RWTH Aachen

Department of Computer Science
Technical Report

Three-Valued Abstraction for Probabilistic Systems

Joost-Pieter Katoen, Daniel Klink, Martin Leucker, Verena Wolf

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2007-20

RWTH Aachen · Department of Computer Science · December 2007

Three-Valued Abstraction for Probabilistic Systems*

Joost-Pieter Katoen[1], Daniel Klink[1], Martin Leucker[2], and Verena Wolf[3]

[1]RWTH Aachen University, [2]TU Munich, [3]University of Mannheim, Germany

Abstract. This paper proposes a novel abstraction technique for fully probabilistic systems. The models of our study are classical discrete-time and continuous-time Markov chains (DTMCs and CTMCs, for short). A DTMC is a Kripke structure in which each transition is equipped with a discrete probability; in a CTMC, in addition, state residence times are governed by negative exponential distributions. Our abstraction technique fits within the realm of three-valued abstraction methods that have been used successfully for traditional model checking. The key ingredients of our technique are a partitioning of the state space combined with an abstraction of transition probabilities by intervals. The uncertainty of intervals is resolved by history-dependent schedulers that may choose extreme values only. It is shown that this provides a conservative abstraction for both negative and affirmative verification results for a three-valued semantics of PCTL (Probabilistic Computation Tree Logic). In the continuous-time setting, the key idea is to apply abstraction on uniform CTMCs which are readily obtained from general CTMCs. In a similar way as for the discrete case, this is shown to yield a conservative abstraction for a three-valued semantics of CSL (Continuous Stochastic Logic). The verification of abstract DTMCs is inspired by the standard MDP (Markov Decision Process) model-checking problem. Abstract CTMCs can be verified by computing time-bounded reachability probabilities in continuous-time MDPs. Some experiments on an infinite-state stochastic Petri net indicate the feasibility of our abstraction technique.

1 Introduction

Model checking of probabilistic systems enjoys a rapid increase of interest. This technique has been successfully applied to case studies from areas such as randomised distributed algorithms, planning and AI, security (such as Crowds anonymity protocol), communication protocols (such as IEEE 802.11), systems biology, and quantum computing. Dedicated model checkers such as PRISM [34], MRMC [31], LiQuor [4], and YMER [47] support verifying a wide class of probabilistic models like Markov chains and Markov decision processes (MDPs). Existing performance and dependability analysis tools for stochastic Petri nets (SMART [11] and GreatSPN [12]), process algebras (the PEPA Workbench [18], a stochastic variant of the CWB [38]), and Statemate [10] have adopted probabilistic model checking as a prominent analysis technique. It is fair to say that probabilistic model checking has been proved to extend and complement long-standing analysis techniques for Markov processes.

Typical properties that are checked are quantitative reachability objectives, such as: "does the probability to reach a certain set of goal states (by avoiding illegal states) exceed $\frac{1}{2}$?". Often bounds are incorporated as well that ensure reaching the goal within a certain number of moves or real-time deadline. For MDPs—exhibiting both transition probabilities and nondeterminism—maximal

* The research has been partially funded by the DFG Research Training Group 1298 (AlgoSyn).

and minimal probabilities are considered. Intricate combinations of numerical or simulation techniques for Markov chains, optimisation algorithms, and traditional CTL model-checking algorithms result in simple, yet very efficient verification procedures. Verifying time-bounded reachability properties on models of tens of millions of states usually is a matter of seconds, cf. [28].

Like in the traditional setting, probabilistic model checking suffers from the state space explosion problem: the number of states grows exponentially in the number of system components and cardinality of data domains. To combat this problem, various techniques have been proposed ranging from the use of (multi-terminal) binary decision diagrams [2] and bisimulation minimization [30] to symmetry reduction [36] and a generalization of Peled's ample set method to MDPs [20]. This paper proposes a novel *abstraction* technique for Markov chains and is an extension of the results in [17] and [32].

The models of our study are discrete-time and continuous-time Markov chains (DTMCs and CTMCs, for short). DTMCs and CTMCs are a contemporary class of stochastic processes that are extensively used to model and analyze random phenomena in application domains such as planning of production lines and safety-critical systems. A DTMC is a Kripke structure in which each transition is equipped with a discrete probability describing the likelihood of moving from one state to another in a single move. In addition, in a CTMC state residence times are governed by negative exponential distributions. That is, the probability to stay in a state for at most t time units is $1-e^{-\lambda \cdot t}$ where λ is uniquely characterizing an exponential distribution. The average state residence time is $\frac{1}{\lambda}$.

Abstraction amounts to obtain smaller models by collapsing sets of concrete states to abstract states. Our abstraction technique is based on a partitioning of the concrete state space. In two-valued semantics, abstraction is typically conservative in the sense that affirmative verification results for abstract models carry over to concrete models. That is to say, if the abstract model satisfies a formula, the concrete one does so too. This does not apply to negative verification results, as false negatives may occur due to over-approximation in the abstraction. Promising results in traditional model checking have been obtained for a three-valued semantics of temporal logic formulae, i.e., an interpretation in which a formula evaluates to either true, false or indefinite. In this setting, abstraction is conservative for both positive and negative verification results. Only if the verification of the abstract model yields an indefinite answer ("don't know"), the validity in the concrete model is unknown. The abstraction techniques proposed in this paper follow this three-valued approach.

For the discrete-time setting, we consider abstractions for the branching-time logic PCTL (Probabilistic Computation Tree Logic [23]), whereas for the continuous-time case the logic CSL (Continuous Stochastic Logic [1, 5]) is regarded. CSL is a real-time probabilistic variant of CTL and is a powerful logic for expressing quantitative time-bounded constrained reachability properties such as the probability to reach a set of goal states (by avoiding bad states) within a maximal time span exceeds $\frac{1}{2}$. Existing abstraction techniques in this setting that have been applied in practice are based on such as bisimulation [30], matrix bounding [9], simulation [48] or symmetry reduction [36]; an extensive discussion of (other) related work is provided in Section 7. (Due to the absence of nondeterminism, techniques such as partial-order reduction do not yield substantial reductions.)

Despite the fact that fairly large reductions have recently been reported, more aggressive abstraction techniques are needed. Such techniques would also be useful to obtain finite abstractions for a large class of infinite-state Markov chains.

In classical model checking, three-valued abstraction yield abstract models (called Kripke modal transition systems [37, 27]) containing *may* and *must* transitions between aggregated states as over- and under-approximation, respectively, of the concrete transition relation. This concept can be lifted to DTMCs in a rather natural way [17, 24, 25] by replacing transition probabilities by *intervals*. Lower and upper bounds of intervals now act as under- and over-approximation, respectively. In fact, the resulting abstract model is of interest on its own, as pointed out in e.g., [29, 46, 33, 42], since often only bounds on probabilities are known rather than precise values. States in abstract DTMCs are thus groups of concrete states and transitions are equipped with intervals. As only certain combinations of intervals are meaningful, abstract DTMCs are normalized. We describe this normalization and present several of its properties, among others, that schedulers on abstract DTMCs and their normalization coincide. It is shown that concrete states are simulated—using Jonsson and Larsen's seminal notion of probabilistic simulation [29]—by their abstract counterparts. Finally, a three-valued semantics of PCTL is provided which is proven to be appropriate for the abstraction considered in the sense that any affirmative or negative verification result on an abstract DTMC carries over to the concrete model. If the verification yields indefinite, no conclusion can be drawn on the validity in the concrete model. Our model-checking algorithm for checking an abstract DTMC against a three-valued PCTL-formula is inspired by verification algorithms for MDPs.

A similar strategy is adopted for CTMCs. The main technical complication, however, is that besides transition probabilities, one has to determine the residence time of an abstract state that results from concrete states with distinct residence times. We show that intervals of transition probabilities, intervals on residence times (or combinations thereof) are not satisfactory in terms of precision. Instead, we suggest to overcome this imprecision by using *uniform* CTMCs, i.e., CTMCs in which all states have equal residence times and use transition probability intervals. Note that this is not a restriction, as any CTMC can be transformed into a weak bisimilar uniform CTMC in linear time and weak bisimulation preserves the validity of CSL formulas except (for the usual reasons) the next-step operator [6]. The resulting abstraction is shown to preserve simulation: concrete states are simulated by their corresponding abstract ones. Then we show that extreme schedulers, i.e. schedulers that only consider lower- and upper bounds, suffice for computing reachability probabilities up to a given tolerance ε rather efficiently [3]. Using a three-valued semantics of CSL it is shown that the abstraction is indeed conservative for affirmative and negative verification results. Besides, we show the relationship with the aforementioned approach for DTMCs.

Finally, the feasibility of the approach is shown by considering abstractions of different granularity for an unbounded stochastic Petri net.

Organization of this paper. Section 2 presents the stochastic models of our study, viz. DTMCs and CTMCs, and introduces the logics CSL and PCTL. Section 3 describes our notion of abstraction, defines abstract Markov chains, the relation between abstracting DTMCs and CTMCs, and presents the result that abstraction preserves probabilistic simulation. Section 4 shows how reachability probabilities can be determined in abstract DTMCs, and how time-bounded reachability probabilities can be obtained for abstract CTMCs. Section 5 presents three-valued semantics of PCTL and CSL, and includes the main result that abstraction preserves affirmative as well as negative verification results. The time complexity of the verification of abstract Markov chains is also analyzed. Finally, Section 6 reports on experiments for the abstraction of an unbounded stochastic Petri net, and Section 7 discusses alternative approaches and a comparison to related work.

2 Markov Chains and Their Logics

Let X be a finite set (for singleton sets, brackets may be omitted). For $Y, Y' \subseteq X$ and function $Q : X \times X \to \mathbb{R}_{\geq 0}$ let $Q(Y, Y') = \sum_{y \in Y, y' \in Y'} Q(y, y')$. The function $Q(x, \cdot)$ is given by $x' \mapsto Q(x, x')$ for all $x' \in X$. Furthermore a function f is called a *distribution on* X iff $f : X \to [0, 1]$ and $f(X) := \sum_{x \in X} f(x) = 1$. The set of all distributions on X is denoted by $distr(X)$. Let AP be a fixed, finite set of atomic propositions and $\mathbb{B}_2 = \{\perp, \top\}$ the two-valued truth domain.

Definition 1 (DTMC). *A DTMC is a tuple* $(S, \mathbf{P}, L)$ *with a finite non-empty set of states* S*, transition probability function* $\mathbf{P} : S \times S \to [0, 1]$ *satisfying* $\mathbf{P}(s, S) = 1$ *for all* $s \in S$*, and labeling function* $L : S \times AP \to \mathbb{B}_2$*.*

$\mathbf{P}(s, s')$ is the (time-independent) transition probability to move from s to s' and $L(s, a)$ states if atomic proposition a holds in s. A DTMC is time-abstract; in contrast, CTMCs are time-aware as they have an explicit reference to time in the form of exit rates which determine, together with the transition probabilities, the stochastic evolution of the system in time.

Definition 2 (CTMC). *A CTMC* $\mathcal{M}$ *is a tuple* $(S, \mathbf{P}, E, L)$ *with* S*,* $\mathbf{P}$ *and* L *as before, and* exit rate $E : S \to \mathbb{R}_{\geq 0}$.

The quantity $E(s)$ determines the random, exponentially distributed residence time of s. That is $1 - e^{-E(s) \cdot t}$ is the probability to take a transition emanating from s within the next t time units. Note that self-loops are admitted. We also say that a transition in state s occurs with an average *pace* of $E(s)$. The *time-dependent* transition probability to move from s to s' within t time units is now given by $\mathbf{P}(s, s', t) := \mathbf{P}(s, s') \cdot (1 - e^{-E(s) \cdot t})$.

The time-abstract probabilistic behaviour of CTMC $\mathcal{M}$ is described by its embedded DTMC: The *embedded* DTMC of CTMC $\mathcal{M} = (S, \mathbf{P}, E, L)$ is simply given by $emb(\mathcal{M}) = (S, \mathbf{P}, L)$.

A CTMC is *uniform* if all its states have the same exit rate, i.e., $E(s) = E_{unif}$ for all states $s \in S$. Each CTMC can be easily transformed into a uniform CTMC by adding self-loops. The idea of this transformation is to fix all exit rates to a value E_{unif}, say, the maximal exit rate occurring in the CTMC. This corresponds

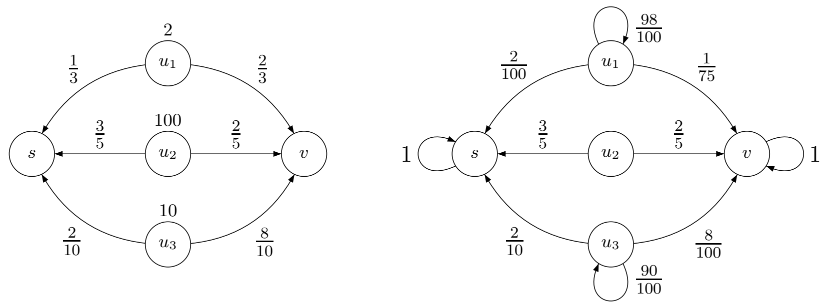


Fig. 1. Uniformization with exit rate $E_{unif} = 100$

to the state with the shortest mean residence time. If $E(s) < E_{unif}$, the mean residence time in s is longer than $\frac{1}{E_{unif}}$, and thus one epoch in the uniform CTMC may not be sufficiently "long". Therefore, state s is equipped with a self-loop whose probability determines the likelihood to stay in s during one epoch in the uniform CTMC. These self-loops slow down, so to speak, state s. The smaller $E(s)$, the higher the probability to reside in s. Intuitively speaking, E_{unif} determines the average "speed" of transitions in the uniform CTMC. Formally:

Definition 3 (Uniformization). *Let $\mathcal{M} = (S, \mathbf{P}, E, L)$ be a CTMC and let $E_{unif} \in \mathbb{R}_{>0}$ with $E_{unif} \geq \max_{s \in S} E(s)$ be the uniformization rate. Then, $unif(\mathcal{M}) = (S, \overline{\mathbf{P}}, \overline{E}, L)$ is a* uniform *CTMC with $\overline{E}(s) = E_{unif}$ for all $s \in S$ and*

$$\overline{\mathbf{P}}(s, s') = \mathbf{P}(s, s') \cdot \frac{E(s)}{E_{unif}} \text{ for } s' \neq s \quad \text{and} \quad \overline{\mathbf{P}}(s, s) = 1 - \overline{\mathbf{P}}(s, S \backslash \{s\}).$$

It should be noted that in the literature [21], uniformization is defined as a transformation of a CTMC $\mathcal{M}$ into the DTMC $emb(unif(\mathcal{M}))$. For technical convenience, we consider uniformization as a CTMC-to-CTMC transformation by basically adding self-loops to slow states. Strictly speaking, we should write $unif_{E_{unif}}(\mathcal{M})$ as the uniformization depends on E_{unif}. It is a known fact [6] that a CTMC is weakly probabilistic bisimilar to its uniformized CTMC.

Example 1. Consider the CTMC in Figure 1 (left) where the exit rates of states s and v are zero and 2, 100 and 10 for u_1, u_2 and u_3 respectively. Let $E_{unif} = 100$, the maximum exit rate. Then $\mathbf{P}(u_1, v) = \frac{2}{3} \cdot \frac{2}{100} = \frac{1}{75}$ and $\mathbf{P}(u_1, v) = \frac{1}{3} \cdot \frac{2}{100} = \frac{1}{150}$. Thus, u_1 is equipped with a self-loop of probability $1 - \frac{1}{75} - \frac{1}{150} = \frac{98}{100}$. The probabilities of self-loops for the other states are determined in a similar way, cf. Figure 1 (right). Note that state u_2 is only equipped with a self-loop if $E_{unif} > 100$.

Probabilistic CTL. PCTL [23] extends CTL by replacing existential and universal path quantification by a probability operator, denoted $\mathcal{P}$. The syntax of PCTL is:

$$\varphi ::= true \mid a \mid \varphi \wedge \varphi \mid \neg \varphi \mid \mathcal{P}_{\bowtie p}(\Psi) \qquad \Psi ::= \varphi \, \mathcal{U} \varphi$$

where $\bowtie \in \{<, \leq, \geq, >\}, p \in [0, 1]$ and $a \in AP$. φ is a *state-formula*, whereas Ψ is a *path-formula*. For the sake of simplicity, we neither consider the next-step operator, nor the bounded until operator of PCTL. For example, formula $\mathcal{P}_{>0.5}(true \, \mathcal{U} goal)$ asserts that the probability to reach a *goal* state eventually is at least $\frac{1}{2}$.

$[\![true]\!](s) \quad = \top$	$[\![a]\!](s) \quad = L(s,a)$
$[\![\varphi_1 \wedge \varphi_2]\!](s) = [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s)$	$[\![\neg\varphi]\!](s) = ([\![\varphi]\!](s))^c$
$[\![\mathcal{P}_{\bowtie p}(\Psi)]\!](s) = \top$, iff $Pr(\{\varsigma \in Paths_s^{\mathcal{M}} \mid [\![\Psi]\!](\varsigma) = \top\}) \bowtie p$	
$[\![\varphi_1 \mathcal{U} \varphi_2]\!](\varsigma) \quad = \top$, iff $\exists\, i \in \mathbb{N} : ([\![\varphi_2]\!](\varsigma[i]) = \top \wedge\ \forall\, 0 \leq j < i : [\![\varphi_1]\!](\varsigma[j]) = \top)$	

Table 1. Semantics of PCTL

A *path* in a DTMC is a sequence $\varsigma = s_0\, s_1\, s_2 \ldots$ with $\mathbf{P}(s_i, s_{i+1}) > 0$. Instead of $s_0\, s_1\, s_2 \ldots$ we also write $s_0 \rightarrow s_1 \rightarrow s_2 \ldots$. Let $\varsigma[i]$ denote the $(i{+}1)$-st state of a path, i.e., $\varsigma[i] = s_i$. Let Pr denote the unique probability measure on sets of paths and let $Paths_s^{\mathcal{M}}$ denote the set of all paths of DTMC $\mathcal{M}$, starting in s. (We use $\mathcal{M}$ to denote a DTMC and a CTMC; it should however be clear from the context which model is meant.) The subscript s is omitted when s is clear from the context; the same applies to superscript $\mathcal{M}$.

The semantics of PCTL is given in Table 1. $\top$ and $\bot$ form a complete lattice where $\bot < \top$ and *meet* $\sqcap$ as well as *complement* $\cdot^c$ are defined as usual.

Continuous Stochastic Logic. CSL [1, 5] extends PCTL by equipping the until-operator with a time bound (as in timed CTL):

$$\varphi ::= true \mid a \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathcal{P}_{\bowtie p}(\Psi) \qquad \Psi ::= \varphi\, \mathcal{U}^I \varphi$$

where $\bowtie\ \in \{<, \leq, \geq, >\}, p \in [0,1], I \in \{[0,t), [0,t], [0,\infty) \mid t \in \mathbb{R}_{>0}\}$ and $a \in AP$. φ is a *state-formula*, whereas Ψ is a *path-formula*. For example, the property to reach a *down* state in a CTMC within 52 time units, via *premium* states, with probability at most 10^{-4} can be formulated by the CSL formula $\mathcal{P}_{\leq 0.0001}(premium\, \mathcal{U}^{[0,52]}\, down)$.

A *path* in a CTMC is an alternating sequence $\sigma = s_0\, t_0\, s_1\, t_1\, s_2 \ldots$ with $\mathbf{P}(s_i, s_{i+1}) > 0$ and $t_i \in \mathbb{R}_{>0}$ for all $i \in \mathbb{N}$. The time stamps t_i denote the amount of time spent in state s_i. As the probability to reside zero time units in a state is zero, t_i is strictly positive. Path $s_0\, t_0\, s_1\, t_1\, s_2 \ldots$ is abbreviated as $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} s_2 \ldots$. The corresponding *time-abstract* path for σ is $\varsigma = s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots$. $\sigma[i]$ denotes the $(i{+}1)$-st state of a path and $\sigma@t$ denotes the state of σ occupied at time t, i.e. $\sigma@t = s_i$ with i the smallest index such that $t < \sum_{j=0}^{i} t_j$. Let Pr denote the unique probability measure on sets of paths and let $Paths_s^{\mathcal{M}}$ denote the set of all paths of $\mathcal{M}$, starting in s. The subscript s is omitted when s is clear from the context; the same applies to superscript $\mathcal{M}$. Note that the probability measure of the set of infinite paths $s_0\, t_0\, s_1\, t_1 \ldots$ with $\sum_{i=0}^{\infty} t_i \in \mathbb{R}$, is zero [5]. We thus may safely assume all paths to be non-Zeno. The parts of the semantics of CSL that differ from PCTL are listed in Table 2.

Time complexity. PCTL (and CSL) model checking is performed inductively on the structure of the state formula φ like for CTL model checking. This yields a time-complexity which is linear in the size of φ. Atomic formulas and boolean

$[\![\mathcal{P}_{\bowtie p}(\Psi)]\!](s) = \top$, iff $Pr(\{\sigma \in Paths_s^{\mathcal{M}} \mid [\![\Psi]\!](\sigma) = \top\}) \bowtie p$
$[\![\varphi_1 \mathcal{U}^I \varphi_2]\!](\sigma) = \top$, iff $\exists\, t \in I : ([\![\varphi_2]\!](\sigma@t) = \top \wedge\ \forall\, t' \in [0,t) : [\![\varphi_1]\!](\sigma@t') = \top)$

Table 2. Semantics of CSL

connectives are dealt with in the usual way. Checking time-bounded and (unbounded) until-formulas reduces to computing reachability probabilities. Standard reachability probabilities involve solving a linear equation system which can be done in $\mathcal{O}(|S|^3)$ where $|S|$ is the number of states in the Markov chain. Time-bounded reachability probabilities can be obtained by solving a Volterra integral equation system. Alternatively, a reduction to transient analysis can be done which yields a time complexity in $\mathcal{O}(|S|^2 \cdot E_{unif} \cdot t)$ where E_{unif} is the uniformization rate and t is the time bound. See [5] for a more detailed discussion.

Three-valued domain. As in our abstraction, states may be grouped that satisfy distinct atomic propositions, we resort to a three-valued interpretation. Let the *three-valued truth domain* $\mathbb{B}_3 = \{\bot, ?, \top\}$ be the complete lattice with ordering $\bot < ? < \top$, meet ($\sqcap$) and join ($\sqcup$) as expected, and complementation $\cdot^c$ is defined such that $\top$ and $\bot$ are complementary to each other and $?^c = ?$. When a formula evaluates to $\bot$ or $\top$, the result is called *definitely* true or false respectively, otherwise it is called *indefinite*.

3 Abstraction

This section discusses abstraction techniques for DTMCs and CTMCs. We first explain and justify the model of *abstract* Markov chains. One of the main principles is to abstract sets of transition probabilities by intervals. To eliminate inconsistent combinations of intervals that may result from an abstraction, we introduce a technique called *normalization*. This is formally defined and several properties of this operation are established. The main theoretical achievement in this section is that concrete states are simulated by their corresponding abstract states. We conclude this section by studying the relationship between abstraction of DTMCs and of CTMCs.

3.1 Abstract Markov Chains

The discrete-time setting. Our aim is to provide an abstraction of DTMCs which is conservative for both positive and negative verification results of PCTL formulas. This is established by adopting a three-valued interpretation. The basic principle is to collapse sets of concrete states into single abstract states such that concrete states are simulated by abstract ones. As opposed to abstract interpretation only disjoint sets of concrete states are collapsed. That is, we consider a partitioning $\mathcal{A} = \{A_1, \ldots, A_n\}$ of the state space S of a Markov chain. The probability to evolve from abstract state A_i to A_j, $i, j \in \{1, \ldots, n\}$ is represented by:

$$\mathbf{P}(A_i, A_j) = \{\mathbf{P}(s, s') \mid s \in A_i, s' \in A_j\}.$$

Taking minimal and maximal probabilities as under- and over-approximation, respectively, suggests to define:

$$\mathbf{P}^l(A_i, A_j) = \inf_{p \in \mathbf{P}(A_i, A_j)} p \quad \text{and} \quad \mathbf{P}^u(A_i, A_j) = \sup_{p \in \mathbf{P}(A_i, A_j)} p$$

as minimal and maximal probabilities. This yields the following abstract model:

Definition 4 (Abstract DTMC). *An* abstract *DTMC (ADTMC for short) is a tuple* $(S, \mathbf{P}^l, \mathbf{P}^u, L)$ *with*

– S*, a finite non-empty set of states,*
– $\mathbf{P}^l, \mathbf{P}^u : S \times S \to [0,1]$*, transition probability bounds such that for all* $s \in S$*:*

$$\mathbf{P}^l(s, S) \leq 1 \leq \mathbf{P}^u(s, S),$$

– $L : S \times AP \to \mathbb{B}_3$*, a labeling function.*

An ADTMC $\mathcal{M}$ has a finite state space and is equipped with a pair of functions describing the lower and upper bound, respectively for the transition probabilities. In contrast to DTMCs, states in an ADTMC may be labeled with ? as in principle it is possible that concrete states labeled with different propositions are grouped and considered as a single abstract state. It follows that an ADTMC $(S, \mathbf{P}^l, \mathbf{P}^u, L)$ with $\mathbf{P}^l = \mathbf{P}^u$ and $L(s,a) \neq ?$ for any $s \in S$ and $a \in AP$ is a DTMC.

ADTMCs are thus DTMCs where the exact transition probability is unknown but provided by an interval. Variants of DTMCs with intervals are also used for different purposes in e.g., [46, 33, 42, 45]. This uncertainty will, as we will discuss in detail later on in this section, be resolved by schedulers (also known as policies, strategies or adversaries), that may select a transition probability from an interval. This *set* of transition probability functions for ADTMC $\mathcal{M}$ is given by:

$$\mathbf{T}_{\mathcal{M}} = \{\mathbf{P} \in \mathit{distr}(S) \mid \mathbf{P}^l \leq \mathbf{P} \leq \mathbf{P}^u\},$$

where $\leq$ is to be interpreted element-wise. We drop subscript $\mathcal{M}$ if $\mathcal{M}$ is clear from the context and write $\mathbf{T}(s, \cdot)$ for the set $\{\mathbf{P}(s, \cdot) \mid \mathbf{P} \in \mathbf{T}\}$ of so-called *valid* distributions for state s.

We often equip an ADTMC with initial distribution $\alpha \in \mathit{distr}(S)$ meaning that $\mathcal{M}$ initially starts with probability $\alpha(s)$ in state s. A *path* in an ADTMC is a sequence $\varsigma = s_0\, s_1\, \ldots$ if there exists $\mathbf{P}_0, \mathbf{P}_1, \ldots \in \mathbf{T}$ such that $\mathbf{P}_i(s_i, s_{i+1}) > 0$ for all $i \in \{0, 1, \ldots\}$. In the sequel, we use the same notations as for paths in DTMCs, e.g. $\varsigma[i]$, $\mathit{Paths}_s^{\mathcal{M}}$ and so forth. If α is the initial distribution of $\mathcal{M}$, we define

$$\mathit{Paths}^{\mathcal{M}} = \bigcup_{s:\alpha(s)>0} \mathit{Paths}_s^{\mathcal{M}}.$$

Before providing a formal definition of what an abstraction is, let us consider an example to convey the intuition.

Example 2. Consider the (A)DTMC in Fig. 2 (left), $AP = \{a\}$, $L(s_0, a) = L(s_1, a) = \top$ and $L(s_0', a) = L(s_2, a) = \bot$. The ADTMC induced by partition $\{\underbrace{\{s_0, s_0'\}}_{=A_0}, \underbrace{\{s_1\}}_{=A_1}, \underbrace{\{s_2\}}_{=A_2}\}$ is depicted in Fig. 2 (right) with $L(A_0, a) = ?$, $L(A_1, a) = \top$, $L(A_2, a) = \bot$.

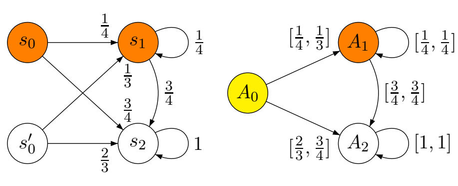


Fig. 2. Abstracting a DTMC

Abstraction of a given ADTMC is formally defined as follows:

Definition 5 (Abstraction). *For ADTMC* $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$*, partitioning* $\mathcal{A} = \{A_1, \ldots, A_n\}$ *of* S *and* $1 \leq i, j \leq n$*, the* abstraction *of* $\mathcal{M}$ *induced by* $\mathcal{A}$ *is the ADTMC* $abstr(\mathcal{A}, \mathcal{M}) := (\mathcal{A}, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L})$ *given by:*

- $\tilde{\mathbf{P}}^l(A_i, A_j) = \min_{s \in A_i} \mathbf{P}^l(s, A_j)$,
- $\tilde{\mathbf{P}}^u(A_i, A_j) = \min\{1, \max_{s \in A_i} \mathbf{P}^u(s, A_j)\}$,
- $\tilde{L}(A_i, a) = \begin{cases} \top & \text{if } L(s,a) = \top \quad \text{for all } s \in A_i, a \in AP, \\ \bot & \text{if } L(s,a) = \bot \quad \text{for all } s \in A_i, a \in AP, \\ ? & \text{otherwise.} \end{cases}$

Note that $\mathbf{P}^u(A, A')$ is defined as the minimum of one and the maximum of all transition probabilities from concrete states in A to states in A'. To cut the upper probability bound at one is necessary as $\mathbf{P}^u(s, A_j)$ may exceed one, which would not yield an ADTMC. E.g., if A_1 and A_2 are merged to $A_{1,2}$ in Example 2, one obtains $\mathbf{P}^u(A_0, A_{1,2}) = \frac{1}{3} + \frac{3}{4} > 1$.

It follows that the class of ADTMCs is closed under the above notion of abstraction, i.e., abstracting an ADTMC yields an ADTMC:

Lemma 1. *For any ADTMC* $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$ *and any partitioning* $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ *of* S*,* $abstr(\mathcal{M}, \mathcal{A})$ *is an ADTMC.*

Proof. S is a finite non-empty set of states, therefore $\mathcal{A}$ is non-empty and finite. For the labeling function there is nothing to show. It remains to prove that probability bounds $\tilde{\mathbf{P}}^l$ and $\tilde{\mathbf{P}}^u$ map to $[0,1]$ and $\tilde{\mathbf{P}}^l(A, \mathcal{A}) \leq 1 \leq \tilde{\mathbf{P}}^u(A, \mathcal{A})$ for all $A \in \mathcal{A}$:

- $$\begin{array}{ll} \mathbf{P}^l(s, S) \leq 1 \text{ and } \mathbf{P}^l(s, s') \in [0,1] & \text{for all } s, s' \in S \\ \Rightarrow \mathbf{P}^l(s, A_j) \in [0,1] & \text{for all } s \in S, A_j \subseteq S \\ \Rightarrow \min_{s \in A_i} \mathbf{P}^l(s, A_j) \in [0,1] & \text{for all } A_i, A_j \subseteq S \\ \Rightarrow \tilde{\mathbf{P}}^l(A_i, A_j) \in [0,1] & \text{for all } A_i, A_j \in \mathcal{A} \end{array}$$
- $$\begin{array}{ll} \mathbf{P}^u(s, s') \in [0,1] & \text{for all } s, s' \in S \\ \Rightarrow \mathbf{P}^u(s, A_j) \geq 0 & \text{for all } s \in S, A_j \subseteq S \\ \Rightarrow \min\{1, \max_{s \in A_i} \mathbf{P}^u(s, A_j)\} \in [0,1] & \text{for all } A_i, A_j \subseteq S \\ \Rightarrow \tilde{\mathbf{P}}^u(A_i, A_j) \in [0,1] & \text{for all } A_i, A_j \in \mathcal{A} \end{array}$$
- for all $A_i \in \mathcal{A}$:
$$\begin{aligned} \tilde{\mathbf{P}}^l(A_i, \mathcal{A}) &= \textstyle\sum_{A_j \in \mathcal{A}} \min_{s \in A_i} \mathbf{P}^l(s, A_j) \\ &\leq \textstyle\sum_{A_j \in \mathcal{A}} \mathbf{P}^l(\hat{s}, A_j) \qquad \text{for all } \hat{s} \in A_i \\ &= \mathbf{P}^l(\hat{s}, S) \leq 1 \end{aligned}$$
- for all $A_i \in \mathcal{A}$:
$$\begin{aligned} \tilde{\mathbf{P}}^u(A_i, \mathcal{A}) &= \textstyle\sum_{A_j \in \mathcal{A}} \min\{1, \max_{s \in A_i} \mathbf{P}^u(s, A_j)\} \\ &\geq \textstyle\sum_{A_j \in \mathcal{A}} \min\{1, \mathbf{P}^u(\hat{s}, A_j)\} \qquad \text{for all } \hat{s} \in A_i \\ &\geq \min\{1, \textstyle\sum_{A_j \in \mathcal{A}} \mathbf{P}^u(\hat{s}, A_j)\} \\ &= \min\{1, \underbrace{\mathbf{P}^u(\hat{s}, S)}_{\geq 1}\} = 1 \end{aligned}$$

Thus, $abstr(\mathcal{M}, \mathcal{A})$ is an ADTMC.

□

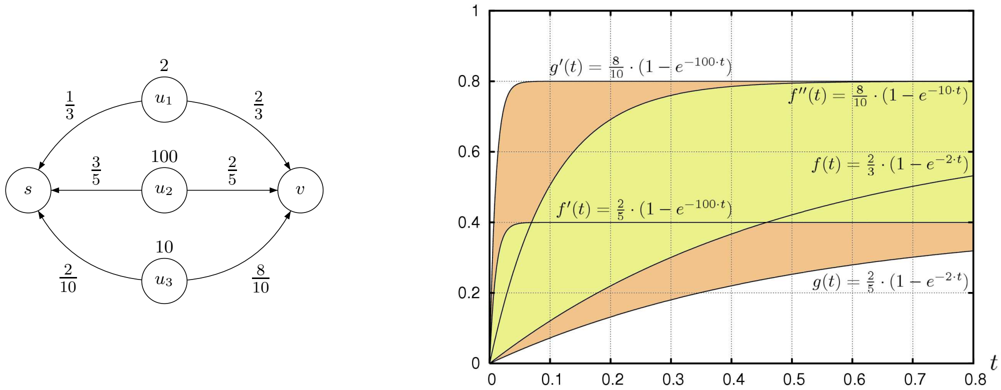


Fig. 3. Abstraction for non-uniform CTMCs

The continuous-time setting. Let us now consider CTMCs. In order to apply a similar abstraction technique to CTMCs, it is natural to group states and consider again intervals of transition probabilities. The main technical complication, however, is that besides the transition probabilities, we have to determine the residence time of an abstract state that results from concrete states with possibly distinct residence times. Let us consider this in a bit more detail, and recall that $\mathbf{P}(s, s', t)$ is the probability to move from state s to s' within t time units. Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be a partitioning of the state space S. The probability to evolve from abstract state A_i to A_j, $i, j \in \{1, \ldots, n\}$ is represented by the functions:

$$\mathbf{P}(A_i, A_j) = \{\mathbf{P}(s, s', \cdot) \mid s \in A_i, s' \in A_j\}.$$

Taking minimal and maximal probabilities as under- and over-approximation—as in the discrete case—respectively, suggests to define:

$$\mathbf{P}^l(A_i, A_j, t) = \inf_{f \in \mathbf{P}(A_i, A_j)} f(t) \quad \text{and} \quad \mathbf{P}^u(A_i, A_j, t) = \sup_{f \in \mathbf{P}(A_i, A_j)} f(t).$$

Observe that the functions $\mathbf{P}^l(A_i, A_j, t)$ and $\mathbf{P}^u(A_i, A_j, t)$ (considered as functions ranging over t) are in general not of the form $p \cdot (1 - e^{-E \cdot t})$ for fixed $p \in [0, 1]$ and $E > 0$. This is illustrated by the following example.

Example 3. Consider the non-uniform CTMC $\mathcal{M} = (\{s, u_1, u_2, u_3, v\}, \mathbf{P}, E, L)$ in Fig. 3 (left). We focus on the transition probabilities of the states u_1, u_2, u_3 (indicated as labeled edges) and their exit rates which appear above the corresponding vertices. Further details of $\mathcal{M}$ are omitted in Fig. 3. Let $\mathcal{A} = \{A_s, A_u, A_v\}$ with $A_u = \{u_1, u_2, u_3\}$, $A_s = \{s\}$ and $A_v = \{v\}$. The set $\mathbf{P}(A_u, A_v) = \{f, f', f''\}$ is plotted in Fig. 3 (right). Note that $\mathbf{P}^l(A_u, A_v, t)$ and $\mathbf{P}^u(A_u, A_v, t)$ are not of the form $p \cdot (1 - e^{-E \cdot t})$. In general, these functions get more complex as the number of transitions between states in A_u and A_v increases.

One might combine the infimum (supremum) of an abstract state's exit rates with the infimum (supremum) of the time-dependent transition probabilities to define an appropriate under- and over-approximation. This yields however a rather coarse abstraction as indicated in Fig. 3 (right) which shows the plot of the functions g and g' resulting from this approach. Increasing the number of parameters to obtain a more accurate approximation results in a far too complex abstraction. Therefore, we propose to abstract a CTMC by generating its uniformised CTMC (cf. Def. 3), and apply abstraction on the uniform CTMC, i.e., CTMCs in which all exit rates are equal to, say, E_{unif}. The advantage of uniform CTMCs is that (for $t \neq 0$):

$$p_l{\cdot}(1 - e^{-E_{unif}{\cdot}t}) \;\leq\; p_u{\cdot}(1 - e^{-E_{unif}{\cdot}t}) \text{ iff } p_l \leq p_u$$

where p_l, p_u are the lower and upper bounds of time-independent transition probabilities. Note that CTMC $\mathcal{M}$ and *unif*($\mathcal{M}$) are weakly bisimilar, and as weak bisimulation preserves CSL equivalence[1] [6], the shift to the uniformized CTMC is correct for CSL. The abstract notion of a CTMC now becomes:

Definition 6 (Abstract CTMC). *An* abstract *CTMC (ACTMC) is a tuple $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E_{unif}, L)$ with S, $\mathbf{P}^l$, $\mathbf{P}^u$, and L as before and with $E_{unif} \in \mathbb{R}_{>0}$, the (global) exit rate for all states.*

As for ADTMCs we sometimes consider ACTMCs with initial distributions $\alpha \in distr(S)$. A *(timed) path* in an ACTMC is a sequence $\sigma = s_0\, t_0\, s_1\, t_1 \ldots$ if $t_i \in \mathbb{R}_{>0}$ and there exists $\mathbf{P}_0, \mathbf{P}_1, \ldots \in \mathbf{T}$ such that $\mathbf{P}_i(s_i, s_{i+1}) > 0$ for all $i \in \{0, 1, \ldots\}$. In the sequel, we use the same notations as for paths in CTMCs, e.g. $\sigma[i], \sigma@t, Paths_s^{\mathcal{M}}$ and so forth. If α is the initial distribution of $\mathcal{M}$ the set $Paths^{\mathcal{M}}$ is the set of all paths σ starting in some s with $\alpha(s) > 0$. Similarly, $Paths_{abs}^{\mathcal{M}}$ denotes the set of all time-abstract paths ς starting in some s with $\alpha(s) > 0$.

An ACTMC with $\mathbf{P}^l = \mathbf{P}^u$ and $L(s, a) \neq ?$ for any $s \in S$ and $a \in AP$ is a *uniform* CTMC.

Definition 7 (Embedded ADTMC). *For ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E, L)$, the* embedded *ADTMC is $emb(\mathcal{M}) = (S, \mathbf{P}^l, \mathbf{P}^u, L)$.*

The abstraction of ACTMC $\mathcal{M}$ with respect to the partitioning $\mathcal{A} = \{ A_1, \ldots, A_n \}$ is denoted $abstr(\mathcal{M}, \mathcal{A})$ and is defined as for ADTMCs where in addition the exit rate of each state equals E_{unif}, the uniform exit rate in $\mathcal{M}$. It follows directly that $abstr(\mathcal{M}, \mathcal{A})$ is indeed an ACTMC.

The next two subsections will be mainly focused on (abstract) DTMCs. All definitions and results apply to (abstract) CTMCs as well unless stated otherwise.

3.2 Normalization

When transitions are labeled with sets (or intervals) of probabilities, it may occur that not every combination of values for different transitions emanating the same state, yield a probability distribution. For instance, in Example 2, a possible choice in state A_0 is to select A_1 with $\frac{1}{4}$ and A_2 with $\frac{2}{3}$, but $\frac{1}{4} + \frac{2}{3} < 1$. The reason is that after fixing one transition probability (by narrowing a transition probability interval to a point interval), the ADTMC is not *delimited* anymore in the following sense.

Definition 8 (Delimited ADTMC). *An ADTMC $\mathcal{M}_{\mathcal{A}}$ is* delimited *iff for any $s, s' \in S$ and $p \in [\mathbf{P}^l(s, s'), \mathbf{P}^u(s, s')]$, there exists $\mathbf{P} \in \mathbf{T}$ with $\mathbf{P}(s, s') = p$.*

[1] Recall that we consider the fragment of CSL without the next-step operator.

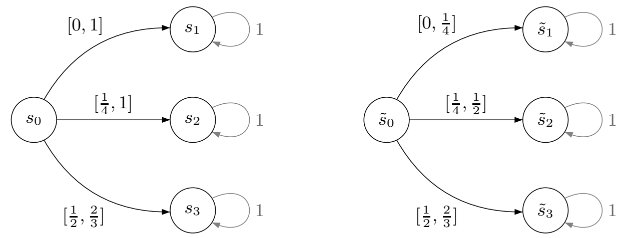


Fig. 4. Transforming an ADTMC (left) into a delimited one (right)

Note that abstracting a DTMC yields a delimited ADTMCs in contrast to abstraction of (even delimited) ADTMCs, where the results are in general ADTMCs that are not delimited. The operation, called *normalization*, aims to transform a given ADTMC into a delimited one. For all $s, s' \in S$, it removes those values $p \in [\mathbf{P}^l(s,s'), \mathbf{P}^u(s,s')]$ for which there is no $\mathbf{P} \in \mathbf{T}$ with $\mathbf{P}(s,s') = p$. Formally,

Definition 9 (Normalization). *For ADTMC* $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$*, let* $norm(\mathcal{M}) = (S, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, L)$ *be the* normalization *of* $\mathcal{M}$ *where for all* $s, s' \in S$*:*

$$\tilde{\mathbf{P}}^l(s,s') = \max\{\mathbf{P}^l(s,s'), 1 - \mathbf{P}^u(s, S \setminus \{s'\})\} \text{ and}$$
$$\tilde{\mathbf{P}}^u(s,s') = \min\{\mathbf{P}^u(s,s'), 1 - \mathbf{P}^l(s, S \setminus \{s'\})\}.$$

For the sake of brevity, we sometimes write $norm(\mathbf{P}^l, \mathbf{P}^u) = (\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u)$ rather than $norm(\mathcal{M}) = (S, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, L)$. In order to determine a valid distribution for a given state s, normalization has to be applied up to $|S|$ times (for every possible successor), and thus should be implemented efficiently.

Example 4. Consider the ADTMC in Fig. 4 (left). Normalization yields new upper bounds for the transitions from s_0 to s_1 and to s_2 (cf. the ADTMC in Fig. 4 right) according to:

$$\tilde{\mathbf{P}}^u(s_0,s_1) = 1 - \mathbf{P}^l(s_0,s_2) - \mathbf{P}^l(s_0,s_3) = 1 - \tfrac{1}{4} - \tfrac{1}{2} = \tfrac{1}{4} \text{ and}$$
$$\tilde{\mathbf{P}}^u(s_0,s_2) = 1 - \mathbf{P}^l(s_0,s_1) - \mathbf{P}^l(s_0,s_3) = 1 - 0 - \tfrac{1}{2} = \tfrac{1}{2}.$$

However, the upper bound from s_0 to s_3 remains unchanged as $(0, \frac{1}{3}, \frac{2}{3}) \in \mathbf{T}$. Furthermore, no lower bound is normalized as the adjustments to upper bounds for transitions from s_0 to s_1 and s_2 ensure that for all $s, s' \in \{s_0, s_1, s_2, s_3\}$ there exists a $\mathbf{P} \in \mathbf{T}$ with $\mathbf{P}(s,s') = \mathbf{P}^l(s,s')$:

$$\tilde{\mathbf{P}}^u(s_0,s_1) + \mathbf{P}^l(s_0,s_2) + \mathbf{P}^l(s_0,s_3) + \overbrace{\mathbf{P}^l(s_0,s_0)}^{=0} = 1 \text{ and}$$
$$\tilde{\mathbf{P}}^u(s_0,s_2) + \mathbf{P}^l(s_0,s_1) + \mathbf{P}^l(s_0,s_3) + \mathbf{P}^l(s_0,s_0) = 1.$$

In general, the need to normalize probability bound $\mathbf{P}^l(s,s')$ (or $\mathbf{P}^u(s,s')$), only depends on the set of bounds $\mathbf{P}^u(s,s'')$ ($\mathbf{P}^l(s,s'')$ respectively) with $s'' \neq s'$ (cf. Def. 9). The following lemma states that for both cases these bounds do not change during normalization, once they have been used to adapt another bound.

Lemma 2. *Let ADTMC* $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$ *and* $norm(\mathbf{P}^l, \mathbf{P}^u) = (\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u)$*. Then it holds for all* $s, s' \in S$*:*

$$\tilde{\mathbf{P}}^l(s,s') \neq \mathbf{P}^l(s,s') \text{ implies } \forall s'' \neq s' : \tilde{\mathbf{P}}^u(s,s'') = \mathbf{P}^u(s,s'') \text{ and}$$
$$\tilde{\mathbf{P}}^u(s,s') \neq \mathbf{P}^u(s,s') \text{ implies } \forall s'' \neq s' : \tilde{\mathbf{P}}^l(s,s'') = \mathbf{P}^l(s,s'').$$

Proof. Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$.

Case 1: Assume $\mathbf{P}^l(s,s')$ has to be raised, i.e. $\mathbf{P}^l(s,s') \leq \tilde{\mathbf{P}}^l(s,s')$, then

$$\begin{aligned}
&\tilde{\mathbf{P}}^l(s,s') = 1 - \mathbf{P}^u(s, S \setminus \{s'\}) \\
\Leftrightarrow\ &\tilde{\mathbf{P}}^l(s,s') + \mathbf{P}^u(s, S \setminus \{s'\}) = 1 \\
\Rightarrow\ &\forall s'' \neq s' : \tilde{\mathbf{P}}^u(s,s'') = \min\{\mathbf{P}^u(s,s''), 1 - \tilde{\mathbf{P}}^l(s, S \setminus \{s''\})\} \stackrel{(*)}{=} \mathbf{P}^u(s,s'') \\
&(*) \text{ since } \quad 1 - \tilde{\mathbf{P}}^l(s, S \setminus \{s''\}) \\
&\qquad\qquad \geq 1 - \mathbf{P}^u(s, S \setminus \{s', s''\}) - \tilde{\mathbf{P}}^l(s,s') \\
&\qquad\qquad = 1 - \mathbf{P}^u(s, S \setminus \{s', s''\}) - (1 - \mathbf{P}^u(s, S \setminus \{s'\})) \\
&\qquad\qquad = \mathbf{P}^u(s,s'')
\end{aligned}$$

Case 2: Assume $\mathbf{P}^u(s,s')$ has to be lowered, i.e. $\mathbf{P}^u(s,s') \geq \tilde{\mathbf{P}}^u(s,s')$, then

$$\begin{aligned}
&\tilde{\mathbf{P}}^u(s,s') = 1 - \mathbf{P}^l(s, S \setminus \{s'\}) \\
\Rightarrow\ &\tilde{\mathbf{P}}^u(s,s') + \mathbf{P}^l(s, S \setminus \{s'\}) = 1 \\
\Rightarrow\ &\forall s'' \neq s' : \tilde{\mathbf{P}}^l(s,s'') = \max\{\mathbf{P}^l(s,s''), 1 - \mathbf{P}^u(s, S \setminus \{s''\})\} \stackrel{(*)}{=} \mathbf{P}^l(s,s'') \\
&(*) \text{ since } \quad 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{s''\}) \\
&\qquad\qquad \leq 1 - \mathbf{P}^l(s, S \setminus \{s', s''\}) - \tilde{\mathbf{P}}^u(s,s') \\
&\qquad\qquad = 1 - \mathbf{P}^l(s, S \setminus \{s', s''\}) - (1 - \mathbf{P}^l(s, S \setminus \{s'\})) \\
&\qquad\qquad = \mathbf{P}^l(s,s'')
\end{aligned}$$

□

Normalization has to be performed only once, since it is idempotent. Together with the above lemma, this shows that the worst case time complexity of normalizing an ADTMC is quadratic in the size of its state space.

Lemma 3. *For any ADTMC $\mathcal{M}$, $norm(\mathcal{M}) = norm(norm(\mathcal{M}))$.*

Proof. We show that $(\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u) = norm(\mathbf{P}^l, \mathbf{P}^u) = norm(norm(\mathbf{P}^l, \mathbf{P}^u)) = (\hat{\mathbf{P}}^l, \hat{\mathbf{P}}^u)$. It follows from the definition of normalization that:

$$\begin{aligned}
\tilde{\mathbf{P}}^l(s,s') &= \max\{\mathbf{P}^l(s,s'), 1 - \mathbf{P}^u(s, S \setminus \{s'\})\} \\
\tilde{\mathbf{P}}^u(s,s') &= \min\{\mathbf{P}^u(s,s'), 1 - \mathbf{P}^l(s, S \setminus \{s'\})\}
\end{aligned}$$

For $\tilde{\mathbf{P}}^l(s,s')$ we investigate two cases:

1. $\tilde{\mathbf{P}}^l(s,s') = 1 - \mathbf{P}^u(s, S \setminus \{s'\})$ and $\tilde{\mathbf{P}}^l(s,s') \neq \mathbf{P}^l(s,s')$. Lemma 2 implies for all $s'' \neq s'$, $\tilde{\mathbf{P}}^u(s,s'') = \mathbf{P}^u(s,s'')$. Thus $\tilde{\mathbf{P}}^l(s,s') = 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{s'\})$ and:

$$\hat{\mathbf{P}}^l(s,s') = \max\{\tilde{\mathbf{P}}^l(s,s'), 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{s'\})\} = \tilde{\mathbf{P}}^l(s,s').$$

2. $\tilde{\mathbf{P}}^l(s,s') = \mathbf{P}^l(s,s')$. Then, for any $s'' \neq s'$:

$$\begin{aligned}
&\tilde{\mathbf{P}}^u(s,s'') \geq 1 - \mathbf{P}^l(s, S \setminus \{s''\}) \\
\Leftrightarrow\ &\tilde{\mathbf{P}}^u(s,s'') \geq 1 - \mathbf{P}^l(s, S \setminus \{s', s''\}) - \mathbf{P}^l(s,s') \\
\Leftrightarrow\ &\mathbf{P}^l(s,s') \ \ \geq 1 - \mathbf{P}^l(s, S \setminus \{s', s''\}) - \tilde{\mathbf{P}}^u(s,s'') \\
\Rightarrow\ &\mathbf{P}^l(s,s') \ \ \geq 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{s'\}).
\end{aligned}$$

This yields:

$$\max\{\underbrace{\tilde{\mathbf{P}}^l(s,s')}_{\mathbf{P}^l(s,s')}, 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{s'\})\} \ = \ \mathbf{P}^l(s,s') \ = \ \tilde{\mathbf{P}}^l(s,s').$$

Analogously, it can be shown that $\hat{\mathbf{P}}^u(s, s') = \tilde{\mathbf{P}}^u(s, s')$.

□

The following result is concerned with the correctness of normalization. It asserts that normalization indeed yields a delimited ADTMC. In the next subsection, we will show that $\mathcal{M}$ and $norm(\mathcal{M})$ are equivalent in a sense that will be detailed later on.

Lemma 4. *For any ADTMC $\mathcal{M}$, $norm(\mathcal{M})$ is delimited.*

Proof. Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$, $norm(\mathbf{P}^l, \mathbf{P}^u) = (\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u)$ and $s, s' \in S$. We show that $norm(\mathcal{M})$ is delimited, i.e., for any lower and upper bound of the probability intervals, there is a corresponding $\bar{\mathbf{P}} \in \mathbf{T}_{norm(\mathcal{M})}$. For each $s \in S$, distinguish two cases:

- $\tilde{\mathbf{P}}^l(s, s') = 1 - \mathbf{P}^u(s, S \setminus \{s'\})$. Then for a proper $\bar{\mathbf{P}}$, for $s'' \in S$:
  $\bar{\mathbf{P}}(s, s'') = \begin{cases} \tilde{\mathbf{P}}^l(s, s'') & \text{if } s'' = s' \\ \mathbf{P}^u(s, s'') & \text{otherwise} \end{cases}$
  Clearly, $\bar{\mathbf{P}}(s, \cdot)$ is a distribution, as $\tilde{\mathbf{P}}^l(s, s') + \mathbf{P}^u(s, S \setminus \{s'\}) = 1$.

- $\tilde{\mathbf{P}}^l(s, s') = \mathbf{P}^l(s, s') > 1 - \mathbf{P}^u(s, S \setminus \{s'\})$. Then:
  - If $\tilde{\mathbf{P}}^u(s, s'') \neq \mathbf{P}^u(s, s'')$ for some $s'' \neq s'$, then $\tilde{\mathbf{P}}^u(s, s'') = 1 - \mathbf{P}^l(s, S \setminus \{s''\})$ yields a valid distribution incorporating $\mathbf{P}^l(s, s')$.
  - Otherwise, $\tilde{\mathbf{P}}^l(s, s') = \mathbf{P}^l(s, s')$ and $\tilde{\mathbf{P}}^u(s, s'') = \mathbf{P}^u(s, s'')$ for all $s'' \neq s'$ implies $\tilde{\mathbf{P}}^l(s, s') > 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{s'\})$ and $\tilde{\mathbf{P}}^l(s, S) \leq 1$ as per definition of ADTMCs $\mathbf{P}^l(s, S) \leq 1$. Together:

    $$\tilde{\mathbf{P}}^l(s, S) \leq 1 < \tilde{\mathbf{P}}^u(s, S \setminus \{s'\}) + \tilde{\mathbf{P}}^l(s, s')$$

    Therefore, a valid probability distribution $\bar{\mathbf{P}}(s, \cdot)$ can be chosen as

    $$\bar{\mathbf{P}}(s, s'') = \begin{cases} \mathbf{P}^l(s, s'') & \text{if } s'' = s' \\ n_s \cdot \mathbf{P}^l(s, s'') + (1 - n_s) \cdot \mathbf{P}^u(s, s'') & \text{otherwise.} \end{cases}$$

    where normalization factor n_s has to be chosen such that $\bar{\mathbf{P}}(s, S) = 1$.

Hence, for any $s, s' \in S$ and $p \in [\tilde{\mathbf{P}}^l(s, s'), \tilde{\mathbf{P}}^u(s, s')]$, there exists some $\bar{\mathbf{P}} \in \mathbf{T}_{norm(\mathcal{M})}$ with $\bar{\mathbf{P}}(s, s') = p$ and thus $norm(\mathcal{M}) = (S, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, L)$ is delimited.

□

3.3 Correctness of Abstraction

This subsection studies the relationship between an ADTMC $\mathcal{M}$ and its abstraction $abstr(\mathcal{M}, \mathcal{A})$ for a given partitioning $\mathcal{A}$ of the state space of $\mathcal{M}$. The central notion for this relationship is a probabilistic variant of (forward) *simulation* on Kripke structures. Simulation relations are preorders on the state space requiring that whenever s' simulates s, denoted $s \preceq s'$, state s' can mimic all stepwise behaviour of s but in addition may also perform steps that cannot be matched by s. In the probabilistic setting, the target of a transition is a probability distribution, and thus the simulation relation needs to be lifted from states to distributions. This is done using so-called weight functions. Jonsson and Larsen's seminal notion of probabilistic simulation [29] is tailored to ADTMCs in the following way:

Definition 10 (Probabilistic simulation). *Let $\mathcal{M}$ be an ADTMC with state space S. $\mathcal{R} \subseteq S \times S$ is a* probabilistic simulation *iff $s\mathcal{R}s'$ implies:*

1. *$L(s', a) \neq ? \Rightarrow L(s', a) = L(s, a)$ for all $a \in AP$.*
2. *For all distributions $\mu \in \mathbf{T}(s, \cdot)$, there is a distribution $\mu' \in \mathbf{T}(s', \cdot)$ and a weight function $\Delta : S \times S \to [0, 1]$ with:*
   (a) *$\Delta(u, v) > 0 \Rightarrow u\mathcal{R}v$,*
   (b) *$\Delta(u, S) = \mu(u)$,*
   (c) *$\Delta(S, v) = \mu'(v)$.*

State s is simulated by s' (written $s \preceq s'$) if there exists a probabilistic simulation $\mathcal{R}$ with $(s, s') \in \mathcal{R}$.

The following result asserts that concrete states are simulated (in the above sense) by their abstract counterparts. More precisely:

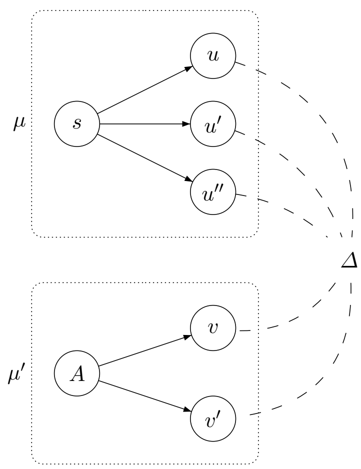
Theorem 1. *For ADTMC $\mathcal{M}$ with state space S, and ADTMC $abstr(\mathcal{A}, \mathcal{M})$:*

$$s \in A \Rightarrow s \preceq A \textit{ for all } s \in S, A \in \mathcal{A}.$$

Proof. For $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$ and abstraction $abstr(\mathcal{A}, \mathcal{M})$, we will show that $\mathcal{R} = \{(s, A) \mid s \in A, A \in \mathcal{A}\}$ is a probabilistic simulation relation. In the following we consider the union of $\mathcal{M}$ and $abstr(\mathcal{A}, \mathcal{M})$.
For condition 1 of Def. 10, we distinguish two cases:

- $L(s, a) = ?$: Follows directly from Def. 5.
- $L(s, a) \in \{\bot, \top\}$: By definition, for any $(s, A) \in \mathcal{R}$: $s \in S$ (since $A \subseteq S$ for all $A \in \mathcal{A}$). By Def. 4, $L(s, a) = \theta \Rightarrow L(A, a) = \theta$ for all $\theta \in \{\top, \bot\}$ and $a \in AP$.

In order to fulfill condition 2 of Def. 10, for every $(s, A) \in \mathcal{R}$ we have to find a distribution $\mu' \in \mathbf{T}(A, \cdot)$ for a given distribution $\mu \in \mathbf{T}(s, \cdot)$ and a suitable weight function Δ for μ and μ'.



For given μ, we define

$$\mu'(v) = \begin{cases} \sum_{u \in v} \mu(u) & \textit{if } v \in \mathcal{A} \\ 0 & \textit{otherwise} \end{cases} \tag{1}$$

and

$$\Delta(u, v) = \begin{cases} \mu(u) & \textit{if } (u, v) \in \mathcal{R} \\ 0 & \textit{otherwise} \end{cases} \tag{2}$$

The figure on the right illustrates the use of weight function Δ, namely *distributing* the distribution μ on S to a distribution μ' on $\mathcal{A}$.
μ' is a probability distribution, if μ is so:

$$\textstyle\sum_{v \in (\mathcal{A} \cup S)} \mu'(v) = \sum_{v \in \mathcal{A}} \mu'(v) = \sum_{v \in \mathcal{A}, u \in v} \mu(u) = \sum_{u \in S} \mu(u) = 1$$

μ' is a valid distribution for A: In the trivial case $v \in S$: $\mathbf{P}^l(A, v) = 0 = \mathbf{P}^u(A, v)$ and $\mu'(v) = 0$. For the case $v \in \mathcal{A}$, we show that $\mathbf{P}^l(A, v) \leq \mu'(v) \leq \mathbf{P}^u(A, v)$:

$$
\begin{aligned}
\mathbf{P}^l(A, v) &= \min_{s^l \in A} \mathbf{P}^l(s^l, v) && \text{(Def. 5)} \\
&\leq \mathbf{P}^l(s, v) \\
&= \textstyle\sum_{s' \in v} \mathbf{P}^l(s, s') && (s \in A) \\
&\leq \textstyle\sum_{s' \in v} \mu(s') = \mu'(v) && (\text{given } \mu, 1) \\
&\leq \textstyle\sum_{s' \in v} \mathbf{P}^u(s, s') \\
&= \mathbf{P}^u(s, v) \\
&\leq \max_{s^u \in A} \mathbf{P}^u(s^u, v) && (s \in A) \\
&= \mathbf{P}^u(A, v) && \text{(Def. 5)}
\end{aligned}
$$

Condition (2a) of Def. 10 is fulfilled trivially, since $\Delta(u, v) = 0$ if $(u, v) \notin \mathcal{R}$. For condition (2b):

$$
\begin{aligned}
\Delta(u, S) &= \textstyle\sum_{v \in S} \Delta(u, v) \\
&= \textstyle\sum_{v \in S} \Delta(u, v) + \sum_{A \in \mathcal{A}} \Delta(u, A) \\
&= \Delta(u, A_u) && (\text{for } A_u \in \mathcal{A}, u \in A_u) \\
&= \mu(u) && \text{(see 2)}
\end{aligned}
$$

Condition (2c) is fulfilled trivially for $v \in S$ since $\Delta(S, v) = 0 = \mu'(v)$. For $v \in \mathcal{A}$ it holds since:

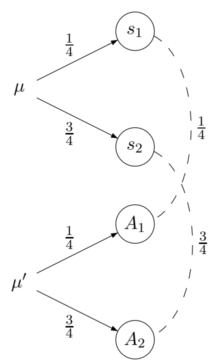
$$
\begin{aligned}
\Delta(S, v) &= \textstyle\sum_{u \in S} \Delta(u, v) \\
&= \textstyle\sum_{u \in v} \Delta(u, v) \\
&= \textstyle\sum_{u \in v} \mu(u) && \text{(see 2)} \\
&= \mu(v) && \text{(see 1)}
\end{aligned}
$$

All conditions are fulfilled by $\mathcal{R}$ and thus $s \preceq A$ for all $s \in A$, $A \in \mathcal{A}$.

□

Let us illustrate the notion of probabilistic simulation on a small example.

Example 5. Consider the DTMC in Fig. 2 (left), the partitioning leading to 2 (right) (see Ex. 2) with $\mathcal{R} = \{(s_0, A_0), (s_0', A_0), (s_1, A_1), (s_2, A_2)\}$. Note that A_i should be considered as a single abstract state. We have $s_0 \mathcal{R} A_0$ because condition 1 of Def. 10 is trivially fulfilled since $L(A_0, a) = ?$. For condition 2 we observe that in s_0 there is only one possible distribution $\mu = (0, 0, \frac{1}{4}, \frac{3}{4})$ to choose. The only distribution in $\mathbf{T}(A_0, \cdot)$, for which there is a weight function Δ fulfilling condition 2, is $\mu' = (0, \frac{1}{4}, \frac{3}{4})$ with $\Delta(s_1, A_1) = \frac{1}{4}$, $\Delta(s_2, A_2) = \frac{3}{4}$ and 0 otherwise (see figure). The conditions of Def. 10 can be checked for the remaining elements of $\mathcal{R}$ similarly.



3.4 Abstraction DTMCs versus CTMCs

To conclude this section, we study the precise relationship between our abstraction in the discrete-time and continuous-time setting. The theorem below states that the following diagram commutes, i.e., that abstraction of CTMCs can be regarded as a conservative extension of abstraction of DTMCs:

$$\begin{array}{ccccc} \mathcal{M} & \xrightarrow{abstr} & \mathcal{M}_{abstr} & \xrightarrow{norm} & \mathcal{M}_{del} \\ emb\downarrow & & & & \downarrow emb \\ \mathcal{N} & \xrightarrow{abstr} & \mathcal{N}_{abstr} & \xrightarrow{norm} & \mathcal{N}_{del} \end{array} \qquad \begin{array}{c} \text{(A)CTMCs} \\ \\ \text{(A)DTMCs} \end{array}$$

In order to make explicit to which type of abstract Markov chains abstraction and normalization is applied we use subscripts with the obvious interpretation.

Theorem 2. *For delimited ACTMC $\mathcal{M}$ and partitioning $\mathcal{A}$:*

$$emb(norm_{ACTMC}(abstr_{CTMC}(\mathcal{A}, \mathcal{M}))) = norm_{ADTMC}(abstr_{DTMC}(\mathcal{A}, emb(\mathcal{M})))$$

Proof. Let $\mathbf{P}^l_{\mathcal{M}}$ and $\mathbf{P}^u_{\mathcal{M}}$ be the lower and upper bound matrices of ACTMC $\mathcal{M}$. By definition, for the embedded Markov chain $\mathcal{N} = emb(\mathcal{M})$, $\mathbf{P}_{\mathcal{N}} = \mathbf{P}_{\mathcal{M}}$. Abstracting $\mathcal{M}$ and $\mathcal{N}$ respectively with respect to the partitioning $\mathcal{A}$ yields:

$$\mathcal{M}_{abstr} = (\mathcal{A}, \mathbf{P}^l_{\mathcal{M}}, \mathbf{P}^u_{\mathcal{M}}, L) \quad \text{and} \quad \mathcal{N}_{abstr} = (\mathcal{A}, \mathbf{P}^l_{\mathcal{N}}, \mathbf{P}^u_{\mathcal{N}}, L)$$

where $(*)$ $\mathbf{P}^l_{\mathcal{M}} = \mathbf{P}^l_{\mathcal{N}}$ and $\mathbf{P}^u_{\mathcal{M}} = \mathbf{P}^u_{\mathcal{N}}$ since for ACTMCs and ADTMCs, lower and upper bounds are calculated in the same way. Normalizing the resulting abstract Markov chains $\mathcal{M}_{abstr}$ and $\mathcal{N}_{abstr}$ yields:

$$\mathcal{M}_{del} = (\mathcal{A}, \mathbf{P}^l_{\mathcal{M}_{del}}, \mathbf{P}^u_{\mathcal{M}_{del}}, E_{unif}, L) \quad \text{and} \quad \mathcal{N}_{del} = (\mathcal{A}, \mathbf{P}^l_{\mathcal{N}_{del}}, \mathbf{P}^u_{\mathcal{N}_{del}}, L)$$

where:

$$\begin{aligned} (\mathbf{P}^l_{\mathcal{M}_{del}}, \mathbf{P}^u_{\mathcal{M}_{del}}) &= norm(\mathbf{P}^l_{\mathcal{M}_{abstr}}, \mathbf{P}^u_{\mathcal{M}_{abstr}}) \\ &\overset{(*)}{=} norm(\mathbf{P}^l_{\mathcal{N}_{abstr}}, \mathbf{P}^u_{\mathcal{N}_{abstr}}) \\ &= (\mathbf{P}^l_{\mathcal{N}_{del}}, \mathbf{P}^u_{\mathcal{N}_{del}}). \end{aligned}$$

Since the transition probability matrices of the embedded Markov chain $\mathcal{N}_{del}$ are the same as for $\mathcal{M}_{del}$ it follows $\mathbf{P}^l_{\mathcal{M}_{del}} = \mathbf{P}^l_{\mathcal{N}_{del}}$ and $\mathbf{P}^u_{\mathcal{M}_{del}} = \mathbf{P}^u_{\mathcal{N}_{del}}$. Hence, $emb(\mathcal{M}_{del}) = \mathcal{N}_{del}$.

□

4 Measures and Reachability

In the previous section, we have introduced and motivated the use of abstract Markov chains, i.e., chains with probability intervals. This section is now concerned with how to analyze such abstract Markov chains. First, we describe how the uncertainty induced by intervals is resolved by means of schedulers. In particular, we show that the set of schedulers on an abstract Markov chain coincides with those on its normalization. This is the correctness argument for normalization. A key result of this section is that so-called extreme schedulers (roughly speaking, they only select extrema of intervals) suffice for considering infima on measurable events. Finally, we detail the main procedures for model-checking of ADTMCs (and ACTMCs), viz. the computation of minimal (or dually, maximal) reachability probabilities in ADTMCs, and the time-bounded variant thereof for ACTMCs.

4.1 Schedulers

The probability to move in one step from s to s' in an ADTMC depends on the chosen probability distribution in s, i.e. on $\mu \in \mathbf{T}(s, \cdot) = \{\mathbf{P}(s, \cdot) \mid \mathbf{P} \in \mathbf{T}\}$. As for Markov decision processes, *schedulers* are used to resolve the nondeterminism. Schedulers come in many different flavors [40]. They may be memoryless (also called *simple* or *stationary*) or make their decisions depending on the history of the systems run (*history-dependent*). They may choose *deterministically* or in a *randomized* fashion, i.e. decide for some combination of several choices, and in the continuous-time setting, they may be aware of the *time* the system resided in each state.

Definition 11 (Schedulers). *Let $\mathcal{M}$ be an ADTMC with state space S, then:*

- $D : S \to \mathbf{T}_\mathcal{M}$ *is a* simple deterministic (SD) scheduler*,*
- $D : S \to distr(\mathbf{T}_\mathcal{M})$ *is a* simple randomized (SR) scheduler*,*
- $D : Paths^\mathcal{M} \to \mathbf{T}_\mathcal{M}$ *is a* history-dependent deterministic (HD) scheduler*,*
- $D : Paths^\mathcal{M} \to distr(\mathbf{T}_\mathcal{M})$ *is a* history-dependent randomized (HR) scheduler*.*

In a similar way, schedulers can be defined for ACTMCs and Markov decision processes (MDPs)[2]. For MDPs it has been shown that all of the above mentioned scheduler classes are equally expressive w.r.t. the infimum and supremum of reachability probabilities (see [7, 15]). In Section 4.4 we will show that this result translates to ADTMCs.

However, in the continuous-time setting scheduler classes for MDPs are much more diversified (see [3]). Though, for ACTMCs we can show that the set of HD-schedulers is as expressive as the set of HR-schedulers, i.e. each HR-scheduler can be simulated by a HD-scheduler. For simple schedulers a similar result follows directly.

Lemma 5. *For any ACTMC $\mathcal{M}$ and any HR-scheduler D, there is an HD-scheduler D' such that:*

$$D(\varsigma) = D'(\varsigma) \quad \textit{for all } \varsigma \in Paths^\mathcal{M}_{abs}.$$

Proof. The general idea is that a choice of an HR-scheduler is a convex combination of elements in a convex and dense set and thus is in this set itself. Per definition $D(\varsigma) \in distr(\mathbf{T}_\mathcal{M})$, i.e. $D(\varsigma) = \sum_{\mathbf{P} \in \mathbf{T}_\mathcal{M}} \alpha_\mathbf{P} \cdot \mathbf{P}$ with $\sum_{\mathbf{P} \in \mathbf{T}_\mathcal{M}} \alpha_\mathbf{P} = 1$. As for all $\mathbf{P} \in \mathbf{T}_\mathcal{M}$ and all $s \in S$, $\mathbf{P}(s, S) = 1$:

$$D(\varsigma) = \sum\nolimits_{\mathbf{P} \in \mathbf{T}_\mathcal{M}} \alpha_\mathbf{P} \cdot \mathbf{P}(s, S) = \sum\nolimits_{\mathbf{P} \in \mathbf{T}_\mathcal{M}} \alpha_\mathbf{P} = 1.$$

Furthermore, for all $\mathbf{P} \in \mathbf{T}_\mathcal{M}$ and all $s, s' \in S$, $\mathbf{P}^l(s, s') \le \mathbf{P}(s, s') \le \mathbf{P}^u(s, s')$. Choose some $\mathbf{P}_{min}, \mathbf{P}_{max} \in \mathbf{T}_\mathcal{M}$ such that $\mathbf{P}_{min}(s, s') \le \mathbf{P}(s, s') \le \mathbf{P}_{max}(s, s')$ for all $\mathbf{P} \in \mathbf{T}_\mathcal{M}$. Then, obviously for all $s, s' \in S$:

$$\mathbf{P}^l(s, s') \le \mathbf{P}_{min}(s, s') \le D(\varsigma) = \sum\nolimits_{\mathbf{P} \in \mathbf{T}_\mathcal{M}} \alpha_\mathbf{P} \cdot \mathbf{P}(s, S) \le \mathbf{P}_{max}(s, s') \le \mathbf{P}^u(s, s')$$

Altogether it follows that $D(\varsigma) \in \mathbf{T}_\mathcal{M}$ and therefore, for all $\varsigma \in Paths_\mathcal{M}$ there is an HD-scheduler D' with $D'(\varsigma) = D(\varsigma)$.

□

[2] In the continuous-time setting, a time-abstract HD-scheduler takes time-abstract paths in ACTMC $\mathcal{M}$ as argument.

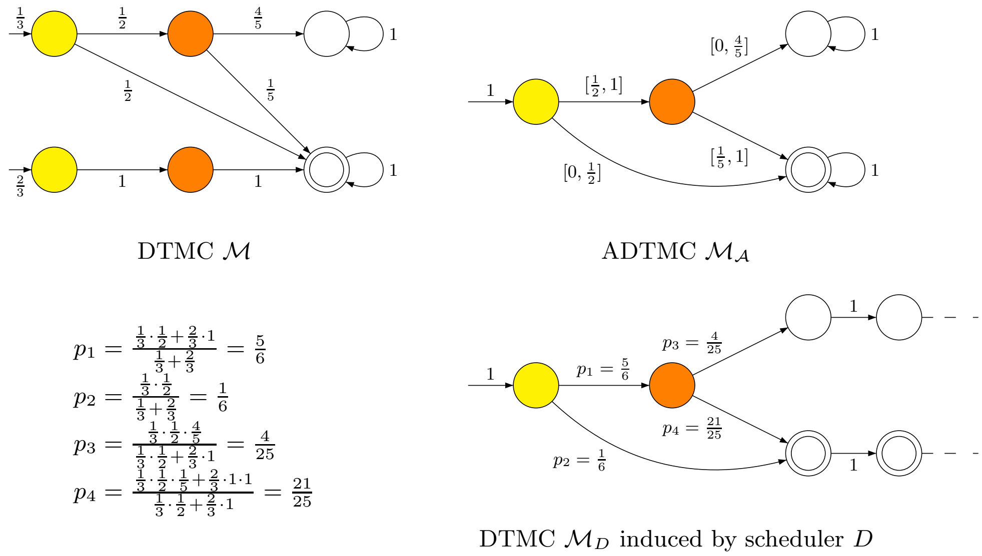


Fig. 5. HD-scheduler mimicking the original behavior

Obviously, this result carries over to the time-abstract setting of ADTMCs.

Let us now focus on (time-abstract) history-dependent schedulers that, given a time-abstract path, deterministically select a transition probability function from the set $\mathbf{T}$. The set of history-dependent schedulers for ADTMC $\mathcal{M}$ is denoted by $\mathit{Sched}^{\mathcal{M}}$.

The function of a scheduler is to determinize the behavior of an ADTMC. This behavior can be described by means of a DTMC as stated in the following lemma.

Definition 12 (Induced DTMC). *For ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$, scheduler $D \in \mathit{Sched}^{\mathcal{M}}$ induces the DTMC $\mathcal{M}_D = (S_D, \mathbf{P}_D, L_D)$ where*[3]

- $S_D = \mathit{Paths}_{abs}(\mathcal{M})$,
- $\mathbf{P}_D(\varsigma, \varsigma') = \begin{cases} D(\varsigma)(\mathit{last}(\varsigma), s) & \text{if } \varsigma' = \varsigma \to s \\ 0 & \text{otherwise} \end{cases}$ *and*
- $L_D(\varsigma, a) = L(\mathit{last}(\varsigma), a)$ *for* $\varsigma \in S_D$, $a \in AP$.

Let us now study how abstractions are simulating their concrete counterparts. Consider a DTMC $\mathcal{M}$ and partitioning $\mathcal{A}$ inducing $\mathcal{M}_{\mathcal{A}} = \mathit{abstr}(\mathcal{A}, \mathcal{M})$. For a fixed initial distribution α on $\mathcal{M}$, one can derive a scheduler D on $\mathcal{M}$ such that reachability probabilities for paths in $\mathcal{M}_{\mathcal{A}}$ and the *corresponding* ones in $\mathcal{M}$ coincide. This can be done by calculating the conditional probabilities for each transition in the induced DTMC, i.e. the scheduling decisions of D, with respect to $\mathcal{M}$ and the initial distribution.

Example 6. Consider DTMC $\mathcal{M}$ in Fig. 5. The partitioning that merges all states with the same gray shading yields abstraction $\mathcal{M}_{\mathcal{A}}$. If we are interested in a scheduler that exactly mimics the behavior of $\mathcal{M}$, we have to compute the conditional

[3] In the continuous-time setting, the exit rate of the induced CTMC is carried over from the given ACTMC.

probabilities $p_1, \ldots, p_4$ as shown in the figure. E.g. p_1, the probability to move from the leftmost state to the middle one in $\mathcal{M}_D$, is calculated by summing up the probabilities to reach the two states in the middle from the initial states in $\mathcal{M}$ and dividing it by the probability to get into one of the corresponding predecessors, here in one of the states on the left in $\mathcal{M}$.

As we have formal means now concider nondeterminism in ADTMCs, we are in a position to show that normalization is preserving all the behavior.

Lemma 6. *For any ADTMC $\mathcal{M}$, $\mathit{Sched}^{\mathcal{M}} = \mathit{Sched}^{norm(\mathcal{M})}$.*

Proof. It holds that[4]

$$\mathit{Sched}^{\mathcal{M}} = \mathit{Sched}^{norm(\mathcal{M})} \iff \mathit{Paths}(\mathcal{M}) = \mathit{Paths}(norm(\mathcal{M})).$$

We prove the second statement.

"$\subseteq$" Proof by contradiction: assume $\mathit{Paths}(\mathcal{M}) \supset \mathit{Paths}(norm(\mathcal{M}))$, then there is a path $\varsigma \in \mathit{Paths}(\mathcal{M})$ with $\varsigma \notin \mathit{Paths}(norm(\mathcal{M}))$. For at least one position $0 \leq i < |\varsigma|$ in ς with $\varsigma[i] = s_i, \varsigma[i+1] = s_{i+1}$, there is $\mathbf{P}_{\mathcal{M}} \in \mathbf{T}_{\mathcal{M}}$ with $p = \mathbf{P}_{\mathcal{M}}(s_i, s_{i+1})$ such that $p \notin [\mathbf{P}^l_{norm(\mathcal{M})}(s_i, s_{i+1}), \mathbf{P}^u_{norm(\mathcal{M})}(s_i, s_{i+1})]$. This can only be the case, if *norm* removed p from the interval, i.e.

- $p > 1 - \mathbf{P}^l_{\mathcal{M}}(s_i, S \setminus \{s_{i+1}\}) \Rightarrow p^l := \mathbf{P}^l_{\mathcal{M}}(s_i, S \setminus \{s_{i+1}\}) + p > 1$ or
- $p < 1 - \mathbf{P}^u_{\mathcal{M}}(s_i, S \setminus \{s_{i+1}\}) \Rightarrow p^u := \mathbf{P}^u_{\mathcal{M}}(s_i, S \setminus \{s_{i+1}\}) + p < 1$.

Since p^l is the lowest and p^u is the largest possible sum for any $\mathbf{P}_{norm(\mathcal{M})}(s, \cdot)$ and $p^u < 1 < p^l$, there is no valid distribution incorporating p. This contradicts $\mathbf{P}_{\mathcal{M}} \in \mathbf{T}_{\mathcal{M}}$. Thus $\mathit{Paths}(\mathcal{M}) \subseteq \mathit{Paths}(norm(\mathcal{M}))$.

"$\supseteq$" At every position $0 \leq i < |\varsigma|$ in each $\varsigma \in \mathit{Paths}(norm(\mathcal{M}))$,

$$p = \mathbf{P}_{norm(\mathcal{M})}(\varsigma[i], \varsigma[i+1]) \in [\mathbf{P}^l_{\mathcal{M}}(\varsigma[i], \varsigma[i+1]), \mathbf{P}^u_{\mathcal{M}}(\varsigma[i], \varsigma[i+1])]$$

for every $\mathbf{P}_{\mathcal{M}} \in \mathbf{T}_{\mathcal{M}}$, as

$$[\mathbf{P}^l_{\mathcal{M}}(s, s'), \mathbf{P}^u_{\mathcal{M}}(s, s')] \supseteq [\mathbf{P}^l_{norm(\mathcal{M})}(s', s'), \mathbf{P}^u_{norm(\mathcal{M})}(s, s')]$$

for all $s, s' \in S$. Thus, $\varsigma \in \mathit{Paths}(\mathcal{M})$.

□

Summarizing, the *norm* function introduced in Def. 9 is efficient due to Lemma 2 (at most $|S|$ bounds have to be adjusted) and Lemma 3 (it is idempotent). Applying it to an ADTMC results in a delimited ADTMC (Lemma 4) for which the set of schedulers is the same (Lemma 6).

4.2 Measure Spaces on Paths

A nonempty set Ω of possible outcomes of an experiment of chance is called a *sample space*. A set $\mathcal{B} \subseteq 2^{\Omega}$ is called *Borel field* (or σ-algebra) over Ω if it contains Ω, $\Omega \backslash E$ for each $E \in \mathcal{B}$ and the union of any countable sequence of sets from $\mathcal{B}$. The subsets of Ω that are elements of $\mathcal{B}$ are called *measurable* with regard to $\mathcal{B}$.

[4] In the continuous-time setting, only time-abstract paths have to be considered.

A Borel field $\mathcal{B}$ is *generated* by an at most countable set $\mathcal{E}$, denoted by $\mathcal{B} = \langle \mathcal{E} \rangle$, if $\mathcal{B}$ is the closure of the elements of $\mathcal{E}$ under complement and countable union.

A *probability space* is a triple $\mathcal{PS} = (\Omega, \mathcal{B}, Pr)$, where Ω is a sample space, $\mathcal{B}$ is a Borel field over Ω, and Pr is a mapping $\mathcal{B} \to [0,1]$ such that $Pr(\Omega) = 1$ and $Pr(\bigcup_{i=1}^{\infty} E_i) = \sum_{i=1}^{\infty} Pr(E_i)$ for any sequence $E_1\, E_2\, \ldots$ of pairwise disjoint sets of $\mathcal{B}$. We call Pr a *probability measure*.

For an ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$ with initial distribution α, we let Ω be the set $Paths^{\mathcal{M}}$ and the Borel field $\mathcal{B}$ is generated by basic cylinder sets $\mathcal{C}(s_0\, \ldots\, s_k)$, where $\mathcal{C}(s_0\, \ldots\, s_k)$ is the set of all paths of $\mathcal{M}$ with prefix $s_0\, \ldots\, s_k$. Recall that $Sched^{\mathcal{M}}$ is the set of all HD-schedulers for $\mathcal{M}$. A scheduler $D \in Sched^{\mathcal{M}}$ induces a probability space $\mathcal{PS}^D = (\Omega, \mathcal{B}, Pr^D)$, where Pr^D is uniquely given by $Pr^D(\Omega) = 1$ and[5]

$$Pr^D(\mathcal{C}(s_0\, \ldots\, s_k)) = \alpha(s_0) \cdot \prod_{j=0}^{k-1} \mathbf{P}^j(s_j, s_{j+1})$$

where $\mathbf{P}^j$ is the probability matrix selected by scheduler D given history $s_0\, \ldots\, s_j$.

For DTMCs, the probability measure follows directly from the definition for ADTMCs:

$$Pr(\mathcal{C}(s_0\, \ldots\, s_k)) = \alpha(s_0) \cdot \prod_{j=0}^{k-1} \mathbf{P}(s_j, s_{j+1}).$$

For an ACTMC, the probability space and probability measure (w.r.t. a scheduler) are defined similarly as for ADTMCs. Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E_{unif}, L)$ be an ACTMC. We set Ω as the set $Paths^{\mathcal{M}}$ of all timed paths of $\mathcal{M}$ given an initial distribution α. The Borel field $\mathcal{B}$ is generated by basic cylinder sets $\mathcal{C}(s_0\, I_0\, \ldots\, I_{k-1}\, s_k)$ where for $i \in \{0, 1, \ldots, k-1\}$ $I_i = (0, x_i]$, $x_i \in \mathbb{R}_{>0}$ and

$$\mathcal{C}(s_0\, I_0\, \ldots\, I_{k-1}\, s_k) = \{u_0\, t_0\, u_1\, t_1 \ldots \mid u_i = s_i, t_i \in I_i \text{ for } 0 \leq i < k, u_k = s_k\}$$

for $s_0, s_1, \ldots, s_k \in S$. A HD scheduler $D \in Sched^{\mathcal{M}}$ induces a probability space $\mathcal{PS}^D = (\Omega, \mathcal{B}, Pr^D)$, where Pr^D is uniquely given by $Pr^D(\Omega) = 1$ and

$$Pr^D(\mathcal{C}(s_0\, I_0\, \ldots\, I_{k-1}\, s_k)) = \alpha(s_0) \cdot \prod_{j=0}^{k-1} (\mathbf{P}^j(s_j, s_{j+1}) \cdot X(I_j))$$

where $\mathbf{P}^j$ is the probability matrix selected by scheduler D for (time-abstract) history $s_0\, \ldots\, s_j$ and

$$X(I) = \int_{t \in I} E_{unif} \cdot e^{-E_{unif}\, t} dt = 1 - e^{-E_{unif}\, x}$$

is the probability for leaving any state within time interval $I = (0, x]$.

For CTMCs, the probability measure follows directly:

$$Pr(\mathcal{C}(s_0\, I_0\, \ldots\, I_{k-1}\, s_k)) = \alpha(s_0) \cdot \prod_{j=0}^{k-1} (\mathbf{P}(s_j, s_{j+1}) \cdot X(I_j))$$

[5] Clearly, Pr^D depends on α but we omit α to improve readability.

4.3 Minimal Probabilities of Measurable Events

In this section we consider the infimum of probabilities of measurable (sets of) events with regard to HD-schedulers. When defining PCTL and CSL semantics on abstract Markov chains in Section 5 these infima are of major interest. We show that instead of regarding HD-schedulers, it suffices to consider only *extreme* ones. Intuitively extreme schedulers only pick *extreme distributions* that result from a one by one minimisation/maximisation of transition probabilities. Note that different priorities for minimising/maximising yield different minimal/maximal probabilities. E.g. in Figure 5 for the leftmost state in the ADTMC there are two extreme distributions: $(\ldots, \frac{1}{2}, \frac{1}{2}, \ldots)$ and $(\ldots, 1, 0, \ldots)$.

Definition 13 (Extreme distributions). *Let $s \in S$ and $S' \subseteq S$. We define $extr(\mathbf{P}^l, \mathbf{P}^u, S', s) \subseteq \mathbf{T}$ such that $\mu \in extr(\mathbf{P}^l, \mathbf{P}^u, S', s)$ iff either $S' = \emptyset$ and $\mu = \mathbf{P}^l(s, \cdot) = \mathbf{P}^u(s, \cdot)$ or one of the following conditions hold*[6]*:*

$$\exists s' \in S' : \mu(s') = \mathbf{P}^l(s, s') \text{ and } \mu \in extr(\mathbf{P}^l, \mathbf{P}^u[s' \mapsto \mu(s')], S' \setminus \{s'\}, s) \text{ or}$$
$$\exists s' \in S' : \mu(s') = \mathbf{P}^u(s, s') \text{ and } \mu \in extr(\mathbf{P}^l[s' \mapsto \mu(s')], \mathbf{P}^u, S' \setminus \{s'\}, s)$$

We call $\mu \in \mathbf{T}(s, \cdot)$ an extreme distribution *if $\mu \in extr(\mathbf{P}^l, \mathbf{P}^u, S, s)$.*

The number of extreme distributions grows exponentially in the size of the state space. However, in general there are *infinitely* many distributions leading from one state to another in an ADTMC, whereas there are only *finitely* many extreme distributions.

A scheduler that only chooses extreme distributions is an *extreme scheduler*. E.g. scheduler D in Fig. 5 is not extreme. Sets of schedulers restricted to extreme ones are subscripted with *extr*. In the remainder of this section we show that for ADTMCs as well as for ACTMCs, HD-schedulers and extreme HD-schedulers yield the same infimum for probability measures of measurable sets. This result reduces the number of schedulers we have to deal with considerably.

Theorem 3 (Extrema). *Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$ be an ADTMC. For every measurable set Q of the induced probability space:*

$$\inf_{D \in Sched^{\mathcal{M}}_{extr}} Pr^D(Q) = \inf_{D \in Sched^{\mathcal{M}}} Pr^D(Q).$$

Proof. Let α be an initial distribution on ADTMC $\mathcal{M}$, and assume $\mathcal{M}$ to be delimited. We show the claim by induction over the structure of the Borel field of the induced probability space on $\mathcal{M}$. In particular, we show the following facts:

1.) For all basic cylinder sets $\mathcal{C}(s_0 \ldots s_k)$ and all $D \in Sched^{\mathcal{M}}$ there exists $E \in Sched^{\mathcal{M}}_{extr}$ such that

$$Pr^D(\mathcal{C}(s_0 \ldots s_k)) \geq Pr^E(\mathcal{C}(s_0 \ldots s_k)).$$

2.) Given two basic cylinder sets $C := \mathcal{C}(s_0 \ldots s_k)$, $C' := \mathcal{C}(s'_0 \ldots s'_l)$ (wlog. $k \leq l$) and $E, E' \in Sched^{\mathcal{M}}_{extr}$ such that $Pr^E(C)$ and $Pr^{E'}(C')$ are minimal, there exists $\hat{E} \in Sched^{\mathcal{M}}_{extr}$ such that for all $D \in Sched^{\mathcal{M}}$

$$Pr^D(C \cup C') \geq Pr^{\hat{E}}(C \cup C').$$

[6] Here, $f[s \mapsto x]$ denotes the function that agrees everywhere with f except at s where it is equal to x.

3.) Given basic cylinder set C and extreme scheduler E minimizing $Pr^E(C)$, we show that there exists $\hat{E} \in \mathit{Sched}^{\mathcal{M}}_{extr}$ for all $D \in \mathit{Sched}^{\mathcal{M}}$ with

$$Pr^D(\Omega \setminus C) \geq Pr^{\hat{E}}(\Omega \setminus C).$$

The first step is the induction basis and 2./3. are induction steps concerning union and complement.

1.) Let $D \in \mathit{Sched}^{\mathcal{M}}$. It holds that

$$Pr^D(\mathcal{C}(s_0 \, \ldots \, s_k)) = \alpha(s_0) \cdot \mathbf{P}^D_0(s_0, s_1) \cdot \ldots \cdot \mathbf{P}^D_{k-1}(s_{k-1}, s_k)$$

where $\mathbf{P}^D_i$ is the choice of D with history $s_0, s_1 \ldots, s_i$, $0 \leq i < k$. Obviously, the product is minimal for D, if each factor $\mathbf{P}_i(s_i, s_{i+1})$ is. We construct E as a scheduler on $\mathcal{M}$ with $E(s_0 \, \ldots \, s_i) = \mathbf{P}^E_i$ such that $\mathbf{P}^E_i(s_i, s_{i+1}) := \mathbf{P}^l(s_i, s_{i+1})$ and $\mathbf{P}^E_i(s_i, \cdot)$ results in an extreme distribution. Then E is extreme and for all $D \in \mathit{Sched}^{\mathcal{M}}$

$$Pr^D(\mathcal{C}(s_0 \, \ldots \, s_k)) \geq Pr^E(\mathcal{C}(s_0 \, \ldots \, s_k)).$$

2.) We distinguish two cases, namely $C \cap C' \neq \emptyset$ and $C \cap C' = \emptyset$.
If $C \cap C' \neq \emptyset$, then $C = \mathcal{C}(s_0 \, \ldots \, s_k)$ is a prefix of $C' = \mathcal{C}(s'_0 \, \ldots \, s'_l)$ since $k \leq l$. Therefore, we let $\hat{E} := E$. Since E is extreme, $\hat{E}$ is so and from induction hypothesis we have that for all $D \in \mathit{Sched}^{\mathcal{M}}$

$$Pr^{\hat{E}}(C \cup C') = Pr^{\hat{E}}(C) = Pr^E(C) \leq Pr^D(C) = Pr^D(C \cup C').$$

If $C \cap C' = \emptyset$, we have that either there exists $j \in \{0, 1, \ldots, k-1\}$ with $s_0 = s'_0$, $s_1 = s'_1$, $\ldots$, $s_j = s'_j$, $s_{j+1} = s'_{j+1}$ or $s_0 \neq s'_0$. In the latter case, we let $\hat{E}(s_0 \, \ldots \, s_n) := E(s_0 \, \ldots \, s_n)$, $n \in \{0, 1, \ldots, k-1\}$ and $\hat{E}(s'_0 \, \ldots \, s'_m) := E'(s'_0 \, \ldots \, s'_m)$, $m \in \{0, 1, \ldots, l-1\}$. All other choices of $\hat{E}$ are arbitrary but extreme. Obviously, $\hat{E}$ is an extreme scheduler and for all $D \in \mathit{Sched}^{\mathcal{M}}$

$$\begin{aligned} Pr^{\hat{E}}(C \cup C') &= Pr^{\hat{E}}(C) + Pr^{\hat{E}}(C') = Pr^E(C) + Pr^{E'}(C') \\ &\leq Pr^D(C) + Pr^D(C') = Pr^D(C \cup C'). \end{aligned}$$

In the former case (i.e. j exists), we calculate

$$\min_{\mu \in \mathbf{T}(s_j, \cdot)} \{\alpha(s_0) \cdot \mu(s_{j+1}) \cdot \prod_{i \neq j} P^D_i(s_i, s_{i+1}) + \alpha(s'_0) \cdot \mu(s'_{j+1}) \cdot \prod_{i \neq j} P^D_i(s'_i, s'_{i+1})\}$$

$$= \alpha(s_0) \cdot \prod_{i<j} P^D_i(s_i, s_{i+1}) \cdot \min_{\mu \in \mathbf{T}(s_j, \cdot)} \{\mu(s_{j+1}) \cdot p + \mu(s'_{j+1}) \cdot p'\}$$

where $p := \prod_{i=j+1}^{k} P^D_i(s_i, s_{i+1})$ and $p' := \prod_{i=j+1}^{l} P^D_i(s'_i, s'_{i+1})$. For the same reason as in case of $C \cap C' \neq \emptyset$ for the first $j-1$ choices we let $\hat{E}$ choose as E or alternatively as E' since

$$Pr^E(C(s_0, \ldots, s_j)) = Pr^{E'}(C(s_0, \ldots, s_j)) = Pr^{E'}(C(s'_0, \ldots, s'_j)).$$

Similarly, as in case $s_0 \neq s'_0$ for $1 \leq m \leq k-(j+1)$, $1 \leq n \leq l-(j+1)$ we let

$$\begin{aligned} \hat{E}(s_0 \, \ldots \, s_j \, s_{j+1} \, \ldots \, s_{j+m}) &:= E(s_0 \, \ldots \, s_j \, s_{j+1} \, \ldots \, s_{j+m}) \\ \hat{E}(s_0 \, \ldots \, s_j \, s'_{j+1} \, \ldots \, s'_{j+n}) &:= E'(s_0 \, \ldots \, s_j \, s'_{j+1} \, \ldots \, s'_{j+n}). \end{aligned}$$

For history $s_0 \ldots s_j$ first observe that p and p' (as defined above but for $D := \hat{E}$) are constant and therefore we choose $\hat{E}(s_0 \ldots s_j) = \mathbf{P}^{\hat{E}}$ such that $\mathbf{P}^{\hat{E}}(s_j, \cdot) =: \eta \in \mathbf{T}(s_j, \cdot)$ fulfills

$$\min_{\mu \in \mathbf{T}(s_j,\cdot)} \{\mu(s_{j+1}) \cdot p + \mu(s'_{j+1}) \cdot p'\} = \eta(s_{j+1}) \cdot p + \eta(s'_{j+1}) \cdot p'.$$

Assume that $p \geq p'$. The proof goes along the same lines if $p < p'$. We choose $\eta \in \mathbf{T}(s_j, \cdot)$ such that it is extreme, $\eta(s_{j+1}) = \mathbf{P}^l(s_j, s_{j+1})$ and $\eta(s'_{j+1})$ is as small as possible. Therefore according to Definition 13

$$\eta(s'_{j+1}) = \max\{\mathbf{P}^l(s_j, s'_{j+1}), 1 - \mathbf{P}^u(s_j, S') - \mathbf{P}^l(s_j, s_{j+1})\}$$

where $S' := S \setminus \{s_{j+1}, s'_{j+1}\}$. We claim that for all $\mu \in \mathbf{T}(s_j, \cdot)$

$$\mu(s_{j+1}) \cdot p + \mu(s'_{j+1}) \cdot p' \leq \eta(s_{j+1}) \cdot p + \eta(s'_{j+1}) \cdot p' \tag{3}$$

We distinguish two cases. Assume that $\eta(s'_{j+1}) = \mathbf{P}^l(s_j, s'_{j+1})$. Then Equation (3) holds trivially. If, however, $\eta(s'_{j+1}) = 1 - q - \mathbf{P}^l(s_j, s_{j+1})$ where $q := \mathbf{P}^u(s_j, S')$ we prove Equation (3) by calculating

$$\begin{aligned}
\eta(s_{j+1}) \cdot p + \eta(s'_{j+1}) \cdot p' &= \mathbf{P}^l(s_j, s_{j+1}) \cdot p + (1 - q - \mathbf{P}^l(s_j, s_{j+1})) \cdot p' \\
&= \mathbf{P}^l(s_j, s_{j+1}) \cdot p + p' - \mathbf{P}^l(s_j, s_{j+1}) \cdot p' - q \cdot p' \\
&= \mathbf{P}^l(s_j, s_{j+1}) \cdot (p - p') + p' - \mathbf{P}^u(s_j, S') \cdot p' \\
&\leq \mu(s_{j+1}) \cdot (p - p') + p' - \textstyle\sum_{s \in S'} \mu(s) \cdot p' \\
&= \mu(s_{j+1}) \cdot p - \big(1 - \mu(s_{j+1}) - \textstyle\sum_{s \in S'} \mu(s)\big) \cdot p' \\
&= \mu(s_{j+1}) \cdot p - \mu(s'_{j+1}) \cdot p'.
\end{aligned}$$

Note that we used the fact that $p - p' \geq 0$. Thus, with $\mathbf{P}_j^{\hat{E}}(s_j, \cdot) = \eta$ we get

$$\begin{aligned}
&Pr^{\hat{E}}(C \cup C') \\
&= \alpha(s_0) \cdot \prod_{i<j} P_i^{\hat{E}}(s_i, s_{i+1}) \big(\eta(s_{j+1}) \cdot p + \eta(s'_{j+1}) \cdot p'\big) \\
&= \alpha(s_0) \cdot \eta(s_{j+1}) \cdot \prod_{i \neq j} P_i^{\hat{E}}(s_i, s_{i+1}) + \alpha(s'_0) \cdot \eta(s'_{j+1}) \cdot \prod_{i \neq j} P_i^{\hat{E}}(s'_i, s'_{i+1}) \\
&\leq \alpha(s_0) \cdot \mu(s_{j+1}) \cdot \prod_{i \neq j} P_i^{D}(s_i, s_{i+1}) + \alpha(s'_0) \cdot \mu(s'_{j+1}) \cdot \prod_{i \neq j} P_i^{D}(s'_i, s'_{i+1}) \\
&= Pr^D(C \cup C')
\end{aligned}$$

for all $D \in \mathit{Sched}^{\mathcal{M}}$ with $\mu = \mathbf{P}_j^D(s_j, \cdot)$. Since η is an extreme distribution and E and E' are extreme schedulers, $\hat{E}$ is so which means that the second induction statement holds. The proof above can be lifted to unions of a countable number of basic cylinder sets in the obvious way.

3.) A scheduler minimizing the measure of the complement of cylinder set C can be derived by construction of a scheduler maximizing the measure of C. Such a scheduler can be constructed analogously to a scheduler minimizing the measure of C. Since the complete probability space Ω has measure 1 and $C \subseteq \Omega$:

$$Pr^D(C^c) = Pr^D(\Omega \setminus C) = Pr^D(\Omega) - Pr^D(C) = 1 - Pr^D(C)$$

This yields:

$$\begin{aligned}\inf_{D\in Sched_{extr}(\mathcal{M})} Pr^D(C^c) &= \inf_{D\in Sched_{extr}(\mathcal{M})}(1 - Pr^D(C)) \\ &= 1 - \sup_{D\in Sched_{extr}(\mathcal{M})} Pr^D(C).\end{aligned}$$

As countable union and complement of cylinder sets are measurable, so are arbitrary measurable sets.

□

Theorem 4 (Extrema). *Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E_{unif}, L)$ be an ACTMC. For every measurable set Q of the induced probability space:*

$$\inf_{D\in Sched^{\mathcal{M}}_{extr}} Pr^D(Q) = \inf_{D\in Sched^{\mathcal{M}}} Pr^D(Q).$$

Proof. The proof is similar to the proof of Theorem 3. Thus, in the following we only highlight the differences.

Induction basis: The cylinder sets are timed now, thus we seek for scheduler D with minimal

$$Pr^D(\mathcal{C}(s_0\, I_0\, \ldots\, I_{k-1}\, s_k)) = (\mathbf{P}_0(s_0, s_1)\cdot X(I_0))\cdot \ldots \cdot (\mathbf{P}_{k-1}(s_{k-1}, s_k)\cdot X(I_{k-1}))$$

The probability to leave any state within time interval I_i is given by $X(I_i)$ since $\mathcal{M}$ has a uniform exit rate. This leaves minimizing the transition probabilities of $\mathbf{P}_i$ as in the discrete-time setting.

Induction step: We show how to construct an extreme scheduler for the union and complement of basic cylinder sets for which *minimal* extreme schedulers are known. Consider timed cylinder sets $C = \mathcal{C}(s_0\, I_0\, \ldots\, s_k)$ and $C' = \mathcal{C}(s'_0, I'_0\, \ldots\, s'_l)$ where wlog. $k \le l$. Again, disjoint and non-disjoint union is treated seperately.

- In the non-disjoint case additional timing information makes no difference for the proof.
- In the disjoint case, $\mathbf{P} \in \mathbf{T}$ has to be determined with minimal:

  $$\begin{aligned}&Pr^D(\mathcal{C}(s_{i-1}\, I_{i-1}\, \ldots\, s_k)) + Pr^D(\mathcal{C}(s'_{i-1}\, I_{i-1} \ldots\, s'_l)) \\ &= (\mathbf{P}(s_{i-1}, s_i)\cdot X(I_{i-1}))\cdot \ldots \cdot (\mathbf{P}(s_{k-1}, s_k)\cdot X(I_{k-1})) \\ &\quad +(\mathbf{P}(s_{i-1}, s'_i)\cdot X(I_{i-1}))\cdot \ldots \cdot (\mathbf{P}(s'_{l-1}, s'_l)\cdot X(I'_{l-1}))\end{aligned}$$

  Since $X(I_{i-1})\ldots$ are not affected by the scheduler, minimization reduces to the minimization of

  $$\begin{aligned}&\mathbf{P}(s_{i-1}, s_i)\cdot \ldots \cdot \mathbf{P}(s_{k-1}, s_k) + \mathbf{P}(s_{i-1}, s'_i)\cdot \ldots \cdot \mathbf{P}(s'_{l-1}, s'_l) \\ &= \mathbf{P}(s_{i-1}, s_i)\cdot p + \mathbf{P}(s_{i-1}, s'_i)\cdot p'\end{aligned}$$

  where no timing information is involved anymore.

The rest of the proof is as in the discrete-time setting.

□

To summarize, the results in this section show that for determining minimal probabilities over abstract Markov chains, it suffices to consider (finitely many) extreme schedulers. The following subsections focus on the main measurable events: reachability and time-bounded reachability. Algorithms for these events provide the basic building bricks for the model checking procedure for PCTL and CSL described in Section 5.

4.4 Unbounded Reachability

The *reachability* problem is about reaching goal states from an initial one via an arbitrary number of transitions. In Markov chains, the probability to reach a goal state is of special interest. Solving reachability problems will be the major task in our model checking algorithm for until formulas presented in Section 5.

Formally, for an ADTMC $\mathcal{M}$, $s \in S$, $B \subseteq S$ and $n \in \mathbb{N}$, define the events:

$$Reach_n(s, B) := \{\varsigma \in Paths^{\mathcal{M}} \mid \varsigma[0] = s, \varsigma[n] \in B \text{ and for all } k < n, \varsigma[k] \notin B\}$$

$$Reach_{\leq n}(s, B) := \bigcup_{i=0}^{n} Reach_i(s, B)$$

$$Reach(s, B) := \bigcup_{n \geq 0} Reach_n(s, B) = \bigcup_{n \geq 0} Reach_{\leq n}(s, B).$$

Let us now compare reachability in two ADTMCs w.r.t. abstraction, i.e. simulation. We first give the intuition of the following theorem. Consider a finite path $s_0 \to \ldots \to s_n$ in an ADTMC. The probability to take such a path is determined by the transition probability bounds for $s_i \to s_{i+1}$ for all $0 \leq i < n$. Let the path in an abstraction of the ADTMC correponding to the given finite path be $s'_0 \to \ldots \to s'_n$ where $s_i \preceq s'_i$ for all $0 \leq i \leq n$. Again, the probability to take such a path is determined by the transition probability bounds, this time for $s'_i \to s'_{i+1}$ for all $0 \leq i < n$. As due to $s_i \preceq s'_i$ there is a mapping from any probability distribution that can be chosen in a state of the original ADTMC to one in the simulating state of the abstraction, the infimum of the path probabilities may not be less than in the abstraction. This basic idea can be lifted to sets of paths (cf. Example 5).

Theorem 5 (Reachability simulation). *Let ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$, $s, s' \in S$ with $s \preceq s'$, sets of goal states $B, B' \subseteq S$ where for all $\bar{s}' \in B'$, $\bar{s} \preceq \bar{s}' \Rightarrow \bar{s} \in B$ and there is no $\bar{s} \in B$ with $\bar{s} \preceq \bar{s}'$ and $\bar{s}' \notin B'$. Then, for all schedulers $D \in Sched^{\mathcal{M}}$ there exists a scheduler $D' \in Sched^{\mathcal{M}}$ with*

$$Pr^D(Reach(s, B)) = Pr^{D'}(Reach(s', B')).$$

Proof. We prove the claim by induction over the maximal path length i from s to states in B. That is, for all $s, s' \in S$ with $s \preceq s'$, for every HD scheduler D there is an HD scheduler D' such that:

$$Pr^D(Reach_{\leq i}(s, B)) = Pr^{D'}(Reach_{\leq i}(s', B')).$$

By induction on i:

- $i = 0$: $Pr^D(Reach_{\leq 0}(s, B))$ is given by the the indicator function

$$\mathsf{i}_B(s) = \begin{cases} 1 & \text{if } s \in B \\ 0 & \text{otherwise.} \end{cases}$$

  $\mathsf{i}_B(s) = \mathsf{i}_{B'}(s')$ follows from the restrictions on B and B' as
  - $\mathsf{i}_B(s) \geq \mathsf{i}_{B'}(s')$ since $s' \in B' \Rightarrow s \in B$ and
  - $\mathsf{i}_B(s) \leq \mathsf{i}_{B'}(s')$ since there is no $\bar{s} \in B$ with $\bar{s} \preceq s'$ and $s' \notin B'$.

Thus, $Pr^{D}(Reach_{\leq 0}(s,B)) = Pr^{D'}(Reach_{\leq 0}(s',B'))$ for all $D, D' \in Sched^{\mathcal{M}}$.

- $i \to i+1$: Let $Pr(D,s,i,B) := Pr^{D}(Reach_{\leq i}(s,B))$ and for all $\bar{s} \in S$, let $D_{\bar{s}}$ such that

$$D_{\bar{s}}(\varsigma) = D(\bar{s} \to \varsigma) \text{ for all } \varsigma \in Paths^{\mathcal{M}}.$$

Similarly define D' such that for all $\bar{s}' \in S$,

$$D'(\bar{s}' \to \varsigma') := D'_{\bar{s}'}(\varsigma') \text{ for all } \varsigma' \in Paths^{\mathcal{M}}.$$

The definition of $D'_{\bar{s}'}$ is derived from:

$$\begin{array}{llll}
Pr(D,s,i+1,B) & & & \\
= & \sum_{\bar{s}\in S} D(s)(\bar{s}) & \cdot Pr(D_{\bar{s}},\bar{s},i,B) & \\
= & \sum_{\bar{s}\in S} \Delta(\bar{s},S) & \cdot Pr(D_{\bar{s}},\bar{s},i,B) & \text{(Def. 10)} \\
= \sum_{\bar{s}'\in S} & \sum_{\bar{s}\in S} \Delta(\bar{s},\bar{s}') & \cdot Pr(D_{\bar{s},\bar{s}'},\bar{s},i,B) & (D_{\bar{s},\bar{s}'} := D_{\bar{s}}) \\
= \sum_{\bar{s}'\in S} \frac{\Delta(S,\bar{s}')}{\Delta(S,\bar{s}')} & \sum_{\bar{s}\in S} \Delta(\bar{s},\bar{s}') & \cdot Pr(D_{\bar{s},\bar{s}'},\bar{s},i,B) & \\
= \sum_{\bar{s}'\in S} \Delta(S,\bar{s}') & \sum_{\bar{s}\in S} \frac{\Delta(\bar{s},\bar{s}')}{\Delta(S,\bar{s}')} & \cdot Pr(D'_{\bar{s},\bar{s}'},\bar{s}',i,B') & \text{(ind. hyp.)} \\
= \sum_{\bar{s}'\in S} \Delta(S,\bar{s}') & & \cdot Pr(D'_{\bar{s}'},\bar{s}',i,B') & (D'_{\bar{s}'} \text{ as below}) \\
= \sum_{\bar{s}'\in S} D'(s')(\bar{s}') & & \cdot Pr(D'_{\bar{s}'},\bar{s}',i,B') & \text{(Def. 10)} \\
= Pr(D',s',i+1,B') & & &
\end{array}$$

The key in constructing $D'_{\bar{s}'}$ is to combine schedulers $D'_{\bar{s},\bar{s}'}$ such that:

$$Pr(D'_{\bar{s}'},\bar{s}',i,B') = \sum_{\bar{s}\in S} \frac{\Delta(\bar{s},\bar{s}')}{\Delta(S,\bar{s}')} \cdot Pr(D'_{\bar{s},\bar{s}'},\bar{s}',i,B')$$

Clearly, this holds if we define $D'_{\bar{s}'}$ by

$$D'_{\bar{s}'}(\bar{s}') := \sum_{\bar{s}\in S} \frac{\Delta(\bar{s},\bar{s}')}{\Delta(S,\bar{s}')} \cdot D'_{\bar{s},\bar{s}'}(\bar{s}')$$

and for histories $\bar{s}' \to \varsigma$ inductively. As $D'_{\bar{s}'}$ is a convex combination of several other schedulers, it actually is a scheduler (see proof of Lemma 5).

Together with $Reach(s,B) = \bigcup_{n\geq 0} Reach_n(s,B)$ the claim follows.

□

The following theorem is a consequence of the result shown above.

Theorem 6. *For ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$, $s, s' \in S$ with $s \preceq s'$, sets of goal states $B, B' \subseteq S$ where for all $\bar{s}' \in B'$, $\bar{s} \preceq \bar{s}' \Rightarrow \bar{s} \in B$ and there is no $\bar{s} \in B$ with $\bar{s} \preceq \bar{s}'$ and $\bar{s}' \notin B'$:*

$$\inf_{D\in Sched^{\mathcal{M}}} Pr(Reach(s,B)) \geq \inf_{D'\in Sched^{\mathcal{M}}} Pr(Reach(s',B')).$$

For our abstraction this implies that to ensure that a certain set of states is reached from initial state s with probability at most p, it suffices to determine the reachability probability of the corresponding macro states in the abstraction. Positive results on the abstract model carry over to the concrete model, but negative results do not. The remainder of this subsection is devoted to computing minimal reachability probabilities in ADTMCs.

For the computation of reachability measures in ADTMCs we will reduce the problem to reachability for *Markov decision processes* (MDPs). To start with, we compare the notion of ADTMCs with the one of MDPs in the three-valued setting.

Definition 14 (MDP). *A Markov decision process is a tuple $\mathcal{M} = (S, Act, \mathbf{P}, L)$, with S and L as for ADTMCs and*

- *Act, a finite set of actions,*
- *$\mathbf{P} : S \times Act \times S \to [0,1]$, an action-dependent probability function with $\mathbf{P}(s, \mathsf{a}, S) \in \{0,1\}$ for all $s \in S$ and $\mathsf{a} \in Act$, and at least one $\mathsf{a} \in Act$ with $\mathbf{P}(s, \mathsf{a}, S) = 1$ for each $s \in S$,*

Note that MDPs are three-valued in contrast to the usual setting.

An ADTMC induces a MDP w.r.t. its extreme behavior.

Definition 15 (Induced MDP). *For ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$, the induced MDP is given by $\mathcal{M}_{\mathrm{I}} = (S, Act, \mathbf{P}_{\mathrm{I}}, L)$ where:*

- *$Act = \{\mathsf{a}_\mu \mid \mu \in \mathbf{T}(s, \cdot)$ and μ is extreme for some $s \in S\}$,*
- *$\mathbf{P}_{\mathrm{I}}(s, \mathsf{a}_\mu, \cdot) = \mu$ if $\mu \in \mathbf{T}(s, \cdot)$ and μ is extreme, otherwise $\mathbf{P}_{\mathrm{I}}(s, \mathsf{a}_\mu, s') = 0$ for all s'.*

The notion of schedulers carries over in the expected manner, i.e. $Sched_{extr}(\mathcal{M}) = Sched(\mathcal{M}_{\mathrm{I}})$. More importantly, the infimum of the measure of some measurable set w.r.t. $Sched_{extr}(\mathcal{M})$ and $Sched(\mathcal{M}_{\mathrm{I}})$ coincides due to Theorem 3.

For reachability properties, it was shown in the setting of MDPs, that the infimum w.r.t. all schedulers agrees with the infimum w.r.t. simple schedulers [15]. Recall that a scheduler $D \in Sched(\mathcal{M})$ is called *simple*, if the choice of a scheduler does not depend on the history but only on the current state. A similar result also holds for ADTMCs. The set of simple schedulers of ADTMC $\mathcal{M}$ that choose only extreme distributions is denoted by $Sched_{simple,extr}(\mathcal{M})$. Since there are only finitely many simple extreme schedulers, the infimum is indeed a minimum.

Lemma 7 (Reachability extrema). *For ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$, $s \in S$, $B \subseteq S$:*

$$\begin{aligned} &\inf_{D \in Sched^{\mathcal{M}}} Pr^D(Reach(s, B)) \\ = &\inf_{D \in Sched^{\mathcal{M}}_{simple,extr}} Pr^D(Reach(s, B)) \\ = &\min_{D \in Sched^{\mathcal{M}}_{simple,extr}} Pr^D(Reach(s, B)) \end{aligned}$$

Theorem 6 and Lemma 7 yield that the lower bound for some reachability probability in the abstract Markov chain is at most the reachability probability in the concrete Markov chain. Thus, whenever the infimum of a reachability probability is at least p in the abstract Markov chain, this applies to the concrete Markov chain as well.

Theorem 7 (Reachability). *For ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L)$, $s, s' \in S$ with $s \preceq s'$, $B \subseteq S$ where for all $\bar{s}' \in B'$, $\bar{s} \preceq \bar{s}' \Rightarrow \bar{s} \in B$:*

$$\inf_{D \in Sched^{\mathcal{M}}} Pr^D(Reach(s, B)) \geq \min_{D \in Sched^{\mathcal{M}}_{simple,extr}} Pr^D(Reach(s', B')).$$

4.5 Time-Bounded Reachability

We now consider the continuous-time setting and focus on minimal probabilities for time-bounded events. In ACTMCs the probability to reach some state in set B from s within t time units is given by:

$$Reach_{\leq t}(s,B) = \{\sigma \in Paths_s^{\mathcal{M}} \mid \sigma@t' \in B \text{ for some } t' \in [0,t]\}.$$

The probability to reach states in B from s in exactly $i \in \mathbb{N}$ steps is given by:

$$Reach'_i(s,B) := \{\sigma \in Paths^{\mathcal{M}} \mid \sigma[0] = s, \sigma[n] \in B \text{ and for all } k < i, \sigma[i] \notin B\}.$$

Note that in contrast to the discrete-time setting, $Reach'_i(s,B)$ is a set of *timed* paths (when considering sets of time-abstract paths, this is indicated by an *abs* superscript). $Reach'_{\leq i}(s,B)$ and $Reach'(s,B)$ are constructed from $Reach'_i(s,B)$ sets as expected.

The following theorems are adaptations of Theorem 5 and Theorem 6 for the discrete-time setting.

Theorem 8 (Time-bounded reachability simulation). *Let ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E_{unif}, L)$, $s, s' \in S$ with $s \preceq s'$, $t \in \mathbb{R}_{\geq 0}$, sets of goal states $B, B' \subseteq S$ where for all $\bar{s} \in S$ and $\bar{s}' \in B'$, $\bar{s} \preceq \bar{s}' \Rightarrow \bar{s} \in B$ and there is no $\bar{s} \in B$ with $\bar{s} \preceq \bar{s}'$ and $\bar{s}' \notin B'$. Then, for all schedulers $D \in Sched^{\mathcal{M}}$ there exists a scheduler $D' \in Sched^{\mathcal{M}}$ with*

$$Pr^D(Reach_{\leq t}(s,B)) = Pr^{D'}(Reach_{\leq t}(s',B')).$$

Proof. As in the discrete-time setting we show this by induction over the maximal path length i to states in B.

1.) Observe that

$$\begin{aligned} Reach_{\leq t}(s,B) &= Reach'(s,B) \cap Reach_{\leq t}(s,B) \\ &= \left(\textstyle\bigcup_{i\geq 0} Reach'_{\leq i}(s,B)\right) \cap Reach_{\leq t}(s,B) \\ &= \textstyle\bigcup_{i\geq 0} \underbrace{\left(Reach'_{\leq i}(s,B) \cap Reach_{\leq t}(s,B)\right)}_{=:Reach_{\leq i,t}(s,B)}. \end{aligned}$$

2.) The probability for i transitions taken within t time units in a stochastic process where events occur uniformly according to an exponential distribution with rate E_{unif} is known as poisson probability $\psi_{E_{unif},t}(i)$. From [3] we have

$$Pr^D(Reach_{\leq i,t}(s,B)) = \sum_{j=0}^{i} \psi_{E_{unif},t}(j) \cdot Pr^D_{emb}(Reach_{\leq j}(s,B))$$

where Pr^D_{emb} is the probability measure w.r.t. the embedded Markov chain $emb(\mathcal{M})$ of $\mathcal{M}$ and scheduler $D \in Sched^{emb(\mathcal{M})}$. From the proof of Theorem 6 we get that for all $i \in \mathbb{N}$ and for all schedulers $D \in Sched^{\mathcal{M}}$ there is some $D' \in Sched^{\mathcal{M}}$ such that:

$$\begin{aligned} Pr^D(Reach_{\leq i,t}(s,B)) &= \textstyle\sum_{j=0}^{i} \psi_{E_{unif},t}(j) \cdot Pr^D_{emb}(Reach^{abs}_{\leq j}(s,B)) \\ &= \textstyle\sum_{j=0}^{i} \psi_{E_{unif},t}(j) \cdot Pr^{D'}_{emb}(Reach^{abs}_{\leq j}(s',B')) \\ &= Pr^{D'}(Reach_{\leq i,t}(s',B')) \end{aligned}$$

From 1. and 2., our claim follows directly.

□

Theorem 9. *For ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E_{unif}, L)$, $s, s' \in S$ with $s \preceq s'$, $t \in \mathbb{R}_{\geq 0}$, sets of goal states $B, B' \subseteq S$ where for all $\bar{s} \in S$ and $\bar{s}' \in B'$, $\bar{s} \preceq \bar{s}' \Rightarrow \bar{s} \in B$ and there is no $\bar{s} \in B$ with $\bar{s} \preceq \bar{s}'$ and $\bar{s}' \notin B'$:*

$$\inf_{D \in Sched^{\mathcal{M}}} Pr^D(Reach_{\leq t}(s, B)) \geq \inf_{D' \in Sched^{\mathcal{M}}} Pr^{D'}(Reach_{\leq t}(s', B'))$$

By Theorem 4, it suffices to consider extreme schedulers if one is interested in lower bounds. As for the induced MDP of an ADTMC, we can interpret an ACTMC as a *continuous-time Markov decision process* (CTMDP), where each extreme distribution can be chosen by some action. Since we consider uniform ACTMCs, it suffices to consider uniform CTMDPs. This allows to expoit the algorithm for the approximation of probability bounds for timed reachability properties (see [3]). An ε-approximation of transient probabilities q_0, the probability distribution for the system to be in a state at time t, can efficiently be computed in an iterative way:

$$\begin{aligned} q_0 &= \psi_{E_{unif},t}(0) \cdot \mathrm{i}_B + q_1 \\ q_i &= \psi_{E_{unif},t}(i) \cdot \mathrm{i}_B + \mathbf{P}_i \cdot q_{i+1}, \text{ for } i \in \{1, \dots, k(\varepsilon, E_{unif}, t)\}, \\ q_{k(\varepsilon, E_{unif}, t)+1} &= \underline{0} \end{aligned}$$

where $k = k(\varepsilon, E_{unif}, t)$ is a proper truncation point such that

$$\sum\nolimits_{i=k+1}^{\infty} \psi_{E_{unif},t}(i) \leq \varepsilon \quad \iff \quad \sum\nolimits_{i=0}^{k} \psi_{E_{unif},t}(i) \geq 1 - \varepsilon.$$

This can be computed a priori (or optionally on-the-fly) as the probability that n events occur in a Poisson process of rate $E_{unif} \cdot t$:

$$\psi_{E_{unif},t}(n) = \frac{(E_{unif} \cdot t)^n \cdot e^{-E_{unif} \cdot t}}{n!} = \begin{cases} \psi_{E_{unif},t}(n-1) \cdot \frac{E_{unif} \cdot t}{n} & \textit{if } n > 0 \\ e^{-E_{unif} \cdot t} & \textit{if } n = 0. \end{cases}$$

Instead of checking for all (exponentially many) extreme distributions in each iteration, we can find a minimizing distribution in polynomial time, by minimizing the vector-product $\mathbf{P}_i(s, \cdot) \cdot q_{i+1}$ with additional constraint $q_{i+1}(S) = 1$. This can be done by successively assigning as much proportion as possible to the transition leading to the successor s' for which $q_{i+1}(s')$ is minimal. For $N := |S|$, sorting the q-vector can be done in $\mathcal{O}(N \log(N))$ and assertion of probabilities takes $O(N^3)$ since the cut has to be applied N times and the *cut* itself has a complexity of $\mathcal{O}(N^2)$. This yields a worst-case complexity of $\mathcal{O}(N^2 \cdot (N \log(N) + N^3) + N) = \mathcal{O}(N^5)$ for every iteration step[7].

Exploiting transient analysis to compute reachability probabilities can be done in the usual way by a slight transformation of the MDP [5, 29]. From now on, we refer to q_0 as the algorithm's result for reachability probabilities.

[7] For unbounded reachability, a variant of the iterative algorithm for time-bounded reachability can be applied as well. An advantage that carries over is, that there is no need to check exponentially many distributions for each state. As a downside, in contrast to time-bounded reachability in the continuous-time setting, one cannot determine the number of iterations needed to get a good approximation in advance.

$$
\begin{array}{ll}
[\![true]\!](s) = \top & [\![a]\!](s) = L(s,a) \\
[\![\varphi_1 \wedge \varphi_2]\!](s) = [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s) & [\![\neg\varphi]\!](s) = ([\![\varphi]\!](s))^c
\end{array}
$$

$$
[\![\varphi_1 \mathcal{U} \varphi_2]\!](\varsigma) = \begin{cases} \top & \text{if } \exists\, i \in \mathbb{N} : ([\![\varphi_2]\!](\varsigma[i]) = \top \wedge\ \forall\, 0 \leq j < i : [\![\varphi_1]\!](\varsigma[j]) = \top) \\ \bot & \text{if } \forall\, i \in \mathbb{N} : ([\![\varphi_2]\!](\varsigma[i]) = \bot \vee\ \exists\, 0 \leq j < i : [\![\varphi_1]\!](\varsigma[j]) = \bot) \\ ? & \text{otherwise} \end{cases}
$$

$$
[\![\mathcal{P}_{\trianglerighteq p}(\Psi)]\!](s) = \begin{cases} \top & \text{if } Pr^l(s,\Psi,\top) \trianglerighteq p \\ \bot & \text{if } Pr^l(s,\Psi,\bot) \triangleright 1-p \\ ? & \text{otherwise} \end{cases} \qquad \trianglerighteq \in \{>, \geq\}, \triangleright = \begin{cases} > \text{ if } \trianglerighteq\, = \,\geq \\ \geq \text{ if } \trianglerighteq\, = \,> \end{cases}
$$

$$
[\![\mathcal{P}_{\trianglelefteq p}(\Psi)]\!](s) = \begin{cases} \top & \text{if } 1-p \trianglelefteq Pr^l(s,\Psi,\bot) \\ \bot & \text{if } \quad p \triangleleft Pr^l(s,\Psi,\top) \\ ? & \text{otherwise} \end{cases} \qquad \trianglelefteq \in \{<, \leq\}, \triangleleft = \begin{cases} < \text{ if } \trianglelefteq\, = \,\leq \\ \leq \text{ if } \trianglelefteq\, = \,< \end{cases}
$$

Table 3. Three-valued semantics of PCTL.

The following lemma, which states that the above algorithm yields an ε-accurate approximation of time-bounded reachability properties, follows directly from [3].

Lemma 8 (Time-bounded reachability extrema). *For ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E_{unif}, L)$, $s \in S$, $B \subseteq S$, $t \in \mathbb{R}_{>0}$ and error margin ε:*

$$
\inf_{D \in Sched^{\mathcal{M}}} (Reach_{\leq t}(s,B)) - \varepsilon \leq q_0(s) \leq \inf_{D \in Sched^{\mathcal{M}}} Pr^D(Reach_{\leq t}(s,B)).
$$

The section is concluded with a result that allows us to use the approximation algorithm presented above to check if a reachability probability is at most p in an abstraction and, in case the result is positive, to deduce that the same holds in the original model.

Theorem 10 (Time-bounded reachability). *Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, E_{unif}, L)$ be an ACTMC and $s, s' \in S$ with $s \preceq s'$, $B, B' \subseteq S$ where for all $\bar{s}' \in B'$, $\bar{s} \preceq \bar{s}' \Rightarrow \bar{s} \in B$ and $t \in \mathbb{R}_{\geq 0}$. For a given maximal error ε one can compute q_0 with:*

$$
\begin{array}{l}
\inf_{D \in Sched^{\mathcal{M}}} Pr^D(Reach(s,B)) \\
\geq \inf_{D \in Sched^{\mathcal{M}}} Pr^D(Reach(s',B')) \\
\geq q_0(s') \\
\geq \inf_{D \in Sched^{\mathcal{M}}} Pr^D(Reach(s',B')) - \varepsilon
\end{array}
$$

5 Model Checking Three-valued Logics

The characterizations in the previous section in terms of minimal reachability and time-bounded reachability probabilities are the key building blocks for model checking of PCTL and CSL. It remains to provide a three-valued semantics for these logics and, more importantly, to show that verification results on abstract Markov chains carry over to their concrete counterparts.

Three-valued semantics. For ADTMC $\mathcal{M}$, we define the satisfaction function $[\![\, \cdot \,]\!] : \text{PCTL} \to (S \cup Paths^{\mathcal{M}} \to \mathbb{B}_3)$ inductively as shown in Table 3, where:

$$
Pr^l(s,\Phi,\theta) = Pr^l(\{\varsigma \in Paths_s^{\mathcal{M}} \mid [\![\Phi]\!](\varsigma) = \theta\}) \textit{ for } \theta \in \mathbb{B}_3
$$

$$[\![\varphi_1 \mathcal{U}^I \varphi_2]\!](\sigma) = \begin{cases} \top & \text{if } \exists\, t \in I : ([\![\varphi_2]\!](\sigma@t) = \top \wedge\ \forall\, t' \in [0,t) : [\![\varphi_1]\!](\sigma@t') = \top) \\ \bot & \text{if } \forall\, t \in I : ([\![\varphi_2]\!](\sigma@t) = \bot \vee\ \exists\, t' \in [0,t) : [\![\varphi_1]\!](\sigma@t') = \bot) \\ ? & \text{otherwise} \end{cases}$$

Table 4. Three-valued semantics of CSL.

For ACTMC $\mathcal{M}$, the satisfaction function $[\![\cdot]\!] : \mathrm{CSL} \to (S \cup \mathit{Paths}^{\mathcal{M}} \to \mathbb{B}_3)$ is defined similar as for ADTMCs[8]. The timed until operator is as shown in Table 4.

Let us have a closer look at the semantics. For the propositional fragment the semantics is clear. A path σ satisfies until formula $\varphi_1 \mathcal{U}^{[0,t]} \varphi_2$ if φ_1 definitely holds until φ_2 definitely holds at the latest at time t. The until-formula is violated, if either before φ_2 holds, φ_1 is violated, or if φ_2 is definitely violated up to time t. Otherwise, the result is indefinite. For untimed until the semantics is similar, but without any time restrictions.

To determine the satisfaction of $\mathcal{P}_{\leq p}(\Psi)$ we consider the probability of the paths for which Ψ is definitely violated. If this probability is larger than $1-p$, then paths where Ψ holds may have measure at most p. Similarly, to show that $\mathcal{P}_{\leq p}(\Psi)$ is violated, we have to consider the measure of all paths definitely satisfying Ψ. If this measure is greater than p, then obviously $\mathcal{P}_{\leq p}(\Psi)$ is violated. The semantics of $\mathcal{P}_{\trianglelefteq p}(\Psi)$ for $\trianglelefteq \in \{<, >, \geq\}$ follows from a similar argumentation.

Example 7. Consider the CTMC with embedded DTMC as in Fig. 2(a) and exit rate 12. Starting in s_0 (s_1), the probability to reach a non-a-state in 0.3 time units is about 0.9037 (0.9328, respectively). Thus, formula $\varphi = a \to \mathcal{P}_{\geq 0.9}(\mathit{true}\,\mathcal{U}^{[0,0.3]} \neg a)$ is true in all states. Consider the abstraction in Fig. 2(b): The lower and upper probability bounds to reach a non-a-state in 0.3 time units from A_0 are about 0.8807 respectively 0.9037. Hence,

$$[\![a \to \mathcal{P}_{\geq 0.9}(\mathit{true}\,\mathcal{U}^{[0,0.3]} \neg a)]\!](A_0) = ? \sqcup [\![\mathcal{P}_{\geq 0.9}(\mathit{true}\,\mathcal{U}^{[0,0.3]} \neg a)]\!](A_0) = ? \sqcup ?\ = ?\,.$$

For $\mathcal{P}_{\geq 0.88}$ instead of $\mathcal{P}_{\geq 0.9}$, the formula would have been satisfied in the abstraction as well, while for $\mathcal{P}_{\geq 0.91}$ the result would still be ? since $? \sqcup \bot = ?$.

Model checking. As for CTL, model checking works by a bottom-up traversal of the parse tree of the formula φ. Boolean combinations of formulas as well as the $\mathcal{P}$-formulas are evaluated as expected. For the latter, however, we need the lower probability bounds for the satisfaction/violation of an until-formula, which remains the only operator to discuss.

For getting the measure of paths definitely satisfying $\Psi = \varphi_1\, \mathcal{U} \varphi_2$ ($\Psi = \varphi_1\, \mathcal{U}^{[0,t]} \varphi_2$ respectively), it suffices to compute the measure of reaching states satisfying φ_2 (in time bounded by t) along paths of states satisfying φ_1. By induction, we know which states do not satisfy φ_1. Removing those from the Markov chain, a path satisfies $\varphi_1\, \mathcal{U}^{[0,t]} \varphi_2$ iff a state φ_2 is reached (within time bound t). In other words, it remains to solve a time-bounded reachability problem in the reduced graph. Getting the measure of paths violating Ψ definitely, is done similarly by exchanging $\top$ and $\bot$ in the argumentation above.

Recall that the given algorithm for computing time-bounded reachability approximates only with error margin ε. However, it can easily be guaranteed

[8] Note that $Pr^l(s, \Phi, \theta)$ concerns timed paths in the CSL semantics.

that the error due to approximation only yields ? in cases where a definite value could be obtained given a smaller error margin.

Also note that until formulas without time bound can be dealt with in the continuous-time setting as well, just by dropping the exit rate and considering the resulting embedded Markov chain.

The following theorems state that our framework developed so far can indeed be used for abstraction-based model checking. Intuitively, the theorems assert that the result of checking a PCTL / CSL formula in the abstract DTMC / CTMC agrees with the one for the more concrete model, unless the result is indefinite.

Theorem 11 (Preservation of PCTL). *Let s and s' be two states of an ADTMC $\mathcal{M}$ with $s \preceq s'$. Then for all PCTL formulas φ:*

$$[\![\varphi]\!](s') \neq ? \ \textit{implies}\ [\![\varphi]\!](s) = [\![\varphi]\!](s').$$

Proof. By induction on the structure of PCTL formulas. Atomic formulas are *true* and $a \in AP$:

- $[\![true]\!](s') = \top = [\![true]\!](s)$
- $[\![a]\!](s') \neq ? \Rightarrow [\![a]\!](s') = L(s', a) = L(s, a) = [\![a]\!](s)$ for $s \preceq s'$.

Induction hypothesis: for all subformulas φ' of φ, and all states s, s' where $s \preceq s'$:

$$[\![\varphi']\!](s') \neq ? \Rightarrow [\![\varphi']\!](s) = [\![\varphi']\!](s') \qquad (*)$$

- $\varphi = \neg\varphi'$:
  For $[\![\varphi']\!](s') = ?$ we have $[\![\neg\varphi']\!](s') = ?$, hence there is nothing to be shown.
  Otherwise, $[\![\neg\varphi']\!](s') = ([\![\varphi']\!](s'))^c \overset{*}{=} ([\![\varphi']\!](s))^c = [\![\neg\varphi']\!](s)$.

- $\varphi = \varphi_1 \wedge \varphi_2$:
  For $[\![\varphi_1]\!](s') = \perp$ (and for $[\![\varphi_2]\!](s') = \perp$ analogously):
  $[\![\varphi_1 \wedge \varphi_2]\!](s') = [\![\varphi_1]\!](s') \sqcap [\![\varphi_2]\!](s') \overset{*}{=} [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s') = \perp \sqcap [\![\varphi_2]\!](s') = \perp$

  For $[\![\varphi_1]\!](s') = [\![\varphi_2]\!](s') = \top$:
  $[\![\varphi_1 \wedge \varphi_2]\!](s') = [\![\varphi_1]\!](s') \sqcap [\![\varphi_2]\!](s') \overset{*}{=} [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s) = \top \sqcap \top = \top$

  For $[\![\varphi_1]\!](s') = ?$, $[\![\varphi_2]\!](s') \neq \top$ (and for $[\![\varphi_2]\!](s') = ?$, $[\![\varphi_1]\!](s') \neq \top$ analogously):
  $[\![\varphi_1 \wedge \varphi_2]\!](s') = [\![\varphi_1]\!](s') \sqcap [\![\varphi_2]\!](s') = ? \sqcap [\![\varphi_2]\!](s') = ?$

  and thus $(*)$ holds trivially in this case.

- $\varphi = \mathcal{P}_{\trianglelefteq p}(\varphi_1\, \mathcal{U} \varphi_2)$:

  As argued before, model checking of an until-formula can be reduced to a reachability analysis on a properly modified ADTMC $\tilde{\mathcal{M}} = (\tilde{S}, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L})$. Let $B = B' := \{s \in \tilde{S} \mid [\![\varphi_2]\!](s) = \top\}$ (or $B = B' := \{s \in \tilde{S} \mid [\![\varphi_2]\!](s) = \perp\}$ respectively) and $s \preceq s'$. Then due to Theorem 6:

$$\inf_{D \in Sched^{\tilde{\mathcal{M}}}} Pr(Reach(s, B)) \geq \inf_{D \in Sched^{\tilde{\mathcal{M}}}} Pr(Reach(s', B'))$$

Intuitively this means that the probability for paths starting in s' and fulfilling (or violating) $\varphi_1\,\mathcal{U}\varphi_2$ is at most the probability of such paths starting in s. Thus:

$$\begin{aligned}&[\![\varphi]\!](s') = \top\\ &\Rightarrow 1-p \trianglelefteq Pr^l(s', \varphi_1\,\mathcal{U}\varphi_2, \bot) \leq Pr^l(s, \varphi_1\,\mathcal{U}\varphi_2, \bot)\\ &\Rightarrow [\![\varphi]\!](s) = \top\end{aligned}$$

and

$$\begin{aligned}&[\![\varphi]\!](s') = \bot\\ &\Rightarrow p \triangleleft Pr^l(s', \varphi_1\,\mathcal{U}\varphi_2, \top) \leq Pr^l(s, \varphi_1\,\mathcal{U}\varphi_2, \top)\\ &\Rightarrow [\![\varphi]\!](s) = \bot\end{aligned}$$

For $\mathcal{P}_{\trianglerighteq p}(\varphi_1\,\mathcal{U}\varphi_2)$ this can be shown analogously.

□

Theorem 12 (Preservation of CSL). *Let s and s' be two states of an ACTMC $\mathcal{M}$ with $s \preceq s'$. Then for all CSL formulas φ:*

$$[\![\varphi]\!](s') \neq ? \ \textit{implies}\ [\![\varphi]\!](s) = [\![\varphi]\!](s').$$

Proof. For atomic formulas and the boolean operators, the proof is as for preservation of CSL and also for $\varphi = \mathcal{P}_{\trianglelefteq p}(\varphi_1\,\mathcal{U}^I\varphi_2)$ the basic idea is as for untimed until. As slight modification, instead of Lemma 6 one has to refer to Lemma 9 when comparing (time-bounded) reachability probabilities for s and s'.

□

Observe that the 3-valued PCTL semantics on a DTMC (viewed as ADTMC) coincides with the 2-valued PCTL semantics for DTMCs as well as the 3-valued CSL semantics on a uniform CTMC (viewed as ACTMC) coincides with the 2-valued CSL semantics for CTMCs (see Section 2). This shows that our abstraction is *conservative* for positive and negative verification results.

Theorem 13. *Given an ADTMC $\mathcal{M}$ and a PCTL formula φ, we can determine $[\![\varphi]\!]$ in time exponential in the size of $\mathcal{M}$ and linear in the size of φ.*

Proof. For the propositional subset of PCTL, $[\![\varphi]\!](s)$ can obviously be checked in time linear to the size of φ. The complexity for checking reachability properties, i.e. the until operator, is polynomial in the size of the induced MDP of $\mathcal{M}$. As for each state state, the induced MDP may have an exponential number of actions to choose from, in the worst case its size is exponential in the size of $\mathcal{M}$.

□

Though complexity results cannot be provided we want to point out that reachability probabilities for ADTMCs can be computed in an iterative way using *value iteration* [40]. In the setting of ADTMCs it is especially favorable as extreme distributions are *calculated on-the-fly* (in polynomial time) in contrast to the MDP approach, where for all (exponentially many) extreme distributions transitions have to be generated in advance.

Theorem 14. *Given an ACTMC $\mathcal{M}$, a CSL formula φ, and an error margin ε, we can approximate $[\![\varphi]\!]$ in time polynomial in the size of $\mathcal{M}$ and linear in size of φ, E_{unif} and the highest time bound t occurring in φ (dependency on ε is omitted as ε is linear in $E_{unif} \cdot t$). In case the approximation yields $\top$ or $\bot$, the result is correct.*

Proof. For the propositional subset of CSL, $[\![\varphi]\!](s)$ can obviously be checked in time linear to the size of φ. The complexity for checking timed reachability properties can be derived from the complexity for uniform CTMDPs (see [3]), which is linear in the size of $\mathcal{M}$, the exit rate E_{unif}, the time bound t and the number of actions. Instead of checking all extreme distributions in an ACTMC, which would yield an exponential number of actions in a corresponding CTMDP, we can use the polynomial algorithm presented earlier to determine a distribution yielding the minimal reachability property. Thus the complexity of checking ACTMCs is as claimed. To ensure correct answers for each, the $\top$ and $\bot$ case, the approximation results may have to be adjusted by ε (cf. Table 3 and Theorem 10). □

Note that, when considering a formula with *nested* until subformulas, even for CTMCs it is a difficult task to ensure a maximal error for the approximation result. In the case of three-valued model checking this means that ? results may not be correct. However, when considering formulas without nested until subformulas, one can assure that the maximal error of the approximation of reachability probabilities will be at most ε. This implies that ? results are computed with an error of at most $2 \cdot \varepsilon$. Moreover, correct results for ? can be determined by adjusting the reachability probabilities according to Table 3 and Theorem 10.

6 Case Study: Quasi-Birth-Death Processes

Consider a simple warehouse with a capacity of m units (that is filled initially) and an attached production facility. As soon as a unit is removed from the warehouse (WH), the current production volume (PV) is increased by one. Note that the sum of units in the warehouse and production volume is always m. We assume that the speed of production and consumption can be described by negative exponential distributions with constant rates γ and λ. If there are no more units in the warehouse, orders are queued. Such delayed orders (DO) are served from the warehouse quickly and therefore, this is described by a negative exponential distribution with rate ε that is much higher than λ.

For $m = 3$, the system is formally described by the *stochastic Petri net* (SPN) [8] in Fig. 6(a). Numbers at edges denote that the corresponding transition consumes or produces the given number of tokens and the transition can not be fired until there are enough tokens to consume. Here, this is used to model that orders are not delayed unless there is no unit left in storage, i.e. the production volume is m.

The semantics of this SPN, which is a typical example for a quasi-birth-death process (QBD), is an infinite CTMC. Uniformization with rate E results in the infinite uniform CTMC (Fig. 6(b)). For $E, x, y, z \in \mathbb{R}_{\geq 0}$, we shortly write $E^x_{yz} = E^x_y - z, E^x_y = E^x - y$ and $E^x = E - x$. State $s_{i,j}$ represents the marking

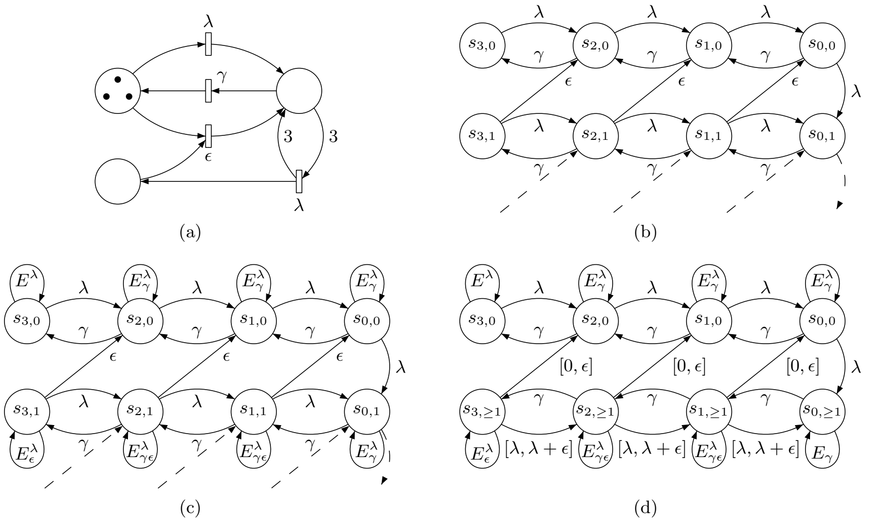


Fig. 6. (a) stochastic Petri net, (b) underlying infinite CTMC, (c) uniformized underlying infinite CTMC, (d) finite abstraction

of the SPN, where i tokens are at WH, $m-i$ at PV and j at DO. Abstracting $\{s_{i,j} \mid j \geq n\}$ by $s_{i,\geq n}$ for all $i \in \{0,\ldots,m\}$ yields Fig. 6(c) ($n=1$).

Consider $\varphi = (\langle l_1 = 0\rangle \wedge \langle l_2 = 0\rangle) \rightarrow \mathcal{P}_{\leq p}(true\,\mathcal{U}^{[0,t]}(\langle l_1 = m\rangle \wedge \langle l_2 = 0\rangle))$ where $\langle l_1 = i\rangle$, $\langle l_2 = j\rangle \in AP$ hold in all states $s_{i,j}$ of the infinite CTMC. Intuitively, φ refers to the probability for completely filling the warehouse within time t, when starting with an empty warehouse and no orders to be served. In Fig. 7, for $\lambda \in \{1, ..., 6\}$, lower and upper probability bounds for φ for abstractions with $n \in \{1, ..., 9\}$ are plotted. As expected, by increasing n, lower and upper bounds are closer, i.e., the accuracy of the abstraction improves.

Increasing capacity m improves the system performance. Less obviously, the probability p for which φ holds decreases for increasing m. This is because upgrading the system with additional capacity m' changes the semantics of φ. The formula is not about raising the storage level by m units anymore, but by $m+m'$. Note that CSL model-checking algorithms for quasi-birth-death processes have also been considered in [41]. Our abstraction technique, though, is not restricted to these (regular) infinite CTMCs and can be applied to any CTMC.

7 Alternative Abstraction Techniques

Abstraction-refinement has been applied to reachability problems in Markov decision processes (MDPs) [13], partial-order reduction techniques using Peled's ample-set method have been generalized to MDPs [20], abstract interpretation has been applied to MDPs [39], and various bisimulation equivalences and simulation pre-orders allow model aggregation prior to model checking, see e. g., [6, 43]. Recent techniques that have been proposed include abstraction of MDPs by two-player stochastic games [35], symmetry reduction [36], and magnifying-lens abstraction [16]. To our knowledge, three-valued abstraction of continuous-time stochastic models has not been considered.

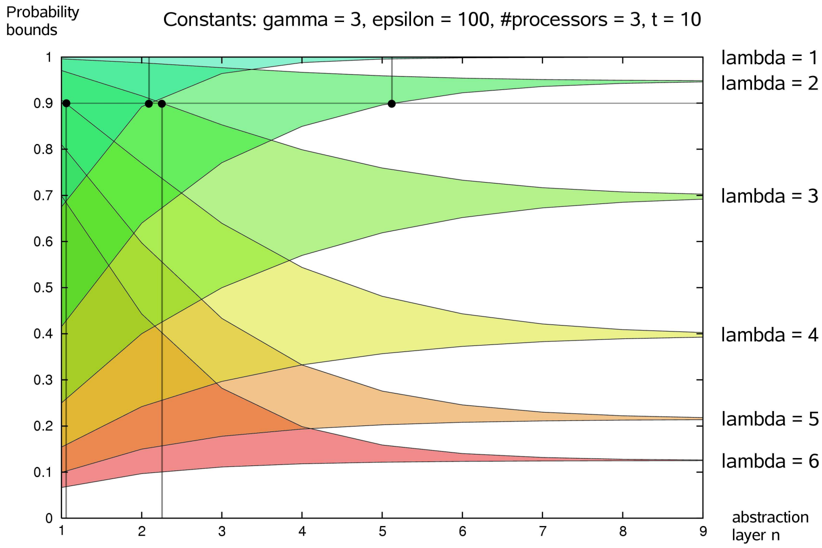


Fig. 7. Probability bounds for φ.

Let us discuss alternative approaches for abstraction of discrete-time Markov chains to the one considered in this paper, hereby comparing our approach, called *AMC-Abstr*, to those found in the literature. For simplicity, we keep the discussion informal.

MDPs as abstractions of DTMCs. Generally, MDPs are considered to be abstractions for Markov chains ([14]). Recall that MDPs extend the model of Markov chains by allowing several distribution functions in each state (see Fig. 8 (c)).

Thus, when merging states to obtain an abstraction, one could define the corresponding distribution functions, as indicated in Figure 8 (a)–(c). Hence, the result would be an MDP. Now, one might be tempted to use existing model checking theory for PCTL and MDPs to reason about the underlying Markov chain. However, this is not possible since, as far as we know, there is no three-valued notion of PCTL for MDPs (not to mention, we need one that suits the role in the abstraction defined here).

When interested in reachability properties, the approach is possible and was pursued in [14]. Let us call the approach *AMDP-Abstr*. Actually, the model checking algorithms presented in the previous section considers the ADTMC

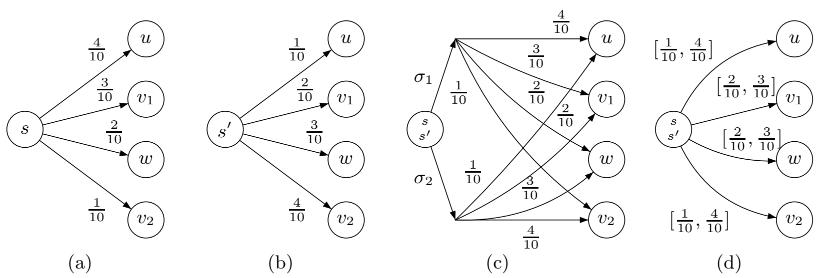


Fig. 8. Abstraction by MDPs vs. AMCs

as an MDP with extreme distributions—but only when computing the minimal probabilities of path properties.

Of course, one could have developed such a three-valued version of PCTL for MDPs as opposed for ADTMCs, as done here. But actually, the three-valued PCTL semantics given in Table 4 can easily be taken over for such a three-valued PCTL semantics for MDPs.

However, there is an intrinsic difference in the approach using ADTMCs and the one based on MDPs. An MDP can easily be abstracted to an ADTMC. For example, for the MDP shown in Figure 8 (c), we would get the ADTMC shown in Figure 8 (d). Observe that using intervals, one *reduces more* information.

This has two implications, one in terms of precision and one terms of memory requirements. Our semantics for PCTL path properties compares *all* extreme distributions. For example, one extreme distribution for the ADTMC in Figure 8 (d) is $(u \mapsto \frac{4}{10}, v_1 \mapsto \frac{2}{10}, w \mapsto \frac{3}{10}, v_2 \mapsto \frac{1}{10})$, which is not present in Figure 8 (c). Now, consider $\phi = [\mathit{true}\, \mathcal{U}\, (a_u \vee a_w)]_{\leq \frac{6}{10}}$, where proposition a_u (a_w) is $\top$ in state u (respectively w) and $\bot$ in all other states. Then the macro state in Figure 8 (c) provides $\top$ for ϕ but for the ADTMC in Figure 8 (d) the result is ?. Thus, our results might sometimes be less precise. In terms of memory, using MDPs, one reduces the number of states but basically all (different) distributions are kept, which may result in only negligible memory savings. In our approach, on the other hand, if, for example, a third distribution denoted by σ_3 with $(u \mapsto \frac{2}{10}, v_1 \mapsto \frac{2}{10}, w \mapsto \frac{3}{10}, v_2 \mapsto \frac{3}{10})$ would be present in Figure 8 (c), we obtain the same ADTMC, thus, reducing the memory requirements.

A different approach was taken in [26], where criteria have been engineered that guarantee an abstraction to be optimal (in some sense not made precise here). Let us call this approach *Optimal-Abstr*. While, of course, such an optimal abstraction sounds preferable, it turns out that neither ADTMCs nor MDPs carry enough information to be optimal. In simple words, the approach loses some of its elegance since it requires to store a lot of information. Furthermore, it is not clear (to us) how to obtain this information without constructing the underlying Markov chain. The author of [26] therefore suggests as well a simpler approximation instead of the optimal abstraction. This approach, which we call *Simple-Abstr*, is similar to *AMC-Abstr* but does not use the normalizing operator to optimize the information present in ADTMCs. Thus, results based on *Simple-Abstr* are less precise than the results obtained with our method: We take a simplified version of Example 15 of [26] to show that *Simple-Abstr* is less precise than *AMC-Abstr*. Consider Figure 9. Although the ADTMC is normalized, the lower bound of reaching one of the states of u, v, or w is less strictly larger than 0 as taking the lower bound 0 in all intervals does not yield a valid distribution. For the approach *Simple-Abstr* the lower bound reaching u, v, or w will be determined as $0 + 0 + 0 = 0$. Consequently, a three-valued PCTL approach based on *Simple-Abstr* would yield ? for the formula $\mathcal{P}_{>0}(\mathit{true}\, \mathcal{U}\, (a_u \vee a_v \vee a_w))$ in state s, while our approach correctly yields $\top$. In terms of memory, *Simple-Abstr* and *AMC-Abstr* are comparable, provided the fixpoint computation method *(value iteration)* is used. Note that [26] does not address the question of model checking.

Summarizing, with abstraction usually one loses information when reducing space requirements. All approaches have in common, that states are grouped together to form an abstract system. They differ in the information that is kept for

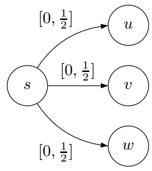


Fig. 9. A normalized ADTMC

transitions. By means of precision, we can order the approaches *Simple-Abstr* < *AMC-Abstr* < *AMDP-Abstr* < *Optimal-Abstr*, where $a < b$ means that a is less precise than b, when for some concrete system, the same states are grouped together. In terms of memory usage, we can order the approaches as *Simple-Abstr* = *AMC-Abstr* < *AMDP-Abstr* < *Optimal-Abstr*, where $a < b$ means that a consumes less memory than b, when for some concrete system, the same states are grouped together.

Abstracting MDPs. Starting from MDPs, one might apply both abstraction schemes, the one presented in [14] (for reachability properties) (see also Figure 8 (c)) and the approach presented in this paper (*AMC-Abstr*), by first translating an MDP to an ADTMC as outlined above. Both methods are applicable due to the following fact: The abstraction operator is an endomorphism, i.e., the abstraction of an MDP or ADTMC is again an MDP or, respectively, an ADTMC.

However, [35] proposes a different abstraction for MDPs. Given a concrete MDP that is to be abstracted by means of aggregating states, the idea is to reflect the non-determinism present in an MDP as on *player* and the non-determinism introduced by means of abstraction as a second *player*. This distinction allows to actually compute boundary values in the abstract system that are closer to the concrete system. Note that however, abstraction is no longer an endomorphism.

Practically obtaining Abstractions. In this paper, we do not consider methods for actually obtaining useful abstractions. In [44] predicate abstraction [19] for probabilistic systems is introduced. However, only an over-approximation of the system is computed allowing only affirmative answers for safe PCTL formulas. It would be interesting to extent these ideas to the three-valued setting hereby supporting full PCTL.

8 Conclusion

In this paper, we have studied the foundations of three-valued abstraction for discrete- and continuous-time Markov chains. The key ingredients of our technique are a partitioning of the state space combined with an abstraction of transition probabilities by intervals. In the continuous-time setting, a similar approach is pursued by turning a CTMC into a (weak bisimilar) model in which all states have an identical exit rate.

The focus of this paper is on the theoretical aspects of our abstraction technique, in particular on properties of the abstraction, model-checking algorithms for abstract Markov chains, three-valued interpretation of probabilistic logics, and preservation results. Although our approach—first and foremost—intends to combat the state-space explosion problem, it is worthwhile to mention that studying model checking of interval Markov chains is of interest in its own when the exact probabilities are not known and, e.g., estimated by experiments, cf. [42]. The results of this paper thus have a broader impact than abstraction.

The feasibility of our abstraction approach has been shown by means of an infinite-state stochastic Petri net model. Experiments with various other models indicate that—like for most other abstraction techniques in traditional model checking—the partitioning of the state space determines the accuracy of the abstraction. For instance, merging "slow" and "fast" states typically yields too coarse abstractions. An important topic for future research is to obtain more practical insight, as well as systematic techniques to obtain appropriate state-space partitions. An interesting approach in this direction is to use predicate abstraction techniques as proposed in [44]. In addition, refinement techniques for probabilistic models are needed that allow refining abstract models that are too coarse (in our setting, their verification would yield indefinite answers). We are currently investigating to adopt counterexample-guided refinement to the probabilistic setting. Promising characterizations of counterexamples in probabilistic model checking have recently been reported [22], and we are currently investigating their use to refine abstract Markov chains.

References

1. Aziz, A., Sanwal, K., Singhal, V., Brayton, R.: Model-checking continuous time Markov chains. *ACM TOCL* **1** (2000) 162–170
2. Baier, C., Clarke, E., Hartonas-Garmhausen, V., Kwiatkowska, M., Ryan, M.: Symbolic model checking for probabilistic processes. In: *Automata, Languages and Programming (ICALP). LNCS*, Vol. 1256. (1997) 430–440
3. Baier, C., Hermanns, H., Katoen, J. P., Haverkort, B. R.: Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. *TCS* **345** (2005) 2–26
4. Baier, C., Ciesinski, F., Größer, M.: ProbMela and verification of Markov decision processes. *Performance Evaluation Review* **32** (2005) 22–27
5. Baier, C., Haverkort, B., Hermanns, H., Katoen, J.-P.: Model-checking algorithms for continuous-time Markov chains. *IEEE TSE* **29** (2003) 524–541
6. Baier, C., Katoen, J.-P., Hermanns, H., Wolf, V.: Comparative branching-time semantics for Markov chains. *Information and Computation* **200** (2005) 149–214
7. Baier, C., Kwiatkoswka, M.: Model checking for a probabilistic branching time logic with fairness. *Distributed Computing* **11** (1998) 125–155
8. Bause, F., Kritzinger, P. S.: Stochastic Petri nets: An introduction to the theory. *SIGMETRICS Performance Evaluation Review* **26** (1998)
9. Ben Mamoun, M., Pekergin, N., Younès, S.: Model checking of continuous-time Markov chains by closed-form bounding distributions. In: *QEST*. IEEE CS (2006) 189–198
10. Böde, E., Herbstritt, M., Hermanns, H., Johr, S., Peikenkamp, T., Pulungan, R., Wimmer, R., Becker, B.: Compositional performability evaluation for Statemate. In: *QEST*. IEEE CS (2006) 167–178
11. Ciardo, G., III, R. L. J., Miner, A., Siminiceanu, R.: Logic and stochastic modeling with SMART. In: *Computer Performance Evaluation. LNCS*, Vol. 2794. (2003) 78–97
12. D'Aprile, D., Donatelli, S., Sproston, J.: CSL model checking for the GreatSPN tool. In: *Computer and Information Sc., ISCIS. LNCS*, Vol. 3280. (2004) 543–553
13. D'Argenio, P. R., Jeannet, B., Jensen, H. E., Larsen, K. G.: Reachability analysis of probabilistic systems by successive refinements. In: *PAPM–PROBMIV. LNCS*, Vol. 2165. (2001) 39–56

14. D'Argenio, P. R., Jeannet, B., Jensen, H. E., Larsen, K. G.: Reduction and refinement strategies for probabilistic analysis. In: *PAPM-PROBMIV. Lecture Notes in Computer Science*, Vol. 2399. Springer (2002) 57–76
15. de Alfaro, L.: *Formal verification of probabilistic systems*. PhD thesis, Stanford, CA, USA (1998) Adviser-Zohar Manna.
16. de Alfaro, L., Pritam, R.: Magnifying-lens abstraction for Markov decision processes. In: *Computer Aided Verification (CAV'07). LNCS*, Vol. 4590. Springer-Verlag (2007) 325–338
17. Fecher, H., Leucker, M., Wolf, V.: Don't know in probabilistic systems. In: *Model Checking Software. LNCS*, Vol. 3925. (2006) 71–88
18. Gilmore, S., Hillston, J.: The PEPA workbench: A tool to support a process algebra-based approach to performance modelling. In: *Computer Performance Evaluation. LNCS*, Vol. 794. (1994) 353–368
19. Graf, S., Saidi, H.: Construction of abstract state graphs with PVS. In: *Computer Aided Verification (CAV'97). LNCS*, Vol. 1254. Springer-Verlag, Haifa, Israel (1997) 72–83
20. Groesser, M., Baier, C.: Partial order reduction for Markov decision processes: a survey. In: *FMCO. LNCS*, Vol. 4111. (2006) 408–427
21. Gross, D., Miller, D.: The randomization technique as a modeling tool and solution procedure for transient Markov chains. *Operations Research* **32** (1984) 343–361
22. Han, T., Katoen, J.-P.: Counterexamples in probabilistic model checking. In: *TACAS*. Vol. 4424. (2007) 72–86
23. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. *Formal Aspects of Computing* **6** (1994) 512–535
24. Huth, M.: An abstraction framework for mixed non-deterministic and probabilistic systems. In: *Validation of Stoch. Systems. LNCS*, Vol. 2925. (2004) 419–444
25. Huth, M.: On finite-state approximants for probabilistic computation tree logic. *TCS* **346** (2005) 113–134
26. Huth, M.: On finite-state approximants for probabilistic computation tree logic. *Theor. Comput. Sci.* **346** (2005) 113–134
27. Huth, M., Jagadeesan, R., Schmidt, D.: Modal transition systems: A foundation for three-valued program analysis. *LNCS* **2028** (2001) 155–169
28. Jansen, D. N., Katoen, J.-P., Oldenkamp, M., Stoelinga, M., Zapreev, I. S.: How fast and fat is your probabilistic model checker? An experimental comparison. In: *Hardware and Software, Verification and Timing*. LNCS. (2008)
29. Jonsson, B., Larsen, K.: Specification and refinement of probabilistic processes. In: *Logic in Computer Science*. IEEE Press (1991) 266–277
30. Katoen, J.-P., Kemna, T., Zapreev, I., Jansen, D. N.: Bisimulation minimisation mostly speeds up probabilistic model checking. In: *TACAS. LNCS*, Vol. 4424. (2007) 87–102
31. Katoen, J.-P., Khattri, M., Zapreev, I. S.: A Markov reward model checker. In: *QEST*. IEEE CS, Los Alamitos (2005) 243–244
32. Katoen, J.-P., Klink, D., Leucker, M., Wolf, V.: Three-valued abstraction for continuous-time Markov chains. In: *Computer-Aided Verification (CAV'07). LNCS*, Vol. 4590. Springer (2007) 316–329
33. Kozine, I. O., Utkin, L. V.: Interval-valued finite Markov chains. *Reliable Computing* **8** (2002) 97–113
34. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic symbolic model checking with PRISM: a hybrid approach. *Int. J. on STTT* **6** (2004) 128–142
35. Kwiatkowska, M., Norman, G., Parker, D.: Game-based abstraction for Markov decision processes. In: *QEST*. IEEE CS (2006) 157–166
36. Kwiatkowska, M., Norman, G., Parker, D.: Symmetry reduction for probabilistic model checking. In: *CAV. LNCS*, Vol. 4144. (2006) 234–248
37. Larsen, K., Thomsen, B.: A modal process logic. In: *Logic in Computer Science (LICS)*. IEEE Computer Society Press (1988) 203–210
38. Moller, F.: *The Edinburgh Concurrency Workbench (Version 6.1)*. Department of Computer Science, University of Edinburgh. (1992)
39. Monniaux, D.: Abstract interpretation of programs as Markov decision processes. *Science of Computer Programming* **58** (2005) 179–205
40. Puterman, M. L.: *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA (1994)
41. Remke, A., Haverkort, B. R., Cloth, L.: Model checking infinite-state Markov chains. In: *TACAS. LNCS*, Vol. 3440. (2005) 237–252

42. Sen, K., Viswanathan, M., Agha, G.: Model-checking Markov chains in the presence of uncertainties. In: *TACAS*. *LNCS*, Vol. 3920. (2006) 394–410
43. Sproston, J., Donatelli, S.: Backward bisimulation in Markov chain model checking. *IEEE TSE* **32** (2006) 531–546
44. Wachter, B., Zhang, K., Hermanns, H.: Probabilistic model checking modulo theories. In: *QEST*. IEEE CS (2007) 129–140
45. Weichselberger, K.: The theory of interval-probability as a unifying concept for uncertainty. *Int. J. of Approximate Reasoning* **24** (2000) 149–170
46. Yi, W.: Algebraic reasoning for real-time probabilistic processes with uncertain information. In: *Formal Techniques in Real-Time and Fault-Tolerant Systems*. *LNCS*, Vol. 863. (1994) 680–693
47. Younes, H.: Ymer: a statistical model checker. In: *Computer-Aided Verificaton*. *LNCS*, Vol. 3567., Berlin (2005) 429–433
48. Zhang, L., Hermanns, H., Eisenbrand, F., Jansen, D. N.: Flow faster: efficient decision algorithms for probabilistic simulations. In: *TACAS*. *LNCS*, Vol. 4424. (2007) 155–170

Aachener Informatik-Berichte

This list contains all technical reports published during the past five years. A complete list of reports dating back to 1987 is available from http://aib.informatik.rwth-aachen.de/. To obtain copies consult the above URL or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: `biblio@informatik.rwth-aachen.de`

2001-01 * Jahresbericht 2000
2001-02 Benedikt Bollig, Martin Leucker: Deciding LTL over Mazurkiewicz Traces
2001-03 Thierry Cachat: The power of one-letter rational languages
2001-04 Benedikt Bollig, Martin Leucker, Michael Weber: Local Parallel Model Checking for the Alternation Free mu-Calculus
2001-05 Benedikt Bollig, Martin Leucker, Thomas Noll: Regular MSC Languages
2001-06 Achim Blumensath: Prefix-Recognisable Graphs and Monadic Second-Order Logic
2001-07 Martin Grohe, Stefan Wöhrle: An Existential Locality Theorem
2001-08 Mareike Schoop, James Taylor (eds.): Proceedings of the Sixth International Workshop on the Language-Action Perspective on Communication Modelling
2001-09 Thomas Arts, Jürgen Giesl: A collection of examples for termination of term rewriting using dependency pairs
2001-10 Achim Blumensath: Axiomatising Tree-interpretable Structures
2001-11 Klaus Indermark, Thomas Noll (eds.): Kolloquium Programmiersprachen und Grundlagen der Programmierung
2002-01 * Jahresbericht 2001
2002-02 Jürgen Giesl, Aart Middeldorp: Transformation Techniques for Context-Sensitive Rewrite Systems
2002-03 Benedikt Bollig, Martin Leucker, Thomas Noll: Generalised Regular MSC Languages
2002-04 Jürgen Giesl, Aart Middeldorp: Innermost Termination of Context-Sensitive Rewriting
2002-05 Horst Lichter, Thomas von der Maßen, Thomas Weiler: Modelling Requirements and Architectures for Software Product Lines
2002-06 Henry N. Adorna: 3-Party Message Complexity is Better than 2-Party Ones for Proving Lower Bounds on the Size of Minimal Nondeterministic Finite Automata
2002-07 Jörg Dahmen: Invariant Image Object Recognition using Gaussian Mixture Densities
2002-08 Markus Mohnen: An Open Framework for Data-Flow Analysis in Java
2002-09 Markus Mohnen: Interfaces with Default Implementations in Java
2002-10 Martin Leucker: Logics for Mazurkiewicz traces
2002-11 Jürgen Giesl, Hans Zantema: Liveness in Rewriting
2003-01 * Jahresbericht 2002
2003-02 Jürgen Giesl, René Thiemann: Size-Change Termination for Term Rewriting
2003-03 Jürgen Giesl, Deepak Kapur: Deciding Inductive Validity of Equations

2003-04 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Improving Dependency Pairs
2003-05 Christof Löding, Philipp Rohde: Solving the Sabotage Game is PSPACE-hard
2003-06 Franz Josef Och: Statistical Machine Translation: From Single-Word Models to Alignment Templates
2003-07 Horst Lichter, Thomas von der Maßen, Alexander Nyßen, Thomas Weiler: Vergleich von Ansätzen zur Feature Modellierung bei der Softwareproduktlinienentwicklung
2003-08 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Mechanizing Dependency Pairs
2004-01 * Fachgruppe Informatik: Jahresbericht 2003
2004-02 Benedikt Bollig, Martin Leucker: Message-Passing Automata are expressively equivalent to EMSO logic
2004-03 Delia Kesner, Femke van Raamsdonk, Joe Wells (eds.): HOR 2004 – 2nd International Workshop on Higher-Order Rewriting
2004-04 Slim Abdennadher, Christophe Ringeissen (eds.): RULE 04 – Fifth International Workshop on Rule-Based Programming
2004-05 Herbert Kuchen (ed.): WFLP 04 – 13th International Workshop on Functional and (Constraint) Logic Programming
2004-06 Sergio Antoy, Yoshihito Toyama (eds.): WRS 04 – 4th International Workshop on Reduction Strategies in Rewriting and Programming
2004-07 Michael Codish, Aart Middeldorp (eds.): WST 04 – 7th International Workshop on Termination
2004-08 Klaus Indermark, Thomas Noll: Algebraic Correctness Proofs for Compiling Recursive Function Definitions with Strictness Information
2004-09 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Parameterized Power Domination Complexity
2004-10 Zinaida Benenson, Felix C. Gärtner, Dogan Kesdogan: Secure Multi-Party Computation with Security Modules
2005-01 * Fachgruppe Informatik: Jahresbericht 2004
2005-02 Maximillian Dornseif, Felix C. Gärtner, Thorsten Holz, Martin Mink: An Offensive Approach to Teaching Information Security: "Aachen Summer School Applied IT Security"
2005-03 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp: Proving and Disproving Termination of Higher-Order Functions
2005-04 Daniel Mölle, Stefan Richter, Peter Rossmanith: A Faster Algorithm for the Steiner Tree Problem
2005-05 Fabien Pouget, Thorsten Holz: A Pointillist Approach for Comparing Honeypots
2005-06 Simon Fischer, Berthold Vöcking: Adaptive Routing with Stale Information
2005-07 Felix C. Freiling, Thorsten Holz, Georg Wicherski: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks
2005-08 Joachim Kneis, Peter Rossmanith: A New Satisfiability Algorithm With Applications To Max-Cut

2005-09 Klaus Kursawe, Felix C. Freiling: Byzantine Fault Tolerance on General Hybrid Adversary Structures

2005-10 Benedikt Bollig: Automata and Logics for Message Sequence Charts

2005-11 Simon Fischer, Berthold Vöcking: A Counterexample to the Fully Mixed Nash Equilibrium Conjecture

2005-12 Neeraj Mittal, Felix Freiling, S. Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Faulty Distributed Systems

2005-13 Carole Delporte-Gallet, Hugues Fauconnier, Felix C. Freiling: Revisiting Failure Detection and Consensus in Omission Failure Environments

2005-14 Felix C. Freiling, Sukumar Ghosh: Code Stabilization

2005-15 Uwe Naumann: The Complexity of Derivative Computation

2005-16 Uwe Naumann: Syntax-Directed Derivative Code (Part I: Tangent-Linear Code)

2005-17 Uwe Naumann: Syntax-directed Derivative Code (Part II: Intraprocedural Adjoint Code)

2005-18 Thomas von der Maßen, Klaus Müller, John MacGregor, Eva Geisberger, Jörg Dörr, Frank Houdek, Harbhajan Singh, Holger Wußmann, Hans-Veit Bacher, Barbara Paech: Einsatz von Features im Software-Entwicklungsprozess - Abschlußbericht des GI-Arbeitskreises “Features”

2005-19 Uwe Naumann, Andre Vehreschild: Tangent-Linear Code by Augmented LL-Parsers

2005-20 Felix C. Freiling, Martin Mink: Bericht über den Workshop zur Ausbildung im Bereich IT-Sicherheit Hochschulausbildung, berufliche Weiterbildung, Zertifizierung von Ausbildungsangeboten am 11. und 12. August 2005 in Köln organisiert von RWTH Aachen in Kooperation mit BITKOM, BSI, DLR und Gesellschaft fuer Informatik (GI) e.V.

2005-21 Thomas Noll, Stefan Rieger: Optimization of Straight-Line Code Revisited

2005-22 Felix Freiling, Maurice Herlihy, Lucia Draque Penso: Optimal Randomized Fair Exchange with Secret Shared Coins

2005-23 Heiner Ackermann, Alantha Newman, Heiko Röglin, Berthold Vöcking: Decision Making Based on Approximate and Smoothed Pareto Curves

2005-24 Alexander Becher, Zinaida Benenson, Maximillian Dornseif: Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks

2006-01 * Fachgruppe Informatik: Jahresbericht 2005

2006-02 Michael Weber: Parallel Algorithms for Verification of Large Systems

2006-03 Michael Maier, Uwe Naumann: Intraprocedural Adjoint Code Generated by the Differentiation-Enabled NAGWare Fortran Compiler

2006-04 Ebadollah Varnik, Uwe Naumann, Andrew Lyons: Toward Low Static Memory Jacobian Accumulation

2006-05 Uwe Naumann, Jean Utke, Patrick Heimbach, Chris Hill, Derya Ozyurt, Carl Wunsch, Mike Fagan, Nathan Tallent, Michelle Strout: Adjoint Code by Source Transformation with OpenAD/F

2006-06 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Divide-and-Color

2006-07 Thomas Colcombet, Christof Löding: Transforming structures by set interpretations

2006-08 Uwe Naumann, Yuxiao Hu: Optimal Vertex Elimination in Single-Expression-Use Graphs
2006-09 Tingting Han, Joost-Pieter Katoen: Counterexamples in Probabilistic Model Checking
2006-10 Mesut Günes, Alexander Zimmermann, Martin Wenig, Jan Ritzerfeld, Ulrich Meis: From Simulations to Testbeds - Architecture of the Hybrid MCG-Mesh Testbed
2006-11 Bastian Schlich, Michael Rohrbach, Michael Weber, Stefan Kowalewski: Model Checking Software for Microcontrollers
2006-12 Benedikt Bollig, Joost-Pieter Katoen, Carsten Kern, Martin Leucker: Replaying Play in and Play out: Synthesis of Design Models from Scenarios by Learning
2006-13 Wong Karianto, Christof Löding: Unranked Tree Automata with Sibling Equalities and Disequalities
2006-14 Danilo Beuche, Andreas Birk, Heinrich Dreier, Andreas Fleischmann, Heidi Galle, Gerald Heller, Dirk Janzen, Isabel John, Ramin Tavakoli Kolagari, Thomas von der Maßen, Andreas Wolfram: Report of the GI Work Group "Requirements Management Tools for Product Line Engineering"
2006-15 Sebastian Ullrich, Jakob T. Valvoda, Torsten Kuhlen: Utilizing optical sensors from mice for new input devices
2006-16 Rafael Ballagas, Jan Borchers: Selexels: a Conceptual Framework for Pointing Devices with Low Expressiveness
2006-17 Eric Lee, Henning Kiel, Jan Borchers: Scrolling Through Time: Improving Interfaces for Searching and Navigating Continuous Audio Timelines
2007-01 * Fachgruppe Informatik: Jahresbericht 2006
2007-02 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, and Harald Zankl: SAT Solving for Termination Analysis with Polynomial Interpretations
2007-03 Jürgen Giesl, René Thiemann, Stephan Swiderski, and Peter Schneider-Kamp: Proving Termination by Bounded Increase
2007-04 Jan Buchholz, Eric Lee, Jonathan Klein, and Jan Borchers: coJIVE: A System to Support Collaborative Jazz Improvisation
2007-05 Uwe Naumann: On Optimal DAG Reversal
2007-06 Joost-Pieter Katoen, Thomas Noll, and Stefan Rieger: Verifying Concurrent List-Manipulating Programs by LTL Model Checking
2007-07 Alexander Nyßen, Horst Lichter: MeDUSA - MethoD for UML2-based Design of Embedded Software Applications
2007-08 Falk Salewski and Stefan Kowalewski: Achieving Highly Reliable Embedded Software: An empirical evaluation of different approaches
2007-09 Tina Kraußer, Heiko Mantel, and Henning Sudbrock: A Probabilistic Justification of the Combining Calculus under the Uniform Scheduler Assumption
2007-10 Martin Neuhäußer, Joost-Pieter Katoen: Bisimulation and Logical Preservation for Continuous-Time Markov Decision Processes
2007-11 Klaus Wehrle: 6. Fachgespräch Sensornetzwerke
2007-12 Uwe Naumann: An L-Attributed Grammar for Adjoint Code

2007-13	Uwe Naumann, Michael Maier, Jan Riehme, and Bruce Christianson: Second-Order Adjoints by Source Code Manipulation of Numerical Programs
2007-14	Jean Utke, Uwe Naumann, Mike Fagan, Nathan Tallent, Michelle Strout, Patrick Heimbach, Chris Hill, and Carl Wunsch: OpenAD/F: A Modular, Open-Source Tool for Automatic Differentiation of Fortran Codes
2007-15	Volker Stolz: Temporal assertions for sequential and current programs
2007-16	Sadeq Ali Makram, Mesut Güneç, Martin Wenig, Alexander Zimmermann: Adaptive Channel Assignment to Support QoS and Load Balancing for Wireless Mesh Networks
2007-17	René Thiemann: The DP Framework for Proving Termination of Term Rewriting

* These reports are only available as a printed version.
Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.