Electronic Communications of the EASST
Volume 53 (2012)

Proceedings of the
12th International Workshop on
Automated Verification of Critical Systems
(AVoCS 2012)

Model Checking Algorithms for Markov Automata

Hassan Hatefi and Holger Hermanns

15 pages

Guest Editors: Gerald Lüttgen, Stephan Merz
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Model Checking Algorithms for Markov Automata *

Hassan Hatefi [1] and Holger Hermanns [2]

[1] hhatefi@depend.cs.uni-saarland.de
Department of Computer Science, Saarland University, Saarbrücken, Germany
Max-Planck-Institut für Informatik, Saarbrücken, Germany
[2] hermanns@cs.uni-saarland.de
Department of Computer Science, Saarland University, Saarbrücken, Germany

Abstract: Markov automata constitute a compositional modeling formalism spanning as special cases the models of discrete and continuous time Markov chains, as well as interactive Markov chains and probabilistic automata. This paper discusses the core algorithmic ingredients of a numerical model checking procedure for Markov automata with respect to a PCTL or CSL like temporal logic. The main challenge lies in the computation of time-bounded reachability probabilities, for which we provide a stable approximation scheme.

Keywords: Markov Automata, Compositionality, Model Checking, Reachability Probability, Markov Decision Process

1 Introduction

Markov automata (*MA*) constitute a compositional behavioural model for continuous time stochastic and nondeterministic systems [EHZ10a, EHZ10b, DH11]. Markov automata are – on the one hand – rooted in interactive Markov chains (*IMC*s) [Her02]. *IMC*s have seen applications in diverse areas [BCS10, HKR+10, CHL+09, EKN+12] where memoryless continuous delays prevail. These practical cases have, however, revealed the need to pair the expressiveness of *IMC* with instantaneous random switching. This observation has led to the conception of Markov automata, which are – on the other hand – based on probabilistic automata (*PA*) [Seg95]. *MA* are expressive enough to capture the complete semantics [EHZ10a] of generalised stochastic Petri nets (GSPNs) [MCB84] and of stochastic activity networks (SANs) [MMS85]. Due to these attractive semantic and compositionality features, there is a growing interest in software tool support for modelling and analysis with *MA*. SCOOP [TKP+12], supports fully compositional construction of *MA* models, and IMCA [Guc12] supports the evaluation of long-run reward properties for *MA*.

In this paper, we discuss temporal logics and model checking algorithms for Markov automata. We focus on branching time logics in the style of CTL. The model checking approach we follow harvests advances in the area of PCTL model checking of probabilistic automata [BA95], and of CSL model checking of interactive Markov chains [Neu10]. The approach however needs

* This work has been supported by the DFG as part of SFB/TR 14 AVACS, by the DFG/NWO Bilateral Research Program ROCKS, and by the European Union FP7-ICT projects MEALS, grant agreement no. 295261, and SENSATION, grant agreement no. 318490.

to intertwine the respective steps in a nontrivial manner. This is rooted in the fact that Markov automata evolve in super dense time [LZ05, MP93]: the time domain is $\mathbb{R}_{\geq 0} \times \mathbb{N}_0$.

We report on empirical results for a queueing system case study, obtained with a prototypical implementation of our technique. Albeit being a small example, that case requires the full expressiveness of Markov automata, and hence could neither be faithfully modelled, nor model checked prior to our work. The experiments demonstrate principal feasibility, and indicate room for further improvement.

Organisation of the paper. After the preliminaries in Section 2, we introduce syntax and semantics of the logic CSL in Section 3. Section 4 is devoted to the core model checking algorithm for CSL. The key step, namely the computation of time interval bounded reachability, is treated in Section 5, where an approximation scheme is developed using a digitisation technique. Algorithmic aspects of this technique are discussed in Section 6. Experimental results are reported in Section 7. Finally, Section 8 concludes the paper.

2 Preliminaries

This section provides the necessary notational background. A distribution μ over a discrete set S is a function $\mu : S \rightarrowtail [0,1]$ such that $\sum_{s \in S} \mu(s) = 1$. We may also use the set representation of μ given by $\{(s,\mu(s)) | s \in S, \mu(s) > 0\}$. If $\mu(s) = 1$ for some $s \in S$ (i.e. $\mu = \{(s,1)\}$), μ is a *Dirac* distribution, and is denoted μ_s. Let $Dist(S)$ be the set of all distributions over set S.

2.1 Markov Automata

Definition 1 A Markov Automaton (*MA*) [EHZ10a] is a tuple $\mathscr{M} = (S, Act, \longrightarrow, \dashrightarrow, \nu)$, where S is a finite set of states, Act is a set of actions, including τ, representing an internal invisible action. Furthermore, $\longrightarrow \subset S \times Act \times Dist(S)$ is a set of interactive transitions, $\dashrightarrow \subset S \times \mathbb{R}_{\geq 0} \times S$ is a set of Markov transitions, and ν is the initial distribution. [1]

A probabilistic automaton is an *MA* where $\dashrightarrow$ is empty. If in addition for each s, $|\{\mu \mid (s,a,\mu) \in \longrightarrow\}| \in \{0,1\}$, we are facing a discrete time Markov chain. An interactive Markov chain (*IMC*) is an *MA* where all distributions occurring in $\longrightarrow$ are Dirac. A continuous time Markov chain is an *MA* where $\longrightarrow$ is empty.

We partition states of *MA* into *interactive*, *Markov* and *hybrid* states, similar to [Her02]. Interactive (Markov) states have only interactive (Markov) outgoing transitions, while hybrid states have transitions of both types. We use S_I, S_M and S_H to denote the set of interactive, Markov and hybrid states respectively. A state without any outgoing transition, is called a *deadlock state*. We turn any such deadlock state into a Markov state by adding a self loop to it with an arbitrary rate. We further assume that *maximum progress* [Her02] is imposed. This means that τ-transitions take precedence over Markov transitions. As for *IMC*, we distinguish between closed and open *MA*. An open *MA* can interact with the environment and in particular, can be composed with other *MA*, e.g. via parallel composition. In contrast, a closed *MA* is not subject to any fur-

[1] We assume that $(s,\alpha,\mu_1),(s,\alpha,\mu_2) \in \longrightarrow$ only if $\mu_1 = \mu_2$, where $s \in S, \alpha \in Act$ and $\mu_1, \mu_2 \in Dist(S)$. This restriction can be relinquished at the price of more involved notations.

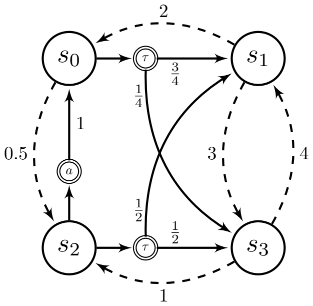

Figure 1: An exemplary Markov automaton

ther communication and composition. In this paper, we assume that the models we are going to analyse are closed, and impose the stronger *urgency assumption* [Her02] which means that any action has precedence over Markov transitions, i.e. interactive transitions are assumed to be taken immediately with probability 1. Consequently, in a closed *MA*, hybrid states can be considered as interactive states.

For uniformity of notation, we use a distinguished action $\perp \notin Act$ to indicate Markov transitions and extend the set of actions to $Act_\perp = Act \,\dot{\cup}\, \{\perp\}$. For $s \in S$, we define $Act_\perp(s)$ as the set of *enabled actions* in state s. If s is a Markov state, $Act_\perp(s) = \{\perp\}$, otherwise $Act_\perp(s) = \{\alpha \mid (s,\alpha,\mu) \in \longrightarrow\}$. The rate between state s and s' is defined by $\mathrm{rate}(s,s') = \sum_{(s,\lambda,s') \in \dashrightarrow} \lambda$. Then $E(s) = \sum_{s' \in S} \mathrm{rate}(s,s')$ denotes the sum of rates going out of state s. Using these concepts, we define the *branching probability* matrix for both interactive and Markov states by

$$\mathbf{P}(s,\alpha,s') = \begin{cases} \mu(s') & s \in S_I \wedge (s,\alpha,\mu) \in \longrightarrow \\ \frac{\mathrm{rate}(s,s')}{E(s)} & s \in S_M \wedge \alpha = \perp \\ 0 & \text{otherwise} \end{cases}$$

Example 1 *Let $\mathcal{M}$ be the MA in Figure 1. s_1 and s_3 are Markov states, while s_2 is an interactive state. State s_0 is a hybrid state, since it has both interactive and Markov transitions emanating. The urgency assumption implies that we can ignore $(s_0, 0.5, s_2) \in \dashrightarrow$ and consider s_0 as an interactive state. Recall that under this assumption, interactive transitions are instantaneously fired after zero time delay. Conversely, the sojourn time in a Markov state s is exponentially distributed with rate $E(s)$. For example, the probability to leave s_1 within δ time unit is $1 - e^{-5\delta}$ $(E(s_1) = 2 + 3 = 5)$. Branching probabilities determine the likelihood of entering particular successor states. For s_1, $\mathbf{P}(s_1,\perp,s_0) = \frac{2}{5}$ and $\mathbf{P}(s_1,\perp,s_3) = \frac{3}{5}$. Therefore, the probabilities to go to s_0 and s_3 after spending at most δ time units in s_1 are $\frac{2}{5}(1-e^{-5\delta})$ and $\frac{3}{5}(1-e^{-5\delta})$ respectively.*

2.2 Behavioural and Measurability Concepts

The behaviour of a model $\mathcal{M}$ when entering a state $s \in S$ can be described by a *time-stamped step*. It represents how long the model stays in state s prior to taking which action and thereby entering which successor state. A time-stamped step m is thus a triple of a sojourn time, an extended action and a state, i.e. $m \in M = \mathbb{R}_{\geq 0} \times Act_\perp \times S$. We often denote a time-stamped step $(t,\alpha,s) \in M$ by $\xmapsto{t,\alpha} s$. For example, when the model is in an interactive state $s \in S_I$, it must immediately (in zero time) choose action $\alpha \in Act_{\perp}(s)$ and go to state s' according to the distribution $\mathbf{P}(s,\alpha,.)$. This boils down to taking the time-stamped step $(0,\alpha,s')$ from s, or $s \xmapsto{0,\alpha} s'$. On the other hand, if $s \in S_M$, the model can stay for $t > 0$ time units and then choose the next state s' based on the distribution $\mathbf{P}(s,\perp,.)$ (i.e. $(s,\lambda,s') \in \dashrightarrow$ for some $\lambda > 0$). Then it takes time-stamped step $(t,\perp,s')$ from s, or $s \xmapsto{t,\perp} s'$. By dropping the sojourn time from time-stamped steps, we obtain time-abstract steps, defined as pairs of extended actions and states ($m \in M_{ta} = Act_{\perp} \times S$).

A finite (possibly time-abstract) path $\pi = s_0 \circ m_0 \circ \cdots \circ m_{n-1}$ is a concatenation of a state and a finite sequence of (time-abstract or) time-stamped steps where $s \in S$ and ($m_i \in M_{ta}$ or) $m_i \in M, i = 0 \cdots n-1$. A (possibly time-abstract) infinite path is an initial state followed by an infinite sequence of (time-abstract or) time-stamped steps. Given Markov automata $\mathscr{M}$, the set of finite and infinite paths in $\mathscr{M}$ are denoted by $Paths^*$ ($\mathscr{M}$) and $Paths^{\omega}$ ($\mathscr{M}$) respectively. We use subscript *ta* to refer to the set of time-abstract finite and infinite paths (i.e. $Paths^*_{ta}$ ($\mathscr{M}$) and $Paths^{\omega}_{ta}$ ($\mathscr{M}$)). When it is clear from the context, $\mathscr{M}$ might be omitted from the notations.

Some further notations for paths are needed. Let $\pi = s_0 \xmapsto{t_0,\alpha_0} s_1 \cdots s_{n-1} \xmapsto{t_{n-1},\alpha_{n-1}} s_n$ be a finite path. We use $|\pi| = n$ as the length of π and $last(\pi) = s_n$ as the last state of π. Assume $k \leq n$ is an index, then $\pi[k] = s_k$ is the $k+1$-th state of π. The time spent on π up to state $\pi[k]$ is calculated by $\Delta(\pi,k)$ which is zero if $k = 0$, and otherwise $\sum_{i=0}^{k-1} t_i$. We use $\Delta(\pi)$ as an abbreviation for $\Delta(\pi,|\pi|)$. For time-abstract paths, Δ_{ta} is similarly defined, however it only counts the number of time-abstract steps with action $\perp$. The prefix of length k is extracted from π by $Pref(\pi,k) = s_0 \xmapsto{t_0,\alpha_0} s_1 \cdots s_{k-1} \xmapsto{t_{k-1},\alpha_{k-1}} s_k$.

Paths in *MA* are constructed by sequences of time-stamped steps originating from an initial state. To define the σ-algebra over a finite paths, first we define it over time-stamped steps. Let $\mathfrak{F}_S = 2^S$ be σ-algebra over S and $\mathfrak{F}_{Act_{\perp}} = 2^{Act_{\perp}}$ be σ-algebra over $Act_{\perp}$. We define the σ-algebra over time-stamped steps using the concept of Cartesian product of σ-algebra [AD00], as $\mathfrak{F}_M = \sigma(\mathfrak{B}(\mathbb{R}_{\geq 0}) \times \mathfrak{F}_{Act_{\perp}} \times \mathfrak{F}_S)$. It can be extended to the σ-algebra over finite paths using the same technique. Let $\mathfrak{F}_{Paths^n} = \sigma(\mathfrak{F}_S \times \overbrace{\mathfrak{F}_M \times \cdots \times \mathfrak{F}_M}^{n \text{ times}})$ be the σ-algebra over finite paths of length n, then the σ-algebra over finite paths is defined as $\mathfrak{F}_{Paths^*} = \cup_{i=0}^{\infty} \mathfrak{F}_{Paths^n}$. The σ-algebra over infinite paths is defined using a standard cylinder set construction. We define the cylinder set of a given base B^n as $Cyl(B^n) = \{\pi \in Paths^{\omega} : Pref(\pi,n) = B^n\}$. $Cyl(B^n)$ is measurable if its base B^n is measurable, i.e. $B^n \in \mathfrak{F}_{Paths^n}$. We further define the σ-algebra over infinite paths, $\mathfrak{F}_{Paths^{\omega}}$, as the smallest σ-algebra over measurable cylinders. And finally the σ-algebra over the set of paths is the disjoint union of the σ-algebras over the finite paths and the infinite paths.

2.3 Resolving Nondeterminism

Nondeterminism in *MA* is resolved by schedulers. The most general scheduler class maps a finite path to a distribution over the set of actions enabled in the last state of the path:

Definition 2 (Generic Scheduler) A generic scheduler over *MA* $\mathscr{M} = (S, Act, \longrightarrow, \dashrightarrow, \nu)$ is a function, $A : Paths^* \times Act_{\perp} \rightarrowtail [0,1]$, where $A(\pi,.) \in Dist(Act_{\perp}(last(\pi))), \pi \in Paths^*$. Scheduler A is measurable if $\forall \alpha \in Act_{\perp}, A(.,\alpha) : Paths^* \rightarrowtail [0,1]$ is measurable.

For a finite path π, a scheduler specifies how to resolve nondeterminism in the last state of π. If $last(\pi)$ is a Markov state, there will be only one enabled action, namely $\perp$, which will be chosen by the scheduler with probability 1. Otherwise, it gives a distribution over the set of enabled actions of $last(\pi)$. Measurability of scheduler A means the preimage of each Borel measuable set in A is a measuable set of paths, i.e. $\forall\alpha \in Act_{\perp}, \forall B \in \mathfrak{B}([0,1])$, $\{\pi | A(\pi,\alpha) \in B\} \in \mathfrak{F}_{Paths^*}$, where $\mathfrak{B}([0,1])$ is Borel σ-algebra over interval $[0,1]$. We use term *GM* to refer to the set of all generic measurable schedulers.

By substituting $Paths^*$ with $Paths^*_{ta}$ in Definition 2, we obtain the less general class of time-abstract schedulers. Measurability issues do not arise for schedulers of this class [ZN10]. The set of all time-abstract schedulers is denoted by *TA*. Time-abstract schedulers can be used to evaluate properties that deal with the order of events without being explicit about their timing. We will use them later for digitised models of *MA*.

2.4 Probability Measure

We now construct a probability measure over the set of finite and infinite paths by extending the probability measure defined on *IMC* [Joh08]. First, we define a probability measure over a measurable set of time-stamped steps, i.e. $TSS \in \mathfrak{F}_M$. Then we extend it to a probability measure on the probability space $(Paths^n, \mathfrak{F}_{Paths^n})$. Let A be a generic measurable scheduler on $\mathscr{M} = (S, Act, \longrightarrow, \dashrightarrow, s_0)$, $\mu_A(\pi,.) : \mathfrak{F}_M \rightarrowtail [0,1], \pi \in Paths^*$ is defined as follows:

$$\mu_A(\pi, TSS) = \begin{cases} \sum\limits_{(0,\alpha,s)\in TSS} \mathbf{P}(last(\pi),\alpha,s)A(\pi,\alpha) & last(\pi) \in S_I \\ \int\limits_{\Lambda} E(last(\pi))e^{-E(last(\pi))t} \sum\limits_{(t,\perp,s)\in TSS} \mathbf{P}(last(\pi),\perp,s)dt & last(\pi) \in S_M \end{cases}$$

where $\Lambda = \{t | (t,\perp,s) \in TSS, s \in S\}$ is the set of all time points defined in *TSS*. Relative to the same definition for *IMC*s [Joh08], the only difference is that for interactive states in the above, $\mathbf{P}(s,\alpha,.)$ can be an arbitrary distribution, whereas in *IMC*, it is a Dirac distribution. We now define the probability measure recursively on the probability space $(Paths^n, \mathfrak{F}_{Paths^n})$ given $A \in GM$ in $\mathscr{M}$ as:

$$Pr^0_{A,\nu}(S_0) = \sum_{s\in S_0} \nu(s), \quad S_0 \in \mathfrak{F}_S$$

$$Pr^n_{A,\nu}(\Pi \circ TSS) = \int\limits_{\pi\in\Pi} \mu_A(\pi, TSS) dPr^{n-1}_{A,\nu}(\{\pi\}), \quad \Pi \in \mathfrak{F}_{Paths^{n-1}}, TSS \in \mathfrak{F}_M$$

The measure $Pr^n_{A,\nu}$, corresponding to the probability of a set of paths with length n, is thus recursively obtained by breaking them into prefixes of length $n-1$ and the n-th time-stamped step. This is included in the above integral by $dPr^{n-1}_{A,\nu}(\{\pi\})$ and $\mu_A(\pi, TSS)$, respectively. It can be shown that this probability measure is unique and can be extended to the probability measure $Pr^{\omega}_{A,\nu}$ on $(Paths^{\omega}, \mathfrak{F}_{Paths^{\omega}})$ [AD00].

Owing to the presence of immediate state changes, an *MA* might exhibit *Zeno behaviour*, where infinitely many interactive transitions are taken in finite time. This is an unrealistic phenomenon, characterised by paths π, where the time spent on π up to state $\pi[n]$, $\Delta(\pi,n)$, does not diverge for diverging n. In other words, the time spent in the system may stop increasing, if the system follows path π. We intend to exclude such unrealistic phenomena. This could safely be achieved by disallowing cyclic structures in $\longrightarrow$. We however take a more relaxed approach and restrict our attention to models where the probability of Zeno behaviour is zero. We require that $\forall A \in GM,\ \forall s \in S.\ Pr^{\omega}_{A,\nu_s}(\{\pi \in Paths^{\omega} | \lim_{n\to\infty} \Delta(\pi,n) < \infty\}) = 0$. This condition implies that starting from any interactive state, we must reach the set of Markov states with probability one. In the remainder of this paper, we therefore restrict to such models.

3 Continuous Stochastic Logic for Markov Automata

This section introduces a logic suitable to express a broad range of performance and dependability measures on Markov automata. Our basis are CTL (Computation Tree Logic [CES86]), its extensions to discrete time Markov models, PCTL [HJ94, BA95], and to continuous time Markov models, CSL [BHH+03, ZN10]. These logics work on *state labelled* Markov models. Given $MA\ \mathscr{M}$ and a finite set of *atomic propositions* AP, a *state labellings function* $L : S \rightarrowtail 2^{AP}$ decorates each state with a set of atomic propositions which do hold in that state. Let $\mathscr{I}$ and $\mathscr{Q}$ be the set of all nonempty nonnegative real intervals with real and rational bounds respectively. Continuous Stochastic Logic (CSL) for Markov automata is defined as follows.

Definition 3 (CSL Syntax) Let $a \in AP$, $p \in [0,1]$, $I \in \mathscr{I}$ an interval and $\trianglelefteq \in \{<,\leq,\geq,>\}$, CSL state and path formula are described by

$$\Phi ::= a \ \mid\ \neg\Phi \ \mid\ \Phi\wedge\Phi \ \mid\ \mathscr{P}_{\trianglelefteq p}(\phi) \quad \text{and} \quad \phi ::= \mathscr{X}^{I}\Phi \ \mid\ \Phi\,\mathscr{U}\,\Phi \ \mid\ \Phi\,\mathscr{U}^{I}\,\Phi$$

Given an infinite path $\pi \in Paths^{\omega}$, π satisfies $\mathscr{X}^{I}\Phi$ if the first transition of π occurs within time interval I and leads to a state that satisfies Φ. Likewise, bounded until formula $\Phi\,\mathscr{U}^{I}\Psi$ is satisfied by π, if π visits states that satisfy formula Φ until at some time point within interval I, it reaches a state that satisfies formula Ψ. In contrast to bounded until, an unbounded until formula does not constrain the time at which π may visits the state satisfying Ψ. This corresponds to the time interval of $[0,\infty)$. The formal semantics of CSL formulae is defined as follows.

Definition 4 (CSL Semantics) Let $\mathscr{M} = (S, Act, \rightarrow, \dashrightarrow, AP, L, \nu)$ be a state labelled MA, $s \in S$, $a \in AP$, $p \in [0,1]$, $I \in \mathscr{I}$, $\trianglelefteq \in \{<,\leq,\geq,>\}$, and $\pi \in Paths^{\omega}$. We define the satisfaction relation $\models$ for state formulae: $s \models a$ iff $a \in L(s)$, $s \models \neg\Phi$ iff $s \not\models \Phi$, $s \models \Phi\wedge\Psi$ iff $s \models \Phi \wedge s \models \Psi$, and $s \models \mathscr{P}_{\trianglelefteq p}(\phi)$ iff $\forall A \in GM.\ Pr^{\omega}_{A,\nu_s}(\{\pi \in Paths^{\omega} \mid \pi \models \phi\}) \trianglelefteq p$. For path formulae:

$$\begin{array}{lll}
\pi \models \mathscr{X}^{I}\Phi & \text{iff} & \pi[1] \models \Phi \wedge \Delta(\pi,1) \in I \\
\pi \models \Phi\,\mathscr{U}^{I}\,\Psi & \text{iff} & \exists n \in \mathbb{N}_0.[\Delta(\pi,n),\Delta(\pi,n+1)] \cap I \neq \emptyset \wedge \pi[n] \models \Psi \wedge \forall k = 0 \cdots n-1.\pi[k] \models \Phi \\
\pi \models \Phi\,\mathscr{U}\,\Psi & \text{iff} & \exists n \in \mathbb{N}_0.\pi[n] \models \Psi \wedge \forall k = 0 \cdots n-1.\pi[k] \models \Phi
\end{array}$$

4 Algorithms for Model Checking

The core model checking approach is as usual. To decide which states of an MA satisfy CSL formula Φ, all sub-formulae Ψ of Φ are recursively processed and for each of them the set $Sat(\Psi) = \{s \in S | s \vDash \Psi\}$ is determined. We have $Sat(a) = \{s \in S | a \in L(s)\}$ for an atomic proposition $a \in AP$, $Sat(\neg\Psi) = S \setminus Sat(\Psi)$ for negation formulae, and $Sat(\Psi_1 \wedge \Psi_2) = Sat(\Psi_1) \cap Sat(\Psi_2)$ for the conjunction of formulae. The proper calculation of $Sat(\mathscr{P}_{\unlhd p}(\phi))$ however needs deeper considerations. $Sat(\mathscr{P}_{\unlhd p}(\phi))$ is naturally defined as $\{s \in S \mid \forall A \in GM.\ Pr^{\omega}_{A,\nu_s}(\{\pi \in Paths^{\omega} \mid \pi \vDash \phi\}) \unlhd p\}$. In a nutshell, determining this set requires the calculation of the maximum or minimum (depending on $\unlhd$) probability measures induced by all ϕ-satisfying paths starting from state s, where the maximum or minimum are to be taken over all measurable schedulers. Let $p^{\mathscr{M}}_{\max}(s,\phi)$ and $p^{\mathscr{M}}_{\min}(s,\phi)$ be those values respectively. We deal with their calculation for the different types of path formulae ϕ separately. In the following, we only consider the maximum cases. The minimum cases can be handled similarly.

4.1 Next Formula

The computation of $p^{\mathscr{M}}_{\max}(s, \mathscr{X}^{I}\Phi)$ is straightforward. First, the set $Sat(\Phi)$ is computed by processing the structure of the formula. Then, if $s \in S_M$, no nondeterminism exists and the computation is done as the next formula in CTMC [BHH+03]. Otherwise, for $s \in S_I$ the computation proceeds as in *PA* [BA95].

4.2 Unbounded Until Formula

The evaluation of a given unbounded until formula in an *MA* can be reduced to the computation of unbounded reachability in a smaller *MA*, which in turn can be reduced to reachability in an untimed model. Assume *MA* $\mathscr{M}$ and a set of goal states G are given, we propose a way to compute maximum and minimum probability of eventually reaching the states in G ($p^{\mathscr{M}}_{\max}(.,\Diamond G)$ and $p^{\mathscr{M}}_{min}(.,\Diamond G)$). We utilise the same technique which is used for computing unbounded reachability in CTMC [BHH+03]. This is because for unbounded reachability, the sojourn time in Markov states is not important. In other words, it does not matter at which time point a transition from a Markov state s to another state s' occurs. It is enough to know what is the probability of eventually reaching s' from s which is $\int_0^{\infty} E(s)e^{-E(s)t}\mathbf{P}(s,\perp,s')dt = \mathbf{P}(s,\perp,s')$. Therefore it suffices to consider only time-abstract paths ($Paths^{\omega}_{ta}$) of a model abstracted from $\mathscr{M}$ by only looking into discrete branching probabilities. The abstract model is an embedded Markov Decision Process (*eMDP*) which is defined as follows.

Definition 5 (Embedded *MDP*) Let $\mathscr{M} = (S, Act, \longrightarrow, \dashrightarrow, \nu)$ be an *MA* as before, $emb(\mathscr{M})$ is an embedded *MDP* induced by $\mathscr{M}$ and defined by $emb(\mathscr{M}) = (S, Act_{\perp}, T, \nu)$, where $T \subset S \times Act_{\perp} \times Dist(S)$ is the set of transition relations defined by $T = \{(s,\perp,\mathbf{P}(s,\perp,.)) \mid s \in S_M\} \cup \longrightarrow$.

The embedded *MDP* mimics all interactive transitions of its original *MA*. However each Markov transition of the original *MA* is replaced with a transition which takes action $\perp$ and enters successor states according to the branching probability matrix. We reduce unbounded reachability in *MA* $\mathscr{M}$ to unbounded reachability in its embedded *MDP* with respect to the set of goal states G by $p^{\mathscr{M}}_{\max}(s,\Diamond G) = p^{emb(\mathscr{M})}_{\max}(s,\Diamond G)$, $\forall s \in S$.

Now we consider $\phi = \Phi\ \mathscr{U}\ \Psi$ is given and the set $Sat(\Phi)$ and $Sat(\Psi)$ are computed. To determine $p^{\mathscr{M}}_{\max}(s, \Phi\ \mathscr{U}\ \Psi)$ we reduce it to unbounded reachability in its embedded *MDP* by the following theorem.

Theorem 1 (Unbounded Until) *Let $\mathscr{M} = (S, Act, \longrightarrow, \dashrightarrow, \nu)$ be an MA as before, and $\phi = \Phi\,\mathscr{U}\,\Psi$ be a CSL formula $G = Sat(\Psi)$ and $emb(\mathscr{M})$ as in Definition 5. Assume we construct $emb(\mathscr{M})_{\neg\Phi}$ from $emb(\mathscr{M})$ by making all states in $Sat(\neg\Phi)$ absorbing. Then $\forall s \in S.\ p^{\mathscr{M}}_{\max}(s, \Phi\,\mathscr{U}\,\Psi) = p^{emb(\mathscr{M})_{\neg\Phi}}_{\max}(s, \Diamond G)$.*

A state is turned into absorbing by replacing all its emanating transition by a single self loop taking action $\perp$. Theorem 1 holds for minimum probabilities as well. The idea of absorbing states has first been used for CTMCs [BHH+03] in the same manner: A path that has entered an absorbing state cannot leave the state anymore. When path π reaches state $s \in Sat(\neg\Phi)$, $\pi \vDash \Phi\mathscr{U}\Psi$ will hold if $s \vDash \Psi$, otherwise it will not hold. In other words, as soon as a path reaches a state in $Sat(\neg\Phi)$, regardless of which states will be visited in future, it can be decided whether the path holds the property or not . As a result, all states in $Sat(\neg\Phi)$ can be made absorbing without altering the truth value of the formulae in the question. Finally, for computation of unbounded reachability in *MDP*, one can use the method discussed in [BK08, Chapter 10].

4.3 Time Bounded Until Formula

The computation of time bounded until is more complicated and requires some innovation. We first reduce it to the computation of time interval bounded reachability with respect to a set of goal states. Let $\mathscr{M}$ be an *MA* and $\phi = \Phi\mathscr{U}^I\Psi$ with $I \in \mathscr{I}$ be a CSL formula. We assume that $Sat(\Phi)$ and $Sat(\Psi)$ have already been computed. Similar to unbounded until, all states in $Sat(\Psi)$ are considered as goal states and also all states in $Sat(\neg\Phi)$ are made absorbing. In this model, time interval bounded until analysis reduces to timed reachability analysis by the following theorem.

Theorem 2 (Bounded Until) *Let $\mathscr{M} = (S, Act, \longrightarrow, \dashrightarrow, \nu)$ be an MA as before, and $\phi = \Phi\,\mathscr{U}^I\,\Psi$ with $I \in \mathscr{I}$ be a CSL formula and $G = Sat(\Psi)$. We construct $\mathscr{M}_{\neg\Phi}$ from $\mathscr{M}$ by making all states in $Sat(\neg\Phi)$ absorbing. Then $\forall s \in S.\ p^{\mathscr{M}}_{\max}(s, \Phi\,\mathscr{U}^I\,\Psi) = p^{\mathscr{M}_{\neg\Phi}}_{\max}(s, \Diamond^I G)$.*

The proof is a direct adaptation of the one in [Neu10, Theorem 6.7]. Theorem 2 says one can compute bounded until using time interval bounded reachability analysis. So we are left with the question how the latter can be computed. The following section discusses this issue.

5 Time Bounded Reachability in *MA*

In this section, we introduce an algorithm that computes an approximation of the time bounded reachability probability in *MA* using a digitisation technique. We start with a fixed point characterisation, and then we explain how that can be approximately computed using a digitisation technique.

5.1 Fixed Point Characterisation

Our goal is to come up with a fixed point characterisation for the maximum (minimum) probability to reach a set of goal states in an interval of time. For $I \in \mathscr{I}$ and $t \in \mathbb{R}_{\geq 0}$, we define $I \ominus t = \{x - t \mid x \in I \wedge x \geq t\}$. If $I \in \mathscr{Q}$ and $t \in \mathbb{Q}_{\geq 0}$, then $I \ominus t \in \mathscr{Q}$. Given *MA* $\mathscr{M}$, a time interval $I \in \mathscr{I}$ and a set of goal states $G \subseteq S$, the set of all paths that reach the goal states within interval I is denoted by $\Diamond^I G$. Let $p^{\mathscr{M}}_{\max}(s, \Diamond^I G)$ be the maximum probability of reaching the goal states within interval I if starting in state s at time 0. The latter indeed is the supremum of all probability measures of the induced paths over all possible *GM* schedulers: $p^{\mathscr{M}}_{\max}(s, \Diamond^I G) = \sup_{A \in GM} Pr^{\omega}_{A,\nu_s}(\Diamond^I G)$. The next definition provides a characterisation of $p^{\mathscr{M}}_{\max}(s, \Diamond^I G)$ as a fixed point.

Definition 6 (Fixed Point Characterisation of *MA*) Let $\mathscr{M}$ be an *MA*, $G \subseteq S$ be a set of goal states and $I \in \mathscr{I}$ with $\inf I = a$ and $\sup I = b$. $p^{\mathscr{M}}_{\max} : S \times \mathscr{I} \rightarrowtail [0,1]$ is the least fixed point of the higher-order operator $\Omega : (S \times \mathscr{I} \rightarrowtail [0,1]) \rightarrowtail (S \times \mathscr{I} \rightarrowtail [0,1])$, which for $s \in S_M$ is defined as:

$$\Omega(F)(s,I) = \begin{cases} \int_0^b E(s) e^{-E(s)t} \sum_{s' \in S} \mathbf{P}(s, \bot, s') F(s', I \ominus t) dt & s \notin G \\ e^{-E(s)a} + \int_0^a E(s) e^{-E(s)t} \sum_{s' \in S} \mathbf{P}(s, \bot, s') F(s', I \ominus t) dt & s \in G \end{cases}$$

For $s \in S_I$, $\Omega(F)(s,I) = 1$ if $s \in G \wedge \inf I = 0$, otherwise it is $\max_{\alpha \in Act_{\bot}(s)} \sum_{s' \in S} \mathbf{P}(s, \alpha, s') F(s', I)$.

For Markov states, the definition of operator Ω in *MA* is the same as for *IMC* [Neu10]. However it is different for interactive states, because a transition from an interactive state in *MA* leads to a distribution over the successor states, which in contrast to *IMC* is not necessarily Dirac.

The above characterisation provides an integral equation system which is in general not directly tractable [BHH+03]. To tackle the problem, we approximate this characterisation using a time digitisation idea, originally developed for *IMC* [ZN10] to *MA*. Intuitively, we divide the time interval into equally sized pieces, which we call digitisation steps. Provided the digitisation steps are small enough, we can derive that with high probability in any digitisation step at most one Markov transition firing happens. Assuming this turns the *MA* into a digitised model which we call digitised Markov Automata.

5.2 Digitised Markov Automata

A digitised Markov automaton (*dMA*) is a specific variation of a semi-*MDP* [Put94], obtained by summarising the behaviour of the Markov Automaton at equidistant time points.

Definition 7 A *dMA* is a tuple $\mathscr{D} = (S, Act, \longrightarrow, \dashrightarrow_d, \nu)$, where S, Act, $\longrightarrow$ are as Definition 1 and $\dashrightarrow_d \subset S \times Dist(S)$ is the set of digitised Markov transitions.

The concepts of closed and open models carry over to *dMA*. As we do not have the notion of continuous time, paths in *dMA* can be seen as time-abstract paths in *MA*, implicitly still counting digitisation steps, and thus discrete time. Also the most general scheduler classes for *dMA* are time-abstract schedulers. The probability measure ($Pr^{\omega}_{A,\nu}$, $A \in TA$) can be derived directly from the one for Interactive Probabilistic Chain (*IPC*) [ZN10].

5.3 A Reduction for Reachability Analysis using Digitisation

We now discuss a digitisation that turns an *MA* into a *dMA*. Afterwards, we show that reachability analysis in an *MA* can be approximated by reachability analysis in its *dMA*, for which we provide an error bound.

Definition 8 (Digitisation) Given *MA* $\mathcal{M} = (S, Act, \longrightarrow, \dashrightarrow, \nu)$ and a digitisation constant $\delta > 0$, $\mathcal{M}_\delta = (S, Act, \longrightarrow, \dashrightarrow_\delta, \nu)$ is a *dMA* constructed from digitisation of $\mathcal{M}$ with respect to digitisation constant δ and $\dashrightarrow_\delta = \{(s, \mu^s) | s \in S_M\}$, where

$$\mu^s(s') = \begin{cases} (1 - e^{-E(s)\delta})\mathbf{P}(s, \perp, s') & s' \neq s \\ (1 - e^{-E(s)\delta})\mathbf{P}(s, \perp, s') + e^{-E(s)\delta} & s' = s \end{cases}$$

The digitisation in Definition 8 approximates the original model by assuming that at most one Markov transition in $\mathcal{M}$ can occurs in each δ step. It is specified by distribution μ^s, which contains the probability of having either one or no Markov transition in $\mathcal{M}$ from state s within the digitisation step δ. Using the fixed point characterisation above, it is possible to relate reachability analysis in an *MA* with reachability analysis in its *dMA*. This is strongly inspired by the work of [Neu10], done for *IMC* and its digitised analogue *IPC*. The author introduces an error bound for reducing time interval bounded reachability problem in an *IMC* to step interval bounded reachability problem in its digitised (there called discretised) *IPC*. Because we rely on literally the same technique, we claim that the same error bound holds for computation of time bounded reachability in *MA*. This is because the fixed point characterisation of *MA* is identical to that of *IMC* for what concerns Markov states (see [ZN10].) Therefore to find the error bound starting from a Markov state, it is possible to reuse the same technique as [Neu10]. So we can focus on proving an error bound for interactive states. Altogether, we extend the error bound for time bounded reachability computation via digitisation from *IMC* to *MA*.

Theorem 3 *Given MA* $\mathcal{M} = (S, Act, \longrightarrow, \dashrightarrow, \nu)$*, a set of goal states* $G \subseteq S$*, an interval* $I = [0, b] \in \mathcal{Q}$ *with* $b > 0$ *and* $\lambda = \max_{s \in S_M} E(s)$*. Assume digitisation step* $\delta > 0$ *is selected such that* $b = k_b\delta$ *for some* $k_b \in \mathbb{N}$*. For all* $s \in S$ *it holds*

$$p_{\max}^{\mathcal{M}_\delta}(s, \Diamond^{[0,k_b]}G) \leq p_{\max}^{\mathcal{M}}(s, \Diamond^I G) \leq p_{\max}^{\mathcal{M}_\delta}(s, \Diamond^{[0,k_b]}G) + k_b\frac{(\lambda\delta)^2}{2}$$

Theorem 3 can be extended to intervals with nonzero lower bound.

Theorem 4 *Given* $\mathcal{M}$*,* G*,* λ *as Theorem 3 and a time interval* $I \in \mathcal{Q}$ *such that* $a = \inf I$ *and* $b = \sup I$ *with* $0 \leq a < b$*. Assume digitisation step* $\delta > 0$ *is selected such that* $b = k_b\delta$ *and* $a = k_a\delta$ *for some* $k_b, k_a \in \mathbb{N}$*. For all* $s \in S$ *it holds*

$$p_{\max}^{\mathcal{M}_\delta}(s, \Diamond^{(k_a,k_b]}G) - k_a\frac{(\lambda\delta)^2}{2} \leq p_{\max}^{\mathcal{M}}(s, \Diamond^I G) \leq p_{\max}^{\mathcal{M}_\delta}(s, \Diamond^{(k_a,k_b]}G) + k_b\frac{(\lambda\delta)^2}{2} + \lambda\delta$$

The full proof of Theorem 3 and 4 is long and basically follows the proof on *IMC*s. In both of the theorems, interval I is restricted to have rational bounds due to decidability issues [ASS+00]. For more details, we refer to [Neu10, Theorem 6.3 and 6.4].

```
Input  : $dMA\ \mathcal{D}$, the set of goal state $G \subseteq S$, the number of steps $k$
Output: Maximum reachability probabilities within $k$ steps ($p^{\mathcal{D}}_{\max}(.,\Diamond^{[0,k]}G)$)
begin
    make all $s \in G$ of $\mathcal{D}$ absorbing ;
    foreach $s \in G$ do $p^{\mathcal{D}}_{\max}(s,\Diamond^{[0,0]}G) := 1$ ;
    foreach $s \in S \setminus G$ do $p^{\mathcal{D}}_{\max}(s,\Diamond^{[0,0]}G) := 0$;
    for $j := 0$ to $k-1$ do
        // $i^*$-phase for step $j$ ;
        foreach $s \in S_I$ do $p^{\mathcal{D}}_{\max}(s,\Diamond^{[0,j]}G) := \sup\limits_{A \in TA} \sum\limits_{s' \in S_M} Pr^{\omega}_{A,v_s}(\Diamond^{[0,0]}\{s'\}) p^{\mathcal{D}}_{\max}(s',\Diamond^{[0,j]}G)$ ;
        // $m$-phase for step $j+1$ ;
        foreach $s \in S_M$ do $p^{\mathcal{D}}_{\max}(s,\Diamond^{[0,j+1]}G) = \sum_{s' \in S} \mathbf{P}(s,\perp,s') p^{\mathcal{D}}_{\max}(s',\Diamond^{[0,j]}G)$ ;
    end
    // $i^*$-phase for step $k$ ;
    foreach $s \in S_I$ do $p^{\mathcal{D}}_{\max}(s,\Diamond^{[0,k]}G) := \sup\limits_{A \in TA} \sum\limits_{s' \in S_M} Pr^{\omega}_{A,v_s}(\Diamond^{[0,0]}\{s'\}) p^{\mathcal{D}}_{\max}(s',\Diamond^{[0,k]}G)$ ;
end
```

Algorithm 1: Computing maximum step bounded reachability

6 Interval Bounded Reachability in Digitised Markov Automata

The results of the preceding section allows us to approximate time-interval bounded reachability in *MA* by step-interval bounded reachability in *dMA*. In this section we discuss how to compute step-interval bounded reachability in a given *dMA*.

Our goal is to compute maximum (or minimum) probability to reach a set of goal states within a given step interval. Let $\mathcal{D}$ be a *dMA*, $G \subseteq S$ be a set of goal state and $[k_1,k_2]$ be a nonempty step interval (i.e. $k_1 \leq k_2 \in \mathbb{N}_0$). We use $\Diamond^{[k_1,k_2]}G = \{\pi \in Paths^{\omega}_{ta} | \exists n \in \mathbb{N}_0 . \pi[n] \in G \wedge \Delta_{ta}(\pi,n) \in [k_1,k_2]\}$ to denote the set of all infinite paths that reach goal states G within the step interval $[k_1,k_2]$. We use $p^{\mathcal{D}}_{\max}(s,\Diamond^{[k_1,k_2]}G)$ to denote the maximum probability of reaching these goal states within $[k_1,k_2]$ step interval. Then we have $p^{\mathcal{D}}_{\max}(s,\Diamond^{[k_1,k_2]}G) = \sup_{A \in TA} Pr^{\omega}_{A,v_s}(\Diamond^{[k_1,k_2]}G)$. First we restrict ourselves to the case that the lower bound of the step interval is zero. Afterwards, we extend it to the general case. The algorithm that will be proposed is a variation of the value iteration algorithm for *MDP*, here adapted to *dMA*.

6.1 Step Bounded Reachability

We consider the case $[0,k]$, $k \in \mathbb{N}_0$ and apply a *modified value iteration algorithm* to compute reachability probabilities. The algorithm proceeds by backward unfolding of the *dMA* in an iterative manner, starting from the goal states. The overall algorithm is depicted in Algorithm 1. We initialise the algorithm by making all goal states absorbing: For these states, all outgoing transitions are removed, and replaced by a digitised Markov self loop (a transition to a Dirac distribution over the source state). The initial value of probability vector ($p^{\mathcal{D}}_{\max}(\cdot,[0,0])$) is set to one for goal states and to zero otherwise.

In the main loop of the algorithm, each iteration consists of an *m-phase* and an i^**-phase*. In an *m-phase*, reachability probabilities from Markov states are updated, while an i^*-phase updates reachability probabilities from interactive states. The algorithm then proceeds by intertwining m-phases and i^*-phases consecutively for k steps. After the j-th execution of the i^**-phase* (and thus after $j-1$ executions of the m-phase), the elements of $p^{\mathscr{D}}_{\max}(.,[0,j-1])$ are up to date. Therefore, a final i^**-phase* is needed. In the following we explain the functioning of the m-phase, respectively the i^*-phase.

In an *m-phase*, we update the reachability probabilities starting from Markov states by taking only one digitised Markov transition. This step is exactly the same as a standard value iteration step for a DTMC. Thus in step j, the m-phase updates the reachability probability for the goal states using the reachability probabilities calculated in step $j-1$.

In an i^**-phase*, we maximise the reachability probabilities starting from interactive states to Markov states by taking arbitrary many interactive transitions. The number of transitions does not matter, because they take zero time. In the algorithm, the j-th iteration operates on the previously computed reachability probabilities from all Markov states up to step j. The i^*-phase then computes the probability to reach Markov states within interval $[0,0]$ which implies that only interactive transitions can be taken.

The key innovation of this approach is that the path from interactive states to goal states is split in two: reaching Markov states from interactive states in zero time and reaching goal states from Markov states in interval $[0,j]$. Due to the memoryless property, the result comes from the product of the probability measure of these two sets of paths, but has to be maximised over all possible schedulers, i.e. $p^{\mathscr{D}}_{\max}(s,[0,j]) = \sup_{A\in TA}\sum_{s'\in S_M} Pr^{\omega}_{A,\nu_s}(\Diamond^{[0,0]}\{s'\})p^{\mathscr{D}}_{\max}(s',[0,j])$. Since the elapse of time is not important in the computation of $Pr^{\omega}_{A,\nu_s}(\Diamond^{[0,0]}\{s'\})$, we can view the problem as an unbounded reachability problem with rewards on Markov states $(p^{\mathscr{D}}_{\max}(s',[0,j]))$. It can be shown that stationary schedulers are sufficient for computing minimum and maximum objectives [BK08, Chapter 10] and these objectives can again be computed by using a value iteration algorithm.

6.2 Step Interval Bounded Reachability

Here we generalise step bounded reachability discussed previously to step interval bounded reachability. Assume we have $\mathscr{D}$, $G \subseteq S$ as before and step interval $[k_1,k_2]$. We break the interval into two pieces, first from k_2 down to k_1 and second from k_1 down to zero. Within the first, we are interested in reaching one of the goal states, as a result we make the goal states absorbing. Nevertheless, within the second, it does not matter that the model is in one of the goal states, which consequently leads us to ignore goal states and reintroduce them as before. Accordingly the algorithm proceeds as follows. In the first k_2-k_1 iterations, goal states are made absorbing and reachability probabilities are computed within $[0,k_2-k_1]$ interval. The result, $p^{\mathscr{D}}_{\max}(.,[0,k_2-k_1])$, will be used as the initial vector of the next k_1 iterations. Goal states are treated as normal states in the next k_1 iterations. Therefore we undo the states that have been made absorbing and set $G=\emptyset$. However other calculations remain the same as the first k_2-k_1 iterations.

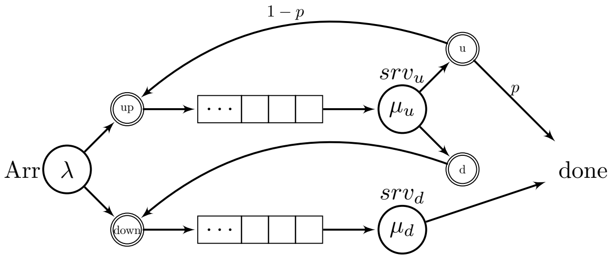

Figure 2: Case study: a system with two queues, two servers, and additional checks

7 Experimental Results

As a case study, we consider a system with two queues as depicted in Figure 2. This system can neither be modelled as a DTMC, nor as a CTMC, nor an *IMC*, nor a probabilistic automaton. The model family closest to the expressiveness needed here is that of continuous-time Markov decision processes. In the system, jobs arrive with rate λ and enter one of the queues according to the very common join-the-shortest-queue queueing strategy. But this strategy is implicitly nondeterministic if both queues are equally filled.

For each queue, jobs are processed by a dedicated server, serving jobs with rates μ_u and μ_d respectively. Jobs leaving the lower server leave the system, while jobs once processed by the upper server are subject to an additional check. Dependent on the (nondeterministic) outcome thereof they are either sent again into the lower queue (action d), or (action u) may leave the queue with probability p or reenter the upper queue with probability $1-p$. In the analysis, we let both queues to have equal capacity. We assume $AP = \{empty, full\}$ where $full$ identifies the states where both queues are full and $empty$ characterises the initial state, in which both queues are indeed empty.

For our first experiment, we set the size of both queues to 3 and compute maximum and minimum probability of reaching $full$ from $empty$ before t time unit, where t ranges from 0 to 10, i.e. $p^{\mathscr{M}}_{min}(empty, \Diamond^{[0,t]} full), p^{\mathscr{M}}_{\max}(empty, \Diamond^{[0,t]} full), t \in [0,10]$. The digitisation step has been set so as to guarantee that the error is at most 10^{-2}. The result depicted in Figure 3 demonstrates that regardless of the scheduler chosen, the chance of reaching $full$ near the end of the interval is relatively high.

For our second experiment, we increase the size of the queues while all other model parameters remain unchanged. We compute maximum and minimum reachability in different intervals with different precision parameters ε. As shown in Table 1, higher precision combined with larger

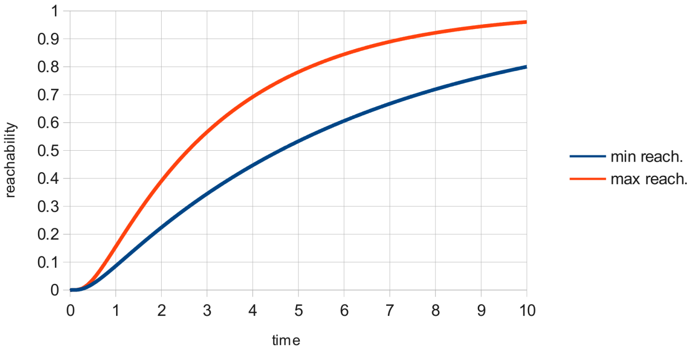

Figure 3: Minimum and maximum reachability for the system with $\lambda = 5$, $\mu_u = 10$, $\mu_d = 4$ and $p = 0.3$, and both queues having capacity 3

Table 1: Computation times for the queueing system with different queue sizes

size	# of states	ε	interval	min prob.	time	max prob.	time
4	69	10^{-5}	$[0,1]$	0.01023	13h	0.02317	13h
		10^{-4}	$[0,2]$	0.0585	5h	0.1412	5h
		10^{-2}	$[5,10]$	0.33	1h19min	0.67	1h15min
5	101	10^{-3}	$[0,1]$	0.001	24min	0.004	27min
		10^{-3}	$[5,10]$	0.164	41h	0.513	44h

interval bounds requires smaller digitisation steps, thereby making the analysis time consuming.

8 Conclusions

The paper has discussed a model checking approach for CSL formulae over Markov automata, with a focus on the most demanding feature, time bounded reachability. To arrive at a numerically stable approximation scheme, we intertwined a digitisation technique, which itself is a generalisation of [Neu10], with an embedded treatment of cascades of interactive transitions taking place in zero time. The resulting algorithm enables a calculation with a predefined accuracy, which can be chosen with respect to the upper bound of the intervals occurring in the formula and the maximum exit rate of the *MA* to be checked. The effectiveness of the proposed method has been demonstrated by some small case studies, that also demonstrate the expressiveness of the *MA* approach.

We have restricted our treatment to models where the probability for an infinite cascade of immediate steps is zero, thereby effectively excluding Zeno behaviour from the analysis. As a future work, we hope to generalise the current algorithms to *MA* with a non-zero risk for Zeno behaviour. Another worthwhile improvement concerns the error bound of the digitisation technique. This may help the solution to scale better for larger intervals and models with higher rates. Furthermore, we plan to integrate our algorithm into the IMCA tool [Guc12] that is being developed for *IMC* and *MA* analysis and to interface with the SCOOP tool [TKP+12], that supports compositional construction of *MA* models.

Bibliography

[ASS+00] Adnan Aziz, Kumud Sanwal, Vigyan Singhal, Robert Brayton: Model-checking continuous-time Markov chains. *ACM Transactions on Computational Logic* 1 (1): 162–170 (2000)

[AD00] Robert B. Ash, Catherine Dolans-Dade: Probability and Measure Theory. Second Edition, Academic Press (2000)

[BHH+03] Christel Baier, Boudewijn Haverkort, Holger Hermanns, Joost-Pieter Katoen: Model-checking algorithms for continuous-time Markov chains. *IEEE Transactions on Software Engineering* 29 (6): 524–541 (2003)

[BK08] Christel Baier, Joost-Pieter Katoen: Principles of Model Checking. MIT Press (2008)

[BA95] Andrea Bianco, Luca de Alfaro: Model checking of probabilistic and nondeterministic systems. In *FST-TCS*, LNCS 1026:499–513, Springer (1995)

[BCS10] Hichem Boudali, Pepijn Crouzen, Mariëlle Stoelinga: A Rigorous, Compositional, and Extensible Framework for Dynamic Fault Tree Analysis. *IEEE Transactions on Dependable and Secure Computing* 7(2): 128-143 (2010)

[CES86] Edmund M. Clarke, E. Allen Emerson, A. Prasad Sistla: Automatic verification of finite state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 8(2):244–263 (1986)

[CHL+09] Nicolas Coste, Holger Hermanns, Etienne Lantreibecq, Wendelin Serwe: Towards Performance Prediction of Compositional Models in Industrial GALS Designs. In *CAV*, LNCS:5643, 204–218, Springer (2009)

[DH11] Yuxin Deng, Matthew Hennessy: On the Semantics of Markov Automata. In *ICALP*. LNCS 6756:307–318, Springer (2011)

[EHZ10a] Christian Eisentraut, Holger Hermanns, Lijun Zhang: On Probabilistic Automata in Continuous Time. In *LICS*, 342–351 (2010)

[EHZ10b] Christian Eisentraut, Holger Hermanns, Lijun Zhang: Concurrency and Composition in a Stochastic World. In *CONCUR*, LNCS 6269:21–39, Springer (2010)

[EKN+12] Marie-Aude Esteve, Joost-Pieter Katoen, Viet Yen Nguyen, Bart Postma, Yuri Yushtein. Formal correctness, safety, dependability and performance analysis of a satellite. In *ICSE*. ACM and IEEE press, 2012.

[Guc12] Dennis Guck: Quantitative Analysis of Markov Automata. Master Thesis, RWTH Aachen University, (2012)

[HJ94] Hans Hansson, Bengt Jonsson: A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535 (1994)

[Her02] Holger Hermanns: Interactive Markov Chains: The Quest for Quantified Quality. LNCS 2428, Springer (2002)

[HJ08] Holger Hermanns, Sven Johr: May we reach it? Or must we? In what time? With what probability? In *14th GI/ITG Conference - Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB)*, 125–140 (2008)

[HKR+10] Boudewijn R. Haverkort, Matthias Kuntz, Anne Remke, S. Roolvink, Mariëlle Stoelinga: Evaluating repair strategies for a water-treatment facility using Arcade. In *DSN*, 419–424 (2010)

[Joh08] Sven Johr: Model checking compositional Markov systems. PhD Thesis, Saarland University, (2008)

[LZ05] Edward A. Lee, Haiyang Zheng: Operational Semantics of Hybrid Systems. In *HSCC*, LNCS 3414:25–53, Springer (2005)

[MP93] Zohar Manna, Amir Pnueli: Verifying Hybrid Systems. In *Hybrid Systems*, LNCS 736:4–35, Springer (1993)

[MCB84] Marco Ajmone Marsan, Gianni Conte, Gianfranco Balbo: A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems. *ACM Transactions on Computer Systems* 2(2):93–122 (1984)

[MMS85] John F. Meyer, Ali Movaghar, William H. Sanders: Stochastic Activity Networks: Structure, Behavior, and Application. In *International Workshop on Timed Petri Nets* IEEE Computer Society: 106–115 (1985)

[Neu10] Martin R. Neuhaeusser: Model Checking Nondeterministic and Randomly Timed Systems. PhD Thesis, RWTH Aachen University and University of Twente, (2010)

[Put94] Martin L. Puterman: Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley and Sons, New York, NY, (1994)

[Seg95] Roberto Segala: Modeling and verification of randomized distributed real-time systems. Ph.D. dissertation, Laboratory for Computer Science, Massachusetts Institute of Technology, (1995)

[TKP+12] Mark Timmer, Joost-Pieter Katoen, Jaco Van De Pol, Mariëlle Stoelinga: Efficient Modelling and Generation of Markov Automata. To appear in CONCUR, LNCS (2012)

[ZN10] Lijun Zhang, Martin R. Neuhaeusser: Model Checking Interactive Markov Chains. In *TACAS*, LNCS 6015:53–68, Springer (2010)