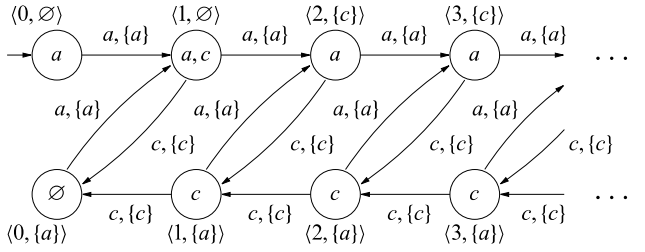

Fig. 8. The translation of the GSMS of Fig. 7.

5. Related work

5.1. Probabilistic transition systems

The notion of probabilistic transition systems studied in the first part of this paper is a generalisation of the discrete probabilistic transition systems, like [47,39,36,17,21,42]. This generalisation allows the representation of soft real-time systems in which time constraints are not necessarily deterministic but have some random nature. PTSs are inspired by the discrete probabilistic models of Hansson [22,21] and Segala [42,43], and are related to the continuous model of Strulo [46,24,25]. Alternating probabilistic and non-deterministic transitions have been introduced by Hansson in a discrete probabilistic model. Strulo considers a continuous extension of Hansson's alternating model. These models do not explicitly consider probability spaces as a structure; instead, probabilistic transitions are labelled with numbers. For continuous distributions this model tends to be cumbersome. The definition of probabilistic transitions as mappings onto probability spaces of states originates from Segala. In his model, there is no distinction between probabilistic and non-deterministic states. As Segala's work is focused on studying randomised distributed algorithms, discrete probability spaces do suffice in his setting. It can be shown that discrete PTSs (in our sense) are as expressive as Segala's *simple* probabilistic automata [11]. Cattani et al. [9] have recently proposed stochastic transition systems as continuous variant of probabilistic automata.

5.2. Probabilistic bisimulation

Probabilistic bisimulation has originally been defined on reactive probabilistic transition systems by Larsen and Skou [36] and has been adapted to generative [16] and stratified probabilistic models [17]. Hansson considered a variant of probabilistic bisimulation tailored to a discrete-time version of his alternating model. Segala and Lynch [43] extended Larsen-Skou probabilistic bisimulation to simple probabilistic automata. Their notion coincides with the discrete variant of the probabilistic bisimulation defined in this paper. All these works consider probabilities in a discrete setting. Notions of probabilistic bisimulation in a continuous setting have received scant attention in the literature. Notable exceptions for the setting with general distributions are the probabilistic bisimulation in a continuous setting by Harrison and Strulo [46,24,25] which coincides with our notion, the co-algebraic characterisation of continuous probabilistic bisimulation by de Vink and Rutten [48], the categorical definition of Desharnais et al. [15], and the recent notion in Cattani et al. [9] that naturally extends trace distributions.

5.3. Stochastic and timed automata

Stochastic automata are a symbolic model for finitely representing continuous-time probabilistic systems. These automata are inspired by the timed automata of Alur and Dill [2] and generalised semi-Markov processes (GSMPs) by Glynn [19] and Whitt [49]. (Extensive introductions to GSMPs can be found in Cassandras [8] and Shedler [44]). Timed automata are extensions of state-transition diagrams with clock variables. Whereas in timed automata clocks are set to a deterministic value (and then run forwards), in stochastic automata clocks are initialised according to a probability distribution function (and run backwards). Similar to timed automata, whose semantics is typically

defined in terms of timed transition systems that are intrinsically infinite, stochastic automata are interpreted in terms of infinite probabilistic transition systems, in fact a probabilistic counterpart to timed transition systems. A translation of stochastic automata into timed automata with deadlines that abstracts from the stochastic information has been studied in [12]. Other probabilistic extensions of timed automata include discrete probabilistic branching (but not stochastic clocks) [30,31], and automata in which the location residence time is governed by a distribution that is positive on a finite set of intervals [1]. Stochastic automata are closely related to the continuous probabilistic timed automata recently considered by Kwiatkowska et al. [32]. Whereas stochastic automata are developed for specification purposes, [1,31,32] focus on the model-checking problem and neither study bisimulation relations nor compositionality issues.

5.4. Stochastic automata and interactive GSMPs

A variant of stochastic automata, called interactive GSMPs, in which clocks run forward has been proposed by Bravetti and Gorrieri in, amongst others, [5,7]. These variants of GSMPs (with a residual lifetime interpretation) generalize Hermanns' interactive Markov chains [27], and equip GSMPs with non-determinism. [5,7] study (strong and weak) bisimulations and compositionality issues (like hiding and parallel composition).

The main technical differences between IGSMPs and stochastic automata are as follows. In IGSMPs, actions are considered as combinations of start and termination events like in ST-semantics [18], a well-studied true concurrency semantics. This allows for, for instance, the transfer of various results from ST-semantics such as weak bisimulations and action refinement to the stochastic setting. In our case, start and termination events do not always occur strictly as pairs, e.g., clocks that are set may never be "used" (i.e., terminated), or clocks may be used in trigger sets more than once. A detailed study reporting extensively on the differences between these approaches has recently been provided by Bravetti and D'Argenio [6].

6. Conclusions and discussion

This paper presented stochastic automata, a new model for symbolically representing probabilistic transition systems that allow to cope with general probability spaces. Stochastic automata are in particular suited for modelling softly timed systems, real-time systems where time constraints are of quantitative nature—"an activity should occur within t time units in 99% of the cases." The semantics of stochastic automata have been defined in terms of probabilistic transition systems with general probability spaces. Notions of (probabilistic) bisimilarity have been defined and their relationship has been established. Furthermore, we proved that generalised semi-Markov processes are a proper subset of stochastic automata. An important property of stochastic automata is their compositional nature. This is discussed in full detail in [13] where stochastic automata are used as semantic model for a process algebra in which action-delays are governed by general distributions, and is also illustrated by their use in providing a semantics to a stochastic extension of UML statecharts [29,20].

This paper does not address the analysis of stochastic automata. Due to the general nature of the distributions involved, quantitative properties, such as the delay between two activities

or the probability of a certain behaviour, can be established by means of discrete-event simulation [8,44]. Prior to the simulation phase, non-determinism is resolved by means of adversaries [43,47]. Simulation of stochastic automata is described in [11,14]. Recently, [50] introduced a simulation-based method to assess the validity on CSL (Continuous Stochastic Logic) formulas [3] on GSMPs. As the process is statistically based, answers may be unsound but the likelihood of error is bounded using statistical hypothesis testing. This technique can therefore be lifted to stochastic automata. For a restricted set of stochastic automata, the insensitive GSMPs [41], numerical methods can be used to assess their steady-state behaviour. Alternatively, in case of absence of non-determinism, approximation results can be employed, and arbitrary distributions can be approximated by phase-type distributions [38]. This results in a continuous-time Markov chain for which efficient numerical methods exist. To assess qualitative properties of stochastic automata such as (untimed) safety properties, standard reachability techniques can be used such as model checking [10], after abstraction of the stochastic ingredients [11,14,12].

Acknowledgments

We thank Ed Brinksma for his collaboration in this research and Gabriel Infante López who unrevealed a flaw in the original definition of symbolic bisimulation [11]. The remarks of the reviewers have significantly improved the presentation of the paper. Part of this work was done while the first author was working for the STW/PROGRESS project TES-4999 "Verification of Hard and Softly Timed Systems (HaaST)" at the University of Twente.

Appendix A. Preliminaries on Probability Theory

This appendix recalls some notions of probabilities which are necessary for the understanding of this article. The reader is referred to [45,33,40] for further reading.

A.1. Probability spaces and measurable functions

Definition 36. Let Ω be a set called *sample space*. A collection $\mathcal{F}$ of subsets of Ω is a *σ-algebra* if

(1) $\Omega \in \mathcal{F}$;
(2) $A \in \mathcal{F} \Rightarrow A^c \in \mathcal{F}$; and
(3) $\forall i \in \mathbb{N}. A_i \in \mathcal{F} \Rightarrow \bigcup_{i\in\mathbb{N}} A_i \in \mathcal{F}$.

The elements of a σ-algebra are called *measurable sets* and the pair $(\Omega, \mathcal{F})$ is a *measurable space*. A *probability measure* on $(\Omega, \mathcal{F})$ is a function $P : \mathcal{F} \to [0,1]$ such that the following properties hold:

(1) $P(\varnothing) = 0$, $P(\Omega) = 1$.
(2) $P\left(\bigcup_{i\in N} A_i\right) = \sum_{i\in N} P(A_i)$ for every pairwise disjoint family $(A_i)_{i\in N}$ of measurable sets in the σ-algebra $\mathcal{F}$ (with $N \subseteq \mathbb{N}$).

The structure $(\Omega, \mathcal{F}, P)$ is called a *probability space*. In particular, if there is a countable set $A \in \Omega$ such that $\{a\} \in \mathcal{F}$ and $\sum_{a \in A} P(\{a\}) = 1$, P is called a *discrete probability measure* and $(\Omega, \mathcal{F}, P)$ is a *discrete* probability space.

The support set of a probability measure is the smallest closed subset of the sample space whose measure is 1. That is, the *support set* of P is the set

$$\mathrm{supp}(P) \stackrel{\text{def}}{=} \Omega - \cup \left\{ A \in \mathcal{F} \mid A \text{ is open}^4 \wedge P(A) = 0 \right\}.$$

Definition 37. Let $(\Omega, \mathcal{F})$ and $(\Omega', \mathcal{F}')$ be measurable spaces. A function $f : \Omega \to \Omega'$ is a *measurable function* if $f^{-1}(A) \stackrel{\text{def}}{=} \{a \mid f(a) \in A\} \in \mathcal{F}$, for all $A \in \mathcal{F}'$.

Observe that, if P is a probability measure on $(\Omega, \mathcal{F})$, $P \circ f^{-1}$ is a probability measure on $(\Omega', \mathcal{F}')$.

Proposition 38. *If $f : \Omega \to \Omega'$ is a measurable function, $(\Omega, \mathcal{F}, P)$ is a probability space and $(\Omega', \mathcal{F}')$ is a measurable space, then $(\Omega', \mathcal{F}', P \circ f^{-1})$ is a probability space.*

Let $\mathcal{P} = (\Omega, \mathcal{F}, P)$ be a probability space and $\mathcal{D} : \Omega \to \Omega'$ be injective. We lift $\mathcal{D}$ to subsets of Ω as usual: $\mathcal{D}(A) \stackrel{\text{def}}{=} \{\mathcal{D}(a) \mid a \in A\}$. Observe that $\mathcal{D}(\mathcal{F}) \stackrel{\text{def}}{=} \{\mathcal{D}(A) \mid A \in \mathcal{F}\}$ is a σ-algebra on the sample space $\mathcal{D}(\Omega)$. As a consequence, $\mathcal{D}$ is a measurable function and, by the previous proposition, $\mathcal{D}(\mathcal{P}) \stackrel{\text{def}}{=} (\mathcal{D}(\Omega), \mathcal{D}(\mathcal{F}), P \circ \mathcal{D}^{-1})$ is a probability space. Since $\mathcal{D}(\mathcal{P})$ is basically the same probability space as $\mathcal{P}$, we say that $\mathcal{D}$ is a *decoration* and we refer to $\mathcal{D}(\mathcal{P})$ as the *decoration of $\mathcal{P}$ according to $\mathcal{D}$*. Decoration is a key concept in the semantics of stochastic automata.

A.2. Borel spaces and probability measures

Borel algebras are an important class of σ-algebras. In particular we are interested in the Borel algebras defined on a Cartesian product of the real numbers.

Definition 39. For every $k \in \{1, \ldots, n\}$, let $I_k = (a_k, b_k]$ with $a_k, b_k \in \mathbb{R} \cup \{-\infty, \infty\}$. The interval $(a, \infty]$ is taken to be (a, ∞).

The set $I = I_1 \times \ldots \times I_n \subseteq \mathbb{R}^n$ is called a *rectangle*.

Let $\mathcal{I}$ be the *set of all rectangles*. The *Borel algebra* of subsets of $\mathbb{R}^n$, denoted by $\mathcal{B}(\mathbb{R}^n)$, is the smallest σ-algebra containing $\mathcal{I}$. The measurable space $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ is called a *Borel space*.

We only consider probability spaces that are isomorphic to some Borel space defined on a real hyperspace (as in Definition 39) whose coordinates are determined by independent random variables. The class of probability measures used is defined by the following theorem [45, II–§3].

Theorem 40. *For $i \in \{1, \ldots, n\}$, let F_i be a distribution function. There is a unique probability measure P on $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ such that*

$$P((a_1, b_1], \ldots, (a_n, b_n]) = \prod_{i=1}^{n} (F_i(b_i) - F_i(a_i))$$

[4] Formally speaking, $(\Omega, \mathcal{F})$ should be a locally compact Hausdorff space (see e.g. [40]). In this article, we only use this kind of measurable spaces.

with $-\infty \leqslant a_i < b_i < \infty$, *for* $i \in \{1,\ldots,n\}$ *and where* $\prod_{i=1}^{n} d_i$ *is a shorthand notation for* $d_1 \cdot d_2 \cdot \cdots \cdot d_n$.

As a consequence of this theorem we uniquely identify by $\mathcal{R}(F_1,\ldots,F_n)$ the probability space $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n), P_n)$ where $\mathcal{B}(\mathbb{R}^n)$ is the Borel algebra on $\mathbb{R}^n$ and P_n is the unique probability measure obtained as in Theorem 40 from a given family of distribution functions $F_1,\ldots,F_n$. In particular, if $n = 0$, $\mathcal{R}()$ is the trivial probability space $(\{\varnothing\}, \{\varnothing, \{\varnothing\}\}, P_0)$ with P_0 in the obvious way.

A.3. Random variables

Definition 41. Let $(\Omega, \mathcal{F})$ be a measurable space and $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$ the Borel space on the real line. A measurable function $X : \Omega \to \mathbb{R}$ is called a *random variable*. If P is a probability measure on $(\Omega, \mathcal{F})$, the probability measure $P_X \stackrel{\text{def}}{=} P \circ X^{-1}$ on $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$ is the *probability distribution of X*. The function F_X, defined by

$$F_X(d) \stackrel{\text{def}}{=} P_X((-\infty, d]) = P(\{a \mid X(a) \leqslant d\})$$

with $d \in \mathbb{R}$, is the *distribution function of X*. The *support set* of F_X is defined by $\text{supp}(F_X) \stackrel{\text{def}}{=} \text{supp}(P_X)$.

Definition 42. Let $\mathcal{P} = (\Omega, \mathcal{F}, P)$ be a probability space. Random variables $X_1,\ldots,X_n$ on $\mathcal{P}$ are *independent* if for all $B_i \in \mathcal{B}(\mathbb{R})$, $i = 1,\ldots,n$:

$$P\left(\bigcap_{i=1}^{n} X_i^{-1}(B_i)\right) = \prod_{i=1}^{n} P_{X_i}(B_i).$$

Proposition 43. *Let* $X_1,\ldots,X_n$ $(n \geqslant 1)$ *be independent random variables on* $(\Omega, \mathcal{F}, P)$ *where* X_i *has distribution function* F_{X_i}, *and* $d \in \mathbb{R}$.

(1) *The distribution function* $Y = \max(X_1,\ldots,X_n)$ *is given by*:

$$F_Y(d) = \prod_{i=1}^{n} F_{X_i}(d).$$

(2) *The distribution function of* $Z = \min(X_1,\ldots,X_n)$ *is given by*:

$$F_Z(d) = 1 - \prod_{i=1}^{n} (1 - F_{X_i}(d)).$$

B. Auxiliary lemma for the proof of $\sim_{\&} \subseteq \sim_o$

Let $\max_v(C) = \max\{v(x) \mid x \in C\}$. In particular, we assume $\max \varnothing = 0$.

Lemma 44. *Let $R_{\&}$ be a symbolic bisimulation on a given stochastic automaton SA. Let R_o be the symmetric closure of $R_p \cup R_n$, where*

$$\begin{aligned} R_p = \big\{ \langle (s_1, v_1), (s_2, v_2) \rangle \mid \langle s_1, s_2, SR \rangle \in R_{\&} \\ \wedge \forall \langle C_1, C_2 \rangle \in SR \ - (\wp(\kappa(s_1)) \times \wp(\kappa(s_2))). \\ \max_{v_1}(C_1) = \max_{v_2}(C_2) \vee (\max_{v_1}(C_1) \leqslant 0 \wedge \max_{v_2}(C_2) \leqslant 0) \big\} \end{aligned} \tag{B.1}$$

$$\begin{aligned} R_n = \big\{ \langle [s_1, v_1], [s_2, v_2] \rangle \mid \langle s_1, s_2, SR \rangle \in R_{\&} \\ \wedge \forall \langle C_1, C_2 \rangle \in SR \ . \ \max_{v_1}(C_1) = \max_{v_2}(C_2) \\ \vee (\ C_1 \cap \kappa(s_1) = \varnothing = C_2 \cap \kappa(s_2) \wedge \max_{v_1}(C_1) \leqslant 0 \wedge \max_{v_2}(C_2) \leqslant 0 \ ) \big\}. \end{aligned} \tag{B.2}$$

Then R_o is a probabilistic bisimulation up to $\sim_p$ in $PTS_o(SA)$.

Proof. By definition, R_o is symmetric. Thus, we directly proceed to prove the transfer properties in Definition 9. We only prove the straight cases. The symmetric ones follow in a similar way.

1. Let $\langle (s_1, v_1), (s_2, v_2) \rangle \in R_o$. Then $\langle s_1, s_2, SR \rangle \in R_{\&}$ satisfies the requirements in (B.1).
   Let $\overrightarrow{\kappa(s_i)} = (x^i_1, \ldots, x^i_{n_i})$ for $i \in \{1, 2\}$. By Definition 16 (rule **Prob**), for $i \in \{1, 2\}$,

$$T(s_i, v_i) = \mathcal{D}^{s_i}_{v_i}(\mathcal{R}(F_{x^i_1}, \ldots, F_{x^i_{n_i}})) = (\Omega_i, \mathcal{F}_i, P_i)$$

Let N be the number of pairs $\langle C_1, C_2 \rangle \in SR$ such that $C_1 \subseteq \kappa(s_1)$ (or equivalently $C_2 \subseteq \kappa(s_2)$). Let $\langle C^1_1, C^1_2 \rangle$, …, $\langle C^N_1, C^N_2 \rangle$ be those tuples. Because SR is a synchronisation relation and satisfies condition 1(ii) of Definition 25, $\kappa(s_i) = \bigcup_{k=1}^{N} C^k_i$ and $C^1_i, \ldots, C^N_i$ are pairwise disjoint. As a consequence, we may assume that, for all $k \in \{1, \ldots, N\}$,

$$C^k_i = \{x^i_{\left(\sum_{l=1}^{k-1} m^i_l\right)+1}, \ldots, x^i_{\sum_{l=1}^{k} m^i_l}\},$$

where $\#C^k_i = m^i_k$ (and $n_i = \sum_{l=1}^{N} m^i_l$).

For $(d_1, \ldots, d_N) \in \mathbb{R}^N$ define

$$[d_1, \ldots, d_N]^{s_i}_{v_i} \stackrel{\text{def}}{=} \{[s_i, v] \mid \forall k \in \{1, \ldots, N\}. \ \max_v(C^k_i) = d_k \wedge \forall x \notin \kappa(s_i). \ v(x) = v_i(x)\}$$

Then, we have the following claim whose proof can be found later.

Claim 44.1.
1. If $[s,v] \in [d_1,\ldots,d_N]^{s_1}_{v_1}$ *and* $\left[s',v'\right] \in [d_1,\ldots,d_N]^{s_2}_{v_2}$, *then* $\langle [s,v], \left[s',v'\right]\rangle \in (\sim_p \cup R_o)^*$.
2. For $i \in \{1,2\}$, *if* $[s,v], \left[s',v'\right] \in [d_1,\ldots,d_N]^{s_i}_{v_i}$, *then* $\langle [s,v], \left[s',v'\right]\rangle \in (\sim_p \cup R_o)^*$.

From this claim it follows that for any $(d_1,\ldots,d_N) \in \mathbb{R}^N$ and any $[s,v] \in [d_1,\ldots,d_N]^{s_1}_{v_1} \cup [d_1,\ldots,d_N]^{s_2}_{v_2}$,

$$[d_1,\ldots,d_N]^{s_1}_{v_1} \cup [d_1,\ldots,d_N]^{s_2}_{v_2} \;\subseteq\; \left[\,[s,v]\,\right]_{(\sim_p \cup R_o)^*} \;\in\; [\mathcal{S}\times\mathbf{V}]/(\sim_p \cup R_o)^*,$$

where $\left[\,[s,v]\,\right]_{(\sim_p \cup R_o)^*}$ is the equivalence class of $[s,v]$ induced by the equivalence relation $(\sim_p \cup R_o)^*$. As a consequence, there is a unique $D^s_v \subseteq \mathbb{R}^N$ such that

$$\begin{aligned}\left[\,[s,v]\,\right]_{(\sim_p \cup R_o)^*} = \quad & \left(\bigcup_{(d_1,\ldots,d_N)\in D^s_v} [d_1,\ldots,d_N]^{s_1}_{v_1} \cup [d_1,\ldots,d_N]^{s_2}_{v_2}\right) \\ \cup \; & \left(\left[\,[s,v]\,\right]_{(\sim_p \cup R_o)^*} - ([\mathbb{R}^N]^{s_1}_{v_1} \cup [\mathbb{R}^N]^{s_2}_{v_2})\right),\end{aligned}$$

where $[\mathbb{R}^N]^s_v = \cup\{[d_1,\ldots,d_N]^s_v \mid (d_1,\ldots,d_N) \in \mathbb{R}^N\}$.
Let $S \subseteq [\mathcal{S}\times\mathbf{V}]/(\sim_p \cup R_o)^*$ and let $\mathbf{D} = \left\{D^s_v \mid \left[\,[s,v]\,\right]_{(\sim_p \cup R_o)^*} \subseteq \cup S\right\}$. Then,

$$\cup S = \left(\bigcup_{D'\in\mathbf{D}} \bigcup_{(d_1,\ldots,d_N)\in D'} [d_1,\ldots,d_N]^{s_1}_{v_1} \cup [d_1,\ldots,d_N]^{s_2}_{v_2}\right) \cup \left(\cup S - ([\mathbb{R}^N]^{s_1}_{v_1} \cup [\mathbb{R}^N]^{s_2}_{v_2})\right).$$

Therefore, we can state that there is a unique set $D \subseteq \mathbb{R}^N$ (taking $D = \cup\mathbf{D}$) such that for $i \in \{1,2\}$,

$$(\Omega_i \cap \cup S) \;=\; \bigcup_{(d_1,\ldots,d_N)\in D} [d_1,\ldots,d_N]^{s_i}_{v_i} \tag{B.3}$$

For $i \in \{1,2\}$, define $M_i : \mathbb{R}^{n_i} \to \mathbb{R}^N$ by

$$\begin{aligned}&M_i(d_1,..,d_{m^i_1},d_{m^i_1+1},..,d_{m^i_1+m^i_2},\ldots,d_{(\sum_{l=1}^{N-1} m^i_l)+1},..,d_{n_i}) \stackrel{\text{def}}{=} \\ &\qquad (\max\{d_1,..,d_{m^i_1}\},\max\{d_{m^i_1+1},..,d_{m^i_1+m^i_2}\},\ldots,\max\{d_{(\sum_{l=1}^{N-1} m^i_l)+1},..,d_{n_i}\})\end{aligned}$$

where $\#C^k_i = m^i_k$, for $k \in \{1,\ldots,N\}$ (and $n_i = \sum_{l=1}^{N} m^i_l$).
The proof of the following claim can be found on page 35.

Claim 44.2. *For all* $(d_1,\ldots,d_N) \in \mathbb{R}^N$, *and any* $i \in \{1,2\}$:

$$[d_1,\ldots,d_N]^{s_i}_{v_i} = (\mathcal{D}^{s_i}_{v_i} \circ M_i^{-1})(d_1,\ldots,d_N)$$

For $i \in \{1,2\}$ and $k \in \{1,\ldots,N\}$, let $\max(C^k_i)$ be the random variable that maximizes all random variables in C^k_i. Its distribution function is given by

$$F_{\max(C_i^k)}(t) = \mathrm{Prob}(\max(C_i^k) \leqslant t)$$
$$= \mathrm{Prob}\left((x^i_{\substack{k-1 \\ (\sum_{l=1} m_l^i)+1}} \leqslant t) \wedge \cdots \wedge (x^i_{\substack{k \\ \sum_{l=1} m_l^i}} \leqslant t) \right)$$
$$= \prod_{h=1}^{m_k^i} \mathrm{Prob}\left(x^i_{\substack{k-1 \\ (\sum_{l=1} m_l^i)+h}} \leqslant t \right) = \prod_{h=1}^{m_k^i} F_{x^i_{\substack{k-1 \\ (\sum_{l=1} m_l^i)+h}}}(t).$$

Therefore, because of condition 1(iii) of symbolic bisimulation (see Definition 25), $F_{\max(C_1^k)} = F_{\max(C_2^k)}$ for all $k \in \{1,\ldots,N\}$. Hence

$$\mathcal{R}(F_{\max(C_1^1)},\ldots,F_{\max(C_1^N)}) = \mathcal{R}(F_{\max(C_2^1)},\ldots,F_{\max(C_2^N)})$$

Notice that $M_i(\mathcal{R}(F_{x_1^i},\ldots,F_{x_{n_i}^i})) = \mathcal{R}(F_{\max(C_i^1)},\ldots,F_{\max(C_i^N)})$ for $i \in \{1,2\}$ is measurable.

As a consequence, if P is the probability measure of $\mathcal{R}(F_{\max(C_1^1)},\ \ldots,F_{\max(C_1^N)})$ (and hence also of $\mathcal{R}(F_{\max(C_2^1)},\ldots,F_{\max(C_2^N)})$), and P_i' is the probability measure of $\mathcal{R}(F_{x_1^i},\ldots,F_{x_{n_i}^i})$ for $i \in \{1,2\}$,

$$P = P_i' \circ M_i^{-1} \tag{B.4}$$

For $i \in \{1,2\}$, we calculate:

$$\begin{aligned}
&\mu((s_i,v_i),\cup S)\\
&\quad = P_i(\Omega_i \cap \cup S) \qquad \text{(Def. of } \mu)\\
&\quad = P_i\left(\bigcup_{(d_1,\ldots,d_N)\in D}[d_1,\ldots,d_N]_{v_i}^{s_i}\right) \qquad \text{(by } (B.3))\\
&\quad = P_i((\mathcal{D}_{v_i}^{s_i} \circ M_i^{-1})(d_1,\ldots,d_N)) \qquad \text{(Claim 44.2)}\\
&\quad = (P_i' \circ (\mathcal{D}_{v_i}^{s_i})^{-1})((\mathcal{D}_{v_i}^{s_i} \circ M_i^{-1})(d_1,\ldots,d_N)) \qquad \text{(Def. of } P_i)\\
&\quad = (P_i' \circ M_i^{-1})(d_1,\ldots,d_N) \qquad ((\mathcal{D}_{v_i}^{s_i})^{-1} \circ \mathcal{D}_{v_i}^{s_i} = id \quad \text{since } \mathcal{D}_{v_i}^{s_i} \text{ is injective})\\
&\quad = P(d_1,\ldots,d_N) \qquad \text{(by } (B.4)).
\end{aligned}$$

Therefore, $\mu((s_1,v_1),\cup S) = \mu((s_2,v_2),\cup S)$, which concludes this part of the proof.

2. Let $\langle [s_1,v_1],[s_2,v_2]\rangle \in R_o$ and suppose that $[s_1,v_1] \xrightarrow{a(d)} (s_1', v_1 - d)$. Then, by rule **Open** in Definition 16, $s_1 \xrightarrow{a,C_1^\star} s_1'$ and $\mathsf{exp}_d(v_1, C_1^\star)$ (i.e., $\forall x \in C_1^\star.\ (v_1 - d)(x) \leqslant 0$). Since $\langle s_1, s_2, SR\rangle \in R_\&$, by item (2) in Definition 25,

$$s_2 \xrightarrow{a,C_2^\star} s_2'. \tag{B.5}$$

By Definition 23, $C_i^\star \subseteq \mathsf{Rel}(s_i)$. From this and items 1(i) and 2(i) in Definition 25, there is a set of pairs $\langle C_1^1, C_2^1\rangle, \ldots, \langle C_1^N, C_2^N\rangle \in SR$, for some $N \geqslant 0$, such that $C_i^\star = \bigcup_k C_i^k$, with $i = 1, 2$. Due to conditions in (B.2), it follows that, for all $k \in \{1, \ldots, N\}$, either $\max_{v_1}(C_1^k) = \max_{v_2}(C_2^k)$ or $\max_{v_1}(C_1^k) \leqslant 0$ and $\max_{v_2}(C_2^k) \leqslant 0$. As a consequence $\max_{v_2}(C_2^\star) \leqslant d$. That is, for all $x \in C_2^\star$, $v_2(x) \leqslant d$, and hence

$$\forall y \in C_2^\star.\ (v_2 - d)(y) \leqslant 0 \quad (\text{i.e., } \mathsf{exp}_d(v_2, C_2^\star)). \tag{B.6}$$

From (B.5), (B.6), and rule **Open**, it follows that $[s_2, v_2] \xrightarrow{a(d)} (s_2', v_2 - d)$.

It remains to prove that $\langle (s_1', v_1 - d), (s_2', v_2 - d)\rangle \in R_o$, which reduces to prove the conditions in (B.1). Because of item 2(ii) in Definition 25, $\langle s_1', s_2', SR'\rangle \in R_\&$ for some SR' satisfying forward compatibility:

$$\begin{aligned}&\left\{\langle C_1, C_2\rangle \in SR \mid (C_1 \cap \mathsf{Fv}(s_1')) \cup (C_2 \cap \mathsf{Fv}(s_2')) \neq \varnothing\right\} - \left(\wp(C_1^\star) \times \wp(C_2^\star)\right)\\ &\quad = \left\{\langle C_1', C_2'\rangle \in SR' \mid (C_1' \cap \mathsf{Fv}(s_1')) \cup (C_2' \cap \mathsf{Fv}(s_2')) \neq \varnothing\right\} - \left(\wp(C_1^\star) \times \wp(C_2^\star)\right).\end{aligned}$$

We consider now two cases. First, let $\langle C_1, C_2\rangle \in SR' - ((\wp(C_1^\star) \times \wp(C_2^\star)) \cup (\wp(\kappa(s_1')) \times \wp(\kappa(s_2'))))$. Then, for $i \in \{1, 2\}$, $C_i \subseteq \mathsf{Rel}(s_i') - \kappa(s_i') = \mathsf{Fv}(s_i')$. Therefore, $\langle C_1, C_2\rangle \in \left\{\langle C_1', C_2'\rangle \in SR' \mid (C_1' \cap \mathsf{Fv}(s_1')) \cup (C_2' \cap \mathsf{Fv}(s_2')) \neq \varnothing\right\} - \left(\wp(C_1^\star) \times \wp(C_2^\star)\right)$, and hence $\langle C_1, C_2\rangle \in SR$. As a consequence

$$\max_{v_1}(C_1) = \max_{v_2}(C_2) \vee (\ \max_{v_1}(C_1) \leqslant 0\ \wedge\ \max_{v_2}(C_2) \leqslant 0\ ),$$

because of (B.2). Now, it is immediate that

$$\max_{(v_1-d)}(C_1) = \max_{(v_2-d)}(C_2) \vee (\ \max_{(v_1-d)}(C_1) \leqslant 0\ \wedge\ \max_{(v_2-d)}(C_2) \leqslant 0\ ). \tag{B.7}$$

Second, let $\langle C_1, C_2\rangle \in SR' \cap (\wp(C_1^\star) \times \wp(C_2^\star))$, which implies that $\mathsf{exp}_d(v_1, C_1)$ and $\mathsf{exp}_d(v_2, C_2)$. Therefore

$$\max_{(v_1-d)}(C_1) \leqslant 0\ \wedge\ \max_{(v_2-d)}(C_2) \leqslant 0. \tag{B.8}$$

From (B.7) and (B.8), for all $\langle C_1, C_2\rangle \in SR' - (\wp(\kappa(s_1')) \times \wp(\kappa(s_2')))$,

$$\max_{(v_1-d)}(C_1) = \max_{(v_2-d)}(C_2)\ \vee\ (\ \max_{(v_1-d)}(C_1) \leqslant 0\ \wedge\ \max_{(v_2-d)}(C_2) \leqslant 0).$$

Since $\langle s_1', s_2', SR'\rangle \in R_\&$, (B.1) is satisfied by $\langle (s_1', v_1 - d), (s_2', v_2 - d)\rangle \in R_o$. □

Proof [*of Claim 44.1*].
1. We show that $\langle [s,v], [s',v'] \rangle \in R_o$, which implies Claim 44.1(1). Let $[s,v] \in [d_1,\ldots,d_N]^{s_1}_{v_1}$ and $[s',v'] \in [d_1,\ldots,d_N]^{s_2}_{v_2}$. Then

$$s = s_1, \quad s' = s_2, \quad \forall k \in \{1,\ldots,N\}.\ \max_v(C_1^k) = d_k = \max_{v'}(C_2^k),$$
$$\forall x \notin \kappa(s_1).\ v(x) = v_1(x) \text{ and } \forall x \notin \kappa(s_2).\ v'(x) = v_2(x)$$

Recall that $\langle (s_1,v_1),(s_2,v_2) \rangle \in R_o$ satisfies the requirements in (B.1). In particular $\langle s_1, s_2, SR \rangle \in R_\&$. So, it suffices to show that $\langle [s,v], [s',v'] \rangle$ satisfies the requirements in (B.2). First, $\langle s, s', SR \rangle = \langle s_1, s_2, SR \rangle \in R_\&$.

Next, take $\langle C_1, C_2 \rangle \in SR$. If $C_1 \cap \kappa(s_1) \neq \varnothing$, then $C_1 \subseteq \kappa(s_1)$ and $C_2 \subseteq \kappa(s_2)$. Therefore there exists $k \in \{1,\ldots,N\}$ s.t. $\langle C_1, C_2 \rangle = \langle C_1^k, C_2^k \rangle$ from which it follows that $\max_v(C_1) = \max_{v'}(C_2)$.

If $C_1 \cap \kappa(s_1) = \varnothing$, then also $C_2 \cap \kappa(s_2) = \varnothing$ and hence

$$\max_v(C_1) = \max\{v(x) \mid x \in C_1\} = \max\{v_1(x) \mid x \in C_1\} = \max_{v_1}(C_1)$$

Similarly, $\max_{v'}(C_2) = \max_{v_2}(C_2)$. Because $\langle (s_1,v_1),(s_2,v_2) \rangle \in R_o$ satisfies the requirements in (B.1),

$$\max_v(C_1) = \max_{v'}(C_2) \vee (\max_v(C_1) \leqslant 0 \wedge \max_{v'}(C_2) \leqslant 0).$$

This completes the proof that $\langle [s,v], [s',v'] \rangle \in R_o$ and hence this part of the claim.

2. Let $[s,v], [s',v'] \in [d_1,\ldots,d_N]^{s_1}_{v_1}$. Notice that $[d_1,\ldots,d_N]^{s_2}_{v_2}$ is never empty. Then there exists $[s'',v''] \in [d_1,\ldots,d_N]^{s_2}_{v_2}$. Due to Claim 44.1(1) $\langle [s,v], [s'',v''] \rangle \in (\sim_p \cup R_o)^*$ and $\langle [s',v'], [s'',v''] \rangle \in (\sim_p \cup R_o)^*$. By symmetry and transitivity $\langle [s,v], [s',v'] \rangle \in (\sim_p \cup R_o)^*$. □

Proof [*of Claim 44.2*].

$$\begin{aligned}
&[s,v] \in [d_1,\ldots,d_N]^{s_i}_{v_i} \\
&\iff \quad \text{(Def. of } [d_1,\ldots,d_N]^{s_i}_{v_i}\text{)} \\
&\quad s = s_i \ \wedge\ \forall k \in \{1,\ldots,N\}.\ \max_v(C_i^k) = d_k \ \wedge\ \forall x \notin \kappa(s_i).\ v(x) = v_i(x) \\
&\iff \quad \text{(Def. of } \max_v\text{)} \\
&\quad s = s_i \ \wedge \\
&\quad \forall k \in \{1,\ldots,N\}.\ \max\{v(x^i_{(\sum_{l=1}^{k-1} m_l^i)+1}),\ldots,v(x^i_{\sum_{l=1}^{k} m_l^i})\} = d_k \ \wedge \\
&\quad \forall x \notin \kappa(s_i).\ v(x) = v_i(x) \\
&\iff \quad \text{(Def. of } M_i\text{)} \\
&\quad s = s_i \ \wedge \\
&\quad M_i(v(x^i_1),\ldots,v(x^i_{m_1^i}),\ldots,v(x^i_{(\sum_{l=1}^{N-1} m_l^i)+1}),\ldots,v(x^i_{n_i})) = (d_1,\ldots,d_N) \ \wedge \\
&\quad \forall x \notin \kappa(s_i).\ v(x) = v_i(x) \\
&\iff \quad \text{(Def. of } \mathcal{D}^{s_i}_{v_i}\text{)} \\
&\quad \mathcal{D}^{s_i}_{v_i}(v(x^i_1),\ldots,v(x^i_{m_1^i}),\ldots,v(x^i_{(\sum_{l=1}^{N-1} m_l^i)+1}),\ldots,v(x^i_{n_i})) = [s,v] \ \wedge
\end{aligned}$$

$$M_i(v(x_1^i),\ldots,v(x_{m_1^i}^i),\ldots,v(x_{(\sum_{l=1}^{N-1} m_l^i)+1}^i),\ldots,v(x_{n_i}^i)) = (d_1,\ldots,d_N)$$
$$\iff \quad \text{(Def. of } \circ \text{ and } {}^{-1}\text{)}$$
$$[s,v] \in (\mathcal{D}_{v_i}^{s_i} \circ M_i^{-1})(d_1,\ldots,d_N). \qquad \square$$

C. Auxiliary lemma for the proof of $\sim_o \subseteq \sim_c$

Lemma 45. *Let SA be a stochastic automaton and let $PTS_c(SA)$ and $PTS_o(SA)$ be its closed and open behaviour, respectively. The two following statements are equivalent*:

(1) $[s,v] \xrightarrow{a(d)}_c (s',v')$

(2) $[s,v] \xrightarrow{a(d)}_o (s',v')$ *and for all* $d' \in [0,d)$, $b \in \mathcal{A}$, $[s,v] \xrightarrow{b(d')}\!\!\!\!\!\!/\;\;_o$,

where $\rightarrow_c$ *and* $\rightarrow_o$ *are the transition relations of $PTS_c(SA)$ and $PTS_o(SA)$, respectively.*

Proof (*1.* $\Rightarrow$ *2.*)**.** By definition of rules **Closed** and **Open**, it immediately follows that $[s,v] \xrightarrow{a(d)}_c (s',v')$ implies $[s,v] \xrightarrow{a(d)}_o (s',v')$.

Moreover, because of rule **Closed**, $\mathsf{mpr}_d(s,v)$ holds, which implies that

$$\forall d' \in [0,d).\ \neg\exists s \xrightarrow{b,C'} .\ \mathsf{exp}_{d'}(v,C').$$

Therefore, by rule **Open**, for all $d' \in [0,d)$ and for all $b \in \mathcal{A}$, $[s,v] \xrightarrow{b(d')}\!\!\!\!\!\!/\;\;_o$.

(*2.* $\Rightarrow$ *1.*). Assume $[s,v] \xrightarrow{a(d)}_o (s',v')$. By contradiction, suppose $[s,v] \xrightarrow{a(d)}\!\!\!\!\!\!/\;\;_c$. By rule **Open**, $s \xrightarrow{a,C} s'$ and $\mathsf{exp}_d(v,C)$. By **Closed**, it must therefore be the case that $\mathsf{mpr}_d(s,v)$ is not valid. As a consequence,

$$\exists d' \in [0,d), b \in \mathcal{A}, C' \subseteq \mathcal{C}.\ \ s \xrightarrow{b,C'} \text{ and } \mathsf{exp}_{d'}(v,C')$$

By **Open**, this implies that $\exists d' \in [0,d), b \in \mathcal{A}.\ [s,v] \xrightarrow{b(d')}_o$ which contradicts item 2. $\square$

References

[1] R. Alur, C. Courcoubetis, D. Dill, Model-checking for probabilistic real-time systems, in: J. Leach Albert, B. Monien, M. Rodríguez (Eds.), Automata, Languages and Programming (ICALP), Lecture Notes in Computer Science, vol. 510, Springer, 1991, pp. 113–126.

[2] R. Alur, D. Dill, A theory of timed automata, Theor. Comput. Sci. 126 (1994) 183–235.

[3] C. Baier, J.-P. Katoen, H. Hermanns, Approximate symbolic model checking of continuous-time Markov chains, in: J.C.M. Baeten, S. Mauw (Eds.), Concurrency Theory (CONCUR), Lecture Notes in Computer Science, vol. 1664, Springer, 1999, pp. 146–161.

[4] A. Bouajjani, S. Tripakis, S. Yovine, On-the-fly symbolic model-checking for real-time systems, in: 18th IEEE Real-Time Systems Symposium (RTSS), IEEE CS Press, Silver Spring, MD, 1997, pp. 25–34.

[5] M. Bravetti, Specification and Analysis of Stochastic Real-Time Systems, PhD thesis, Università di Bologna, Padova, Venezia, 2002.
[6] M. Bravetti, P.R. D'Argenio, Tutte le algebre insiemi: concepts, discussions and relations of stochastic process algebras with general distributions, in: C. Baier (Ed.), et al., Validation of Stochastic Systems, Lecture Notes in Computer Science, vol. 2925, Springer, 2004, pp. 44–89.
[7] M. Bravetti, R. Gorrieri, The theory of interactive generalized semi-Markov processes, Theor. Comput. Sci. 282 (1) (2002) 5–32.
[8] C.G. Cassandras, Discrete Event Systems. Modeling and Performance Analysis, Aksen Associates, Irwin, 1993.
[9] S. Cattani, R. Segala, M.Z. Kwiatkowska, G. Norman, Stochastic transition systems for continuous state spaces and non-determinism, in: V. Sassone (Ed.), Foundations of Software Science and Computation Structures (FOSSACS), Lecture Notes in Computer Science, 3441, Springer, 2005, pp. 125–139.
[10] E.M. Clarke, O. Grumberg, D. Peled, Model Checking, MIT Press, Cambridge, MA, 1999.
[11] P.R. D'Argenio, Algebras and Automata for Timed and Stochastic Systems, PhD thesis, University of Twente, 1999.
[12] P.R. D'Argenio, A compositional translation of stochastic automata into timed automata, CTIT Technical Report CTIT 00-08, University of Twente, 2000.
[13] P.R. D'Argenio, J.-P. Katoen, A theory of stochastic systems. Part II: Process algebra, Inform. Comput. 203 (2005) 39–74.
[14] P.R. D'Argenio, J.-P. Katoen, E. Brinksma, Specification and analysis of soft real-time systems: quantity and quality, in: 20th IEEE Real-Time Systems Symposium (RTSS), IEEE CS Press, Silver Spring, MD, 1999, pp. 104–114.
[15] J. Desharnais, A. Edalat, P. Panangaden, Bisimulation for labelled Markov processes, Inform. Comput. 179 (2) (2002) 163–193.
[16] A. Giacalone, C.-C. Jou, S.A. Smolka, Algebraic reasoning for probabilistic concurrent systems, in: M. Broy, C.B. Jones (Eds.), IFIP Working Conference on Programming Concepts and Methods (PROCOMET), North-Holland, Amsterdam, 1990, pp. 443–458.
[17] R.J. van Glabbeek, S.A. Smolka, B. Steffen, Reactive, generative, and stratified models of probabilistic processes, Inform. Comput. 121 (1995) 59–80.
[18] R.J. van Glabbeek, F.W. Vaandrager, Petri net models for algebraic theories of concurrency, in: J.W. de Bakker, A. Nijman, P.C. Treleaven (Eds.), Parallel Architectures and Languages Europe (PARLE), Lecture Notes in Computer Science, vol. 259, 1987, pp. 224–243.
[19] P.W. Glynn, A GSMP formalism for discrete event simulation, Proc. IEEE 77 (1) (1989) 14–23.
[20] S. Gnesi, D. Latella, M. Massink, A stochastic extension of a behavioural subset of UML statechart diagrams, in: Symp. on High-Assurance Systems Engineering (HASE), IEEE CS Press, Silver Spring, MD, 2000, pp. 55–64.
[21] H.A. Hansson, Time and Probability in Formal Design of Distributed Systems, PhD thesis, Uppsala University, 1991.
[22] H.A. Hansson, B. Jonsson, A calculus for communicating systems with time and probabilities, in: 11th IEEE Real-Time Systems Symposium (RTSS), IEEE CS Press, Silver Spring, MD, 1990, pp. 278–287.
[23] P.G. Harrison, N.M. Patel, Performance Modelling of Communication Networks and Computer Architectures, Addison-Wesley, Reading, MA, 1992.
[24] P.G. Harrison, B. Strulo, Stochastic process algebra for discrete event simulation, in: F. Bacelli, A. Jean-Marie, I. Mitrani (Eds.), Quantitative Methods in Parallel Systems, Esprit Basic Research Series, Springer, 1995, pp. 18–37.
[25] P.G. Harrison, B. Strulo, SPADES: Stochastic process algebra for discrete event simulation, J. Logic Comput. 10 (1) (2000) 3–42.
[26] T.A. Henzinger, X. Nicollin, J. Sifakis, S. Yovine, Symbolic model checking for real-time systems, Inform. Comput. 111 (1994) 193–244.
[27] H. Hermanns, Interactive Markov Chains and The Quest for Quantified Quality, Lecture Notes in Computer Science, Springer, 2002.
[28] R. Jain, The Art of Computer Systems Performance Analysis, John Wiley, New York, 1991.

[29] D.N. Jansen, H. Hermanns, J.-P. Katoen, A QoS-oriented extension of UML statecharts, in: P. Stevens, J. Whittle (Eds.), Unified Modeling Language (UML), Lecture Notes in Computer Science, vol. 2863, 2003, pp. 76–92.
[30] H.E. Jensen, Model checking probabilistic real-time systems, in: B. Bjerner, M. Larsson, B. Nordström (Eds.), 7th Nordic Workshop on Programming Theory, Report 86, Chalmers University of Technology, 1996, pp. 247–261.
[31] M. Kwiatkowska, G. Norman, R. Segala, J. Sproston, Automatic verification of real-time systems with probability distributions, Theor. Comput. Sci. 282 (1) (2002) 101–150.
[32] M. Kwiatkowska, G. Norman, R. Segala, J. Sproston, Verifying quantitative properties of continuous probabilistic timed automata, in: C. Palamidessi (Ed.), Concurrency Theory, Lecture Notes in Computer Science, vol. 1877, Springer, 2000, pp. 123–137.
[33] S. Lang, Real and Functional Analysis, Graduate Texts in Mathematics, Springer, 1993.
[34] K.G. Larsen, P. Pettersson, W. Yi, Compositional and symbolic model-checking for real-time systems, in: 16th IEEE Real-Time Systems Symposium (RTSS), IEEE CS Press, Silver Spring, MD, 1995, pp. 76–87.
[35] K.G. Larsen, P. Pettersson, W. Yi, UPPAAL in a nutshell, J. Software Tools Technol. Transfer 1 (1/2) (1997) 134–152.
[36] K.G. Larsen, A. Skou, Bisimulation through probabilistic testing, Inform. Comput. 94 (1991) 1–28.
[37] R. Milner, Communication and Concurrency, Prentice-Hall, Englewood Cliffs, NJ, 1989.
[38] M.F. Neuts, Matrix-geometric Solutions in Stochastic Models—An Algorithmic Approach, The Johns Hopkins University Press, Baltimore, MD, 1981.
[39] A. Pnueli, L.D. Zuck, Probabilistic verification, Inform. Comput. 103 (1993) 1–29.
[40] W. Rudin, Real and Complex Analysis, Series in Higher Mathematics, McGraw-Hill, 1974.
[41] R. Schassberger, Insensitivity of steady-state distributions of GSMPs, Ann. Prob. 5 (1977) 87–89.
[42] R. Segala, Modeling and Verification of Randomized Distributed Real-Time Systems, PhD thesis, MIT, 1995.
[43] R. Segala, N. Lynch, Probabilistic simulations for probabilistic processes, Nordic J. Comput. 2 (2) (1995) 250–273.
[44] G.S. Shedler, Regenerative Stochastic Simulation, Academic Press, 1993.
[45] A.N. Shiryaev, Probability, Graduate Texts in Mathematics, Springer, 1996.
[46] B. Strulo, Process Algebra for Discrete Event Simulation, PhD thesis, Imperial College, 1993.
[47] M.Y. Vardi, Automatic verification of probabilistic concurrent finite state programs, in: 26th Symp. on Foundations of Computer Science (FOCS), IEEE CS Press, Silver Spring, MD, 1985, pp. 327–338.
[48] E.P. de Vink, J.J.M.M. Rutten, Bisimulation for probabilistic transition systems: a coalgebraic approach, Theor. Comput. Sci. 221 (1–2) (1999) 271–293.
[49] W. Whitt, Continuity of generalized semi-Markov processes, Math. Oper. Res. 5 (1980) 494–501.
[50] H.L.S. Younes, R.G. Simmons, Probability verification of discrete event systems using acceptance sampling, in: E. Brinksma, K.G. Larsen (Eds.), Computer Aided Verification (CAV), Lecture Notes in Computer Science, vol. 2404, Springer, 2002, pp. 223–236.
[51] S. Yovine, KRONOS: a verification tool for real-time systems, J. Software Tools Technol. Transfer 1 (1/2) (1997) 123–133.