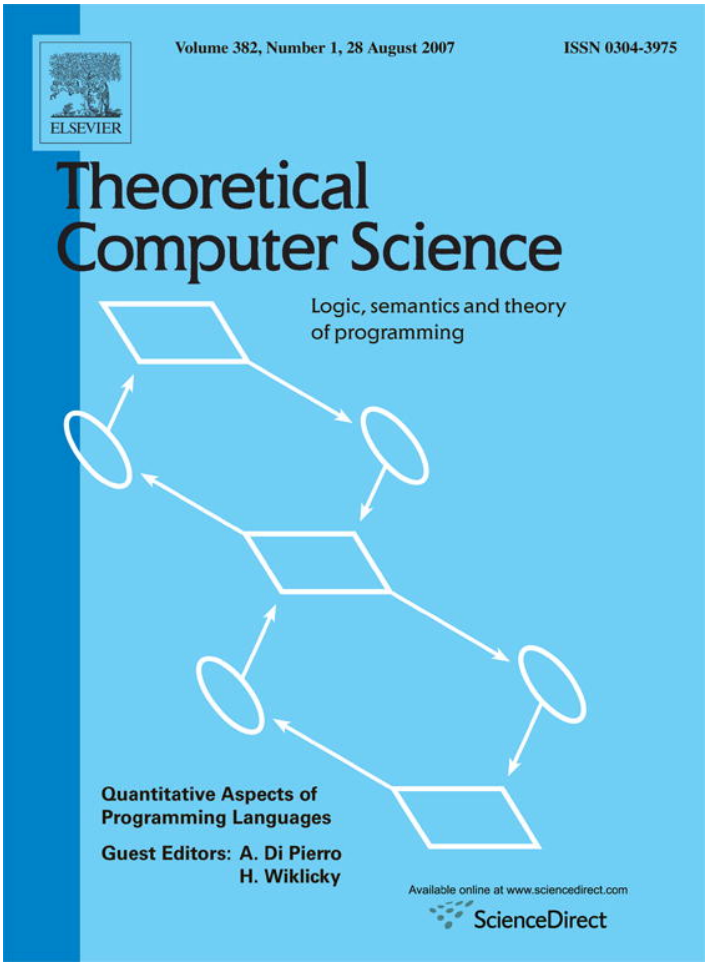


Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 382 (2007) 24–41
Theoretical Computer Science
www.elsevier.com/locate/tcs

CSL model checking algorithms for QBDs☆

Anne Remke*, Boudewijn R. Haverkort, Lucia Cloth

University of Twente, Faculty for Electrical Engineering, Mathematics and Computer Science, PO Box 217, 7500 AE Enschede, The Netherlands

Abstract

We present an in-depth treatment of model checking algorithms for a class of infinite-state continuous-time Markov chains known as quasi-birth death processes. The model class is described in detail, as well as the logic CSL to express properties of interest. Using a new property-independency concept, we provide model checking algorithms for all the CSL operators. Special emphasis is given to the time-bounded until operator for which we present a new and efficient computational procedure named uniformization with representatives. By the use of an application-driven dynamic stopping criterion, the algorithm stops whenever the property to be checked can be certified (or falsified). A comprehensive case study of a connection management system shows the versatility of our new algorithms.

Keywords: CSL; Model checking; Infinite-state; Quasi-birth death processes; Continuous-time Markov chains; Uniformization; Transient analysis

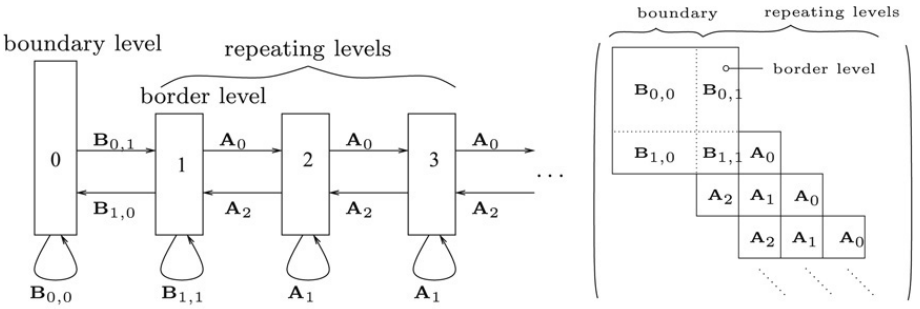
1. Introduction

Continuous-time Markov chains (CTMCs) have been used widely for Modeling, performance, and dependability evaluations of computer and communication systems. CTMCs are well understood and mathematically attractive, while at the same time being flexible enough to model complex systems. The logic CSL [3,6] has been proposed as a stochastic extension of CTL to express quantitative properties on CTMCs. Efficient computational algorithms have been developed for checking these models against formally specified properties expressed in these logics, cf. [5,6], as well as supporting tools, cf. PRISM [25] and ETMC^2 [20], and, recently, also the APNN toolbox [9]. Other tools, like GreatSPN [11] are being used as front-ends to model checking tools like PRISM and MRMC [22]. So far, the work on model checking continuous-time Markov chains has focused on *finite*-state models. However, there are many applications for which infinite-state models are more appropriate: think of modeling systems with unbounded buffers, of models including variables, or of approximating the behavior of very large-but-finite systems. Model checking CSL properties on *general* infinite-state CTMCs is, however, beyond reach. Therefore, we restrict the model class

☆ Anne Remke is supported through the MC = MC project (612.000.311) and Lucia Cloth is supported through the MOQS project (642.000.505), both financed by the Netherlands Organization for Scientific Research (NWO). We would like to thank the anonymous reviewers for their constructive comments.

* Corresponding address: University of Twente, Design and Analysis of Communication Systems, Fac. of Electrical Engineering, Mathematics and Computer Science, PO Box 217, 7500 AE Enschede, The Netherlands. Tel.: +31 53 489 3612.
E-mail addresses: anne@ewi.utwente.nl (A. Remke), brh@ewi.utwente.nl (B.R. Haverkort), lucia@ewi.utwente.nl (L. Cloth).

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2007.05.007



(a) levels in a QBD (b) generator matrix $\mathbf{Q}$

Fig. 1. Regular structure of QBDs.

to so-called *quasi-birth-death models* (QBDs), cf. [28]; QBDs comprise a very versatile yet well-understood class of infinite-state CTMCs. It is not necessary to specify QBDs manually at the state level, as high-level specifications, like, e.g., *infinite stochastic Petri nets*, do exist [29].

In this paper, we provide a complete description of CSL model checking algorithms for QBDs. We show that the syntax and semantics of CSL as for the finite case apply here as well. To facilitate the model checking algorithms, we introduce a new independency concept for CSL formulas, called *level independence as of level k*, and show how the CSL operators affect this level independence. For model checking the steady-state and the probabilistic operator of CSL, we have to develop new algorithms. For the steady-state operator, we have to compute steady-state probabilities for QBDs; we can resort to well-known algorithms for that purpose. However, for model checking the time-bounded until operator of CSL, we also need efficient algorithms for the transient analysis of QBDs, for any possible starting state. This can be done with a new and efficient uniformization-based method, called *uniformization with representatives* [34], which we present in the context of model checking in this paper. The current paper extends our conference contribution [32] by providing this new and efficient algorithm for computing *all* transient state probabilities in a QBD, in the context of CSL model checking. Furthermore, we explain how to model check the CSL until operator with all its different time intervals.

The paper is organized as follows. We introduce labeled infinite-state CTMCs, and QBDs in particular, in Section 2. We then describe the syntax and semantics of CSL in Section 3. Section 4 addresses in detail the model checking algorithms for the CSL operators. We present an efficient algorithm to compute all required transient state probabilities in Section 5. We provide an elaborate case study showing the versatility of the approach in Section 6, as well as detailed links to related work in Section 7 before the paper is concluded in Section 8.

2. Quasi birth death processes

A **labeled QBD** $\mathcal{Q}$ of order (N_0, N) (with $N_0, N \in \mathbb{N}^+$) is a labeled infinite-state continuous-time Markov chain. From a fixed set AP of atomic propositions, the labeling function $L : S \to 2^{AP}$ assigns to each state the set of valid atomic propositions in that state. The infinite state space of a QBD can be viewed as a two-dimensional strip, which is finite in one dimension and infinite in the other. The states in this strip are grouped in so-called *levels*, according to their identity in the infinite dimension. Fig. 1(a) gives a graphical representation of a QBD. Transitions, represented by positive entries in the generator matrix $\mathbf{Q}$, can only occur between states of the same level or between states of neighboring levels. All repeating levels have the same inter-level and intra-level transition structure.

The set of states S can be partitioned into an infinite number of finite sets S^j, $j = \{0, 1, \ldots\}$, each containing the states of one level, such that $S = \bigcup_{j=0}^{\infty} S^j = \{0, \ldots, N_0 - 1\} \times \{0\} \cup \{0, \ldots, N - 1\} \times \mathbb{N}^+$, where the first part represents the boundary level with N_0 states, and the second part the infinite number of repeating levels, each with N states. We call the first repeating level the border level. Two states (i_1, j_1) and (i_2, j_2) are *corresponding states*, if $i_1 = i_2$ and $j_1, j_2 > 0$.

The block-tri-diagonal generator matrix $\mathbf{Q}$ consists of the following finite matrices describing the inter- and intra-level transitions, as shown in Fig. 1(b):

$\mathbf{B}_{0,0} \in \mathbb{R}^{N_0 \times N_0}$: intra-level transition structure of the boundary level,
$\mathbf{B}_{0,1} \in \mathbb{R}^{N_0 \times N}$: inter-level transitions from the boundary to the border level,
$\mathbf{B}_{1,0} \in \mathbb{R}^{N \times N_0}$: inter-level transitions from the border to the boundary level,

$\mathbf{B}_{1,1} \in \mathbb{R}^{N\times N}$: intra-level transition structure of the border level.
$\mathbf{A}_0 \in \mathbb{R}^{N\times N}$: inter-level transitions from one repeating level to the next higher repeating level,
$\mathbf{A}_1 \in \mathbb{R}^{N\times N}$: intra-level transitions for the repeating levels,[1] and
$\mathbf{A}_2 \in \mathbb{R}^{N\times N}$: inter-level transitions from one repeating level to the next lower repeating level.

The states of each level S^i for $i > 0$ are divided into three not necessarily disjoint sets of states: $S^i = S^{i,\uparrow}_{in} \cup S^{i,\uparrow}_{center} \cup S^{i,\uparrow}_{out}$. The set $S^{i,\uparrow}_{in}$ comprises the states that can be reached from the next lower level in one step, $S^{i,\uparrow}_{center}$ comprises the states from which level $i+1$ cannot be reached in one step, and $S^{i,\uparrow}_{out}$ comprises the states from which the next higher level can be reached in one step. Similarly, we define $S^{i,\downarrow}_{in}$ to comprise the states that can be reached from the next higher level in one step, $S^{i,\downarrow}_{center}$ to comprise the states from which level $i-1$ cannot be reached in one step and $S^{i,\downarrow}_{out}$ to comprise all states from which the next lower level can be reached in one step. Note that for the boundary level we define $S^0 = S^{0,\uparrow}_{center} \cup S^{0,\uparrow}_{out}$ and $S^0 = S^{0,\downarrow}_{center} \cup S^{0,\downarrow}_{in}$. The minimum number of steps that has to be undertaken to reach s_2 from s_1 is given by $g(s_1, s_2) = |shortestpath(s_1, s_2)|$. Let $d^\uparrow \geq 1$ be the so-called *upward level diameter*, that is, the minimum number of state transitions needed to reach the next higher repeating level from a state in $S^{i,\uparrow}_{in}$: $d^\uparrow = \min\{g(s_1, s_2) \mid s_1 \in S^{i,\uparrow}_{in}, s_2 \in S^{i+1,\uparrow}_{in}\}$. The *downward level diameter* $d^\downarrow$ is defined along the same lines as $d^\downarrow = \min\{g(s_1, s_2) \mid s_1 \in S^{i,\downarrow}_{in}, s_2 \in S^{i-1,\downarrow}_{in}\}$. We define d, the *symmetric level diameter*, as the minimum of the upward and downward level diameter. As the repeating levels of a QBD all exhibit the same structure, they all have the same level diameter. However, the number of steps needed to cross l levels may be larger than $l \cdot d$, depending on the structure of the QBD.

An *infinite path* σ is a sequence $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} s_2 \xrightarrow{t_2} \cdots$ with, for $i \in \mathbb{N}$, $s_i \in S$ and $t_i \in \mathbb{R}_{>0}$ such that $\mathbf{Q}(s_i, s_{i+1}) > 0$ for all i. A *finite path* σ of length $l+1$ is a sequence $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \cdots s_{l-1} \xrightarrow{t_{l-1}} s_l$ such that s_l is absorbing,[2] and $\mathbf{Q}(s_i, s_{i+1}) > 0$ for all $i < l$. For an infinite path σ, $\sigma[i] = s_i$ denotes for $i \in \mathbb{N}$ the $(i+1)$st state of path σ. The time spent in state s_i is denoted by $\delta(\sigma, i) = t_i$. Moreover, with i the smallest index with $t \leq \sum_{j=0}^{i} t_j$, let $\sigma@t = \sigma[i]$ be the state occupied at time t. For finite paths σ with length $l+1$, $\sigma[i]$ and $\delta(\sigma, i)$ are defined in the way described above for $i < l$ only and $\delta(\sigma, l) = \sigma[l] = \infty$ and $\delta@t = s_l$ for $t > \sum_{j=0}^{l-1} t_j$. $Path^{\mathcal{Q}}(s)$ is the set of all finite and infinite paths of the QBD $\mathcal{Q}$ that start in state s and $Path^{\mathcal{Q}}$ includes all (finite and infinite) paths of the QBD $\mathcal{Q}$.

As for finite CTMCs, a probability measure on paths can now be defined, depending on the starting state [6]. Starting from there, two different types of state probabilities can be distinguished for the QBDs. The **transient state probability** is a time-dependent measure that considers the QBD at a given time instant t. The probability of being in state s' at time instant t, given initial state s, is denoted as $\mathbf{V}^{\mathcal{Q}}(s, s', t) = \Pr(\sigma \in Path^{\mathcal{Q}}(s) \mid \sigma@0 = s \wedge \sigma@t = s')$. The transient probabilities are characterized by a linear system of differential equations of infinite size. Let $\mathbf{V}(t)$ be the matrix of the transient state probabilities at time t for all possible starting states s and for all possible goal states s' (we omit the superscript $\mathcal{Q}$ for brevity here); then we have $\mathbf{V}'(t) = \mathbf{V}(t) \cdot \mathbf{Q}$. Using a standard differential equation solver is impossible, since the number of differential equations is infinite. Later in this paper, we propose a technique called uniformization with representatives, which deals in an efficient way with this differential equation system of infinite size. The **steady-state probabilities** to be in state s', given initial state s, are then defined as $\pi^{\mathcal{Q}}(s, s') = \lim_{t\to\infty} \mathbf{V}^{\mathcal{Q}}(s, s', t)$, and indicate the probabilities to be in some state s' "in the long run". If a steady-state is reached, the above mentioned derivatives will approach zero. Furthermore, if the QBD is ergodic, the initial state does not influence the steady-state probabilities (we therefore often write $\pi(s')$ instead of $\pi(s, s')$ for brevity). The steady-state probability vector $\underline{\pi}$ then follows from the infinite system of linear equations $\underline{\pi} \cdot \mathbf{Q} = \underline{0}$, and $\sum_s \pi_s = 1$ (normalization). This system of equations can be solved using so-called matrix–geometric methods which exploit the repetitive structure in the matrix $\mathbf{Q}$. Details on these methods in general can be found in [28] and in a model checking context in [31].

3. The logic CSL

We apply the logic CSL [6] on the QBDs. The syntax and semantics are the same, with the only difference being that we now interpret the formulas over states and paths of the QBDs. Let $p \in [0, 1]$ be a real number, $\bowtie \in \{\leq, <, >, \geq\}$

[1] Note that $\mathbf{B}_{1,1}$ differs from $\mathbf{A}_1$ only in the diagonal entries.
[2] A state s is called absorbing if for all s' the rate $\mathbf{Q}(s, s') = 0$.

a comparison operator, $t_1, t_2, t \in \mathbb{R}^+$ real numbers and AP a set of atomic propositions with $ap \in AP$. CSL state formulas Φ are defined by

$$\Phi ::= \mathtt{tt} \mid ap \mid \neg\Phi \mid \Phi \wedge \Phi \mid \mathcal{S}_{\bowtie p}(\Phi) \mid \mathcal{P}_{\bowtie p}(\phi),$$

where ϕ is a CSL path formula[3] constructed by

$$\phi ::= \mathcal{X}^{[t_1,t_2]}\Phi \mid \Phi\, \mathcal{U}^{[t_1,t_2]}\Phi.$$

For a CSL state formula Φ on a QBD $\mathcal{Q}$, the satisfaction set $Sat(\Phi)$ contains all states of $\mathcal{Q}$ that fulfill Φ. The satisfaction set can be considered as the infinite union of finite *level satisfaction sets* $Sat(\Phi) = \bigcup_{j=0}^{\infty} Sat^j(\Phi)$. The level satisfaction set $Sat^j(\Phi)$ contains only those Φ-states that are situated in level j. Satisfaction is stated in terms of a satisfaction relation $\models$. The relation $\models$ for states and CSL state formulas is defined as:

$s \models \mathtt{tt}$	for all $s \in S$,	$s \models \Phi \wedge \Psi$	iff $s \models \Phi$ and $s \models \Psi$,
$s \models ap$	iff $ap \in L(s)$,	$s \models \mathcal{S}_{\bowtie p}(\Phi)$	iff $\pi^{\mathcal{Q}}(s, Sat(\Phi)) \bowtie p$,
$s \models \neg\Phi$	iff $s \not\models \Phi$,	$s \models \mathcal{P}_{\bowtie p}(\phi)$	iff $Prob^{\mathcal{Q}}(s, \phi) \bowtie p$,

where $\pi^{\mathcal{Q}}(s, Sat(\Phi)) = \sum_{s' \in Sat(\Phi)} \pi^{\mathcal{Q}}(s, s')$, and $Prob^{\mathcal{Q}}(s, \phi)$ describes the probability measure of all paths $\sigma \in Path(s)$ that satisfy ϕ when the system is starting in state s, that is, $Prob^{\mathcal{Q}}(s, \phi) = \Pr\{\sigma \in Path^{\mathcal{Q}}(s) \mid \sigma \models \phi\}$.

The relation $\models$ for paths and CSL path formulas is defined as:

$$\sigma \models \mathcal{X}^{[t_1,t_2]}\Phi \quad \text{iff } \sigma[1] \text{ is defined and } \sigma[1] \models \Phi \text{ and } t_1 \le \delta(\sigma, 0) \le t_2,$$

$$\sigma \models \Phi\, \mathcal{U}^{[t_1,t_2]}\Psi \quad \text{iff } \exists t (t_1 \le t \le t_2)\ (\sigma @ t \models \Psi \wedge (\forall t' \in [0, t)(\sigma @ t' \models \Phi))).$$

The steady-state operator $\mathcal{S}_{\bowtie p}(\Phi)$ denotes that the steady-state probability for Φ-states meets the bound p. $\mathcal{P}_{\bowtie p}(\phi)$ asserts that the probability measure of the paths satisfying ϕ meets the bound p. The next operator $\mathcal{X}^{[t_1,t_2]}\Phi$ states that a transition to a Φ-state is made during the time interval $[t_1, t_2]$. The until operator $\Phi\, \mathcal{U}^{[t_1,t_2]}\Psi$ asserts that Ψ is satisfied at some time instant in between $[t_1, t_2]$, and that at all preceding time instants Φ holds.

4. Algorithms for CSL model checking QBDs

This section addresses in detail the model checking algorithms for the CSL operators. In 4.1, we explain our restriction of level independent atomic properties, in 4.2, we discuss whether the CSL formulas are level independent in general, and 4.3 covers the level independence of atomic propositions and logical operators. In Section 4.4, we discuss how to model check the steady-state operator, and in Section 4.5 how to model check the next operator. Section 4.6 then covers the until operator with its different time intervals.

4.1. Level independent atomic properties

In the following, we limit ourselves to strongly connected QBDs with so-called *level independent* atomic propositions. That is, if an atomic proposition $ap \in AP$ is valid in a certain state of an arbitrary repeating level, it has to be valid in the corresponding states of all repeating levels. This limitation poses a true restriction on the set of formulas we are able to check. In practice, this means that atomic propositions must not refer to the level index in order to be level independent. Let $i \in \{0, \ldots, N-1\}$, be an atomic proposition $ap \in AP$ is *level independent* if and only if for all $l, k \ge 1$, $L(i, k) = L(i, l)$.

In order to develop efficient CSL model checking algorithms for QBDs, we need to exploit the connection between the validity of state formulas and the special structure of QBDs. At first glance one could think that in corresponding states of all repeating levels, the same CSL formulas hold. Unfortunately this is not the case, which can easily be seen when considering the time-bounded next operator. In the border level, other next-formulas might be satisfied than in the other repeating levels, because the boundary level is still reachable from the border level, but not from any other repeating level. Thus, if we want to check for example the formula $\phi = \mathcal{X}^{[t_1,t_2]}red$ and the property *red*

[3] We comment on the different time intervals in Section 4.6.

is only valid in the boundary level, this property ϕ can be fulfilled by a path starting in the border level, but not when starting in any other repeating level. A similar reasoning holds for the until operator, where not only the border level is concerned but even more repeating levels, because with the until operator not just one step is considered, but potentially an infinite number. Thus, no two repeating levels can *a priori* be considered to satisfy the same path-formulas.

4.2. Level independence of CSL formulas

Even though the CSL formulas are not level independent in general, their validity does not change arbitrarily between levels. Remember that we assume level independence of atomic propositions for the QBDs we consider. For the CSL formulas, we generalize the idea of level independence: we show that the validity in a state is level independent for repeating levels with an index of at least k for some $k > 0$. Thus, the validity of a CSL formula changes between corresponding states of repeating levels, but only up to repeating level $k-1$. From level k onwards, the validity remains unchanged.

Let $\mathcal{Q}$ be a QBD of order (N_0, N). A CSL state formula Φ is *level independent as of level* $k \geq 1$ (in QBD $\mathcal{Q}$) if and only if for levels above and including k, the validity of Φ in a state does not depend on the level; that is, for all $i \in \{0, \ldots, N-1\}$ and for all $l \geq k : (i, l) \models \Phi \iff (i, k) \models \Phi$.

The following proposition states, under the assumption of level independent atomic propositions, that such a k exists for any CSL state formula. We will justify this proposition inductively over the structure of the logic in the sections that discuss the model checking of the different types of CSL state formulas. Note that the requirement for level-independent atomic propositions is not necessary. In case the atomic propositions are level independent as of level k, we just extend the boundary level to the first $k-1$ repeating levels.

Proposition 1. *Let $\mathcal{Q}$ be a QBD with level independent atomic propositions, and let Φ be a CSL state formula other than $\mathcal{P}_{\bowtie p}(\Phi\, \mathcal{U}^I\, \Psi)$. Then there exists a $k \in \mathbb{N}$, such that Φ is level independent as of level k in $\mathcal{Q}$.*

For the until operator $\mathcal{P}_{\bowtie p}(\Phi\, \mathcal{U}^I\, \Psi)$, we require that for no state s is the probability measure exactly equal to p; hence, $\mathrm{Prob}(s, \Phi \mathcal{U}^I\, \Psi) \neq p$. *Under this assumption, there exists a $k \in \mathbb{N}$, such that $\mathcal{P}_{\bowtie p}(\Phi\, \mathcal{U}^I\, \Psi)$ is level independent as of level k in $\mathcal{Q}$.* □

For model checking a property Φ, we compute the set $Sat(\Phi)$ with a recursive descent procedure over the parse tree of Φ. For a state formula Φ that is level independent as of level k, only the first k level satisfaction sets have to be computed. $Sat^k(\Phi)$ then acts as a representative for all following levels.

4.3. Atomic propositions and logical operators

Computing the satisfaction set for an atomic proposition ap proceeds as follows. $Sat^0(ap)$ consists of those states of the boundary level where ap is contained in the labeling. We model check all states in the border level in order to obtain $Sat^1(ap)$, and, hence, $Sat^j(ap)$ for $j \geq 1$. Let Φ be a CSL state formula that is level independent as of level k. Its negation $\neg\Phi$ is clearly also level independent as of level k. The level satisfaction sets of $\neg\Phi$ are computed by complementing the corresponding satisfaction set of Φ: $Sat^j(\neg\Phi) = S^j \backslash Sat^j(\Phi)$, for all $j \geq 0$. Let Φ and Ψ be two CSL state formulas, level independent as of level k_Φ and k_Ψ, respectively. The conjunction $\Phi \wedge \Psi$ is level independent as of level $\max(k_\Phi, k_\Psi)$. The level satisfaction sets are computed by intersecting the corresponding satisfaction sets of Φ and Ψ: $Sat^j(\Phi \wedge \Psi) = Sat^j(\Phi) \cap Sat^j(\Psi)$, for all $j \geq 0$.

4.4. Steady-state operator

A state s satisfies $\mathcal{S}_{\bowtie p}(\Phi)$ if the sum of the steady-state probabilities of all Φ-states reachable from s meets the bound p. Since we assume a strongly connected QBD, the steady-state probabilities are independent of the starting state. It follows that either all states satisfy a steady-state formula or none of the states does, which implies that a steady-state formula is always level independent as of level 1. We first determine the satisfaction set $Sat(\Phi)$ and then compute the accumulated steady-state probability. If the accumulated steady-state probability meets the bound p, we

have $Sat(\mathcal{S}_{\bowtie p}(\Phi)) = S$; otherwise, $Sat(\mathcal{S}_{\bowtie p}(\Phi)) = \varnothing$. Exploiting the special structure of QBDs, the accumulated probability is given by

$$\pi(Sat(\Phi)) = \sum_{s \in Sat(\Phi)} \pi(s) = \sum_{j=0}^{\infty} \sum_{s \in Sat^j(\Phi)} \underline{\pi}_j(s),$$

where the vectors $\underline{\pi}_j = (\ldots, \underline{\pi}_j(s), \ldots)$ can be computed one after the other, using the matrix–geometric method, cf. [28]. In essence, $\underline{\pi}_j = \underline{\pi}_{j-1} \cdot \mathbf{R}$ with $\mathbf{R}$ a square matrix that follows from $\mathbf{A}_0\mathbf{R}^2 + \mathbf{A}_1\mathbf{R} + \mathbf{A}_2 = 0$.

In order to deal with the infinite summation, we iterate through the repeating levels and accumulate the steady-state probabilities in a level-wise fashion. We denote with $\tilde{\pi}^l(Sat(\Phi))$ the accumulated steady-state probabilities of all Φ-states up to level l, that is,

$$\tilde{\pi}^l(Sat(\Phi)) = \sum_{j=0}^{l} \sum_{s \in Sat^j(\Phi)} \underline{\pi}_j(s).$$

Starting with $l = 0$, we compute $\tilde{\pi}^l(Sat(\Phi))$ and $\tilde{\pi}^l(Sat(\neg\Phi))$, respectively. The computation of the steady-state probabilities of $\neg\Phi$-states introduces no additional cost, since we have to compute the whole vector $\underline{\pi}_j$ anyway. At every step, we have to check whether we can already decide on the validity of the steady-state formula $\mathcal{S}_{\bowtie p}(\Phi)$. The following implications hold:

(a) $\tilde{\pi}^l(Sat(\Phi)) > p \quad \Rightarrow \quad \pi(Sat(\Phi)) > p,$
(b) $\tilde{\pi}^l(Sat(\neg\Phi)) > 1 - p \quad \Rightarrow \quad \pi(Sat(\Phi)) < p.$

As soon as one of the left-hand side inequalities becomes true, we can stop. For the interpretation, we distinguish the cases $\mathcal{S}_{<p}(\Phi)$ and $\mathcal{S}_{>p}(\Phi)$. For $\mathcal{S}_{<p}(\Phi)$ the interpretation is as follows. If inequality (a) holds, the condition $\pi(Sat(\Phi)) < p$ is clearly not accomplished and $Sat(\mathcal{S}_{<p}(\Phi)) = \varnothing$. If inequality (b) holds, the condition $\pi(Sat(\Phi)) < p$ is accomplished and $Sat(\mathcal{S}_{<p}(\Phi)) = S$. For $\mathcal{S}_{>p}(\Phi)$ the same conditions need to be checked in every iteration step l, but they need to be interpreted differently; if inequality (a) holds, the probability bound is met and $Sat(\mathcal{S}_{>p}(\Phi)) = S$. If inequality (b) holds, the bound is not met and $Sat(\mathcal{S}_{>p}(\Phi)) = \varnothing$. For $\mathcal{S}_{\geq p}(\Phi)$ or $\mathcal{S}_{\leq p}(\Phi)$ the equations need to be modified accordingly.

The satisfaction set of Φ might be finite. For a CSL formula Φ that is level independent as of level k, this is the case when no state in level k satisfies Φ. The iteration then ends at level $k-1$ and $\pi(Sat(\Phi)) = \tilde{\pi}^{k-1}(Sat(\Phi))$. In case $Sat(\Phi)$ is infinite, the iteration stops as soon as one of the inequalities is satisfied. Unfortunately, if the bound p is exactly equal to the steady-state probability $\pi(Sat(\Phi))$, the approximations $\tilde{\pi}^l(Sat(\Phi))$ and $\tilde{\pi}^l(Sat(\neg\Phi))$ will never fulfill one of the inequalities. In an implementation of this algorithm, some care must be taken to detect this case in order to avoid a non-stopping iteration.

Instead of the just-sketched iterative process, we can also develop a closed-form matrix expression for the probability $\pi(Sat(\Phi))$ by exploiting properties of the matrix–geometric solution, i.e., by using the fact that $\sum_i \mathbf{R}^i = (\mathbf{I} - \mathbf{R})^{-1}$. In doing so, the infinite summation disappears; however, this comes at the cost of a matrix inversion. In practice, this is therefore not always a more efficient approach, but it avoids the stopping problem.

4.5. Time-bounded next operator

Recall that a state s satisfies $\mathcal{P}_{\bowtie p}(\mathcal{X}^{[t_1,t_2]}\Phi)$ if the one-step probability to reach a state that fulfills Φ within a time $t \in [t_1, t_2]$, outgoing from s meets the bound p; that is,

$$s \models \mathcal{P}_{\bowtie p}(\mathcal{X}^{[t_1,t_2]}\Phi) \Leftrightarrow \Pr\{\sigma \in Path(s) \mid \sigma \models \mathcal{X}^{[t_1,t_2]}\Phi\} \bowtie p$$
$$\Leftrightarrow \left(\left(e^{\mathbf{Q}(s,s)\cdot t_1} - e^{\mathbf{Q}(s,s)\cdot t_2} \right) \cdot \sum_{\substack{s' \in Sat(\Phi) \\ s' \neq s}} \frac{\mathbf{Q}(s,s')}{-\mathbf{Q}(s,s)} \right) \bowtie p, \tag{1}$$

where $e^{Q(s,s)\cdot t_1} - e^{Q(s,s)\cdot t_2}$ is the probability of residing at s for a time $t \in [t_1, t_2]$, and $\frac{Q(s,s')}{-Q(s,s)}$ specifies the probability to step from state s to state s'. Note that the above inequality contains a summation over all Φ-states. We only need to

sum over the states of $Sat(\Phi)$ that are reachable from s in one step. That is, for $s = (i, j)$, we only have to consider the Φ-states from levels $j-1$, j, and $j+1$; the one-step probabilities for all other states are zero, thus making this summation finite.

Now, let the inner formula Φ of the next-formula be level independent as of level k. Hence, the validity of the state formula $\mathcal{P}_{\bowtie p}(\mathcal{X}^{[t_1,t_2]}\,\Phi)$ might be different in corresponding states for all levels up to $k-1$. Therefore, unfortunately, level k can still have different states satisfying $\mathcal{P}_{\bowtie p}(\mathcal{X}^{[t_1,t_2]}\,\Phi)$, since level $k-1$ is reachable in one step. But, as of level $k+1$, only those levels can be reached where the validity of the state formula Φ is equal for the corresponding states. Hence, if Φ is level independent as of level k, $\mathcal{P}_{\bowtie p}(\mathcal{X}^{[t_1,t_2]}\,\Phi)$ is level independent as of level $k+1$. For the construction of the satisfaction set of such a formula, we therefore have to compute explicitly the satisfying states up to level $k+1$. Subsequently, $Sat^{k+1}(\mathcal{P}_{\bowtie p}(\mathcal{X}^{[t_1,t_2]}\,\Phi))$ can be seen as a representative for all following repeating levels.

4.6. Time-bounded until operator

For model checking $\mathcal{P}_{\bowtie p}(\Phi\ \mathcal{U}^I\ \Psi)$, we adopt the general approach for finite CTMCs [6]. The idea is to use a transformed QBD where several states are made absorbing. Recall, that the CSL path formula $\varphi = \Phi\ \mathcal{U}^I\ \Psi$ is valid if a Ψ-state is reached on a path during the time interval I via only Φ-states. We discuss model checking the until operator for the interval $[0, t]$ in Section 4.6.1, for the interval $[t_1, t_2]$ in Section 4.6.2, for the interval $[0, \infty)$ in Section 4.6.3 and for the interval $[t, \infty)$ in Section 4.6.4. Section 4.6.5 presents the connection between these four cases and the involved numerical algorithms to be discussed in Section 5.

4.6.1. Case $I = [0, t]$

First, we restrict the time interval to a time interval $I = [0, t]$. In this case, the future behavior of the QBD is irrelevant for the validity of φ, as soon as a Ψ-state is reached. Thus, all Ψ-states can be made absorbing without affecting the satisfaction set of formula φ. On the other hand, as soon as a $(\neg\Phi \wedge \neg\Psi)$-state is reached, φ will be invalid, regardless of the future evolution. As a result, we may switch from checking the Markov chain $\mathcal{Q}$ to checking the Markov chain $\mathcal{Q}[\Psi][\neg\Phi \wedge \neg\Psi] = \mathcal{Q}[\neg\Phi \vee \Psi]$, where all states satisfying the formula in $[\cdot]$ are made absorbing. Model checking a formula involving the until operator then reduces to calculating the transient probabilities $\pi^{\mathcal{Q}[\neg\Phi\vee\Psi]}(s, s', t)$ for all Ψ-states s'. Exploiting the special structure of QBDs yields

$$s \models \mathcal{P}_{\bowtie p}(\Phi\,\mathcal{U}^{[0,t]}\,\Psi) \Leftrightarrow Prob^{\mathcal{Q}}(s,\ \Phi\,\mathcal{U}^{[0,t]}\,\Psi) \bowtie p \Leftrightarrow \left(\sum_{i=0}^{\infty} \sum_{s'\in Sat^i(\Psi)} \pi^{\mathcal{Q}[\neg\Phi\vee\Psi]}(s, s', t)\right) \bowtie p.$$

4.6.2. Case $I = [t_1, t_2]$

Considering a time interval $[t_1, t_2]$ with $0 < t_1 < t_2$, we can split the computation in two parts. The first part then addresses the path from the starting state s to a Φ-state s' at time t_1 via only Φ states. The second part of the computation addresses the path from s' to a Ψ-state s'' via only Φ states. This leads us to two transformed QBDs: $\mathcal{Q}[\neg\Phi]$ that is used in the first part and $\mathcal{Q}[\neg\Phi \vee \Psi]$ in the second part. To calculate the probability for such a path, we accumulate the multiplied transition probabilities for all triples (s, s', s''), where $s' \models \Phi$ is reached before time t_1 and $s'' \models \Psi$ is reached before time $t_2 - t_1$. This can be done, because the QBDs are time homogeneous.

$$s \models \mathcal{P}_{\bowtie p}(\Phi\,\mathcal{U}^{[t_1,t_2]}\,\Psi) \Leftrightarrow Prob^{\mathcal{Q}}(s,\ \Phi\,\mathcal{U}^{[t_1,t_2]}\,\Psi) \bowtie p$$
$$\Leftrightarrow \left(\sum_{i=0}^{\infty} \sum_{s'\in Sat^i(\Phi)} \sum_{j=0}^{\infty} \sum_{s''\in Sat^j(\Psi)} \pi^{\mathcal{Q}[\neg\Phi]}(s, s', t_1) \cdot \pi^{\mathcal{Q}[\neg\Phi\vee\Psi]}(s', s'', t_2 - t_1)\right) \bowtie p.$$

4.6.3. Case $I = [0, \infty)$

For the unbounded case (interval $[0, \infty)$), the probability $Prob^{\mathcal{Q}}(s,\ \Phi\mathcal{U}^{[0,\infty)}\,\Psi)$ equals the probability of eventually reaching a Ψ-state. Since these Ψ-states are absorbing, this is exactly the steady-state probability to be in a Ψ-state,

so we have

$$s \models \mathcal{P}_{\bowtie p}(\Phi \mathcal{U}^{[0,\infty)} \Psi) \Leftrightarrow Prob^{\mathcal{Q}}(s, \Phi \mathcal{U}^{[0,\infty)} \Psi) \bowtie p$$
$$\Leftrightarrow \pi^{\mathcal{Q}[\neg\Phi \vee \Psi]}(s, Sat(\Psi)) \bowtie p$$
$$\Leftrightarrow \left(\sum_{i=0}^{\infty} \sum_{s' \in Sat(\Psi)} \pi^{\mathcal{Q}[\neg\Phi \vee \Psi]}(s, s') \right) \bowtie p.$$

4.6.4. Case $I = [t, \infty)$

For the interval $[t, \infty)$ the computation is split into two parts, just as for $[t_1, t_2]$. The first part addresses the path from the starting state s to a Φ-state s' via only Φ-states at time t, whereas the second part addresses the path that eventually leads from s' to a Ψ-state. Note that we combine the transient probabilities in the transformed QBD $\mathcal{Q}[\neg\Phi]$ for the first part with the steady-state probabilities in $\mathcal{Q}[\neg\Phi \vee \Psi]$ for the second part as follows:

$$s \models \mathcal{P}_{\bowtie p}(\Phi \, \mathcal{U}^{[t,\infty)} \Psi) \Leftrightarrow Prob^{\mathcal{Q}}(s, \Phi \, \mathcal{U}^{[t,\infty)} \Psi) \bowtie p$$
$$\Leftrightarrow \left(\sum_{i=0}^{\infty} \sum_{s' \in Sat^i(\Phi)} \sum_{j=0}^{\infty} \sum_{s'' \in Sat^j(\Psi)} \pi^{\mathcal{Q}[\neg\Phi]}(s, s', t) \cdot \pi^{\mathcal{Q}[\neg\Phi \vee \Psi]}(s', s'') \right) \bowtie p.$$

4.6.5. Algorithms

The transient probability of being in each state of the infinite-state QBD for any possible initial state (as needed for the bounded until operators) can be calculated with a new iterative uniformization-based method, which we present in the next section. To calculate the satisfaction set for $\mathcal{P}_{\bowtie p}(\Phi \, \mathcal{U}^{[t_1,t_2]} \Psi)$, we need to understand how this algorithm works; therefore we postpone this discussion to Section 5.6. The algorithm for the unbounded until operator is briefly discussed in Section 5.7. The justification of Proposition 1 for the until operators is postponed to Section 5.8, as we need a better understanding of how the probabilities are calculated first.

5. Uniformization with representatives

We first describe the main principles of uniformization for the QBDs in Section 5.1. In Section 5.2 we then describe how to exploit the QBD structure to obtain a finite data representation. We address the growth of the involved data structures in Section 5.3. The actual iterative algorithm is then presented in Section 5.4 before we discuss the complexity in Section 5.5. How many steps have to be undertaken is explained in Section 5.6 and the algorithm for the unbounded until operator is briefly discussed in Section 5.7. Furthermore, we provide a proof of Proposition 1 for the until operator in Section 5.8.

5.1. Uniformization

Uniformization is a well-established technique to determine the transient-state probabilities $\mathbf{V}(t)$ in a continuous-time Markov chain via an embedded discrete-time Markov chain subordinated to a Poisson process [16]. The parameter of this Poisson process corresponds to the maximum outgoing transition rate in the CTMC. This so-called uniformization rate λ can easily be determined because $\mathbf{Q}$ has only a finite number of different diagonal entries (originating from the matrices $\mathbf{B}_{0,0}$, $\mathbf{B}_{1,1}$, and $\mathbf{A}_1$).

The probability matrix $\mathbf{P}$ for the embedded DTMC then is computed as $\mathbf{I} + \mathbf{Q}/\lambda$, and it follows the same tri-diagonal structure as $\mathbf{Q}$ (where the sub-matrices are replaced by $\hat{\mathbf{B}}_{0,0}$, $\hat{\mathbf{B}}_{0,1}$, $\hat{\mathbf{B}}_{1,0}$, $\hat{\mathbf{B}}_{1,1}$, $\hat{\mathbf{A}}_0$, $\hat{\mathbf{A}}_1$ and $\hat{\mathbf{A}}_2$, respectively). The sub-matrices are calculated as follows:

$$\hat{\mathbf{B}}_{i,j} = \begin{cases} \mathbf{I} + \frac{\mathbf{B}_{i,j}}{\lambda}, & i = j, \\ \frac{\mathbf{B}_{i,j}}{\lambda}, & i \neq j, \end{cases} \quad \text{and} \quad \hat{\mathbf{A}}_i = \begin{cases} \mathbf{I} + \frac{\mathbf{A}_i}{\lambda}, & i = 1, \\ \frac{\mathbf{A}_i}{\lambda}, & i \neq 1. \end{cases}$$

Let $\mathbf{U}^{(k)}$ be the state probability distribution matrix after k epochs in the DTMC with transition matrix $\mathbf{P}$. That is, entry (i, j) of $\mathbf{U}^{(k)}$ is the probability that j is reached from i in k steps. $\mathbf{U}^{(k)}$ can be derived recursively as:

$$\mathbf{U}^{(0)} = \mathbf{I}, \quad \text{and} \quad \mathbf{U}^{(k)} = \mathbf{U}^{(k-1)}\mathbf{P}, \quad k \in \mathbb{N}^+. \tag{2}$$

Then, the matrix of transient state probabilities for the original CTMC at time t, can be calculated as:

$$\mathbf{V}(t) = \sum_{k=0}^{\infty} \psi(\lambda t; k)\mathbf{P}^k = \sum_{k=0}^{\infty} \psi(\lambda t; k)\mathbf{U}^{(k)}, \tag{3}$$

where $\psi(\lambda t; k)$ is the probability of k events occurring in the interval $[0, t)$ in a Poisson process with rate λ. The probability distribution in the DTMC after k steps is described by $\mathbf{V}(0)\mathbf{P}^k$ (note that $\mathbf{V}(0) = \mathbf{I}$). Recall that the matrices $\mathbf{V}(t)$ and $\mathbf{U}^{(k)}$, $k \in \mathbb{N}$, have infinite size. To avoid the infinite summation over the number of steps k, the sum (3) needs to be truncated. We denote the approximation of $\mathbf{V}(t)$ that has been calculated with up to n terms of the summation with $\mathbf{V}^{(n)}(t)$:

$$\mathbf{V}^{(n)}(t) = \sum_{k=0}^{n} \psi(\lambda t; k)\mathbf{U}^{(k)}. \tag{4}$$

We can compute $\mathbf{V}^{(n+1)}(t)$ as:

$$\mathbf{V}^{(n+1)}(t) = \mathbf{V}^{(n)}(t) + \psi(\lambda t; n+1)\mathbf{U}^{(n+1)}. \tag{5}$$

Note that $\mathbf{V}^{(n)}(t)$ follows the structure of the previous $\mathbf{U}^{(m)}$ $(m \leq n)$ in terms of zeroes and non-zeroes, because any non-zero entry in $\mathbf{V}^{(n)}$ corresponds to a non-zero in $\mathbf{U}^{(m)}(m \leq n)$. We denote the maximum error that possibly occurs in an entry of $\mathbf{V}(t)$ when the series is truncated after n steps as $\varepsilon_{t,\lambda}^{(n)}$. We have

$$\varepsilon_{t,\lambda}^{(n)} = \left\| \sum_{k=n+1}^{\infty} \psi(\lambda t; k)\mathbf{U}^{(k)} \right\| \leq 1 - \sum_{k=0}^{n} \mathrm{e}^{-\lambda t} \frac{(\lambda t)^k}{k!}. \tag{6}$$

Note that for a given $\varepsilon_{t,\lambda}^{(n)}$, λ and t, n can be computed *a priori*, cf. [16,17] and for a given n, $\varepsilon_{t,\lambda}^{(n)}$ increases linearly with $\lambda \cdot t$.

Finally, observe that the matrices $\mathbf{V}^{(n)}(t)$ and $\mathbf{U}^{(n)}$ are of infinite size. However, exploiting the repetitive structure of the QBDs and the truncation given by uniformization, we can give a finite representation that depends on the number of considered steps n for a given error bound, as will be presented next.

5.2. Finite representation

From every single state, only a finite number of states are reachable in n steps. The transient probability computed by uniformization to reach one of the non-reachable states is zero. Hence, for a single starting state, it is sufficient to consider only the finite set of reachable states. This idea was already mentioned in [15,27,37]. When simultaneously considering every state of the infinite state space as starting state, one would have to consider an infinite number of finite parts of the QBD, which is not feasible. However, given a finite number n of steps, there is a repeating level l from which onwards the boundary level cannot be reached anymore. Therefore, the finite part of the QBD that needs to be considered for starting states from repeating levels l onward does not contain states of the boundary level. The structure of all these finite parts is identical, only shifted appropriately. This implies that we obtain identical transient probabilities (shifting appropriately) for corresponding states in repeating levels at least l, within the error bounds of uniformization given n steps. Therefore, we can use the states of level l as *representatives* for all corresponding states of higher levels. In fact, we restrict the computation to a finite number of starting states and still perform a comprehensive transient analysis for every possible state as a starting state.

For a finite representation of the matrices $\mathbf{V}^{(n)}(t)$ and $\mathbf{U}^{(n)}$, it is now sufficient to store all non-zero entries for starting states of levels up to l. The size of the finite representation depends on the considered number of steps n, and hence, on the time, the uniformization rate, and the required accuracy.

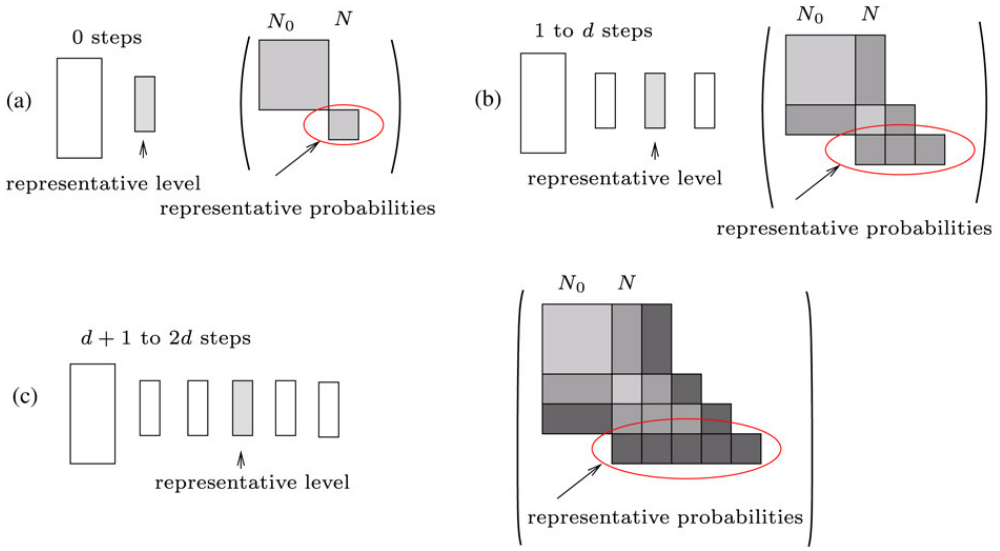


Fig. 2. Considered part of the state space (left) and finite representation of $\mathbf{U}^{(n)}$ and $\mathbf{V}^{(n)}(t)$ (right), depending on the number of considered steps.

5.3. Probability distribution after n epochs

We now address the growth of the matrices $\mathbf{U}^{(n)}$ in the course of the computation. Fig. 2(a) shows that the dimension of the finite representation of $\mathbf{U}^{(0)}$ is $(N_0+N)\times(N_0+N)$. Since $n=0$, we cannot leave a level, and the first repeating level is already representative for all (other) repeating levels. In the case $n=1$, we can reach the next higher or the next lower level. Since the next lower level is the boundary level, the first repeating level cannot be used as representative, but we can use the second repeating level as representative, as shown in Fig. 2(b). Since $n=1$, it is possible to reach the next higher level as well; thus we have to consider starting in one of the first three levels (including the boundary level), and ending up in one of the first four levels. The dimension of the finite representation of $\mathbf{U}^{(1)}$ therefore equals $(N_0+2N)\times(N_0+3N)$. With a symmetric level diameter d, we will need at least another $d-1$ steps before possibly reaching the next higher repeating level. Thus, the size of all $\mathbf{U}^{(n)}$, for $n=1,\ldots,d$, will be the same as for $n=1$. Fig. 2(c) shows the finite representation of matrices $\mathbf{U}^{(n)}$, for $n=d+1,\ldots,2d$. From a given level, we can reach at most the next two higher or lower levels. Therefore, we have to pick a new representative: the third repeating level. Starting from this representative, we can reach the next two higher repeating levels. We have to attach another row (of blocks) for the new representative, and in every other row, we have to attach one block to the left (the next lower) and one to the right (the next higher level), wherever possible. The dimension of the finite representation is then $(N_0+3N)\times(N_0+5N)$, for all $\mathbf{U}^{(n)}$, for $n=d+1,\ldots,2d$. In general, for a given number of steps $n\geq 1$ and level diameter d, the maximum number l of levels reachable from a representative level in one direction (up or down) is given by

$$l=((n-1)\ \mathrm{div}\ d)+1. \tag{7}$$

The size of the matrix $\mathbf{U}^{(n)}$ is then determined by l. The dimension of its finite representation is $(N_0+(l+1)N)\times(N_0+(2l+1)N)$. As before, the finite representation of the matrix $\mathbf{V}^{(n)}$ has the same dimension.

5.4. Uniformization with representatives

We now proceed with the actual computation of $\mathbf{U}^{(n)}$ and $\mathbf{V}^{(n)}(t)$ according to Eq. (5). Starting with $n=0$, and thus with a small finite portion of the QBD, cf. Fig. 2(a), we increase n step-by-step, thus increasing the accuracy and size of the considered finite representation of the QBD. However, in each iteration we always use the smallest possible representation. The matrices $\mathbf{U}^{(n)}$ and $\mathbf{V}^{(n)}(t)$ have a block structure, according to the levels of a QBD; we denote the blocks that give the probabilities from states in level i to states in level j as $\mathbf{U}^{(n)}_{i,j}$ and $\mathbf{V}_{i,j}(t)$.

Starting with $\mathbf{U}^{(0)}$ (with dimension of the finite representation $(N_0+N)\times(N_0+N)$), the computation of $\mathbf{U}^{(1)}$ is visualized in Fig. 3: we multiply the finite representation of $\mathbf{U}^{(0)}$, where one row of blocks is added for the new

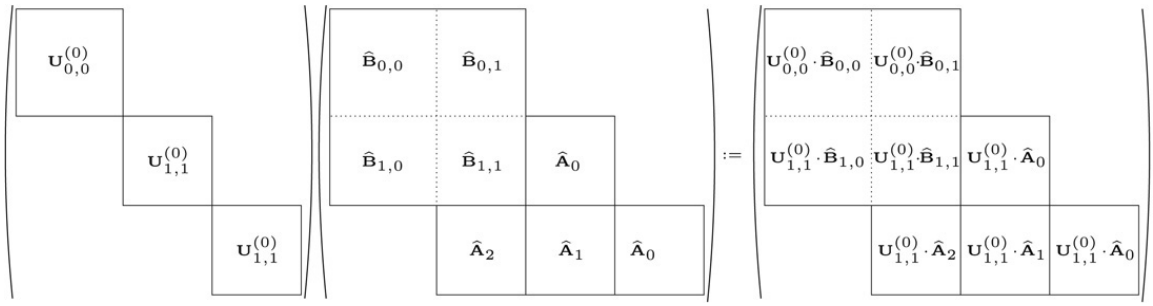


Fig. 3. Computation of $\mathbf{U}^{(0)} \cdot \mathbf{P} = \mathbf{U}^{(1)}$.

representative repeating level, with a finite portion of $\mathbf{P}$ that consists of three block rows (for the three considered starting levels) and of four block columns (for the four levels that can be reached). In general, for $n \geq 1$, $\mathbf{U}^{(n)}$ is computed as $\mathbf{U}^{(n-1)} \cdot \mathbf{P}$, cf. (2) as follows:

$$\begin{aligned}
\mathbf{U}^{(n)}_{i,0} &= \mathbf{U}^{(n-1)}_{i,0} \cdot \hat{\mathbf{B}}_{0,0} + \mathbf{U}^{(n-1)}_{i,1} \cdot \hat{\mathbf{B}}_{1,0}, && \text{for } i = 0, \ldots, l, \\
\mathbf{U}^{(n)}_{i,1} &= \mathbf{U}^{(n-1)}_{i,0} \cdot \hat{\mathbf{B}}_{0,1} + \mathbf{U}^{(n-1)}_{i,1} \cdot \hat{\mathbf{B}}_{1,1} + \mathbf{U}^{(n-1)}_{i,2} \cdot \hat{\mathbf{A}}_2, && \text{for } i = 0, \ldots, l+1, \\
\mathbf{U}^{(n)}_{i,j} &= \mathbf{U}^{(n-1)}_{i,j-1} \cdot \hat{\mathbf{A}}_0 + \mathbf{U}^{(n-1)}_{i,j} \cdot \hat{\mathbf{A}}_1 + \mathbf{U}^{(n-1)}_{i,j+1} \cdot \hat{\mathbf{A}}_2, && \text{for } i = 0, \ldots, l+1,\ j = 2, \ldots, i+l,
\end{aligned} \tag{8}$$

where l is computed as in (7). Due to the block structure of $\mathbf{V}^{(n)}(t)$, we can rewrite (4) as:

$$\mathbf{V}^{(n)}_{i,j}(t) = \mathbf{V}^{(n-1)}_{i,j}(t) + \psi(\lambda t; n)\mathbf{U}^{(n)}_{i,j}, \tag{9}$$

for $i = 0, \ldots, l+1$, and for $j = \max\{0, i-l\}, \ldots, i+l$. After d steps, the size of $\mathbf{V}^{(n)}(t)$ will have to be adapted. This comes at no computational cost, since the block matrices that need to be appended are either zero, or just copies of block matrices that have been computed before already.

5.5. Complexity

The level index l_k of the representative increases with the number of considered steps k, and decreases with the symmetric level diameter d. In the k-th iteration, we actually consider the states of the boundary level and of $l_k + 1$ repeating levels as starting states, and the states of the boundary level and of $2l_k + 1$ repeating levels as end states. The boundary level has N_0 states and each repeating level has N states, resulting in matrices with $(N_0 + (l_k + 1)N) \times (N_0 + (2l_k + 1)N)$ entries, so that the storage requirement grows with the level index of the representative. If n is the maximum number of steps considered, the overall storage complexity for the three probability matrices $\mathbf{U}^{(n-1)}$, $\mathbf{U}^{(n)}$ and $\mathbf{V}^{(n)}$ is $\mathcal{O}(3(N_0^2 + l_n N_0 N + l_n^2 N^2))$.

Let ν denote the average number of transitions originating from a single state in the QBD. Assuming a sparse representation, the discrete transition matrix $\mathbf{P}$ has storage complexity $\mathcal{O}(\nu(N_0 + 2N))$. In the kth iteration, the multiplication of matrix $\mathbf{U}^{(n-1)}$ with $\mathbf{P}$ is carried out in $\mathcal{O}(\nu(N_0 + (l_k + 1)N))$. For n, the maximum number of considered steps, the overall time complexity therefore equals $\mathcal{O}(n \cdot \nu(N_0 + (\frac{l_n}{2} + 1)N))$.

Note that the iteration costs per level increase. However, when probability matrices of the size $\mathbf{U}^{(n)}$ and $\mathbf{V}^{(n)}$ are used throughout the complete computation, the iteration costs are much higher.

5.6. How many steps are to be taken?

As for any uniformization-based technique, we can compute the number of steps that need to be taken into account *a priori*. However, this may introduce several problems in a model checking context. First of all, such a statically computed number of steps might be larger than is really needed to determine whether a CSL property is satisfied or not. Furthermore, in the other cases, the preset accuracy (hence, number of steps) might not be sufficient to decide whether the computed probability meets the required bound. To overcome both problems, we propose a dynamic stopping criterion, which we claim to be optimal in the current setting.

For model checking an until-formula $\mathcal{P}_{\bowtie p}(\Phi\, \mathcal{U}^{[t_1,t_2]}\, \Psi)$, we have to compare for each starting state the probability to take a $(\Phi\, \mathcal{U}^{[t_1,t_2]}\, \Psi)$-path with the probability bound p. In the transformed QBD $\mathcal{Q}[\neg\Phi \vee \Psi]$, the set of goal states consists of all Ψ-states. We denote the probability to end up in a Ψ-state, given starting state s, as $\gamma_s(t)$. For the time interval $I = [0, t]$, we have:

$$\gamma_s(t) = \sum_{i=0}^{\infty} \sum_{s' \in Sat^i(\Psi)} \pi^{\mathcal{Q}[\neg\Phi\vee\Psi]}(s, s', t).$$

Similarly, for the time interval $I = [t_1, t_2]$, we obtain

$$\gamma_s(t) = \sum_{i=0}^{\infty} \sum_{s' \in Sat^i(\Phi)} \sum_{j=0}^{\infty} \sum_{s'' \in Sat^j(\Psi)} \pi^{\mathcal{Q}[\neg\Phi]}(s, s', t_1) \cdot \pi^{\mathcal{Q}[\neg\Phi\vee\Psi]}(s', s'', t_2 - t_1),$$

starting in a state s in $\mathcal{Q}[\neg\Phi]$. Note that $\gamma(t)$ consists of sub-vectors corresponding to the levels of the QBD. The approximation of $\gamma_s(t)$ after n iterations is called $\gamma^{(n)}(t) = \mathbf{V}^{(n)}(t) \cdot \gamma(0)$, and

$$\gamma_s(0) = \begin{cases} 1, & s \models \Psi, \\ 0, & \text{otherwise.} \end{cases}$$

In principle, $\gamma^{(n)}(t)$ is of infinite size, but we can cut it to a finite representation, as from a representative level onwards, all levels contain the same values. It is also possible to derive Eq. (8) directly for $\gamma(t)$ and then use this vector for the computation. When increasing the number of considered steps n, the entries of $\gamma^{(n)}(t)$ increase monotonously. Thus, comparing entries of the probability vector $\gamma^{(n)}(t)$ with the bound p on a regular basis, we might be able to decide whether the probability meets the bound p after a smaller number of iterations than computed a priori. With uniformization with representatives, the computed approximation after n steps always underestimates the actual probability.

Recall that $\varepsilon_{t,\lambda}^{(n)}$ is the maximum error of uniformization after n iteration steps (cf. (6)), such that $\gamma_s(t) \leq \gamma_s^{(n)}(t) + \varepsilon_{t,\lambda}^{(n)}$ for time interval $I = [0, t]$ and $\gamma_s(t) \leq \gamma_s^{(n)}(t) + \varepsilon_{t_1,\lambda_1}^{(n)} + \varepsilon_{t_2-t_1,\lambda_2}^{(n)}$ for time interval $I = [t_1, t_2]$. We define $\xi^{(n)}$ to be the appropriate maximum error for each time interval, respectively. From (6) it follows that the value of $\varepsilon_{t,\lambda}^{(n)}$ decreases as n increases. Exploiting the above inequality, we obtain the following stopping criteria:

$$\begin{array}{lrcl} \text{(a)} & \gamma_s^{(n)}(t) \geq p & \Rightarrow & \gamma_s(t) \geq p, \\ \text{(b)} & \gamma_s^{(n)}(t) \leq p - \xi^{(n)} & \Rightarrow & \gamma_s(t) \leq p. \end{array}$$

These criteria can be exploited as follows. Starting with a small number of steps, we check whether for the current approximation one of the inequalities (a) or (b) holds for all starting states. If this is not the case, we continue, check again, etc., until one of the stopping criteria holds. However, if for one of the starting states $s \in S$, we have $\gamma_s(t) = p$, the iteration never stops, as neither of the stopping criteria ever holds. However, this is highly unlikely to occur in practice.

5.7. The unbounded case ($I = [0, \infty]$)

In Section 4.6.3, we showed how the model checking of an unbounded until operator relies on the computation of the steady-state probabilities in the absorbing QBD $\mathcal{Q}[\neg\Phi \vee \Psi]$. Note that $\mathcal{Q}[\neg\Phi \vee \Psi]$ is not irreducible, so we cannot compute the steady-state probabilities using an MGM-based algorithm. However, the steady-state probability of the set of absorbing Ψ-states is independent of the residence time in each state-visit, but only depends on the branching probabilities and the starting state. We can therefore switch to the embedded discrete-time QBD with infinite tri-diagonal probability matrix $\mathbf{P}$ where $\mathbf{P}(s, s') = \mathbf{Q}(s, s') / \sum_{s'' \neq s} \mathbf{Q}(s, s'')$. If the level of the starting state is l, the

desired steady-state probability is

$$\begin{aligned}\pi(s, Sat(\Psi)) &= \sum_{i=0}^{\infty} \sum_{s' \in Sat^i(\Psi)} \pi(s, s') \\ &= \sum_{i=0}^{\infty} \sum_{s' \in Sat^i(\Psi)} \sum_{n=0}^{\infty} \mathbf{P}^n(s, s') \\ &= \sum_{n=0}^{\infty} \sum_{i=\max(0,l-n)}^{l+n} \sum_{s' \in Sat^i(\Psi)} \mathbf{P}^n(s, s').\end{aligned}$$

The last equality follows from the fact that only a finite number of levels is reachable in n steps. Using this equation, representative probabilities can be computed for an increasing number of considered steps n as is the case for uniformization. The stopping criterion is adopted from Section 4.4: the approximations computed with n steps, $\tilde{\pi}^n(s, Sat(\Psi))$ and $\tilde{\pi}^n(s, Sat(\neg\Psi))$, are compared to p and $1-p$, respectively, until we can stop the iteration.

5.8. Proof of Proposition 1 for the until operator

After having presented the algorithms for the model checking of until formulas with different time intervals, we discuss the justification of Proposition 1. We want to show that any until formula $\mathcal{P}_{\bowtie p}(\Phi \mathcal{U}^I \Psi)$ is independent as of level k, under the assumption that for no state s the probability measure is exactly equal to p, hence, $Prob(s, \Phi \mathcal{U}^I \Psi) \neq p$.

We do this for bounded until formulas in more detail. Such formulas are level independent as of level k, if for every repeating state $i \in \{0, \ldots, N-1\}$, there exists an $N_i \geq 1$ such that either

$$\forall l \geq N_i : \gamma_{(i,l)}(t) > p$$

or

$$\forall l \geq N_i : \gamma_{(i,l)}(t) < p.$$

The maximum of all N_i is then the index k for level independence. The stopping criteria (cf. Section 5.6) of uniformization with representatives ensure that we will find such N_i for all states. However, we still have to ensure that the algorithm always stops.

Assume that it does not do so, then

$$\forall l : \lim_{n\to\infty} \gamma^{(n)}_{(i,l)}(t) < p \quad \text{and} \quad \forall l : \lim_{n\to\infty} \left(\gamma^{(n)}_{(i,l)}(t) + \underbrace{\xi^{(n)}}_{\to 0} \right) > p.$$

But $\lim_{n\to\infty} \xi^{(n)} = 0$ and so we get

$$p < \gamma_{(i,l)}(t) = \lim_{n\to\infty} \gamma^{(n)}_{(i,l)}(t) < p,$$

which is not possible. Consequently, the algorithm will always stop, thereby having computed an N_i for each repeating state, and so the corresponding until formula is level independent as of level $\max_i(N_i)$. For the unbounded until formula, level independence as of level k can be proven similarly.

6. Case study: Connection management

The transport protocol TCP offers a connection-oriented service on the Internet, which implies that a connection should be established prior to any application data is exchanged [24]. Arriving application-layer protocol data units, e.g., HTTP requests, therefore potentially suffer a delay from connection establishment, unless an existing connection can be (re)used.

We analyze the behavior of the connection management mechanism, known as "on-demand connection with delayed release" (OCDR) [19], as sketched in Fig. 4(a). Packets that have to be transported are generated by an abstract

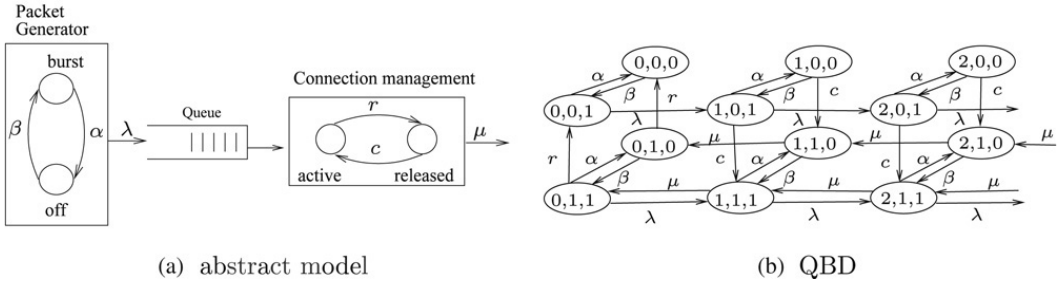


Fig. 4. OCDR mechanism for connection management.

Table 1
Numerical values for the parameters of the model

Parameter	λ	μ	α	β	c	r
s^{-1}	100	125	1	0.04	10	10

packet generator and submitted to the transfer queue. The *connection* can be in one of two modes: (i) it can be *active*, so that an arriving packet can be served immediately, at the cost of maintaining a possibly unused connection; (ii) the connection can be *released*, so that an arriving packet can only be transmitted after the connection is re-established, but there are no costs for maintaining an unused connection. Arriving packets at a released connection suffer an extra *connection-establishment delay*. Once active, all queued packets, as well as those newly arriving, will be transported.

In this specific application, the packet generator cycles through periods in which packets are generated with high intensity (in bursts), followed by periods in which no packets are generated at all. The connection management switches between modes, so as to find the right balance between good performance (low delays) and low costs. Having served the last packet of a burst, the connection will be held active for some time. If no new burst starts within some time-out period, the OCDR mechanism releases the connection.

To keep the model simple and illustrative, we assume an exponentially distributed connection-establishment delay with rate c, as well as an exponentially distributed time-out for release with mean $1/r$. The model could be extended easily to more deterministic delays, e.g., by using Erlangian approximations [29]. Packets take an exponentially distributed amount of time to be transmitted, with rate μ. In a burst, packets are generated according to a Poisson process with rate λ. The generator switches between epochs of activity and idleness, both exponentially distributed, with rates α and β, respectively. Under these conditions, Fig. 4(b) provides the corresponding QBD. In this model, the state space is $S = \{(i, j, k) \mid i \in \mathbb{N}, j, k = 0, 1\}$, where i denotes the number of packets queued (and being transmitted), j denotes whether the connection is *active* ($j = 1$) or *released* ($j = 0$), and k denotes whether the packet arrival process is in a burst ($k = 1$) or not ($k = 0$). Clearly, each level i (= i packets present) consists of four states: $S^i = \{(i, 0, 0), (i, 0, 1), (i, 1, 0), (i, 1, 1)\}$. The symmetric level diameter is $d = 1$. Table 1 shows the numerical values of the parameters as presented in [19].

Model checking steady-state properties. We want to know whether the steady-state probability of being in the different phases with atomic properties $\Phi_1 =$ *active and no burst*, $\Phi_2 =$*released and burst* or $\Phi_3 =$*active and burst*, is greater than a given probability bound p. For each Φ_i, each level contains exactly one state where this atomic property holds. Hence, the sets $Sat(\Phi_i)$ have infinite size. Fig. 5 shows the number of iterations (as discussed in Section 4.4) needed to verify the property, depending on the probability bound p. If the actual steady-state probability of Φ_i-states comes close to the given bound p, more iterations are needed. This explains the peak at $p = 0.0065$ for Φ_1, at $p = 0.0071$ for Φ_2, and at $p = 0.0313$ for Φ_3. Depending on the chosen probability bound p, the satisfaction sets $Sat(\mathcal{S}_{>p}(\Phi_i))$ are either empty or consist of the complete state space. The switching probability is the same as the peak probability.

Model checking time-bounded until. Fig. 6 shows the number of uniformization steps n needed for the computation of $Sat(\mathcal{P}_{\geq p}(tt\ \mathcal{U}^{[0,t]}\ \Psi))$ for $\Psi =$ *released and no burst* and for $t \in \{0.5, 1.0, 2.0\}$, depending on the probability bound p.

To analyze the efficiency gain using the dynamic stopping criterion as presented in Section 5.6, we obtain the number of iterations with the dynamic stopping criterion, as well as the a priori computed number of steps required

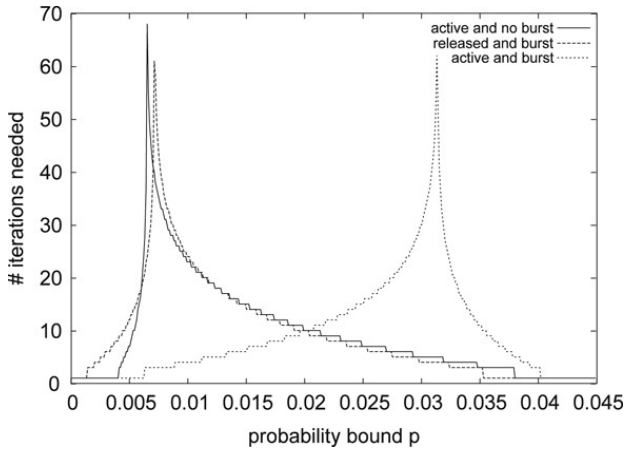


Fig. 5. Number of iterations needed for checking $s \models \mathcal{S}_{>p}(\Phi_i)$, $i = 1, 2, 3$.

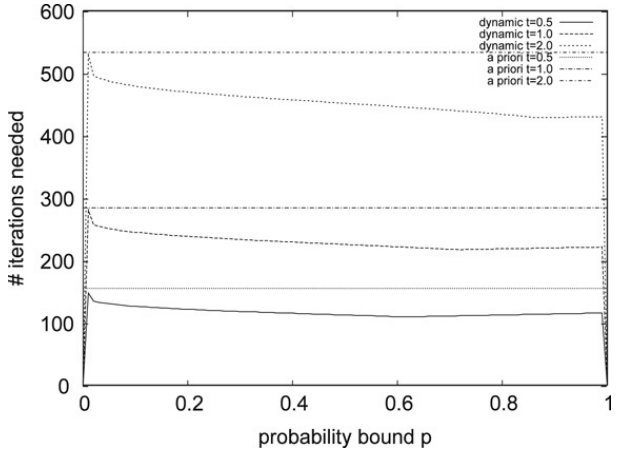


Fig. 6. Number of iterations needed for checking $s \models \mathcal{P}_{\geq p}(tt\ \mathcal{U}^{[0,t]}\ \Psi)$.

for an error $\varepsilon_{t,\lambda}^{(n)} = 10^{-4}$. Clearly, the a priori number of steps is independent of the probability bound p, and increases with a growing time bound t.

After 0 steps, the comparison can be evaluated for $p = 0$ for all time bounds when using the dynamic stopping criterion: every probability is at least 0. For an increasing probability bound p, the number of iterations first increases steeply. This is because the Poisson probabilities are de facto equal to zero in the first few iterations, and no decision can be made when comparing with any $p > 0$.

Then, for a very low probability bound $p_{\text{peak}} \approx 0.02$, the number of iterations jumps to a peak value for all three time bounds. A peak occurs whenever the computed probability for some state gets really close to the probability bound p we have to compare with. The peak number of iterations with the dynamic stopping criterion approximately equals the a priori computed number of steps for $\varepsilon_{t,\lambda}^{(n)} = 10^{-4}$. Hence, we conclude that the computed probability for one of the starting states lies in $[p_{\text{peak}} - 10^{-4}, p_{\text{peak}}]$.

For an increasing probability bound p, the number of iteration steps using the dynamic stopping criterion decreases. For larger time bounds t, the gap between the curves for the dynamic and the a priori number of iteration steps increases, showing the efficiency gain using the dynamic stopping criterion.

The execution time per iteration is the same for the dynamic as for the a priori stopping criterion. Hence, for the same number of iterations, the execution time is the same for one measure. For an increasing number of iterations, the execution time per iteration grows, as for large n the iterations take longer due to the larger matrices involved. For the measure $\mathcal{P}_{\geq p}(tt\ \mathcal{U}^{[0,t]}\ \Psi)$, the execution time per iteration ranges from $1.26 \cdot 10^{-3}$ s for the time bound 0.5 and $4.92 \cdot 10^{-3}$ s for the time bound 2.0. Even though the curves in Fig. 6 look smooth, small variations in the number of iterations occur when using the dynamic stopping criterion.

7. Related work

We are not aware of any other work that addresses the continuous-time model checking problems addressed in this paper. There is, however, some *related* work on model-checking infinite state systems, as well as on the steady-state and transient analysis of infinite-state Markov chains. We comment on these below.

As for **model checking infinite-state Markov chains**, the following results are in some sense related to our work, although none of them addresses the specific problems we address.

Work on the LTL model checking for so-called *probabilistic lossy channel* systems (PLCSs) has been proposed for the evaluation of asynchronous buffer systems [7]. The main idea, as presented in [4], is to reduce the LTL model checking problem to a reachability problem in a (non-probabilistic) labeled transition system. Every PLCS with just one message type and one channel with global fault semantics can be seen as a discrete time QBD, but not conversely. The results in this area do not, however, refer to continuous time, nor do they provide model checking algorithms for full CSL. Of particular interest in this context is [21], in which the same PLCS as in [4] is addressed. This paper proposes path-enumeration algorithms to compute, with a given small error tolerance, whether another state can be reached with probability at least p. Similarly, using an iterative step-count-based state space exploration scheme, the probability for more general CTL formulae is computed, again for individual starting states and a given error tolerance. These algorithms bear a resemblance with our algorithm for the untimed-until operator. In [2], the algorithms of [21] are refined and made simpler; furthermore, some new decidability results are given. Similarly, [30] addresses also PLCS, and two very specific reachability properties for individual starting states, in an untimed setting.

None of the above papers addressing PLCS addresses continuous-time, the logic CSL, or provides a model checking algorithm for all possible starting states, as we do. By restricting to the special class of QBDs, we have been able to provide a richer set of model checking algorithms.

Regular model checking comprises a set of techniques for symbolic reachability analysis for parameterized and non-probabilistic infinite-state systems, based on the automata theory [23,1]. Words are used to represent states and finite-state transducers describe transitions between states. Every discrete-time QBD can be easily expressed within the regular model checking framework. As above, the results in this area do not refer to continuous time, nor do they provide model checking algorithms for full CSL.

Recursive Markov chains (RMCs) have been proposed to model probabilistic procedural programs. The main goal of model checking RMCs is to find the probability of eventually reaching a given terminating state of the RMC, starting from a given initial state. In [14], this probability is defined as the least fixed-point solution of a system of polynomial equations. Discrete-time QBDs are a subclass of RMCs. Again, the RMC work does not address continuous time, and nor are complete CSL model checking algorithms provided.

Finally, there is also related work on *probabilistic pushdown automata* (pPDA); as stated in [8], these models coincide with RMCs. [8] focuses on decidability results for such automata, whereas [13] presents an algorithm for evaluating the time-unbounded until operator for a single starting state only. These models do not address continuous time, and nor are complete CSL model checking algorithms provided in them. The decidability results presented are not necessarily valid for QBDs, as QBDs comprise only a (structured) subset of pPDA.

As for the **steady-state and transient analysis of infinite-state Markov chains**, some work has been done in the past, however, not in the context as we need it. We refer to the seminal work by Neuts [28] on matrix–geometric solutions for computing the steady-state probabilities in infinite-state quasi-birth-death processes. In particular, we employ the logarithmic reduction algorithm as proposed in [26].

For transient-state probabilities, there is much less work available. Zhang et al. [38] describe a Laplace-transform based technique to obtain these probabilities for QBDs; however, they do not provide the required back transformation. In his Ph.D. thesis, Van Moorsel hints at an approach called dynamic uniformization [37], and so does Grassmann [15], as a technique to evaluate systems with infinite state spaces. In their well-known 1984 paper [16], Gross and Miller already refer to a possible use of uniformization for infinite-state systems. These three papers have in common that hints towards evaluating transient-state probabilities in infinite-state systems are given, but that no true algorithms or data structures are presented. Furthermore, the issue of having an infinite number of possible starting states is not addressed. With step-wise uniformization [10], it is possible to calculate transient-state probabilities in large, or even infinite-size, CTMCs and DTMCs. This is done by step-wisely extending the considered state space, i.e. on-the-fly while generating the state space. Interestingly, this approach bears resemblance with the probabilistic reachability algorithm presented in [21], however, it requires (as does [21]) an unique starting state. Recently, Van

Houdt and Blondia [35,36] addressed the transient analysis of QBDs, however, in a discrete-time setting, and using an approximation technique with unknown a priori error bounds. Furthermore, it addresses just a unique single starting state.

The way we compute the transient probabilities comes closest to the work by Le Ny and Sericola [27], in which they compute the transient queue length distribution in the very specific context of an BMAP/PH/1 queue (their equations (3–5,10,11) closely resemble our recursion (8)). Their approach is tailored toward two very specific queuing-related measures of interest, which are less general than the transient-state probabilities we need.

In conclusion, what all the above approaches for the computation of transient-state probabilities in infinite-state Markov chains have in common is that they assume a single unique starting state (or a single unique starting distribution). By contrast, we compute all such transient-state probabilities for all possible starting states in a single computation, as required for the CSL model checking procedure. Hence, none of the previously published results is applicable in our context.

8. Conclusions

In this paper we have presented new algorithms for model checking CSL properties against infinite-state CTMCs, in particular for QBDs. The model checking algorithms make extensive use of uniformization for transient analysis (for time-bounded until), and matrix–geometric methods for determining steady-state probabilities (for the steady-state operator). The model checking algorithms as presented are new. We are aware of the fact that when checking nested formulas, the number of levels that have level-dependent properties grows, which makes the algorithms less efficient. On the other hand, practice reveals that the nesting depth of logical expressions to be checked is typically small [12], so that this is not so much of a disadvantage after all. By restricting ourselves to level-independent and periodic formulas, we restrict the set of CSL formulas that can be checked. For model checking truly level-*dependent* CSL formulas, new model checking algorithms will be needed, since in such cases we cannot exploit the level-independent QBD structure to cut the infinite set of states.

Our approach to analyze the transient state probabilities of QBDs with uniformization with representatives is also new. We claim the stopping criterion as presented to be optimal when checking a CSL until-formula. Uniformization with representatives is both computationally and memory efficient. The only drawback the approach suffers from is that the number of considered steps can become large, as n depends on the product λt; but this is a general drawback of uniformization. Large n lead to large matrices $\mathbf{U}^{(n)}$ and $\mathbf{V}^{(n)}$; however, these matrices are very sparse, due to the block-structure. We have shown the feasibility of our approach by a case study and discussed related work in detail.

Our model checking approach can also be applied to a number of extensions of QBDs, making our model checking algorithms even more versatile [33]. QBDs with *resets* additionally allow for transitions that lead from states in repeating levels to the boundary level. This requires a straightforward extension of our matrix–geometric algorithms to calculate the steady-state probabilities. The transient probabilities can again be calculated with uniformization with representatives. We can also allow for transitions between states in non-neighboring levels, provided the number of levels skipped remains finite, and the structure of the level skipping is the same for all repeating levels. We need additional transition matrices to account for the possible skips. By regrouping as many states into one level as necessary to guarantee that transitions entering or leaving a level are restricted to neighboring levels only, we can transform any QBD with finite skipping into a standard QBD [18]. Finally, we can also deal with level dependent atomic properties as long as these show periodicity. These can, as well, be regrouped to QBDs with level independent atomic propositions by combining several levels into one bigger level. The regrouped QBDs can be model checked with the standard procedure.

References

[1] P. Abdulla, B. Jonsson, M. Nilsson, M. Saksena, A survey of regular model checking, in: Proc. 10th International Conference on Concurrency Theory, Concur'04, in: Lecture Notes in Computer Science, vol. 3170, 2004, pp. 35–48.

[2] P.A. Abdulla, N.B. Henda, R. Mayr, Verifying infinite Markov chains with a finite attractor or the global coarseness property, in: Proc. 20th IEEE Symposium on Logic in Computer Science, LICS'05, IEEE Computer Society, 2005, pp. 127–136.

[3] A. Aziz, K. Sanwal, R. Brayton, Model checking continuous-time Markov chains, ACM Transactions on Computational Logic 1 (1) (2000) 162–170.

[4] C. Baier, B. Engelen, Establishing qualitative properties for probabilistic lossy channel systems: An algorithmic approach, in: Lecture Notes in Computer Science, vol. 1601, 1999, pp. 34–52.

[5] C. Baier, B.R. Haverkort, H. Hermanns, J.-P. Katoen, On the logical characterisation of performability properties, in: Proc. 27th International Colloquium on Automata, Languages and Programming, ICALP'00, in: Lecture Notes in Computer Science, vol. 1853, 2000, pp. 780–792.
[6] C. Baier, B.R. Haverkort, H. Hermanns, J.-P. Katoen, Model-checking algorithms for continuous-time Markov chains, IEEE Transactions on Software Engineering 29 (7) (2003) 524–541.
[7] N. Bertrand, Ph. Schnoebelen, Model checking lossy channel systems is probably decidable, in: Proc. 6th Foundations of Software Science and Computation Structures, FOSSACS'03, in: Lecture Notes in Computer Science, vol. 2060, 2003, pp. 120–135.
[8] T. Brázdil, A. Kucera, O. Strazovský, On the decidability of temporal properties of probabilistic pushdown automata, in: Proc. 22nd Annual Symposium on Theoretical Aspects of Computer Science, STACS'05, in: Lecture Notes in Computer Science, vol. 3404, 2005, pp. 145–157.
[9] P. Buchholz, J.P. Katoen, P. Kemper, C. Tepper, Model-checking large structured Markov chains, The Journal of Logic and Algebraic Programming 56 (1–2) (2003) 69–97.
[10] G. Ciardo, Discrete-time Markovian stochastic Petri nets, in: Computations with Markov Chains, Raleigh, 1995, pp. 339–358.
[11] D. d'Aprile, S. Donatelli, J. Sproston, CSL model checking for the GreatSPN tool, in: Proc. 19th International Symposium on Computer and Information Sciences, ISCIS'04, in: Lecture Notes in Computer Science, vol. 3280, 2004, pp. 543–552.
[12] M.B. Dwyer, G.S. Avrunin, J.C. Corbett, Patterns in property specification for finite-state verification, in: Proc. 21st International Conference on Software Engineering, IEEE CS Press, 1999, pp. 411–420.
[13] J. Esparza, A. Kucera, R. Mayr, Model checking probabilistic pushdown automata, in: Proc. 19th IEEE Symposium on Logic in Computer Science, LICS'04, IEEE CS Press, 2004, pp. 12–21.
[14] K. Etessami, M. Yannakakis, Recursive Markov chains, stochastic grammars, and monotone systems of nonlinear equations, in: Proc. 22nd Annual Symposium on Theoretical Aspects of Computer Science, STACS'05, in: Lecture Notes in Computer Science, vol. 3404, 2005, pp. 340–352.
[15] W. Grassmann, Finding transient solutions in Markovian event systems through randomization, in: Numerical Solution of Markov Chains, Dekker, 1991, pp. 411–420.
[16] D. Gross, D.R. Miller, The randomization technique as a modeling tool and solution procedure for transient Markov processes, Operations Research 32 (2) (1984) 343–361.
[17] B.R. Haverkort, Performance of Computer Communication Systems, John Wiley & Sons, 1998.
[18] B.R. Haverkort, A. Ost, Steady-state analysis of infinite stochastic petri nets: A comparing between the spectral expansion and the matrix–geometric method, in: Proc. of the 7th International Workshop on Petri Nets and Performance Models, PNPM'97, IEEE CS Press, 1997.
[19] G.J. Heijenk, B.R. Haverkort, Designing and evaluation of a connection management mechanism for an ATM-based connectionless service, Distributed System Engineering Journal 3 (1) (1996) 53–67.
[20] H. Hermanns, J.-P. Katoen, J. Meyer-Kayser, M. Siegle, A tool for model-checking Markov chains, International Journal on Software Tools for Technology Transfer 4 (2) (2003) 153–172.
[21] S.P. Iyer, M. Narasimha, Probabilistic lossy channel systems, in: Proc. Theory and Practice of Software Development, TAPSOFT'97, in: Lecture Notes in Computer Science, vol. 1214, 1997, pp. 667–681.
[22] J.-P. Katoen, M. Khattri, I.S. Zapreev, A Markov reward model checker, in: Proc. 2nd International Conference on Quantitative Evaluation of Systems, QEST'05, IEEE CS Press, 2005, pp. 243–244.
[23] Y. Kesten, O. Maler, M. Marcus, A. Pnueli, E. Shahar, Symbolic model checking with rich assertional languages, in: O. Grumberg (Ed.), Proc. 9th International Conference on Computer Aided Verification, CAV'97, in: Lecture Notes in Computer Science, vol. 1254, 1997, pp. 424–435.
[24] J. Kurose, K. Ross, Computer Networking, Addison Wesley, 2005.
[25] M. Kwiatkowska, G. Norman, D. Parker, Probabilistic symbolic model checking with PRISM: A hybrid approach, International Journal on Software Tools for Technology Transfer 6 (2) (2004) 128–142.
[26] G. Latouche, V. Ramaswami, A logarithmic reduction algorithm for quasi birth and death processes, Journal of Applied Probability 30 (1993) 650–674.
[27] L. Le Ny, B. Sericola, Transient analysis of the BMAP/PH/1 queue, International Journal of Simulation 3 (3–4) (2003) 4–15.
[28] M.F. Neuts, Matrix Geometric Solutions in Stochastic Models: An Algorithmic Approach, Johns Hopkins University Press, 1981.
[29] A. Ost, Performance of communication systems. A model-based approach with matrix–geometric methods, Ph.D. Thesis, Dept. of Computer Science, RWTH Aachen, 2001.
[30] A.M. Rabinovich, Quantitative analysis of probabilistic lossy channel systems, in: Proc. Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, in: Lecture Notes in Computer Science, vol. 2719, 2003, pp. 1008–1021.
[31] A. Remke, Model checking quasi birth death processes, Master's Thesis, Dept. of Computer Science, RWTH Aachen. http://www.cs.utwente.nl/~anne/pub/modelchecking.pdf, 2004.
[32] A. Remke, B.R. Haverkort, L. Cloth, Model checking infinite-state Markov chains, in: Proc. 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'05, in: Lecture Notes in Computer Science, vol. 3440, 2005, pp. 237–252.
[33] A. Remke, B.R. Haverkort, Beyond model-checking CSL for QBDs: Resets, batches and rewards, in: 7th International Workshop on Performability Modeling of Computer and Communication Systems, PMCCS'05, 2005, pp. 23–26.
[34] A. Remke, B.R. Haverkort, L. Cloth, Uniformization with representatives, in: Proc. Workshop on Tools for solving structured Markov chains, 2006.
[35] B. Van Houdt, C. Blondia, Approximated transient queue length and waiting time distributions via steady-state analysis, Stochastic Models 21 (2005) 725–744.
[36] B. Van Houdt, C. Blondia, QBDs with marked time epochs: A framework for transient performance measures, in: Proc. 2nd International Conference on the Quantitative Evaluation of Systems, IEEE CS Press, 2005, pp. 210–219.
[37] A. van Moorsel, Performability evaluation concepts and techniques, Ph.D. Thesis, Dept. of Computer Science, University Twente, 1993.
[38] J. Zhang, E.J. Coyle, Transient analysis of quasi-birth-death processes, Stochastic Models 5 (3) (1989) 459–496.