Approximate Symbolic Model Checking of Continuous-Time Markov Chains[⋆]
(Extended Abstract)

Christel Baier[1], Joost-Pieter Katoen[2,3], and Holger Hermanns[3]

[1] Lehrstuhl für Praktische Informatik II, University of Mannheim
68131 Mannheim, Germany
[2] Lehrstuhl für Informatik 7, University of Erlangen-Nürnberg
Martensstraße 3, 91058 Erlangen, Germany
[3] Systems Validation Centre, University of Twente
P.O. Box 217, 7500 AE Enschede, The Netherlands

Abstract. This paper presents a symbolic model checking algorithm for continuous-time Markov chains for an extension of the continuous stochastic logic **CSL** of Aziz et al [1]. The considered logic contains a time-bounded until-operator and a novel operator to express steady-state probabilities. We show that the model checking problem for this logic reduces to a system of linear equations (for unbounded until and the steady state-operator) and a Volterra integral equation system for time-bounded until. We propose a symbolic approximate method for solving the integrals using MTDDs (multi-terminal decision diagrams), a generalisation of MTBDDs. These new structures are suitable for numerical integration using quadrature formulas based on equally-spaced abscissas, like trapezoidal, Simpson and Romberg integration schemes.

1 Introduction

The mechanised verification of a given (usually) finite-state model against a property expressed in some temporal logic is known as *model checking*. For probabilistic systems, transition systems where branching is governed by discrete probability distributions, qualitative and quantitative model checking algorithms have been investigated extensively [2,5,6,7,11,14,15,16,19,23,27]. In a qualitative setting it is checked whether a property holds with probability 0 or 1; in a quantitative setting it is typically verified whether the probability for a certain property meets given lower- or upper-bounds. For discrete-time systems, the quantitative approach has been investigated quite thoroughly: model checking algorithms have been developed for fully probabilistic transition systems [2,5,14,19], like discrete-time Markov chains or generative transition systems, as well as for probabilistic systems that contain non-determinism [6,7,8,16].

In this paper we consider real-time probabilistic systems, that is, we consider the model checking problem for continuous-time Markov chains (CTMCs)

[⋆] The first and second author are sponsored by the DAAD-Project AZ 313-ARC-XII-98/38 on stochastic modelling and verification.

that are at the basis of contemporary performance evaluation and reliability analysis methodologies. A branching-time logic called *continuous-time stochastic logic* (**CSL**) is used to express properties over CTMCs. This logic is an extension of the (equally named) logic by Aziz et al [1] with an operator to reason about steady-state probabilities: e.g. the formula $\mathcal{S}_{\geqslant p}(\Phi)$ asserts that the steady-state probability for a Φ-state is at least p, for $p \in [0,1]$. Apart from the usual path-formulas like next and until, a time-bounded until $\mathcal{U}^{\leqslant t}$, for t a non-negative real, is incorporated, together with standard derivatives, such as a time-bounded eventually $\diamond^{\leqslant t}$. The usual path quantifiers $\forall$ and $\exists$ are replaced by the probabilistic operator $\mathcal{P}_{\bowtie p}(.)$ for comparison operator $\bowtie$ and $p \in [0,1]$. For instance, $\mathcal{P}_{<0.001}(\diamond^{\leqslant 4} error)$ asserts that the probability for a system error within 4 time-units is less than 10^{-3}.

The model checking problem for **CSL** is known to be decidable [1] (for rational time bounds), but to the best of our knowledge no algorithms have been considered yet to verify CTMCs automatically, let alone symbolically. This paper investigates which numerical methods can be adapted to "model check" **CSL**-formulas over CTMCs as models. We show that next and (unbounded) until-formulas can be treated similarly as in the discrete-time probabilistic setting. Checking steady-state probability-properties reduces to solving a linear equation system combined with standard graph analysis methods, while checking the time-bounded until reduces to solving a (recursive) Volterra integral equation system. These integrals are characterised as least fixed points of appropriate higher-order functions, and can thus be approximated by an iterative approach.

One of the major reasons for the success of model checking tools in practice is the efficient way to cope with the state-space explosion problem. A prominent technique is to adopt a compact representation of state spaces using (reduced ordered) *binary decision diagrams*, BDDs for short [9]. This paper follows this line by proposing an alternative variant, referred to as *multi-terminal decision diagrams* (MTDDs), that is suited for the necessary real-time probability calculations. MTDDs are a novel generalisation of multi-terminal binary decision diagrams (MTBDDs [12], also called algebraic decision diagrams [3]), variants of BDDs that can efficiently deal with real matrices. MTBDDs (and MTDDs) allow arbitrary real numbers in the terminal nodes instead of just 0 and 1 (like in BDDs). Whereas MTBDDs are defined on boolean variables, MTDDs allow both boolean and real variables. This generalisation is suitable for numerical integration — needed for time-bounded until — using quadrature formulas based on equally-spaced abscissas (i.e. interval points). This includes well-known methods like trapezoidal, Simpson and Romberg integration schemes [24]. Due to their suitability for numerical integration, the potential application of MTDDs is much wider than model checking CTMCs. For the other temporal operators in **CSL** we show that slight modifications of the MTBDD-approach for discrete-time probabilistic systems [5] can be adopted.

The paper introduces MTDDs, defines appropriate operators on them and presents a symbolic model checking algorithm for **CSL** using these structures. Although it is difficult to obtain precise estimates for the time complexity of model checking using MTDDs (as with BDDs and MTBDDs), the success of (MT)BDD-based model checkers for large-scale examples (for BDDs [10] and for MTBDDs [18,20]) provides sufficient evidence to investigate MTDDs for our setting. For instance, [18] reports experimental results of the computation of steady-state probabilities for discrete-time Markov chains of over 10^{27} states.

Organisation of the Paper. Section 2 introduces the necessary concepts of CTMCs. Section 3 presents the logic **CSL** and provides some useful characterisations of **CSL**-formulas that facilitate a model checking procedure. Section 4 introduces MTDDs, describes how CTMCs can be encoded as MTDDs, and presents several operators on these structures. Section 5 presents the approximative symbolic model checking algorithm. Finally, Section 6 concludes the paper. A (small) running example is used throughout the paper to illustrate the key concepts.

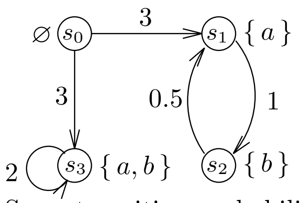
2 Continuous-Time Markov Chains

Basic Definitions. Let AP be a fixed, finite set of atomic propositions. A (labelled) *continuous-time Markov chain* (CTMC for short) is a tuple $\mathcal{M} = (S, \mathbf{Q}, L)$ where S is a finite set of *states*, $\mathbf{Q} : S \times S \to \mathbb{R}_{\geqslant 0}$ the *generator matrix*[1], and $L : S \to 2^{AP}$ the *labelling* function which assigns to each state $s \in S$ the set $L(s)$ of atomic propositions $a \in AP$ that are valid in s.

Intuitively, $\mathbf{Q}(s, s')$ specifies that the probability of moving from state s to s' within t time-units (for positive t) is $1 - e^{-\mathbf{Q}(s,s')\cdot t}$, an exponential distribution with rate $\mathbf{Q}(s, s')$. If $\mathbf{Q}(s, s') > 0$ for more than one state s', a competition between the transitions is assumed to exist, known as the *race condition*. Let $\mathbf{E}(s) = \sum_{s' \in S} \mathbf{Q}(s, s')$, the total rate at which any transition emanating from state s is taken. This rate is the reciprocal of the mean sojourn time in s. More precisely, $\mathbf{E}(s)$ specifies that the probability of leaving s within t time-units (for positive t) is $1 - e^{-\mathbf{E}(s)\cdot t}$, due to the fact that the minimum of exponential distributions (competing in a race) is characterised by the sum of their rates. Consequently, the probability of moving from state s to s' by a single transition, denoted $\mathbf{P}(s, s')$, is determined by the probability that the delay of going from s to s' finishes before the delays of other outgoing edges from s; formally, $\mathbf{P}(s, s') = \mathbf{Q}(s, s')/\mathbf{E}(s)$ (except if s is an absorbing state, i.e. if $\mathbf{E}(s) = 0$; in this case we define $\mathbf{P}(s, s') = 0$). Remark that the matrix $\mathbf{P}$ describes an embedded discrete time Markov chain. (For a more extensive treatment of CTMCs see [25].)

Example 1. As a running example we consider $AP = \{ a, b \}$, $S = \{ s_0, \ldots, s_3 \}$ with $L(s_0) = \varnothing$, $L(s_1) = \{ a \}$, $L(s_2) = \{ b \}$ and $L(s_3) = \{ a, b \}$. The details of the CTMC are:

[1] Whereas usually the diagonal elements are defined as $\mathbf{Q}(s, s) = -\sum_{s' \neq s} \mathbf{Q}(s, s')$ we allow self-loops. This does not affect the transient and steady state behaviour of the chain, but allows the standard interpretation of the next-state operator of the logic.



$$\mathbf{Q} = \begin{pmatrix} 0 & 3 & 0 & 3 \\ 0 & 0 & 1 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix} \text{ and } \mathbf{E} = \begin{pmatrix} 6 \\ 1 \\ 0.5 \\ 2 \end{pmatrix}.$$

Some transition probabilities are $\mathbf{P}(s_0, s_3) = \mathbf{P}(s_0, s_1) = \frac{1}{2}$ and $\mathbf{P}(s_1, s_2) = 1$.

A *path* is a (finite or infinite) sequence $s_0, t_0, s_1, t_1, s_2, t_2, \ldots$, written as

$$\sigma \ = \ s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} s_2 \xrightarrow{t_2} \ \ldots,$$

with for natural i, $s_i \in S$ and $t_i \in \mathbb{R}_{>0}$ such that $\mathbf{Q}(s_i, s_{i+1}) > 0$, if σ is infinite. Otherwise, if $\sigma = s_0 \xrightarrow{t_0} \ldots \xrightarrow{t_{l-1}} s_l$ is finite, we require that s_l is absorbing, and $\mathbf{Q}(s_i, s_{i+1}) > 0$ for all $i < l$. For σ a path, $t \in \mathbb{R}_{>0}$ and natural i let $\sigma[i] = s_i$, the i-th state of σ, $\delta(\sigma, i) = t_i$, the time spent in state s_i, and $\sigma(t) = s_0$ for $t < t_0$, and $\sigma(t) = \sigma[i]$ where i is the smallest index i with $t \leqslant \sum_{0 \leqslant j \leqslant i} t_j$, otherwise. (For σ a finite path with absorbing state s_l, $\sigma[i]$ and $\delta(\sigma, i)$ are only defined for $i \leqslant l$, $\delta(\sigma, l) = \infty$, and $\sigma(t) = s_l$ for $t > t_1 + \ldots + t_{l-1}$.) Let *Path*$(s)$ denote the set of paths in $\mathcal{M}$ starting in s, and *Reach*(s) the set of states reachable from s.

Borel Space. Let $s_0, \ldots, s_k \in S$ with $\mathbf{Q}(s_i, s_{i+1}) > 0$, $(0 \leqslant i < k)$, and $I_0, \ldots, I_{k-1}$ non-empty intervals in $\mathbb{R}_{\geqslant 0}$. Then, $C(s_0, I_0, \ldots, I_{k-1}, s_k)$ denotes the *cylinder set* consisting of all paths $\sigma \in$ *Path*(s_0) such that $\sigma[i] = s_i$ $(i \leqslant k)$, and $\delta(\sigma, i) \in I_i$ $(i < k)$. Let $\mathcal{F}(\mathit{Path}(s))$ be the smallest σ-algebra on *Path*(s) which contains all sets $C(s, I_0, \ldots, I_{k-1}, s_k)$ where $s_0, \ldots, s_k$ ranges over all sequences of states such that $s = s_0$ and $\mathbf{Q}(s_i, s_{i+1}) > 0$ $(0 \leqslant i < k)$ and $I_0, \ldots, I_{k-1}$ ranges over all sequences of non-empty intervals in $\mathbb{R}_{\geqslant 0}$. The probability measure Pr on $\mathcal{F}(\mathit{Path}(s))$ is the unique measure defined by induction on k by $\Pr(C(s_0)) = 1$ and for $k \geqslant 0$:

$$\Pr(C(s_0, \ldots, s_k, I', s')) = \Pr(C(s_0, \ldots, s_k)) \cdot \mathbf{P}(s_k, s') \cdot \left(e^{-\mathbf{E}(s_k) \cdot a} - e^{-\mathbf{E}(s_k) \cdot b}\right)$$

where $a = \inf I'$ and $b = \sup I'$. (For $b = \infty$ and $\lambda > 0$ let $e^{-\lambda \cdot \infty} = 0$.)

3 The Continuous Stochastic Logic CSL

Syntax. CSL is a branching-time, CTL-like temporal logic where the state-formulas are interpreted over states of a CTMC. It adopts operators of PCTL [19], like a time-bounded until operator and a probabilistic operator asserting that the probability for a certain event meets given bounds. We treat a variant of the (equally named) logic of [1] with, for reasons of simplicity, an unnested time-bounded until operator plus a novel steady-state probability operator.

Definition 1. *For $a \in AP$, $p \in [0, 1]$ and $\bowtie \in \{ \leqslant, <, \geqslant, > \}$, the state-formulas of* **CSL** *are defined by the grammar*

$$\varPhi ::= \mathrm{tt} \ \Big| \ a \ \Big| \ \varPhi \wedge \varPhi \ \Big| \ \neg \varPhi \ \Big| \ \mathcal{S}_{\bowtie p}(\varPhi) \ \Big| \ \mathcal{P}_{\bowtie p}(\varphi)$$

where for $t \in I\!R_{\geqslant 0}$ path-formulas are defined by

$$\varphi ::= X\Phi \;\Big|\; \Phi\,\mathcal{U}\,\Phi \;\Big|\; \Phi\,\mathcal{U}^{\leqslant t}\,\Phi.$$

The other boolean connectives are derived in the usual way, i.e. $\mathrm{ff} = \neg\mathrm{tt}$, $\Phi_1 \vee \Phi_2 = \neg(\neg\Phi_1 \wedge \neg\Phi_2)$, and $\Phi_1 \rightarrow \Phi_2 = \neg\Phi_1 \vee \Phi_2$. The intended meaning of the temporal operators $\mathcal{U}$ ("until") and X ("next step") is standard. The temporal operator $\mathcal{U}^{\leqslant t}$ is the real-time variant of $\mathcal{U}$; $\Phi_1\,\mathcal{U}^{\leqslant t}\,\Phi_2$ asserts that $\Phi_1\,\mathcal{U}\,\Phi_2$ will be satisfied in the time interval $[0,t]$; i.e. there is some $x \in [0,t]$ such that Φ_1 continuously holds during the interval $[0,x[$ and Φ_2 becomes true at time instant x. The state formula $\mathcal{S}_{\bowtie p}(\Phi)$ asserts that the steady-state probability for a Φ-state falls in the interval $I_{\bowtie p} = \{\, q \in [0,1] \mid q \bowtie p \,\}$. $\mathcal{P}_{\bowtie p}(\varphi)$ asserts that the probability measure of the paths satisfying φ falls in the interval $I_{\bowtie p}$.

Temporal operators like $\Diamond$, $\Box$ and their real-time variants $\Diamond^{\leqslant t}$ or $\Box^{\leqslant t}$ can be derived, e.g. $\mathcal{P}_{\bowtie p}(\Diamond^{\leqslant t}\,\Phi) = \mathcal{P}_{\bowtie p}(\mathrm{tt}\,\mathcal{U}^{\leqslant t}\,\Phi)$ and $\mathcal{P}_{\geqslant p}(\Box\,\Phi) = \mathcal{P}_{\leqslant 1-p}(\Diamond\,\neg\Phi)$. For example, $\mathcal{P}_{\geqslant 0.99}(\Box\,(\mathit{req} \rightarrow \mathcal{P}_{\geqslant 1}(\Diamond^{\leqslant 5}\,\mathit{resp})))$ asserts that there is a probability of at least 99% that every request will be responded within the next 5 time-units.

Semantics. The state-formulas are interpreted over the states of a CTMC. Let $\mathcal{M} = (S, \mathbf{Q}, L)$ with proposition labels in AP. The definition of the satisfaction relation $\models\; \subseteq S \times \mathbf{CSL}$ is as follows. Let $\mathit{Sat}(\Phi) = \{\, s \in S \mid s \models \Phi \,\}$.

$$\begin{array}{llll}
s \models \mathrm{tt} & \text{for all } s \in S & s \models \Phi_1 \wedge \Phi_2 & \text{iff } s \models \Phi_i, i{=}1,2 \\
s \models a & \text{iff } a \in L(s) & s \models \mathcal{S}_{\bowtie p}(\Phi) & \text{iff } \pi_{\mathit{Sat}(\Phi)}(s) \in I_{\bowtie p} \\
s \models \neg\Phi & \text{iff } s \not\models \Phi & s \models \mathcal{P}_{\bowtie p}(\varphi) & \text{iff } \mathit{Prob}(s,\varphi) \in I_{\bowtie p}.
\end{array}$$

Here, $\pi_{S'}(s)$ denotes the steady-state probability for $S' \subseteq S$ wrt. state s, i.e.

$$\pi_{S'}(s) \;=\; \lim_{t\rightarrow\infty} \Pr\{\, \sigma \in \mathit{Path}(s) \mid \sigma(t) \in S' \,\}.$$

The limit exists, a consequence of S being finite [25]. Obviously, $\pi_{S'}(s) = \sum_{s' \in S'} \pi_{s'}(s)$, where we write $\pi_{s'}(s)$ instead of $\pi_{\{\, s' \,\}}(s)$. We let $\pi_{\varnothing}(s) = 0$. $\mathit{Prob}(s,\varphi)$ denotes the prob. measure of all paths $\sigma \in \mathit{Path}(s)$ satisfying φ, i.e.

$$\mathit{Prob}(s,\varphi) \;=\; \Pr\{\, \sigma \in \mathit{Path}(s) \mid \sigma \models \varphi \,\}.$$

The fact that, for each state s, the set $\{\, \sigma \in \mathit{Path}(s) \mid \sigma \models \varphi \,\}$ is measurable, follows by easy verification. The satisfaction relation (also denoted $\models$) for the path-formulas is defined as usual:

$$\begin{array}{ll}
\sigma \models X\Phi & \text{iff } \sigma[1] \text{ is defined and } \sigma[1] \models \Phi \\
\sigma \models \Phi_1\,\mathcal{U}\,\Phi_2 & \text{iff } \exists k \geqslant 0.\ (\sigma[k] \models \Phi_2 \wedge \forall 0 \leqslant i < k.\, \sigma[i] \models \Phi_1) \\
\sigma \models \Phi_1\,\mathcal{U}^{\leqslant t}\,\Phi_2 & \text{iff } \exists x \in [0,t].\ (\sigma(x) \models \Phi_2 \wedge \forall y \in [0,x[.\, \sigma(y) \models \Phi_1)\,.
\end{array}$$

In the remainder of this section we present alternative characterisations for $\pi_{S'}(s)$ and $\mathit{Prob}(s,\varphi)$ that will serve as a basis for our model checking algorithm. Since the derivation of these characterisations from the theory of CTMCs and DTMCs is not much involved, proofs are omitted.

Computing Steady-State Probabilities. It is well known that the steady state probabilities exist for arbitrary CTMCs. For a strongly connected CTMC $\mathcal{M}$ and (non-absorbing) state s', the steady state probability $\pi_S(s')$ can be obtained by solving a *linear equation system* [25], i.e.

$$\pi_S(s') = \frac{\pi'_S(s')/\mathbf{E}(s')}{\sum_{s \in S} \pi'_S(s)/\mathbf{E}(s)}$$

where $\pi'_S(s'')$ satisfies the linear equation system

$$\pi'_S(s'') = \sum_{s \in S} \mathbf{P}(s, s'') \cdot \pi'_S(s) \text{ such that } \sum_{s \in S} \pi'_S(s) = 1.$$

For the general case we reformulate this as follows. Let G be the underlying directed graph of $\mathcal{M}$ where vertices represent states and where there is an edge from s to s' iff $\mathbf{Q}(s, s') > 0$. Sub-graph B is a *bottom strongly connected component* (bscc) of G if it is a strongly connected component such that for any $s \in B$, $\mathit{Reach}(s) \subseteq B$. We have $\pi_{s'}(s) = 0$ iff s' does not occur in any bscc reachable from s. Let B be a bscc of G with $\mathit{Reach}(s) \cap B \neq \varnothing$ (or equivalently, $B \subseteq \mathit{Reach}(s)$) and assume that a_B is an atomic proposition such that $a_B \in L(s)$ iff $s \in B$. Then $\Diamond a_B$ is a path-formula in **CSL** and $\mathit{Prob}(s, \Diamond B) = \mathit{Prob}(s, \Diamond a_B)$ is the probability of reaching B from s at some time t. For $s' \in B$, $\pi_{s'}(s)$ is given by $\pi_{s'}(s) = \mathit{Prob}(s, \Diamond B) \cdot \pi_B(s')$ where $\pi_B(s') = 1$ if $B = \{s'\}$, and otherwise

$$\pi_B(s') = \frac{\pi'_B(s')/\mathbf{E}(s')}{\sum_{s \in B} \pi'_B(s)/\mathbf{E}(s)}$$

for which $\pi'_B(s'')$ satisfies the linear equation system

$$\pi'_B(s'') = \sum_{s \in B} \mathbf{P}(s, s'') \cdot \pi'_B(s) \text{ such that } \sum_{s \in B} \pi'_B(s) = 1.$$

Example 2. Consider $\mathcal{S}_{>0.5}(\Phi)$ where $\Phi = (a \wedge b) \vee \mathcal{P}_{\leqslant 0.8}(a \,\mathcal{U}^{\leqslant 2}\, b)$, for the CTMC of Example 1. Note that the CTMC is not strongly connected, since e.g. s_3 cannot be reached from s_2 (and vice versa). Assume that Φ is valid in states s_2 and s_3, and invalid otherwise (as we will see later on). Then we have $s_0 \models \mathcal{S}_{>0.5}(\Phi)$, since from s_0 both the bscc $B_1 = \{ s_3 \}$ and the bscc $B_2 = \{ s_1, s_2 \}$ can be reached with probability $\mathbf{P}(s_0, s_3) = \mathbf{P}(s_0, s_2) = 1/2$, and s_2 has a non-zero steady-state probability in B_2. Thus, the steady-state probability for Φ exceeds 0.5. Formally: $\pi_{\mathit{Sat}(\Phi)}(s_0) = \pi_{\{ s_2, s_3 \}}(s_0) = \pi_{s_2}(s_0) + \pi_{s_3}(s_0)$ where $\pi_{s_2}(s_0) = \mathit{Prob}(s_0, \Diamond B_2) \cdot \pi_{B_2}(s_2)$ and $\pi_{s_3}(s_0) = \mathit{Prob}(s_0, \Diamond B_1) \cdot \pi_{B_1}(s_3)$. We have $\mathit{Prob}(s_0, \Diamond B_1) = \mathit{Prob}(s_0, \Diamond B_2) = 1/2$, $\pi_{B_1}(s_3) = 1$, and obtain $\pi'_{B_2}(s_2) = 1/2$ by solving the equation system $\pi'_{B_2}(s_2) = \pi'_{B_2}(s_1)$, $\pi'_{B_2}(s_1) = \pi'_{B_2}(s_2)$, $\pi'_{B_2}(s_1) + \pi'_{B_2}(s_2) = 1$. Subsequently, calculation of $\pi_{B_2}(s_2)$ yields $2/3$. Thus, $\pi_{\{ s_2, s_3 \}}(s_0) = 1/2 \cdot 2/3 + 1/2 \cdot 1 = 5/6$ which indeed exceeds 0.5.

Computing $\mathit{Prob}(s, \varphi)$**.** The basis for calculating the probabilities $\mathit{Prob}(s, \varphi)$ is the following result.

Theorem 1. *For $s \in S$, $t \in I\!R_{\geqslant 0}$ and Φ, Φ_1, Φ_2 state-formulas in* **CSL***:*

1. *$Prob(s, X\Phi) = \sum_{s' \in Sat(\Phi)} \mathbf{P}(s, s')$.*
2. *The function $S \to [0,1]$, $s \mapsto Prob(s, \Phi_1 \mathcal{U} \Phi_2)$ is the least fixed point of the higher-order operator $\Theta : (S \to [0,1]) \to (S \to [0,1])$ where*[2]

$$\Theta(F)(s) = \begin{cases} 1 & \text{if } s \models \Phi_2 \\ \sum_{s' \in S} \mathbf{P}(s,s') \cdot F(s') & \text{if } s \models \Phi_1 \wedge \neg \Phi_2 \\ 0 & \text{otherwise.} \end{cases}$$

3. *The function $S \times I\!R_{\geqslant 0} \to [0,1]$, $(s,t) \mapsto Prob(s, \Phi_1 \mathcal{U}^{\leqslant t} \Phi_2)$ is the least fixed point of the higher-order operator $\Omega : (S \times I\!R_{\geqslant 0} \to [0,1]) \to (S \times I\!R_{\geqslant 0} \to [0,1])$ where*[3]

$$\Omega(F)(s,t) = \begin{cases} 1 & \text{if } s \models \Phi_2 \\ \sum_{s' \in S} \mathbf{Q}(s,s') \cdot \int_0^t e^{-\mathbf{E}(s) \cdot x} \cdot F(s', t-x)\, dx & \text{if } s \models \Phi_1 \wedge \neg \Phi_2 \\ 0 & \text{otherwise.} \end{cases}$$

The first two results of Theorem 1 are identical to the discrete-time probabilistic case, cf. [14,19,4]. This entails that model checking for these formulas can be carried out by well-known methods:

- $(Prob(s, X\Phi))_{s \in S}$ can be obtained by multiplying the transition probability matrix $\mathbf{P}$ with the (boolean) vector $\mathbf{i}_\Phi = (i_\Phi(s))_{s \in S}$ characterising $Sat(\Phi)$, i.e. $i_\Phi(s) = 1$ if $s \models \Phi$, and 0 otherwise.
- $(Prob(s, \Phi_1 \mathcal{U} \Phi_2))_{s \in S}$ can be obtained by solving a linear equation system of the form $\mathbf{x} = \overline{\mathbf{P}} \cdot \mathbf{x} + \mathbf{i}_{\Phi_2}$ where $\overline{\mathbf{P}}(s,s') = \mathbf{P}(s,s')$ if $s \models \Phi_1 \wedge \neg \Phi_2$ and 0 otherwise. $Prob(s, \Phi_1 \mathcal{U} \Phi_2)$ is the least solution of this set of equations. Note, however, that this system of equations can, in general, have more than one solution. The least solution can be obtained by applying an iterative approximative method or a graph analysis combined with standard methods (like Gaussian elimination) to solve regular linear equation systems. The worst case time complexity of this step is linear in the size of φ and polynomial in the number of states.

Example 3. Consider our running CTMC example, $\Phi = (a \wedge b) \vee \mathcal{P}_{\leqslant 0.8}(a \mathcal{U}^{\leqslant 2} b)$ and suppose we want to check $s_1 \models \Phi$. It follows from $\mathbf{Q}(s_1, s_2) = 1$ that the probability of reaching b-state s_2 from s_1 within two time-units equals $1 - e^{-1 \cdot 2} \approx 0.864664$. Formally, we have $s_2 \models \Phi$, since $s_2 \models b$, and $s_1 \not\models \Phi$, since $s_1 \not\models a \wedge b$ and using Theorem 1 we have that $Prob(s_1, a \mathcal{U}^{\leqslant 2} b)$ equals

$$\sum_{s' \in S} \mathbf{Q}(s_1, s') \cdot \int_0^2 e^{-\mathbf{E}(s_1) \cdot x} \cdot F(s', 2-x)\, dx = \int_0^2 e^{-x} dx = [-e^{-x}]_0^2 = 1 - e^{-2}$$

which exceeds 0.8.

[2] The underlying partial order on $S \to [0,1]$ is defined for F_1, $F_2 : S \to [0,1]$ by $F_1 \leqslant F_2$ iff $F_1(s) \leqslant F_2(s)$ for all s.

[3] The underlying partial order on $S \times \mathbb{R}_{\geqslant 0} \to [0,1]$ is defined for F_1, $F_2 : S \times \mathbb{R}_{\geqslant 0} \to [0,1]$ by $F_1 \leqslant F_2$ iff $F_1(s,t) \leqslant F_2(s,t)$ for all s, t.

The last result of Theorem 1 is due to the fact that the probability density function of the sojourn time in state s is given by $\mathbf{E}(s) \cdot e^{-\mathbf{E}(s) \cdot t}$. The resulting recursive integral formula can be reformulated into a *heterogeneous linear differential equation* of the form

$$\mathbf{y}'(t) \;=\; \overline{\mathbf{Q}} \cdot \mathbf{y}(t) \;+\; \mathbf{b}(t)$$

where $\mathbf{y}(t)$ denotes the vector $(Prob(s, \Phi_1 \,\mathcal{U}^{\leqslant t} \Phi_2))_{s \in S}$, and $\overline{\mathbf{Q}}$ is derived from $\mathbf{Q}$, by $\overline{\mathbf{Q}}(s, s') \;= \mathbf{Q}(s, s')$ if $s, s' \models \Phi_1 \wedge \neg\Phi_2$, and otherwise $\overline{\mathbf{Q}}(s, s') = 0$. The vector $\mathbf{b}(t) \;=\; (b_s(t))_{s \in S}$ is given by $b_s(t) \;=\; \sum_{s' \in Sat(\Phi_2)} \mathbf{Q}(s, s') \cdot e^{-\mathbf{E}(s)t}$, if $s \models \Phi_1 \wedge \neg\Phi_2$, and otherwise $b_s(t) = 0$. The vector $(Prob(s, \Phi_1 \,\mathcal{U}^{\leqslant t} \Phi_2))_{s \in S}$ agrees with the following solution of the above heterogeneous linear differential equation:

$$\mathbf{y}(t) \;=\; e^{\overline{\mathbf{Q}}t} \cdot \left(\mathbf{i}_{\Phi_2} \;+\; \int_0^t e^{-\overline{\mathbf{Q}}x} \cdot \mathbf{b}(x)\; dx \right), \text{ where } \quad e^{\overline{\mathbf{Q}}x} \;=\; \sum_{k=0}^{\infty} \frac{(\overline{\mathbf{Q}}x)^k}{k!}.$$

Unfortunately, it is not clear (at least to the authors) how to obtain a closed solution for the above integral. Using a numerical approximation method instead is also not an accurate way out, essentially because known approximative methods for computing $e^{\mathbf{A}x}$ (for some square matrix $\mathbf{A}$) are instable, yet computationally expensive [25]. For that reasons, our algorithm to compute $Prob(s, \Phi_1 \,\mathcal{U}^{\leqslant t}\, \Phi_2)$ is directly based on the last result of Theorem 1. The result suggests the following *iterative* method to approximate $Prob(s, \Phi_1 \mathcal{U}^{\leqslant t}\, \Phi_2)$: let $F_0(s, t) = 0$ for all s, t and $F_{k+1} = \Omega(F_k)$. Then,

$$\lim_{k \to \infty} \; F_k(s, t) \;=\; Prob(s, \Phi_1 \mathcal{U}^{\leqslant t}\, \Phi_2).$$

(The general nested time-bounded until in [1] can be treated in a similar way.) Each step in the iteration amounts to solve an integral of the following form:

$$F_{k+1}(s, t) = \int_0^t \sum_{s' \in S} \mathbf{Q}(s, s') \cdot e^{-\mathbf{E}(s) \cdot x} \cdot F_k(s', t{-}x)\; dx,$$

if $s \models \Phi_1 \wedge \neg\Phi_2$. These integrals can be solved numerically based on quadrature formulas of the type

$$\int_a^b \; f(x)\; dx \quad \approx \quad \sum_{j=0}^{N} \; \alpha_j \cdot f(x_j)$$

with interval points $x_0, \ldots, x_N \in [a, b]$ and weights $\alpha_0, \ldots, \alpha_N$ that do not depend on f (but may be on N). In our model checking algorithm we focus on equally-spaced abscissas, i.e. $x_j \;= a + j \cdot h$ where $h \;=\; (b{-}a)/N$. Well-known methods applied in practice, like trapezoidal, Simpson, and Romberg integration schemes belong to this category [24]. For instance, for the trapezoidal method $\alpha_0 = \alpha_N = \frac{h}{2}$ and $\alpha_i = h$ for $0 < i < N$.

4 Multi-terminal Decision Diagrams

BDDs and MTBDDs. While (ordered) binary decision diagrams (BDDs) are data structures for representing boolean functions $f : \{0,1\}^n \to \{0,1\}$, multi-terminal BDDs (MTBDDs [12], also called algebraic decision diagrams [3]) allow terminals to be labelled with values of some domain D (usually $\mathbb{R}$ or $[0,1]$), i.e. they represent functions of the type $f : \{0,1\}^n \to D$. The main idea behind the MTBDD representation is the use of acyclic rooted directed graphs for a simplified (more compact) representation of the (binary) decision tree which results from the Shannon expansion: $f(b_1,\ldots,b_n) = (1-b_1)\cdot f(0,b_2,\ldots,b_n) + b_1 \cdot f(1,b_2,\ldots,b_n)$.

For model checking discrete-time Markov chains against PCTL-formulas [19] it has been shown that MTBDDs can be effectively used [5,20]. These techniques can potentially be adapted to our continuous-time setting, but are not able to cope with numerical integration, a technique needed for the time-bounded until operator of **CSL** with the iterative method sketched above. Therefore, we introduce a variant of MTBDDs that is focussed on dealing with numerical integration.

MTDDs. *Multi-terminal decision diagrams* (MTDDs) are a variant of MTBDDs that yield a discrete representation of real-valued functions whose arguments are either boolean variables (called *state variables*, since they represent the encoding of states) or real variables (called *integral variables*, since they represent variables over which numerical integration takes place). For instance, MTDDs can represent functions of the type $\{0,1\}^n \times \mathbb{R} \to \mathbb{R}$. For the state variables, the aforementioned Shannon expansion is used. For an integral variable x, a finite set $\{x_0,\ldots,x_N\}$ is chosen from the range of x. The function $(\ldots,x,\ldots) \mapsto f(\ldots,x,\ldots)$ is represented by the function values $f(\ldots,x_j,\ldots)$, for $0 \leqslant j \leqslant N$. To accomplish this, we use a representation of f that is based on a discrete fragment of the decision tree where the branches for the integral variables represent the cases where $x \in \{x_0,\ldots,x_N\}$.

Formally, with each integral variable x (over interval $[0,t]$) the following components are associated: (i) a natural number $N(\mathsf{x})$ that denotes the number of abscissas of x, (ii) a set of abscissas where $abs_j(\mathsf{x})$ denotes the j-th abscissa, (iii) a range $rng(\mathsf{x}) = \{abs_0(\mathsf{x}),\ldots,abs_{N(\mathsf{x})}(\mathsf{x})\}$, and (iv) a number of weights $wt_j^J(\mathsf{x})$ for $J \leqslant N(\mathsf{x})$, and $0 \leqslant j \leqslant J$. The basic idea is that this representation facilitates numerical integration based on the quadrature formula:

$$\int_0^{\mathsf{x}_J} f(\mathsf{x})\, d\mathsf{x} \;\approx\; \sum_{j=0}^{J} wt_j^J(\mathsf{x}) \cdot f(abs_j(\mathsf{x}))$$

where $abs_j(\mathsf{x}) = x_j = j \cdot h$ for step-size $h = t/N(\mathsf{x})$. This corresponds to a quadrature formula in which the interval points $abs_j(\mathsf{x})$ are *equally-spaced* abscissas [24].

For state variable z, we define $rng(\mathsf{z}) = \{0,1\}$ and $N(\mathsf{z}) = 1$. Let $<$ be a fixed total order on Var, the set of state and integral variables, such that $\mathsf{z} < \mathsf{x}$ for all state variables z and integral variables x.

Definition 2. *A* multi-terminal decision diagram *(MTDD) over* $\langle \mathsf{Var}, < \rangle$ *is a rooted acyclic directed graph with vertex set* V *containing 3 types of vertices:*

- *each* state vertex v *is labelled by a state variable* $var(v)$ *and has two children* $child_0(v)$, $child_1(v) \in V$.
- *each* integral vertex v *is labelled by an integral variable* $var(v) = \mathsf{x}$ *and a natural number* $epnt(v) \leqslant N(\mathsf{x})$, *(endpoint) and has* $N(\mathsf{x}){+}1$ *children* $child_0(v)$, $\ldots, child_{N(\mathsf{x})}(v)$.
- *each* terminal vertex v *is labelled by a real number* $val(v)$,

such that $var(v) < var(w)$ *for each non-terminal vertex* v *and non-terminal child* w *of* v.

(For $\mathsf{Var} = \{\mathsf{v}_1, \ldots, \mathsf{v}_n\}$ with $\mathsf{v}_i < \mathsf{v}_{i+1}$ we refer to an MTDD over $\langle \mathsf{Var}, < \rangle$ as an MTDD over $(\mathsf{v}_1, \ldots, \mathsf{v}_n)$.) The constraint on the labelling of the non-terminal vertices is standard for (ordered) BDDs, and requires that on any path from the root to a terminal vertex, the variables respect the given ordering $<$. An MTDD M over $(\mathsf{v}_1, \ldots, \mathsf{v}_n)$ represents a partial function f_{M}, the values of which are obtained by traversing M starting at the root vertex as follows. For state vertex v, the edge from v to $child_0(v)$ represents the case $var(v)$ is false; the edge from v to $child_1(v)$ the case $var(v)$ is true. For integral vertex v, the edge from v to $child_j(v)$ stands for the case where the value of the real variable $var(v) = \mathsf{x}$ is $abs_j(\mathsf{x})$. The value $epnt(v)$ is needed to perform the operator INTEGRATE (defined below). If $epnt(v) = J$ then in vertex v the range of integration is $[0, x_J]$ where $x_J = abs_J(\mathsf{x})$.[4] For efficiency reasons, an implementation will internally represent MTDDs in a *reduced* form [9], a compact and canonical representation.

The relationship between BDDs, MTBDDs and MTDDs is as follows. An MTBDD is an MTDD without integral vertices; a BDD is an MTBDD with $val(v) \in \{0, 1\}$ for all terminal vertices v.

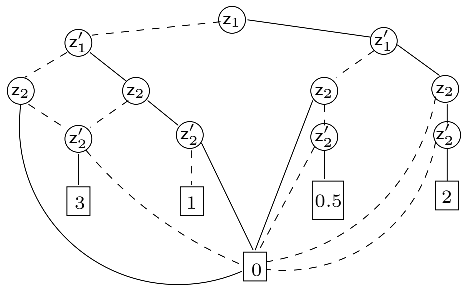
Remark 1. Note that an MTDD over $\langle \mathsf{Var}, < \rangle$ is also an MTDD over $\langle \mathsf{Var}', <' \rangle$ for any superset Var' of Var and total order $<'$ on Var' such that $\mathsf{v}_1 < \mathsf{v}_2$ iff $\mathsf{v}_1 <' \mathsf{v}_2$ for all $\mathsf{v}_1, \mathsf{v}_2 \in \mathsf{Var}$.

Encoding CTMCs by MT(B)DDs. In BDD-approaches transition systems are symbolically represented by encoding states by bit vectors, and encoding the transition relation by its characteristic function. To represent the generator matrix of a CTMC by a MTDD we abstract from the names of the states, and instead, similar to [13], use binary tuples of atomic propositions that are true in that state. Using this scheme, CTMCs are encoded as MTDDs as follows. Let $\mathcal{M} = (S, \mathbf{Q}, L)$ be a labelled CTMC. We assume that $|S| = 2^n$ and that the labelling function L is injective. (Any labelled CTMCs may be transformed into one satisfying these conditions by adding dummy states and new propositions.)

[4] In our model checking procedure, for any integral vertex v with $var(v) = \mathsf{x}$ and $epnt(v) = J < N(\mathsf{x})$, the branches representing the cases where $\mathsf{x} = x_j$ and $j > epnt(v)$ (i.e. the edges to the children $child_j(v)$) are not of importance. Accordingly, we may assume that any integral vertex v with $epnt(v) = J$ has exactly $J{+}1$ children $child_0(v), \ldots, child_J(v)$.

We fix an enumeration $a_1, \ldots, a_n$ of atomic propositions and identify each state s with the boolean n-tuple $(b_1, \ldots, b_n)$ where $b_i = 1$ iff $a_i \in L(s)$. In what follows, we assume that $S = \{0,1\}^n$ where we identify each state s with its encoding and the generator matrix $\mathbf{Q}$ with the function $F : \{0,1\}^{2n} \to \mathbb{R}$ where $F(\mathsf{z}_1, \mathsf{z}'_1, \ldots, \mathsf{z}_n, \mathsf{z}'_n) = \mathbf{Q}((\mathsf{z}_1, \ldots, \mathsf{z}_n), (\mathsf{z}'_1, \ldots, \mathsf{z}'_n))$. We represent $\mathcal{M}$ by the MTBDD Q for $\mathbf{Q}$ over $(\mathsf{z}_1, \mathsf{z}'_1, \ldots, \mathsf{z}_n, \mathsf{z}'_n)$, in other words $f_{\mathsf{Q}} = F$. Note that Q does not contain integral variables and hence is a MTBDD.

Example 4. Consider the CTMC of Example 1. According to the above scheme we encode the states by $s_0 \mapsto 00$, $s_1 \mapsto 01$, $s_2 \mapsto 10$ and $s_3 \mapsto 11$. The function $F = f_{\mathsf{Q}}$ and the MTDD Q are given by:



$$
\begin{aligned}
F(0,1,0,1) &= 3\\
F(0,0,0,1) &= 3\\
F(0,1,1,0) &= 1\\
F(1,0,0,1) &= 0.5\\
F(1,1,1,1) &= 2\\
F(z_1, z'_1, z_2, z'_2) &= 0 \text{ otherwise}
\end{aligned}
$$

where dotted lines denote zero-edges and solid lines one-edges.

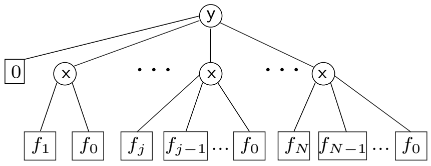
Operators on MTDDs. The symbolic model checking algorithm in this paper uses several operators on MTDDs that are slight modifications of equivalent operators on BDDs [9] and MTBDDs [12,3]. For space reasons we only briefly describe these operators and focus on the new operators, in particular substitution and computing integrals. As it is standard in the BDD setting, hash tables can be used to generate a reduced MTDD during its construction.

- *Combining MTDDs via binary operators.* Operator APPLY allows a pointwise application of the binary operator *op* (like summation or multiplication) to two MTDDs. For MTDDs M_1 and M_2 over $(\mathsf{v}_1, \ldots, \mathsf{v}_n)$, APPLY$(\mathsf{M}_1, \mathsf{M}_2, op)$ yields an MTDD M over $(\mathsf{v}_1, \ldots, \mathsf{v}_n)$ for the function $f_{\mathsf{M}} = f_{\mathsf{M}_1} \, op \, f_{\mathsf{M}_2}$.
- *Variable renaming.* Operator RENAME changes the variable labelling of any v_i-labelled vertex of MTDD M over $(\mathsf{v}_1, \ldots, \mathsf{v}_n)$ into w, for $\mathsf{w} \neq \mathsf{v}_j$, $0 < j \leqslant n$. RENAME$(\mathsf{M}, \mathsf{v}_i, \mathsf{w})$ yields a MTDD over $(\mathsf{v}_1, \ldots, \mathsf{v}_{i-1}, \mathsf{w}, \mathsf{v}_{i+1}, \ldots, \mathsf{v}_n)$.
- *Restriction.* For state variable $\mathsf{v}_i = \mathsf{z}$ and boolean b, RESTRICT$(\mathsf{M}, \mathsf{z}, b)$ denotes the MTDD over $(\mathsf{v}_1, \ldots, \mathsf{v}_{i-1}, \mathsf{v}_{i+1}, \ldots, \mathsf{v}_n)$ that is obtained from M by replacing any edge from a vertex v to an z-labelled vertex w by an edge from v to $child_b(w)$, followed by removing all z-labelled vertices. In a similar way, RESTRICT$(\mathsf{M}, \mathsf{x}, x_j)$ is defined for $\mathsf{v}_i = \mathsf{x}$ an integral variable, $0 \leqslant j \leqslant N(\mathsf{x})$ and $x_j = abs_j(\mathsf{x})$.
- *Comparison operators.* Given MTDD M without integral vertices and over n state variables, and interval I, COMPARE(M, I) is the BDD representing the function that equals 1 if $f_{\mathsf{M}}(b_1, \ldots, b_n) \in I$ and 0, otherwise.
- *Matrix/vector multiplication.* Let MTBDDs Q and B without integral vertices over $2n$ and n state variables, respectively, represent the matrix $\mathbf{Q}$

and vector $\mathbf{b}$. Then $\textsc{Multi}(\mathsf{Q},\mathsf{B})$ denotes the MTBDD over n variables that represents the vector $\mathbf{Q}\cdot\mathbf{b}$. This operator can easily be modified for MTDDs. E.g. if Q is a MTDD over $(\mathsf{v}_1,\ldots,\mathsf{v}_n,\mathsf{w}_1,\ldots,\mathsf{w}_m)$ and B a MTDD over $(\mathsf{w}_1,\ldots,\mathsf{w}_m)$ then $\mathsf{MULTI}(\mathsf{Q},\mathsf{B})$ represents the function

$$(\mathsf{v}_1,\ldots,\mathsf{v}_n)\mapsto\sum_{\mathsf{w}_1,\ldots,\mathsf{w}_m} f_{\mathsf{Q}}(\mathsf{v}_1,\ldots,\mathsf{v}_n,\mathsf{w}_1,\ldots,\mathsf{w}_m)\cdot f_{\mathsf{B}}(\mathsf{w}_1,\ldots,\mathsf{w}_m).$$

– *Substitution.* Let M be a MTDD over $(\mathsf{v}_1,\ldots,\mathsf{v}_n)$ where $\mathsf{v}_n=\mathsf{x}$ is an integral variable with $N(\mathsf{x})=N$ and assume $\mathsf{y}\neq\mathsf{v}_i$ for all i. Assume that for any x-labelled vertex v in M we have $epnt(v)=N$. Then $\textsc{Subst}(\mathsf{M},\mathsf{y},\mathsf{x})$ denotes the MTDD over $(\mathsf{v}_1,\ldots,\mathsf{v}_{n-1},\mathsf{y},\mathsf{x})$ which represents the partial function that equals $f_{\mathsf{M}}(\ldots,\mathsf{y}{-}\mathsf{x})$ for $0\leqslant\mathsf{x}\leqslant\mathsf{y}$ and is undefined otherwise. $\textsc{Subst}(\mathsf{M},\mathsf{y},\mathsf{x})$ results from M by replacing any x-labelled vertex v by the subgraph depicted as:



In the figure (visualising the decision tree instead of the reduced MTDD), children are depicted from left to right and vertices $child_i(v'_j)$ where $i>j$ are omitted. Here $f_j=val(child_j(v))$ for $0\leqslant j\leqslant N$. More precisely, new vertices v', $v'_1,\ldots,v'_N$ are introduced with $var(v')=\mathsf{y}$, $var(v'_j)=\mathsf{x}$, $child_j(v')=v'_j$, $epnt(v')=N$ and $epnt(v'_j)=j$.

– *Computing integrals.* Let $\mathsf{v}_n=\mathsf{x}$ be an integral variable with $N(\mathsf{x})=N$, $rng(\mathsf{x})=\{x_0,\ldots,x_N\}$ and $wt^J_j(\mathsf{x})=\alpha^J_j$. Then, $\textsc{Integrate}(\mathsf{M},\mathsf{x})$ denotes the MTDD over $(\mathsf{v}_1,\ldots,\mathsf{v}_{n-1})$ that results from M by replacing any x-labelled vertex v with $epnt(v)=J$ by the terminal vertex labelled by

$$\sum_{j=0}^{J}\alpha^J_j\cdot val(child_j(v))$$

where $\alpha^0_0=0$. [5] Thus, if $epnt(v)=N(\mathsf{x})$ for each x-labelled vertex v in M then $\textsc{Integrate}(\mathsf{M},\mathsf{x})$ represents an approximation of the function $(\ldots)\mapsto\int_0^t f_{\mathsf{M}}(\ldots,\mathsf{x})d\mathsf{x}$.

Besides the described operators on MTDDs, our model checking algorithm uses methods for boolean combinators and for a BDD-based graph analysis, e.g. to obtain the bottom strongly connected components of the graph underlying a CTMC, and MTBDD-based methods for solving linear equation systems, e.g. to compute the probabilities $\pi_{s'}(s)$. For these algorithms we refer to [9,12,3].

[5] Note that we assume that $\mathsf{x}=\mathsf{v}_n$; hence v_n is the largest in the variable ordering, and the children of a x-labelled vertex are terminal vertices.

5 Symbolically Model Checking CSL

Our symbolic model checking algorithm for **CSL** works as follows. Let $\mathcal{M} = (S, \mathbf{Q}, L)$ be a CTMC which is represented by a MT(B)DD Q over $2n$ variables as explained in the previous section. For each **CSL**-state-formula Φ we define a BDD $\mathsf{Sat}[\![\, \Phi \,]\!]$ over $(\mathsf{z}_1, \ldots, \mathsf{z}_n)$ that represents the characteristic function of the set $Sat(\Phi)$; for each **CSL**-path formula φ we define a MTBDD $\mathsf{PR}[\![\, \varphi \,]\!]$ representing the function $s \mapsto Prob(s, \varphi)$. By applying standard operators on MTBDDs we determine the MTBDDs P, representing the transition probability matrix $\mathbf{P}$ of $\mathcal{M}$, and SP representing the steady-state probabilities $\pi_{s'}(s)$ for $s, s' \in S$. $\mathsf{Sat}[\![\, \Phi \,]\!]$ is defined as follows:

$$\begin{aligned}
\mathsf{Sat}[\![\, \mathrm{tt} \,]\!] &= \mathbf{1} \\
\mathsf{Sat}[\![\, a_i \,]\!] &= \text{the BDD for the boolean function } (\mathsf{z}_1, \ldots, \mathsf{z}_n) \mapsto \mathsf{z}_i \\
\mathsf{Sat}[\![\, \neg\Phi \,]\!] &= \neg\mathsf{Sat}[\![\, \Phi \,]\!] \\
\mathsf{Sat}[\![\, \Phi_1 \wedge \Phi_2 \,]\!] &= \mathsf{Sat}[\![\, \Phi_1 \,]\!] \wedge \mathsf{Sat}[\![\, \Phi_2 \,]\!] \\
\mathsf{Sat}[\![\, \mathcal{S}_{\bowtie p}(\Phi) \,]\!] &= \textsc{Compare}(\textsc{Multi}(\mathsf{SP}, \mathsf{Sat}[\![\, \Phi \,]\!]'), I_{\bowtie p}) \\
\mathsf{Sat}[\![\, \mathcal{P}_{\bowtie p}(\varphi) \,]\!] &= \textsc{Compare}(\mathsf{PR}[\![\, \varphi \,]\!], I_{\bowtie p}).
\end{aligned}$$

Here, $\mathbf{1}$ denotes the BDD consisting of a single, terminal vertex labelled by 1. $\mathsf{Sat}[\![\, a_i \,]\!]$ is a BDD consisting of a single state-vertex v labelled with z_i such that $child_0(v)$ and $child_1(v)$ are labelled with 0 and 1, respectively. $\mathsf{Sat}[\![\, \Phi \,]\!]'$ denotes $\mathsf{Sat}[\![\, \Phi \,]\!]$ where z_i is renamed into z'_i (using nested applications of Rename). The definition of $\mathsf{Sat}[\![\, \mathcal{S}_{\bowtie p}(\Phi) \,]\!]$ is justified by the characterisation of $\pi_{Sat(\Phi)}(s)$ in Section 3.

MTBDD $\mathsf{PR}[\![\, \varphi \,]\!]$ is defined by induction over the structure of φ. For $\varphi = X\Phi$ and $\varphi = \Phi_1 \, \mathcal{U} \, \Phi_2$, the MTBDD $\mathsf{PR}[\![\, \varphi \,]\!]$ can be obtained in the same way as for the discrete-time probabilistic case [5]. This follows directly from the first two clauses of Theorem 1. For the time-bounded until-operator we define:

$$\mathsf{PR}[\![\, \Phi_1 \mathcal{U}^{\leqslant t} \Phi_2 \,]\!] = \textsc{BoundedUntil}(\mathsf{Q}, \mathsf{Sat}[\![\, \Phi_1 \,]\!], \mathsf{Sat}[\![\, \Phi_2 \,]\!], t, k_{max}, \epsilon)$$

where k_{max} indicates the maximum number of iterations and ϵ is the maximum desired tolerance of the approximation. The algorithm for BoundedUntil is listed in Table 1. Here, F_0 represents the first approximation $F_0(s,t) = 0$. First the MT(B)DD H for the function $H(s, s', x) = \mathbf{Q}(s, s') \cdot e^{-\mathbf{E}(s) \cdot x}$ is constructed. This requires as input the MTBDD-representations of $\mathbf{E}$ that can easily be obtained from the MTDD Q representing the generator matrix (cf. Section 2). X consists of a single, integral vertex labelled by x with $N{+}1$ terminal vertices labelled with the values $x_0, \ldots, x_N$. Here we assume that N is sufficiently large. In the first five steps of the iteration, the MTDD-representation of F_{k+1} is constructed systematically. More precisely, F_k represents (approximations) for the values $F_k(s, x_j)$ $(0 \leqslant j \leqslant N)$ where $x_j = j \cdot h$ and $h = t/N$. I' represents the function $f_{\mathsf{I}'}(s', y, x) = F_k(s', y{-}x)$. MTDD J represents the function

$$f_{\mathsf{J}}(s, s', y, x) = \mathbf{Q}(s, s') \cdot e^{-\mathbf{E}(s) \cdot x} \cdot F_k(s', y{-}x).$$

```
algorithm BoundedUntil $(\mathsf{Q}, \mathsf{B}_1, \mathsf{B}_2, t, k_{max}, \epsilon)$ :
begin $\mathsf{F}_0 := \mathbf{0}; k := 0;$
        $\mathsf{H} := \text{Apply}(\mathsf{Q}, \text{Apply}(\mathsf{E}, \mathsf{X}, (q_1, q_2) \mapsto e^{-q_1 \cdot q_2}), \cdot);$
        repeat
            $\mathsf{I} := \text{Subst}(\mathsf{F}_k, \mathsf{x}, \mathsf{y});$
            $\mathsf{J} := \text{Apply}(\mathsf{H}, \mathsf{I}', \cdot);$
            $\mathsf{K} := \text{Multi}(\mathsf{J}, \mathbf{1});$
            $\mathsf{L} := \text{Rename}(\text{Integrate}(\mathsf{K}, \mathsf{x}), \mathsf{y}, \mathsf{x});$
            $\mathsf{F}_{k+1} := \text{Apply}(\text{Apply}(\mathsf{L}, \mathsf{B}_1, min), \mathsf{B}_2, max);$
            $\mathsf{D}_{k+1} := \text{Apply}(\mathsf{F}_{k+1}, \mathsf{F}_k, -);$
            $\Delta_{k+1} := \max_{s,j} \left| f_{\mathsf{D}_{k+1}}(s, x_j) \right|;$
            $k := k + 1;$
        until $(k = k_{max}$ or $\Delta_k \leqslant \epsilon);$
        if $\Delta_k \leqslant \epsilon$ then return Restrict$(\mathsf{F}_k, \mathsf{x}, t)$ else return $error$;
end.
```

Table 1. Algorithm for BoundedUntil

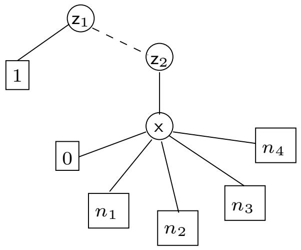
For this, we consider J as the MTDD representation of a matrix whose rows are indexed by triples (s, y, x), and whose columns are indexed by s'. Matrix-vector multiplication with $\mathbf{1}$, the MTDD over $(\mathsf{z}'_1, \ldots, \mathsf{z}'_n)$ that represents the constant function $s' \mapsto 1$, yields MTDD K over $(\mathsf{z}_1, \ldots, \mathsf{z}_n, \mathsf{y}, \mathsf{x})$ representing

$$f_{\mathsf{K}}(s, y, x) = \sum_{s' \in S} f_{\mathsf{J}}(s, s', y, x).$$

By Integrate(K, x) the integrals $\int_0^{x_J} f_{\mathsf{K}}(s, x_J, x)dx$ are approximated by $\sum_j \alpha_j^J \cdot f_{\mathsf{K}}(s, x_J, x_j)$. For generating the MTDD F_{k+1} that represents function $(s, x) \mapsto 1$ if $s \models \Phi_2$, $(s, x) \mapsto f_{\mathsf{L}}(s, x)$ if $s \models \Phi_1 \wedge \neg\Phi_2$ and $(s, x) \mapsto 0$ otherwise, we use the fact that $F_{k+1}(s, x) = \max\{ \min\{ f_{\mathsf{Sat}[\![\Phi_1]\!]}(s), f_{\mathsf{L}}(s, x) \}, f_{\mathsf{Sat}[\![\Phi_2]\!]}(s) \}$.

Finally, after the calculation of F_{k+1}, the result is compared with the result of the previous iteration, by an inspection of the terminal nodes of D_{k+1} which represents the difference between F_k and F_{k+1}. The iteration is finished if either the indicated maximum number of iterations is reached, or the tolerance of an "acceptable" approximation results.

Example 5. Consider our running example and check $s_1 \models \mathcal{P}_{\leqslant 0.8}(a\,\mathcal{U}^{\leqslant 2}\, b)$. We assume that N equals 4 and adopt the trapezoidal method for numerical integration. The MTBDD Q is the same as in Example 4. In the first iteration we obtain for F_1 a single state vertex v labelled z_1 with $child_1(v) = \mathbf{1}$, the terminal vertex labelled 1. In the second iteration we obtain the MTDD F_2 depicted on the rigth. Here $n_1 = \frac{1}{2} \cdot e^{-0} + \frac{1}{2} \cdot e^{-0.5}$, $n_2 = \frac{1}{4} \cdot e^{-0} + \frac{1}{2} \cdot e^{-0.5} + \frac{1}{4} \cdot e^{-1}$, $n_3 = \frac{1}{4} \cdot e^{-0} + \frac{1}{2} \cdot (e^{-0.5} + e^{-1}) + \frac{1}{4} \cdot e^{-1.5}$, and



$n_4 = \frac{1}{4} \cdot e^{-0} + \frac{1}{2} \cdot (e^{-0.5} + e^{-1} + e^{-1.5}) + \frac{1}{4} \cdot e^{-2}$. The third iteration reveals that $\mathsf{F}_3 = \mathsf{F}_2$, so the algorithm finishes and returns $\mathsf{R} = \textsc{Restrict}(\mathsf{F}_3, \mathsf{x}, 2)$, obtained from F_3 by replacing the subgraph starting in vertex x by the terminal vertex with label $n_4 \approx 0.882604$. Finally, $\textsc{Compare}(\mathsf{R}, I_{\leqslant 0.8})$ reveals that $\mathcal{P}_{\leqslant 0.8}(a\,\mathcal{U}^{\leqslant 2}\, b)$ is indeed violated in s_1. Increasing the number of abscissas increases the accuracy: e.g. $N = 64$ leads to 0.8647350 as an approximation for $1 - e^{-2}$.

6 Concluding Remarks

We have presented a symbolic model checking algorithm for verifying properties stated in **CSL** over continuous-time Markov chains. The basis of this model checking procedure is a characterisation of time-bounded until in terms of a Volterra integral equation system that can be solved by iteration. To solve the integrals in a symbolic way we generalised MTBDDs into multi-terminal decision diagrams (MTDDs) and presented suitable operators on these structures that facilitate a numerical integration using quadrature formulas based on equally-spaced abscissas. Due to their suitability for numerical integration, the potential application of MTDDs is much wider than model checking CTMCs.

An important direction for future research is the implementation of the proposed algorithm that should provide evidence about the adequacy of our approach. Amongst others, the size of intermediate MTDDs is unclear yet, and we want to compare our technique with standard methods to extract performance measures from Markov chains [26]. In fact, **CSL** and the algorithm can be generalised such that transient and steady state measures are expressible and can be approximated. We also plan to consider CTMCs that may contain non-determinism, like stochastic transition systems [17] or interactive Markov chains [21,22]. We believe that by extending our approach with schedulers [27] in a similar way as for the discrete-time probabilistic case this is feasible.

Acknowledgement. We thank Markus Siegle for discussion about our initial ideas concerning MTDDs. Ed Brinksma provided valuable comments on an earlier version.

References

1. A. Aziz, K. Sanwal, V. Singhal and R. Brayton. Verifying continuous time Markov chains. In *CAV*, LNCS 1102, pp. 269–276, 1996.
2. A. Aziz, V. Singhal, F. Balarin, R. Brayton and A. Sangiovanni-Vincentelli. It usually works: the temporal logic of stochastic systems. In *CAV*, LNCS 939, pp. 155–165, 1995.
3. I. Bahar, E. Frohm, C. Gaona, G. Hachtel, E. Macii, A. Padro and F. Somenzi. Algebraic decision diagrams and their applications. *Formal Methods in System Design*, **10**(2/3): 171–206, 1997.
4. C. Baier. On algorithmic verification methods for probabilistic systems. Habilitation thesis (submitted), Univ. Mannheim, 1998.
5. C. Baier, E. Clarke, V. Hartonas-Garmhausen, M. Kwiatkowska, and M. Ryan. Symbolic model checking for probabilistic processes. In *ICALP*, LNCS 1256, pp. 430-440, 1997.

6. C. Baier and M. Kwiatkowska. Model checking for a probabilistic branching-time logic with fairness. *Distr. Comp.*, **11**(3): 125–155, 1998.
7. D. Beauquier and A. Slissenko. Polytime model checking for timed probabilistic computation tree logic. *Acta Inf.*, **35**: 645–664, 1998.
8. A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *FSTTCS*, LNCS 1026, pp. 499–513, 1995.
9. R. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. on Comp.*, **C-35**(8): 677-691, 1986.
10. W. Chan, R. Anderson, P. Beame, S. Burns, F. Modugno, D. Notkin and J.D. Reese. Model checking large software specifications. *IEEE Trans. on Softw. Eng.*, **24**(7): 498–519, 1998.
11. I. Christoff and L. Christoff. Reasoning about safety and liveness properties for probabilistic systems. In *FSTTCS*, LNCS 652, pp 342-355, 1992.
12. E. Clarke, M. Fujita, P. McGeer, J. Yang and X. Zhao. Multi-terminal binary decision diagrams: an efficient data structure for matrix representation. In *Proc. IEEE Int. Workshop on Logic Synthesis*, pp. 1–15, 1993.
13. E. Clarke, O. Grumberg and D. Long. Verification tools for finite-state concurrent programs. In *A Decade of Concurrency*, LNCS 803, pp. 124–175, 1993.
14. C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In *FOCS*, pp. 338–345, 1988.
15. C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *J. ACM*, **42**(4): 857–907, 1995.
16. L. de Alfaro. How to specify and verify the long-run average behavior of probabilistic systems. In *LICS*, 1998.
17. L. de Alfaro. Stochastic transition systems. In *CONCUR*, LNCS 1466, pp. 423–438, 1998.
18. G. Hachtel, E. Macii, A. Padro and F. Somenzi. Markovian analysis of large finite-state machines. *IEEE Trans. on Comp. Aided Design of Integr. Circ. and Sys.*, **15**(12): 1479–1493, 1996.
19. H. Hansson and B. Jonsson. A logic for reasoning about time and probability. *Form. Asp. of Comp.*, **6**: 512–535, 1994.
20. V. Hartonas-Garmhausen, S. Campos and E.M. Clarke. ProbVerus: probabilistic symbolic model checking. In *Formal Methods for Real-Time and Probabilistic Systems (ARTS)*, LNCS 1601, pp. 96–110, 1999.
21. H. Hermanns. *Interactive Markov Chains.* Ph.D thesis, U. Erlangen-Nürnberg, 1998.
22. H. Hermanns and J.-P. Katoen. Automated compositional Markov chain generation for a plain-old telephone system. *Sci. of Comp. Programming*, 1999.
23. A. Pnueli, L. Zuck. Probabilistic verification. *Inf. and Comp.*, **103,**: 1–29, 1993.
24. W. Press, B. Flannery, S. Teukolsky and W. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge Univ. Press, 1989.
25. W. Stewart. *Introduction to the Numerical Solution of Markov Chains.* Princeton Univ. Press, 1994.
26. K.S. Trivedi, J.K. Muppala, S.P. Woolet, and B.R. Haverkort. Composite performance and dependability analysis. *Performance Evaluation*, **14**: 197–215, 1992.
27. M.Y. Vardi. Automatic verification of probabilistic concurrent finite state programs. In *FOCS*, pp 327–338, 1985.