Time-bounded reachability in tree-structured QBDs by abstraction

Daniel Klink\textsuperscript{a,}\textsuperscript{*}, Anne Remke\textsuperscript{b}, Boudewijn R. Haverkort\textsuperscript{b,c}, Joost-Pieter Katoen\textsuperscript{a,b}

\textsuperscript{a} RWTH Aachen University, Germany
\textsuperscript{b} University of Twente, The Netherlands
\textsuperscript{c} Embedded Systems Institute, The Netherlands

\textbf{Article Info}

Article history:
Received 19 February 2010
Accepted 29 April 2010
Available online xxxx

Keywords:
Reachability
Abstraction
Markov chains
Probabilistic simulation
Queueing theory

\textbf{Abstract}

This paper studies quantitative model checking of infinite tree-like (continuous-time) Markov chains. These tree-structured quasi-birth death processes are equivalent to probabilistic pushdown automata and recursive Markov chains and are widely used in the field of performance evaluation. We determine time-bounded reachability probabilities in these processes – which with direct methods, i.e., uniformization, result in an exponential blow-up – by applying abstraction. We contrast abstraction based on Markov decision processes (MDPs) and interval-based abstraction; study various schemes to partition the state space, and empirically show their influence on the accuracy of the obtained reachability probabilities. Results show that grid-like schemes, in contrast to chain- and tree-like ones, yield extremely precise approximations for rather coarse abstractions.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

Probabilistic model checking is a verification technique for Kripke structures in which time and transition probabilities are specified stochastically. Popular models are discrete- and continuous-time Markov Chains (DTMC and CTMCs, respectively) and variants thereof that exhibit nondeterminism such as Markov decision processes (MDPs). CTMCs are heavily used in the field of performance evaluation, and since model checking offers various advantages to traditional CTMC analysis techniques, tools such as SMART \cite{1}, GreatSPN \cite{2}, PEPA Workbench \cite{3}, and so on, have been enriched with model checking facilities. Time-bounded reachability probabilities are at the heart of these model-checkers and are reduced to transient analysis \cite{4}. Hence, efficient algorithms are available for finite-state CTMCs, however, they are not applicable to classes of infinite-state Markov chains such as tree-structured quasi-birth death (QBD) processes \cite{5,6}. Tree-structured QBDs have been applied to model single-server queues with a LIFO service discipline \cite{7}, to analyze random access algorithms \cite{8}, as well as priority queueing systems \cite{9}. Discrete-time tree-structured QBDs are equivalent to probabilistic pushdown automata \cite{10} and recursive Markov chains \cite{11}. The analysis of (tree-structured) QBDs mostly focuses on steady-state probabilities as these can be obtained using matrix-geometric techniques \cite{12}. Transient analysis has received scant attention; the only existing approach is approximate \cite{13}. Recently, direct techniques based on uniformization or variants thereof \cite{14}, have been proposed for reachability properties for general infinite-state CTMCs \cite{15} and for highly structured infinite-state CTMCs, such as normal QBDs \cite{16} and Jackson queueing networks \cite{17}. However, they all lead to an exponential blow-up when applied to tree-structured QBDs. Other related work includes model checking discrete-time infinite-state probabilistic systems \cite{18,10,19,20}. Other abstraction techniques for MDPs such as magnifying lens abstraction \cite{21} and game-based...
abstraction [22] cannot be easily adapted to the continuous-time setting and are thus not applicable here. Whereas our technique guarantees to yield upper and lower bounds of reachability probabilities, [13] yields arbitrary approximations. In addition, applying transient analysis to compute timed reachability probabilities requires an amendment of the CTMC which destroys the tree structure; therefore [13] cannot be directly applied to our setting.

In this paper, we determine time-bounded reachability probabilities in tree-structured QBDs by applying CTMC abstraction. To that end, we consider two techniques, interval-based [23] and MDP-based abstraction [24], and compare them. A major issue in state-based abstraction is to come up with an adequate, i.e., small though precise, partitioning of the state space. In fact, we identify various possibilities to partition the state space of a tree-structured QBD, relate them formally using simulation relations [25], and empirically investigate their influence on the accuracy of the obtained time-bounded reachability probabilities. The partitioning methods range from simple schemes such as counter abstraction (group states with equal number of customers) to more advanced schemes in which the ordering of customers, and/or the service phase is respected. This yields tree-, chain-, and grid-like abstractions. We perform extensive experiments for phase-type service distributions of different orders and analyze the influence of parameters setting and partitioning scheme on the quality of the results and on the size of the resulting abstract state space. Our experiments show that grid-like schemes yield extremely precise approximations for rather coarse abstractions. For settings in which uniformization-based techniques would require about 10^{300} states to obtain a given accuracy of 10^{-6}, say, grid-like abstractions of less than 10^6 states suffice.

Organization of the paper. Section 2 introduces CTMCs and tree-structured QBDs and summarizes how to compute time-bounded reachability properties. Section 3 contrasts interval and MDP abstraction and provides some theoretical underpinnings. Different state space partitionings of tree-structured QBDs are considered in Section 4. Experimental results on these abstractions are provided in Section 5. Finally, Section 6 concludes the paper.

2. Tree-structured QBDs

**Notation.** Let $X$ be a countable set. For $x \in X$, $X' \subseteq X$ and $f : X \times X \to \mathbb{R}_{\geq 0}$ (and similarly for $n$-dimensional functions), let $f(x, X') = \sum_{x' \in X'} f(x, x')$ and let $f(x, \cdot)$ be given by $y \mapsto f(x, y)$ for all $x, y \in X$. A probability distribution $\mu : X \to [0, 1]$ assigns a probability to each $x \in X$ such that $\sum_{x \in X} \mu(x) = 1$. We may write $\{x_1 \mapsto p_1, x_2 \mapsto p_2, \ldots\}$ for distribution $\mu$ with $\mu(x_i) = p_i$ for all $x_i \in X, i \in \mathbb{N}^+$. The set of all distributions on $X$ is denoted $\text{distr}(X)$.

**Continuous-time Markov chains.** A CTMC is a Kripke structure with randomized transitions and exponentially distributed delays. Formally, it is denoted as tuple $(\delta, R, (\mu_0))$ with countable state space $\delta$, transition rate function $R : \delta \times \delta \to \mathbb{R}_{\geq 0}$ with $\sup_{s \in \delta} R(s, \cdot) \in \mathbb{R}_{\geq 0}$, and initial distribution $\mu_0 \in \text{distr}(\delta)$. Fig. 1 shows a CTMC with $\delta = \{s_1, s_2, s_3\}$, $R(s_1, s_2) = L$, $R(u, s) = 0$, $\mu_0(s_1) = 1/4$, and so forth.

By adding self-loops, a CTMC can be transformed into a weakly bisimilar, uniform CTMC $(\delta', R', (\mu_0))$ where for all $s \in \delta$, $R'(s, \delta') = \lambda$ for some $\lambda \in \mathbb{R}_{\geq 0}$ with $\lambda \geq \sup_{s \in \delta} R(s, \cdot)$, cf. [25]. Such a uniform CTMC can be seen as a discrete-time Markov chain $(\delta, P, (\mu_0))$ with transition probabilities $P(s, s') = R(s, s')/\lambda$ where the probability to perform a certain number of steps within $[0, t]$ follows a Poisson distribution with parameter $\lambda t$ [26]; in the following, for convenience, we may denote a CTMC by $(\delta, P, \lambda, (\mu_0))$. The probability to reach state $s'$ within $t$ time units is now given by:

$$\sum_{s \in \delta} \mu_0(s) \cdot P(s, s', t), \quad \text{with } P(s, s', t) = \sum_{i=0}^{\infty} P^i(s, s') \cdot e^{-\lambda t} \frac{(\lambda t)^i}{i!},$$

where $P^i$ is the $i$th power of $P$. To avoid the infinite summation over the number of steps $i$, the sum is truncated. For an a priori defined maximum error bound $\varepsilon > 0$, a given time bound $t \in \mathbb{R}_{\geq 0}$ and the uniformization rate $\lambda$ one can compute the truncation point $n_t$. The overall time complexity for computing reachability probabilities up to $\varepsilon$ is in $\mathcal{O}(N^2 \lambda t)$ where $N$ is the number of states in the CTMC that are reachable in $n_t$ steps.

**Queueing theory.** Large classes of distributions can be approximated arbitrarily closely by phase-type distributions. A phase-type distribution of order $d$ (written $\text{PH}_d$) is a probability distribution that can be characterized as the time until absorption in an $(d + 1)$-state CTMC [27]. The time until absorption in the CTMC in Fig. 1 is $\text{PH}_2$ distributed. For example, an $M|PH|1$ queueing station describes a station with one processing unit serving jobs according to a phase-type distribution and with Markovian, i.e., exponentially distributed job interarrival times.

Please cite this article in press as: D. Klink, et al., Time-bounded reachability in tree-structured QBDs by abstraction, Performance Evaluation (2010), doi:10.1016/j.peva.2010.04.002
To compute the utilization, i.e. the fraction of time the processing unit is busy serving jobs, the mean service time \( E[X] \) is needed. This can be seen as the mean time to absorption in the CTMC representing the phase-type distribution. For a PH\(_d\) distribution, consider the corresponding CTMC with state space \( \mathcal{S} = \{1, \ldots, d, d+1\} \), where the states 1, \ldots, \( d \) are transient and state \( d+1 \) is the only absorbing state, regardless of the initial probability vector. The generator matrix of this CTMC is then given as

\[
Q = \begin{pmatrix} V & V^0 \\ 0 & 0 \end{pmatrix},
\]

where \( V \) is a \( d \times d \) matrix with \( v_{i,j} < 0 \) for \( i = 1, \ldots, d, v_{i,j} \geq 0 \) for \( i \neq j \) and \( V^0 \) is a column vector with nonnegative elements. Let the initial probability vector be given by \((\mu, \mu_{d+1})\). The mean time to absorption is then given by \( E[X] = -\mu V^{-1} \cdot 1 \), where 1 is defined as the vector that consists of 1s and the utilization \( \rho \) of an M\(|\text{PH}|1 \) queue can be computed by multiplying the arrival rate of the Markovian arrival process with the mean service time \( E[X] \) of the phase-type distribution.

CTMCs representing PH\(|\text{PH}|1 \) queuing stations with first in first out (FIFO) service discipline correspond to QBD processes [27]. Therefore their state spaces just grow linearly with the number of queued jobs. This stems from the fact that jobs are not preemted, i.e., jobs are served until completion and only the service phase of the job currently in service needs to be encoded in the state. For such systems, uniformization with representatives is a feasible technique to compute transient probabilities [16]. For queuing stations with preemptive last in first out (LIFO) service discipline, however, the underlying CTMCs are so-called tree-structured QBDs, whose number of states grows exponentially with the queue length. In the following, for simplicity we restrict ourselves to M\(|\text{PH}|1 \) queues, however, our approach can also be applied to PH\(|\text{PH}|1 \) queues in the same manner.

In principle, uniformization with representatives [16] can be adapted to the analysis of tree-structured QBDs. However, for PH\(_d\) distributed service times, \( n \) uniformization steps and a single starting state, one would have to consider \( \mathcal{O}(d^n) \) states, which is practically infeasible. The same holds for techniques based on uniformization like in [28, 15].

**Definition 1.** A \( d \)-ary tree-structured QBD \( \mathcal{T} \) is a CTMC \((\mathcal{S}, \mathcal{R}, \mu_0)\) with state space \( \mathcal{S} = \{(x_1, \ldots, x_n) \mid n \in \mathbb{N} \land \forall i \leq n : x_i \in \{1, \ldots, d\}\}\). A state \( (x_1, \ldots, x_n) \) represents a queue of length \( n \) where jobs 1, 2, \ldots, \( n - 1 \) have been preemted in phase \( x_i \in \{1, \ldots, d\} \) and job \( n \) is currently in service in phase \( x_n \in \{1, \ldots, d\} \).

Transitions, represented by positive entries in \( \mathcal{R} \), can only occur between a state and its parent, its children and its siblings (cf. Fig. 2). For \( \vec{x}, \vec{y} \in \mathcal{S} \):

\[
\mathcal{R}(\vec{x}, \vec{y}) = \begin{cases} \small r_{x_{m+1},y} & \text{if } \vec{x} = (x_1, \ldots, x_m), \text{ and} \\
\small r_{x_{m+1},\uparrow} & \text{if } \vec{x} = (x_1, \ldots, x_{m+1}), \text{ and} \\
\small r_{x_{m},y_{m}} & \text{if } \vec{x} = (x_1, \ldots, x_m), \text{ and} \\
0 & \text{otherwise.} \end{cases}
\]

The underlying state space of a preemptive LIFO M\(|\text{PH}|1 \) queue with overall arrival rate \( L_\downarrow + R_\downarrow \) and service time distribution as depicted in Fig. 1 is the (binary) tree-structured QBD shown in Fig. 2, where \( r_1 \downarrow = L_\downarrow, r_2 \downarrow = R_\downarrow, r_1 \uparrow = l \uparrow, r_2 \uparrow = R \uparrow, r_{1,2} = L, r_{2,1} = R \). Note that, in contrast to ordinary trees, in tree-structured QBDs, transitions between siblings are allowed.

State \( \varnothing = (\cdot) \) represents the empty queue. Arriving jobs can either enter service phase 1 or 2, represented by the states (1) and (2). Due to the preemptive LIFO service discipline, a new job arrival causes the preemption of the job currently in service and the service phase of the preempted job needs to be stored. This results in a tree-structured state space.

**Fig. 2.** An example (binary) tree-structured QBD.
Fig. 3. A uniform CTMC.

Fig. 4. An interval abstraction.

Note that it has been shown in [29], that every tree-structured QBD can be embedded in a binary tree-structured Markov chain with a special structure.

A measure of interest for performance evaluation of M|PH|1 queues is: “if the queue is filled up to a certain level, what is the probability for the system to process all but \( k \) jobs within \( t \) time units?”. New jobs that arrive while serving older jobs should be completed as well. This cannot be answered with steady-state analysis that is typically conducted on such queues. Hence, we resort to computing time-bounded reachability probabilities on tree-structured QBDs.

3. Abstraction

Here, we discuss two abstraction techniques for CTMCs that preserve time-bounded reachability probabilities and allow for huge reductions of the state space. The first technique has been introduced in [23] and will be referred to as interval abstraction in the following. The second one is based on [24] and is referred to as MDP abstraction.

As a preprocessing step for both techniques, the given concrete CTMC is transformed into a weakly bisimilar uniform CTMC. This can be done in linear time and preserves time-bounded reachability probabilities. For \( d \)-ary tree-structured QBDs, the uniformization rate \( \lambda \) can be chosen as

\[
\max_{i \in \{1, \ldots, d\}} \left( r_i \uparrow + \sum_{j \in \{1, \ldots, d\}} (r_j \downarrow + r_{i,j}) \right).
\]

The self-loop probabilities need to be adapted accordingly. To explain the abstraction concepts, in the remainder of the section a finite-state uniform CTMC will be considered.

Interval abstraction. The main idea behind interval abstraction is to partition the state space and to keep track of the minimal and maximal probabilities for taking exactly one transition leading from a partition to a successor partition (possibly the same). In the abstract model, these minimal and maximal probabilities form the lower and upper bounds of transition probability intervals.

To exemplify how to obtain an abstraction of a concrete model, we consider the uniform CTMC depicted in Fig. 3. Given the partitioning \( \mathcal{A} \) defined by abstraction function \( \alpha : S \rightarrow \mathcal{A} \) with \( \alpha(s_i) = s, \alpha(u_i) = u \) and \( \alpha(v_i) = v \) for all \( i \in \{1, \ldots, 5\} \), the abstract model is depicted in Fig. 4 with abstract states \( s, u, \) and \( v \). To compute, say, the probability bounds for taking a transition from abstract state \( s \) to abstract state \( u \), the minimal and maximal probabilities to take corresponding transitions in the concrete model have to be determined. The minimal probability of such a transition in the concrete model is \( \frac{1}{5} \) for leaving \( s_4 \) towards \( u_2 \). The maximum is \( \frac{4}{5} \) as for \( s_2 \) there are two ways to reach a state in \( u \), the overall probability to do so is just the sum of the probabilities \( \left( \frac{1}{5} + \frac{2}{5} \right) \), and there is no other state in \( s \) for which there is a larger probability to reach states in \( u \). This yields the transition probability interval \( \left[ \frac{1}{5}, \frac{4}{5} \right] \) for the abstract transition from \( s \) to \( u \). The intervals for the other abstract transitions are computed similarly.

The probability to start in an abstract state is just the sum of probabilities to start in the represented concrete states (the same applies to MDP abstraction).
Consider ACTMC and the set of all paths with 5 23. The Fig. 5 yields the same statespace as for the interval abstraction, 30 37 added to the abstract state s. s decision process [37], however, the transition structure is quite different. The resulting abstract model is a uniform continuous-time Markov when infinitely many probability distributions have to be associated to an abstract state. Instead of keeping track of the extreme behavior in terms of minimal and maximal transition probabilities, for each MDP abstraction by Definition 2. we obtain 3 3. 4 other hand, when choosing ut [37] MDP abstraction to the example CTMC from Fig. 4, we obtain 3 3. 4. Further, let Path M be the set of all paths in M and let Pr inf M(X) (Pr sup M(X)) for some measurable set X Path M denote the infimum (supremum) of all probability measures that can be obtained by resolving the nondeterministic choice for transition probability distributions. For details on probability measures for ACTMCs, we refer to [23].

Example 1. Consider ACTMC M in Fig. 4 and the set of all paths with time-abstract prefix s s u u A = {s t0 s t1 u t2 u t3 u t4 ... | t i ∈ ]0, ∞[ for all i ∈ N}. As it is possible to decide not to take the self-loop in state s, we obtain Pr sup M(A) = 0. On the other hand, when choosing 3 5 for the self-loop probability in the initial state and 5 5 for moving from s to u after the first step, we obtain Pr sup M(σ) = 5 5 · 5 4 · 1 · 1 · ... = 12 25.

Definition 2. The interval abstraction abstr inf (C, A) of CTMC C = (δ, P, λ, μ0) with respect to partitioning A of δ, is given by (A, P, P, λ, μ0) where

- for all s', u' ∈ A,

\[ \tilde{P}(s', u') = \inf_{s \in s'} \sum_{u \in u'} P(s, u), \]

\[ \tilde{P}(s', u') = \min \left\{ 1, \sup_{s \in s'} \sum_{u \in u'} P(s, u) \right\}, \]

- for all u' ∈ A it holds μ0(u') = \sum_{u \in u'} μ0(u).

MDP abstraction. The idea behind MDP abstraction is to include sets of distributions for each state in the abstract model. Instead of keeping track of the extreme behavior in terms of minimal and maximal transition probabilities, for each concrete state that is represented by an abstract one, the transition probabilities are stored. Note that this technique is not applicable when infinitely many probability distributions have to be associated to an abstract state.

Applying MDP abstraction to the example CTMC from Fig. 3 yields the same state space as for the interval abstraction, however, the transition structure is quite different. The resulting abstract model is a uniform continuous-time Markov decision process [31] (CTMDP; see Fig. 5, dashed arcs connect states to the associated distributions). For s1 and s2, we obtain the same probability distributions for taking a transition to the sets of states that are mapped to s, u and v by abstraction function α, therefore, they can be collapsed in the abstract model. For all other states in s, a distinct distribution has to be added to the abstract state s.

Formally, a uniform CTMDP is a tuple (δ, A, P, λ, μ0) with set of states δ, action set A, the three-dimensional probability matrix P : δ × A × δ → [0, 1] with P(s, a, δ) = 1 for all s ∈ δ, a ∈ A, uniform exit rate λ ∈ ]0, ∞[ and initial distribution

\( \mu_0 \in \text{distr}(\delta) \). The set of transition distributions for CTMDP \( \mathcal{M} \) is given by \( T_M : \delta \rightarrow 2^{\text{distr}(\delta)} \) where for all \( s \in \delta \),
\[
T_M(s) = \{ P(s, a, \cdot) \in \text{distr}(\delta) \mid a \in A \}.
\]

ACTMCs and uniform CTMDPs are conservative extensions of uniform CTMCs, i.e., a uniform CTMC can be represented as an ACTMC with \( P = P_0 \) and as uniform CTMDP with \( |A| = 1 \). The notion of paths as well as the infimum (supremum) of probability measures on CTMDPs is just as for ACTMCs.

**Definition 3.** The MDP abstraction \( \text{abstr}_{\text{MDP}}(\mathcal{C}, \mathcal{A}) \) of CTMC \( \mathcal{C} = (\delta, P, \lambda, \mu_0) \) w.r.t. partitioning \( \mathcal{A} \) of \( \delta \) is given by \((A, \tilde{A}, \tilde{P}, \lambda, \tilde{\mu}_0)\) where:

- \( \tilde{A} = \{ a_{\mu'} \mid \mu' \in \text{distr}(A) \wedge \exists s \in \delta, \mu = P(s, \cdot) : \forall u' \in A : \mu'(u') = \sum_{u \in u'} \mu(u) \} \) is finite,
- for all \( s', u' \in S' \),
  \[
  \tilde{P}(s', a_{\mu'}, u') = \begin{cases} 
  \mu'(u') = \sum_{u \in u'} \mu(u) & \text{if } \mu = P(s, \cdot) \text{ for some } s \in s' \\
  0 & \text{otherwise}
  \end{cases}
  \]
- for all \( u' \in A \) it holds \( \tilde{\mu}_0(u') = \sum_{u \in u'} \mu_0(u) \).

**Nondeterminism.** Both abstract models have a nondeterministic component. In ACTMCs, the transition probabilities have to be chosen from the intervals in each step and in CTMDPs an action, i.e., a distribution, has to be chosen in a state. Depending on how these choices are made by the so-called scheduler (also-called strategy, policy, adversary) the system may behave differently. Also time-bounded reachability probabilities depend on the scheduler [31]. However, by computing the minimal and maximal reachability probabilities in the abstract model, one obtains lower and upper bounds for the value in the concrete model. If the partitioning of the state space has been chosen properly, enough information is preserved in the abstract model to guarantee the desired minimal/maximal reachability probabilities, also for the concrete model.

**Comparison.** First we intuitively explain the relation between both abstractions using the examples above. Then we formalize this relationship.

In principle, interval abstraction has more potential for reduction of the model’s size and MDP abstraction preserves more information from the original model. This can be observed in the diagram in Fig. 6 where the possible choices for transition probabilities to leave \( s \) towards \( u \) and \( v \), respectively, are plotted for both abstractions. The choices of MDP abstraction are marked by the concrete states they represent (cf. Fig. 5). For interval abstraction, all possible choices \( \mu \) are given by the intersection of the rectangle (all choices for \( \mu(u) \) and \( \mu(v) \) out of \([\frac{1}{2}, \frac{3}{2}]\) and \([\frac{1}{10}, \frac{3}{2}]\) and the trapezoid shape (the probability mass left for distribution amongst \( \mu(u) \) and \( \mu(v) \) after choosing the self-loop probability \( \mu(s) \) from \([0, \frac{1}{2}]\), i.e. \( 1 - [0, \frac{3}{2}] = [\frac{3}{2}, 1] \)).

Removing the dark area (all the behavior under a randomized scheduler in MDP abstraction) from that intersection yields the three marked triangles that represent all the behavior in interval abstraction that is not present in MDP abstraction.

We formalize this observation using the concept of probabilistic simulation and give a variant of the definition in [23] that is compatible with ACTMCs and uniform CTMDPs. Intuitively, simulation relations are used to describe relations between
Simulation as defined above can also be understood as a maximal flow problem. The propositions above suggest that as long as \( \text{abstr}_{\text{MDP}}(\mathcal{C}, \mathcal{A}) \) does exist and is not significantly larger than \( \text{abstr}_{\text{Int}}(\mathcal{C}, \mathcal{A}) \), MDP abstraction is to be favored since the abstract model is at least as accurate as in case of interval abstraction. Otherwise, interval abstraction would be the first choice.

**Lemma 1.** Let \( \mathcal{C} \) be a uniform CTMC with state space \( \mathcal{S} \) and let \( \mathcal{A} \) be a partitioning of \( \mathcal{S} \) such that there exists \( \text{abstr}_{\text{MDP}}(\mathcal{C}, \mathcal{A}) \), then:

\[
\mathcal{C} \leq \text{abstr}_{\text{MDP}}(\mathcal{C}, \mathcal{A}) \leq \text{abstr}_{\text{Int}}(\mathcal{C}, \mathcal{A}).
\]

**Proof.** The propositions \( \mathcal{C} \leq \text{abstr}_{\text{MDP}}(\mathcal{C}, \mathcal{A}) \) (and \( \mathcal{C} \leq \text{abstr}_{\text{Int}}(\mathcal{C}, \mathcal{A}) \)) have been shown in [24] (and [23]). It remains to be shown that \( \text{abstr}_{\text{MDP}}(\mathcal{C}, \mathcal{A}) = (\mathcal{A}, \mathcal{P}, \lambda, \mu_0) \leq (\hat{\mathcal{A}}, \hat{\mathcal{P}}, \hat{\lambda}, \hat{\mu}_0) = \text{abstr}_{\text{Int}}(\mathcal{C}, \mathcal{A}) \). Let \( \mathcal{R} \subseteq \mathcal{A} \times \mathcal{A} \) with \( s' \mathcal{R} s \) iff \( s' = \hat{s} \), that is, we compare abstract states representing the same set of concrete states. We show that for all \( s' \mathcal{R} s \) it holds that for any \( \mu' \in T_{\text{MDP}}(s') = T_{\text{abstr}_{\text{MDP}}(\mathcal{C}, \mathcal{A})}(s') \), there exists \( \hat{\mu} \in T_{\text{Int}}(\hat{s}) = T_{\text{abstr}_{\text{Int}}(\mathcal{C}, \mathcal{A})}(\hat{s}) \) with \( \hat{\mu'} \mathcal{R} \hat{\mu} \). As \( s' \mathcal{R} s \) iff \( s' = \hat{s} \), it suffices to show that \( T_{\text{MDP}}(s') \subseteq T_{\text{Int}}(\hat{s}) \) for all \( s' \in \mathcal{A} \).

By the definition of \( T_{\text{MDP}}(s') \), we have that for any \( \mu' \in \text{distr}(s') \) it holds \( \mu' \in T_{\text{MDP}}(s') \) iff there exists \( s \in s' \) with \( \mu'(u') = \sum_{u \in u'} \mu(u) \) for \( \mu = \mathcal{P}(s, \cdot) \). For any \( s \in s' \) and distribution \( \mu \in \mathcal{P}(s, \cdot) \), by the definition of interval abstraction,
holds for all $u' \in \mathcal{A}$:
\[
\hat{P}(s, u') = \inf_{\delta \in \mathcal{E}} \sum_{u \in u'} P(\delta, u) \\
\leq \sum_{u \in u'} \mu(u) = \mu'(u') \\
\leq \min \left(1, \sup_{\delta \in \mathcal{E}} \sum_{u \in u'} P(\delta, u) \right) = \tilde{P}(s, u').
\]

Further, the set $T_{\text{int}}(s')$ contains all distributions respecting the intervals given by $\hat{P}$ and $\tilde{P}$ in the interval abstraction, i.e.
\[
T_{\text{int}}(s') = \{ \mu' \in \text{distr}(s') \mid \hat{P}(s', u') \leq \mu'(u') \leq \tilde{P}(s', u') \text{ for all } u' \in \mathcal{A} \},
\]
and thus it follows $T_{\text{int}}(s') \subseteq T_{\text{int}}(s')$ for all $s' \in \mathcal{A}$, concluding the proof. $\square$

Note that simulation preserves time-bounded reachability probabilities, that is, the probability for reaching a set of goal states $B \subseteq S$ within $t \in \mathbb{R}_{\geq 0}$ time units. Formally, time-bounded reachability is defined by the set of paths
\[
\diamondsuit \leq t B = \{ \sigma \in \text{Path} \mid \exists \sigma' \leq t : \sigma @ \sigma' \in B \}.
\]

In order to compare the reachability probabilities in two (abstract) models $\mathcal{M}$ and $\mathcal{M}'$ with sets of states $\delta$ and $\delta'$, we rely on the notion of compatible sets of goal states. We say that $B \subseteq \delta$ and $B' \subseteq \delta'$ are compatible, if for any $s \in \delta$, $s' \in \delta'$ with $s \leq s'$ it holds $s \in B \iff s' \in B'$. For more details, see [23].

**Theorem 1** (See [24]). Let $\mathcal{M}$ and $\mathcal{M}'$ be two CTMDPs/ACTMCs with $\mathcal{M} \leq \mathcal{M}'$ and let $t \in \mathbb{R}_{\geq 0}$ and $B, B'$ be compatible sets of goal states. Then
\[
Pr_{\text{int}}^{\mathcal{M}'}(\diamondsuit \leq t B') \leq Pr_{\text{int}}^{\mathcal{M}}(\diamondsuit \leq t B) \leq Pr_{\text{sup}}^{\mathcal{M}'}(\diamondsuit \leq t B) \leq Pr_{\text{sup}}^{\mathcal{M}}(\diamondsuit \leq t B').
\]

### 4. Partitioning a tree-structured QBD

In order to apply abstraction to a tree-structured QBD, first a suitable partitioning of the state space has to be found. Recall that the state space of the tree-structured QBD results from a PH service distribution with preemptive LIFO scheduling. Every state of the tree represents (i) the number of jobs in the queue, (ii) the service phases of the preempted jobs, (iii) the phase of the job that is currently in service and (iv) the precise order of jobs in the queue. The states with $m$ jobs, that are situated in the same layer of the tree, have the form $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ where $x_i$ gives the service phase of the $i$th job in the queue. We abbreviate the prefix of $\mathbf{x}$ of length $n$ by $\mathbf{x}_{\downarrow n}$ and the number of jobs in $\mathbf{x}$ in phase $i$ by $\#_i \mathbf{x}$.

In the following, we present abstractions that preserve some of the above mentioned properties from (i) to (iv). In order to obtain a finite abstract state space, we also have to apply counter abstraction to the infinite-state tree-structured QBD, i.e., we cut the state space at layer $n$ (denoted cut level in the following), which implies that property (i) is only preserved for less than $n$ customers.

Let $\mathcal{T}$ be a tree-structured QBD with state space $\delta$. For partitioning scheme $\mathcal{ps} \in \{\text{tree}, \text{grid}, \text{qbld}, \text{bd}\}$ and cut level $n \in \mathbb{N}^+$, we define partitioning $\mathcal{A}_{\mathcal{ps}, n}$ by abstraction function $\alpha_{\mathcal{ps}, n} : \delta \rightarrow \mathcal{A}_{\mathcal{ps}, n}$.

For example, when using the grid scheme, all states with the same number of jobs (up to the $n$th queued job) in phases 1, 2, $\ldots$, $d$, respectively, are grouped.

Scheme bd preserves property (i) only, grid additionally preserves (ii), whereas qbld additionally preserves (iii). Scheme tree preserves all properties and grid preserves all but (iv). Interval abstractions of the binary tree-structured QBD from
Fig. 8. Interval abstractions \( M_{\text{tree}, 2} \), \( M_{\text{qgrid}, 3} \), \( M_{\text{grid}, 3} \), \( M_{\text{qbd}, 3} \) and \( M_{\text{bd}, 3} \) for \( L^\uparrow + L^\omega = R^\uparrow \), \( L + L^\circ = R \), and \( L^\circ = L^\omega + L^\circ \).

Fig. 2 are shown in Fig. 8. From those, it becomes clear that the partitioning schemes are named after the structure of the abstract models. Schemes \( \text{bd} \) and \( \text{qbd} \) yield chain-like structures similar to (quasi)-birth death processes, where the \( \text{qbd} \) scheme enhances the \( \text{bd} \) scheme by storing the phase of the job currently in service. Similarly the \( \text{qgrid} \) scheme enhances the \( \text{grid} \) scheme. For sufficiently large \( n \), the size of the abstract models decreases in the order of the partitionings as presented in the first row in Table 1. Positive results on the relationship of abstractions induced by the partitionings proposed above are given in the following lemma. Note that for \( \text{grid} \) and \( \text{qbd} \) abstraction, a formal relationship in terms of probabilistic simulation cannot be established as each abstraction preserves information that is not preserved by the other, (ii) and (iii) respectively.

Please cite this article in press as: D. Klink, et al., Time-bounded reachability in tree-structured QBDs by abstraction, Performance Evaluation (2010), doi:10.1016/j.peva.2010.04.002
Table 1
Sizes of abstract models and average numbers of distributions per state (for \( d > 1 \)).

<table>
<thead>
<tr>
<th>( \mathcal{A}_{ps,n} )</th>
<th>tree</th>
<th>qgrid</th>
<th>grid</th>
<th>qbd</th>
<th>bd</th>
</tr>
</thead>
<tbody>
<tr>
<td>#distrs</td>
<td>( \Theta(d^3) )</td>
<td>( \Theta(d \cdot \left( \frac{d^2 + d}{d} \right)) )</td>
<td>( \Theta\left( \frac{d^2 + d}{d} \right) )</td>
<td>( \Theta(d \cdot n) )</td>
<td>( \Theta(n) )</td>
</tr>
</tbody>
</table>

Lemma 2. Let \( T \) be a tree-structured QBD, \( x \in \{\text{Int, MDP}\} \) and \( n \in \mathbb{N}^+ \), then:

1. \( \text{abstr}_x(T, \mathcal{A}_{\text{tree},n}) \preceq \text{abstr}_x(T, \mathcal{A}_{\text{qgrid},n}) \)
2. \( \text{abstr}_x(T, \mathcal{A}_{\text{qgrid},n}) \preceq \text{abstr}_x(T, \mathcal{A}_{\text{grid},n}) \)
3. \( \text{abstr}_x(T, \mathcal{A}_{\text{grid},n}) \preceq \text{abstr}_x(T, \mathcal{A}_{\text{qbd},n}) \)

Proof. Let \( T = (\delta, R, \mu_0) \) be a \( d \)-ary tree-structured QBD with \( R \) induced by rates \( r_i, r_i^\uparrow \) and \( r_i, r_i^\downarrow \) for \( i, j \in \{1, \ldots, d\} \) as in Definition 1. In the following, we consider the uniformized CTMC with exit rate \( \lambda \). Let \( p_i^\downarrow = \frac{r_i}{\lambda}, p_i^\uparrow = \frac{r_i}{\lambda}, p_i^\downarrow = \frac{r_i}{\lambda} \) for all \( i, j \in \{1, \ldots, d\} \). Further, we define \( p_i^\uparrow = \sum_{j=0}^d p_i^\downarrow, p_m^\uparrow = \sum_{j=0}^d p_m^\downarrow \) and indicator function \( 1 : \mathbb{N} \times \mathbb{N} \to \{0, 1\} \) with \( 1(i,j) = 1 \) iff \( i = j \).

We will now show that \( \text{abstr}_{\text{MDP}}(T, \mathcal{A}_{\text{qbd},n}) \preceq \text{abstr}_{\text{MDP}}(T, \mathcal{A}_{\text{bd},n}) \) holds. For simplicity, by \( T_{ps} \), we denote \( T_{\text{abstr}_{\text{MDP}}(T,A_{ps,n})} \) for \( ps \in \{\text{qbd, bd}\} \). Let \( R \in \mathcal{A}_{\text{qbd},n} \times \mathcal{A}_{\text{bd},n} \) such that \( s' \, R \, \tilde{s} \) iff there exists \( s \in \delta \) such that \( s' = \omega_{\text{qbd},n}(s) \) and \( \tilde{s} = \omega_{\text{bd},n}(s) \). More precisely, we define

\[
R = \{(0, 0)\} \cup \{(m; i), (m)\} \mid m \in \{1, \ldots, n\}, i \in \{1, \ldots, d\}.
\]

To show that \( R \) is indeed a simulation relation, we have to prove that for \( s' R \tilde{s} \): for any \( \mu' \in T_{\text{qbd}}(s') \) there exists \( \tilde{\mu} \in T_{\text{bd}}(\tilde{s}) \) and weight function \( \Delta \) such that for all \( u' \in \mathcal{A}_{\text{qbd},n} \) and \( \tilde{u} \in \mathcal{A}_{\text{bd},n} \), conditions (a)–(c) from Definition 4 hold. First, let us consider \( [0]R[0] \). The set

\[
T_{\text{qbd}}([0]) = \{([1; 1] \mapsto p_1^\downarrow, \ldots, [1; d] \mapsto p_d^\downarrow, [0] \mapsto 1 - p_d^\downarrow)\}
\]

is a singleton. For this distribution \( \mu' \in T_{\text{qbd}}([0]) \), we “choose” \( \tilde{\mu} \) from the singleton set

\[
T_{\text{bd}}([0]) = \{([1] \mapsto p_1^\downarrow, [0] \mapsto 1 - p_1^\downarrow)\}
\]

and weight function \( \Delta([0], [0]) = \mu'([0]) \) and

\[
\Delta([m; i], [m']) = \begin{cases} 
\mu'([m; i]) & \text{for all } 0 < m < m' < n \text{ and } i \in \{1, \ldots, d\} \\
0 & \text{otherwise.}
\end{cases}
\]

Then, conditions (a)–(c) from Definition 4 are fulfilled:

(a) \( \Delta(s', \tilde{s}) > 0 \Rightarrow s' \, R \, \tilde{s} \)
(b) \( \Delta([1; 1], [1; i]) = \sum_{m=0}^d \Delta([1; i], [m]) = \Delta([1; i], [1]) = \mu'([1; i]) = p_i^\downarrow \) for all \( i \in \{1, \ldots, d\} \), and \( \Delta([0], \mathcal{A}_{\text{bd},n}) = \Delta([0], [0]) = \mu'([0]) = 1 - p_1^\downarrow \)
(c) \( \Delta(\mathcal{A}_{\text{qbd},n}, [1]) = \sum_{i=1}^d \Delta([1; i], [1]) = \sum_{i=1}^d p_i^\downarrow = \tilde{\mu}([1]) = p_1^\downarrow \) and \( \Delta(\mathcal{A}_{\text{bd},n}, [0]) = \Delta([0], [0]) = \mu'([0]) = 1 - p_1^\downarrow = \tilde{\mu}([0]) \).

Now, we consider \([m; i]R[m]\) for \( 0 < m < n \) and \( i \in \{1, \ldots, d\} \). For any distribution \( \mu' \) in

\[
T_{\text{qbd}}([m; i]) = \{([m+1; 1] \mapsto p_1^\downarrow, \ldots, [m+1; d] \mapsto p_d^\downarrow, \\
[m; 1] \mapsto p_{i,1}, \ldots, [m; i-1] \mapsto p_{i,i-1}, \\
[m; i] \mapsto 1 - p_d^\downarrow - p_{i,m-1} \uparrow - p_i^\uparrow, \\
[m; i+1] \mapsto p_{i,i+1}, \ldots, [m; d] \mapsto p_{i,d}, \\
[x_{m-1} \in \{1, \ldots, d\}]\}
\]

we choose the corresponding distribution \( \tilde{\mu} \) with the same \( x_{m-1} \in \{1, \ldots, d\} \) as for \( \mu' \) from

\[
T_{\text{bd}}([m]) = \{([m+1] \mapsto p_1^\downarrow, \\
[m-1] \mapsto p_{m-1}^\uparrow, \\
[m] \mapsto 1 - p_d^\downarrow - p_{m-1}^\uparrow \\
[x_{m-1} \in \{1, \ldots, d\}]\}
\]

and further let weight function \( \Delta \) be defined as before. Then conditions (a)–(c) from Definition 4 are fulfilled:

(a) \( \Delta(s', \tilde{s}) > 0 \Rightarrow s' \, R \, \tilde{s} \)
(b) $\Delta([m; i], \mathcal{A}_{bd,n}) = \Delta([m; i], [m]) = \mu'([m; i])$ for all $0 < m < n$ and $i \in \{1, \ldots, d\}$.

c. $\Delta(\mathcal{A}_{qbd,n}, [m + 1]) = \sum_{i=1}^{d} \Delta([m + 1; i], [m + 1]) = \sum_{i=1}^{d} \mu'( [m + 1; i]) = \sum_{i=1}^{d} p_i \downarrow = p \downarrow = \tilde{\mu}([m + 1]),$ and $\Delta(\mathcal{A}_{qbd,n}, [m - 1]) = \sum_{i=1}^{d} \Delta([m - 1; i], [m - 1]) = \sum_{i=1}^{d} \mu'([m - 1; i]) = \mu'([m - 1; x_{n-1}]) = p_{x_{n-1}} \uparrow = \tilde{\mu}([m - 1]),$ and $\Delta(\mathcal{A}_{qbd,n}, [m]) = \sum_{i=1}^{d} \Delta([m; i], [m]) = \sum_{i=1}^{d} \mu'([m; i]) = 1 - p \downarrow - p_{x_{n-1}} \uparrow - p_i + \sum_{j=1}^{d} p_{ij} = 1 - p \downarrow - p_{x_{n-1}} \uparrow = \tilde{\mu}([m]).$

Finally, we consider $[n; i]|\mathcal{R}[n]$ for $i \in \{1, \ldots, d\}$. For any distribution $\mu'$ in $\mathcal{T}_{qbd}(n; i) = \{$\{[n; 1] \mapsto p_{1 \downarrow} + p_{1,1}, \ldots, [n; i - 1] \mapsto p_{i-1 \downarrow} + p_{i,i-1} [n; i] \mapsto 1 - p \downarrow + p_{i \downarrow} - p_i + p_{i-1} - p_{x_{n-1}} \uparrow, [n; i + 1] \mapsto p_{i+1 \downarrow} + p_{i,i+1}, \ldots, [n; d] \mapsto p_d \downarrow + p_{d,i} \uparrow, [n; d-1; x_{n-1}] \mapsto p_{x_{n-1}} \uparrow, [x_{n-1} \in \{1, \ldots, d\}]$\}$ $\cup \{$\{[n; 1] \mapsto p_{1 \downarrow} + p_{1,1} + \mathbf{1}(x_{n-1} = 1) \cdot p_{x_{n-1}} \uparrow, \ldots, [n; i - 1] \mapsto p_{i-1 \downarrow} + p_{i,i-1} + \mathbf{1}(x_{n-1} = 1 - 1) \cdot p_{x_{n-1}} \uparrow [n; i] \mapsto 1 - p \downarrow + p_{i \downarrow} - p_i + p_{i-1} - p_{x_{n-1}} \uparrow + \mathbf{1}(x_{n-1} = 1) \cdot p_{x_{n-1}} \uparrow, [n; i + 1] \mapsto p_{i+1 \downarrow} + p_{i,i+1} + \mathbf{1}(x_{n-1} = 1 + 1) \cdot p_{x_{n-1}} \uparrow, \ldots, [n; d] \mapsto p_d \downarrow + p_{d} \uparrow, [n; d-1; x_{n-1}] \mapsto p_{x_{n-1}} \uparrow, [x_{n-1} \in \{1, \ldots, d\}]$\}$

with $\mu'([n-1; x_{n-1}]) > 0$ we choose the corresponding distribution $\tilde{\mu}$ with $\tilde{\mu}([n-1; x_{n-1}]) > 0$ from $\mathcal{T}_{bd}([m]) = \{$\{[n] \mapsto 1 - p_{x_{n-1}} \uparrow, [n-1] \mapsto p_{x_{n-1}} \uparrow [x_{n-1} \in \{1, \ldots, d\}]$\}$ $\cup \{$\{[n] \mapsto 1\}$\}$ and otherwise, if $\sum_{i=1}^{d} \mu'([n-1; i]) = 0$, we choose $\tilde{\mu} = \{[n] \mapsto 1\}$. Further let weight function $\Delta$ be as before. Then conditions (a)–(c) from Definition 4 hold:

(a) $\Delta(s', s) > 0 \Rightarrow s' \preceq s$.

(b) $\Delta([n; i], \mathcal{A}_{bd,n}) = \sum_{m=0}^{n} \Delta([n; i], [m]) = \Delta([n; i], [n]) = \Delta([n; i], [n])$ for all $i \in \{1, \ldots, d\}$.

(c) For all $\mu' \in \mathcal{T}_{qbd}(n; i)$ with $\mu'([n-1; x_{n-1}-1]) > 0$ we calculate:

$\Delta(\mathcal{A}_{qbd,n}, [n]) = \sum_{m=1}^{d} \Delta([n; i], [m]) = \sum_{m=1}^{d} \mu'([m; i]) = 1 - p \downarrow - p_{x_{n-1}} \uparrow + \sum_{j=1}^{d} (p_{i \downarrow} + p_{ij}) = 1 - p_{x_{n-1}} \uparrow = \tilde{\mu}([n]),$ and $\Delta(\mathcal{A}_{qbd,n}, [n - 1]) = \sum_{m=1}^{d} \Delta([n - 1; i], [n - 1]) = \sum_{m=1}^{d} \mu'([n - 1; i]) = \mu'([n - 1; x_{n-1}]) = p_{x_{n-1}} \uparrow = \tilde{\mu}([n - 1]),$.

For all $\mu' \in \mathcal{T}_{qbd}(n; i)$ with $\mu'([n-1; i]) = 0$ for all $i \in \{1, \ldots, d\}$ we calculate:

$\Delta(\mathcal{A}_{qbd,n}, [n]) = \sum_{m=1}^{d} \Delta([n; i], [m]) = \sum_{m=1}^{d} \mu'([m; i]) = 1 - p \downarrow - p_{x_{n-1}} \uparrow + \sum_{j=1}^{d} (p_{i \downarrow} + p_{ij}) = 1 - p_{x_{n-1}} \uparrow + \sum_{j=1}^{d} \mathbf{1}(x_{n-1} = 1) \cdot p_{x_{n-1}} \uparrow = 1 = \tilde{\mu}([n]),$.

This shows that $\mathcal{R}$ as defined above is indeed a simulation relation, that is, $[m; i] \preceq [m]$ for all $m \in \{0, \ldots, n\}, i \in \{1, \ldots, d\}$. It remains to show that for the initial distributions $\mu'_0$ and $\mu_0$ of the two abstractions it holds $\mu'_0 \preceq \mu_0$. With $\Delta$ defined by $\Delta([0], [0]) = \mu'_0([0]) = \mu_0([0])$ and

$$\Delta([m; i], [m']) = \begin{cases} \mu'_0([m; i]) & \text{for all } 0 < m = m' < n \text{ and } i \in \{1, \ldots, d\} \\ 0 & \text{otherwise,} \end{cases}$$

conditions (a)–(c) from Definition 4 hold:

(a) $\Delta(s', s) > 0 \Rightarrow s' \preceq s$ follows directly from the definition of $\Delta$.

(b) $\Delta([m; i], \mathcal{A}_{bd}) = \Delta([m; i], [m]) = \mu'_0([m; i])$ for all $m \in \{0, \ldots, n\}, i \in \{1, \ldots, d\}$.

(c) $\Delta(\mathcal{A}_{qbd,n}, [m]) = \sum_{i=1}^{d} \Delta([m; i], [m]) = \sum_{i=1}^{d} \mu'_0([m; i]) = \sum_{i=1}^{d} \mu_{\mathcal{A}_{qbd,n}}([s;m; i]) = \sum_{i=1}^{d} \mu_{\mathcal{A}_{bd,n}}([s;m; i]) = \mu'_0([m]).$

This concludes the proof for $\text{abstr}_{\text{MDP}}(\mathcal{T}, \mathcal{A}_{qbd,n}) \preceq \text{abstr}_{\text{MDP}}(\mathcal{T}, \mathcal{A}_{bd,n})$. Please cite this article in press as: D. Klink, et al., Time-bounded reachability in tree-structured QBDs by abstraction, Performance Evaluation (2010), doi:10.1016/j.peva.2010.04.002.
Now, we prove that this also holds when applying interval abstraction, i.e., we show \( \text{abstr}_{\text{int}}(\mathcal{T}, \mathcal{A}_{\text{qbd}, n}) \preceq \text{abstr}_{\text{int}}(\mathcal{T}, \mathcal{A}_{\text{bd}, n}) \). Let \( \mathcal{P}'_{\varepsilon} \) and \( \mathcal{P}''_{\varepsilon} \) be the lower and upper bound transition matrices for \( \text{abstr}_{\text{int}}(\mathcal{T}, \mathcal{A}_{\text{qbd}, n}) \) and let \( \mathcal{P}'_{u} \) and \( \mathcal{P}''_{u} \) be the ones for \( \text{abstr}_{\text{int}}(\mathcal{T}, \mathcal{A}_{\text{bd}, n}) \).

The same relation \( \mathcal{R} \) as defined in the proof for MDP abstraction is also a simulation relation in this setting. For \([0] \mathcal{R} [0]\) the proof is exactly as before as \( \mathcal{T}_{\text{qbd}}([0]) \) and \( \mathcal{T}_{\text{bd}}([0]) \) are the same singleton sets as for MDP abstraction. For \( 0 < m < n \) and \( i \in \{1, \ldots, d\} \) we now show that for any \( \mu' \in \mathcal{T}_{\text{qbd}}([m, i]; \cdot) \) we choose the corresponding distribution \( \tilde{\mu} \) from

\[
\mathcal{T}_{\text{bd}}([m]) = \{ \mathcal{P}(m), \cdot \in \text{distr}(\mathcal{A}_{\text{bd}, n}) \}
\]

\[
| \mathcal{P}'(m, [m + 1]) = \mathcal{P}(m, [m + 1]) = \mathcal{P}'(m, [m + 1]) = p \downarrow
\]

\[
\mathcal{P}'(m, [m]) = \min_{j=1,\ldots,d} (1 - p_j \downarrow - p_j \uparrow) \leq \mathcal{P}(m, [m])
\]

\[
\mathcal{P}'(m, [m - 1]) = \min_{j=1,\ldots,d} (p_j \uparrow) \leq \mathcal{P}(m, [m - 1])
\]

\[
\mathcal{P}'(m, [m - 1]) = \max_{j=1,\ldots,d} (p_j \downarrow)
\]

Finally, we consider \([n] \mathcal{R} [n]\) for \( i \in \{1, \ldots, d\} \). For any distribution \( \mu' \) in

\[
\mathcal{T}_{\text{qbd}}([n, i]; \cdot) = \{ \mathcal{P}(n, i, \cdot) \in \text{distr}(\mathcal{A}_{\text{qbd}, n}) \}
\]

\[
| \mathcal{P}'(n, i, [n, j]) = p_j \downarrow + p_{i,j} \leq \mathcal{P}(n, i, [n, j])
\]

\[
\mathcal{P}'(n, i, [n, i]) = 1 - p_j \downarrow + p_{i-j} - p_i \uparrow + p_{i,i}
\]

\[
\mathcal{P}'(n, i, [n - 1, i]) = 1 - p_j \downarrow + p_{i-j} - p_i \uparrow + p_{i,i}
\]

we choose the corresponding distribution \( \tilde{\mu} \) from

Please cite this article in press as: D. Klink, et al., Time-bounded reachability in tree-structured QBDs by abstraction, Performance Evaluation (2010), doi:10.1016/j.peva.2010.04.002
Lemma 2. If the distributions per state never exceed bounds for gradually increasing time bounds up to a point where the system approaches an equilibrium. The average number we stress that our abstractions guarantee that these bounds are always safe, i.e., the results for the concrete model lie within the computed bounds, cf. Theorem 1. To get an impression of how the system evolves over time, we compute probability bounds for gradually increasing time bounds up to a point where the system approaches an equilibrium. The average number of distributions per state never exceeds \( d \) (cf. second row in Table 1). Since we investigate phase-type service distributions of the order \( d \leq 5 \), with MDP abstraction, we obtain models that are not much larger than the ones obtained by interval abstraction. Due to Lemma 1, we therefore stick to MDP abstraction in the following. Abstract models are generated using a Java tool, and the analysis is done using MRMC version 1.4 [33]. All experiments were run on a desktop computer (Athlon X2 3800+, 2 GB RAM).

5. Experimental results

We compute lower and upper bounds for the probability to reach a certain set of states within a given amount of time. We stress that our abstractions guarantee that these bounds are always safe, i.e., the results for the concrete model lie within the computed bounds, cf. Theorem 1. To get an impression of how the system evolves over time, we compute probability bounds for gradually increasing time bounds up to a point where the system approaches an equilibrium. The average number of distributions per state never exceeds \( d \) (cf. second row in Table 1). Since we investigate phase-type service distributions of the order \( d \leq 5 \), with MDP abstraction, we obtain models that are not much larger than the ones obtained by interval abstraction. Due to Lemma 1, we therefore stick to MDP abstraction in the following. Abstract models are generated using a Java tool, and the analysis is done using MRMC version 1.4 [33]. All experiments were run on a desktop computer (Athlon X2 3800+, 2 GB RAM).

Recall from Section 2 that we analyze the probability of an \( M|PH|1 \) queue with preemptive LIFO scheduling, with a given number of waiting jobs, to process all but \( k \) jobs within \( t \) time units, cf. Fig. 9. We carry out the following six experiments. We also provide the utilizations for each setting, i.e., the fraction of time the server is busy serving jobs.

**E1.** Firstly, we compare the precision of the abstractions induced by the partitionings presented in the previous section, check if the results are consistent with Lemma 2 and provide results for two sets of rates. We choose a low cut level for all partitioning schemes and \( k = 0 \). Note that \( k = 0 \) is a special case as the ordering of the jobs is of no relevance in the concrete model as all jobs need to be served anyway before \( k \) is reached. Hence, for all partitioning schemes preserving the number of customers per phase, as tree, qgrid and grid, the imprecisions result from the choice of the cut level.

**E2.** In the second experiment, we investigate the influence of the cut level on the imprecisions occurring for the special case \( k = 0 \). We focus on the grid scheme in this experiment.

**E3.** In a third experiment we compare the precision of the grid-abstractions for varying goal levels \( k \) and fixed cut level.

**E4.** We then present a refinement of the grid abstraction, where the tree is built in full breadth up to level \( c \) and the grid abstraction is applied for higher levels. We present results for varying \( c \), fixed \( k = 4 \) and given cut level.

**E5.** In an additional experiment, we choose the cut levels such that the resulting abstract state spaces are of approximately the same size. Apart from the quality of the results, we compare the time it takes to generate and analyze the abstract models.
E6. To emphasize the generality of our approach we present results for the refined grid abstraction, as presented in Experiment 4, for a phase-type 5 distribution.

As standard parameters we choose $L_\downarrow = 2, R_\downarrow = 3, L = 4, R = 5, L_\uparrow = 7.5, R_\uparrow = 10$, with a resulting utilization of $\rho = 57\%$.

E1. Partitioning schemes:

We compare three abstractions, $\mathcal{M}_{\text{tree},12}$, $\mathcal{M}_{\text{grid},12}$ and $\mathcal{M}_{\text{bd},12}$, where we choose initial states that simulate $(1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2)$ in the concrete model, namely $[1, 2, 1, 2, 1, 2, 1, 2], [4, 4]$ and $[8]$ respectively, and take the empty state as goal state. The resulting lower and upper probability bounds, for these abstractions are shown in Fig. 10(top) where the x-axis shows the time bounds. However, the difference between the obtained upper and lower bound is rather large for medium and large time bounds, since the cut level 12 is too close to the initial state. The second best abstraction is grid abstraction, that is almost as good, especially for low time bounds. The bd abstraction, while having the least memory requirements, is clearly outperformed on the whole range of time bounds. Note that for large time bounds results are not as bad as for medium ones. This comes from the fact that upper bounds are derived by assuming that jobs are processed rather quickly while for lower bounds it is assumed that jobs are processed slowly. However, in both cases all jobs are processed eventually, due to a utilization which is less than one. The results we described above reflect the relation between the different abstractions as stated in Lemma 2.

The results for the other partitionings are presented in Fig. 10(bottom). The initial states $[4, 4, 2], [4, 4]$ and $[8, 2]$ simulate $(1, 2, 1, 2, 1, 2, 1, 2)$. Qbd abstraction performs significantly better than bd abstraction (top), but still is outperformed by grid abstraction for low and medium time bounds. For large time bounds (starting at about 4.7), qbd abstraction yields better lower bounds, even though the state space is significantly smaller than in case of grid abstraction. This comes from
the fact that qbd abstraction is aware of the job that is currently processed. If that job is processed rather quickly, a transition to a higher level (away from the cut level) is more probable. This shows that grid and qbd abstractions are incomparable (cf. Lemma 2). Finally, combining the advantages of grid and qbd yields the inner pair of curves (qgrid). This scheme outperforms the other two abstractions and comes close to the tree scheme’s results (top), however, with a drastically lower memory consumption (with a factor of 217).

Now, we consider an alternative set of parameters where rates are much more diverse: \( L^\downarrow = 1, R^\downarrow = 5, L = 3, R = 15, L^\uparrow = 8, R^\uparrow = 20 \), with a resulting utilization of \( \rho = 46\% \). As Fig. 11 indicates, both bd and qbd abstractions perform badly with these parameters. The reason is that only the total number of jobs is stored and therefore in the best (worst) case, it is assumed that only short (long) jobs are present in the queue. On the contrary, grid and qgrid abstraction perform excellent (lower and upper bounds are overlapping in the graph).

**E2. Influence of the cut level:**

In our second experiment, we investigate the influence of the cut level on the quality of the results. We show upper and lower bounds on the probability to reach the empty state from initial state \([4,4]\) with grid abstraction and increasing cut level. The parameters are chosen as in experiment 1, except \( L^\downarrow = \frac{10}{3} \) and \( R^\downarrow = 5 \) are larger, resulting in a very high utilization of \( \rho = 96\% \) and, hence, a much larger range on the time-axis.

Fig. 12 shows that grid abstraction is capable of providing lower and upper bounds that differ marginally, for large enough cut levels. As all the jobs need to be served to reach the empty state, the ordering of the jobs does not matter. Hence, for cut level \( n \to \infty \), grid abstraction contains all the necessary information, if the empty state is chosen as goal state.

Another aspect we are covering with this experiment, is how the quality of results depends on the cut level for different utilizations. Varying the utilization can be done, e.g., by varying the arrival rate \( L^\downarrow + R^\downarrow \). By decreasing the arrival rate by a factor of \( \frac{2}{3} \), we obtain a utilization of 32\%, a factor of \( \frac{1}{3} \) results in 64\% utilization and increasing the arrival rate by \( \frac{1}{3} \) and \( \frac{2}{3} \) respectively yields utilizations of 128\% and 159\%. For these parameter sets, Fig. 13 shows lower and upper bounds for reaching the empty state from \([4,4]\) within \( t = 5 \) time units for cut levels from 8 to 16. As expected, probabilities for reaching the empty state decrease with increasing utilization; the probability masses attraction towards the cut level increases. Further, it can be observed (e.g. for cut level 16) that for small and large utilizations, the lower and upper bounds converge considerably faster than for utilization \( \rho \approx 100\% \).

**E3. Influence of the goal level:**

We compare results obtained with grid abstraction for a large enough cut level for increasing goal levels \( k \). Fig. 14 shows that with growing goal levels the difference between upper and lower probability bounds increases. This is due to the fact, that in this setting grid abstraction does not contain all necessary information, as the ordering of the jobs in the queue influences the time that is needed to serve all but \( k \) jobs.

**E4. Refinement:**

To obtain more precise results, the grid partitioning scheme is refined. Lack of information on the first jobs in the queue leads to less tight bounds. Hence, the abstract states are refined according to the ordering of the first jobs. We define \( \alpha_{grid,c,n} \) such that states are merged for which the order of the first \( c \) job phases is the same and where the numbers of job phases
Fig. 12. E2: Upper and lower bounds for grid abstractions with increasing cut levels and \( k = 0 \).

Fig. 13. E2: Upper and lower bounds for grid abstraction for a fixed time bound \( t = 5 \), increasing cut levels and parameter sets with utilizations from 32\% to 159\%.

for jobs \( c + 1 \) up to \( n \) are equal as well:

\[
\alpha_{\text{grid},c,n}(\vec{x}) = \begin{cases} 
[\vec{x}_d; \#_1(x_{c+1}, \ldots, x_n), \ldots, \#_d(x_{c+1}, \ldots, x_n)] & \text{if } m < n, \\
[\vec{x}_d; \#_1(x_{c+1}, \ldots, x_n), \ldots, \#_d(x_{c+1}, \ldots, x_n)] & \text{otherwise}.
\end{cases}
\]

This still is a very natural partitioning scheme. For example, for a binary tree-structured QBD, \( \alpha_{\text{grid},2,8}((1)) = [(1); 0, 0] \) and \( \alpha_{\text{grid},2,8}((1, 2, 1, 1, 2, 1)) = [(1, 2); 3, 1] \).

In Fig. 15, results for goal level \( k = 4 \) and refinement levels \( c \in \{0, 2, 4\} \) are shown. If the refinement level \( c \) equals the goal level \( k \), and if the cut level is large enough, the obtained lower and upper bounds are extremely tight.

E5. Similar sized state spaces:

While in the first experiment the cut level remained constant, we now compare the different abstractions, while keeping the size of the abstract state space approximately the same. We select the same initial states and the same goal state as in the first experiment, as well as the first set of rates. The goal of this experiment is to determine which properties are more important for the quality of the results, given a maximal number of states.

The cut level, the number of states in the abstract model and the time for generating the abstract state space and for computing probability bounds are given in the table in Fig. 16. While tree and bd abstraction use the least computation time, their results, shown in Fig. 16, are rather imprecise compared to the other abstractions. Long-term behavior is not captured.
very well by tree abstraction as the cut level is close to the initial state. For small time bounds, bd abstraction is the worst. Yet, it catches up with increasing time bounds, as it has the largest depth and therefore very little probability mass is lost in the cut level. However, neither one can compete with grid and qgrid abstractions for which the lower and upper bounds are almost identical. Here, the grid scheme is favorable as the computation time is 12% smaller than for qgrid.

**E6. Phase-type 5 service distribution:**

To emphasize the generality of our approach, we present results on a M|PH$_5$|1 queue with preemptive LIFO and an utilization of $\rho = 77\%$. The phase-type distribution and its parameters are shown below.

$$
\begin{align*}
  r_{\downarrow} &= (0.5 \ 1 \ 0.75 \ 1.25 \ 1.5) \\
  r_{\uparrow} &= (4 \ 5 \ 6 \ 7 \ 8) \\
  r &= \begin{pmatrix}
    2 & 0.5 & 2 & 1 & 1.5 \\
    0 & 2 & 1.5 & 2.5 & 1 \\
    0 & 0 & 2 & 2.5 & 2.5 \\
    0 & 0 & 0 & 4 & 3 \\
    0 & 0 & 0 & 0 & 7 \\
  \end{pmatrix}.
\end{align*}
$$

Please cite this article in press as: D. Klink, et al., Time-bounded reachability in tree-structured QBDs by abstraction, Performance Evaluation (2010), doi:10.1016/j.peva.2010.04.002
Fig. 16. E5: Upper and lower bounds for bd, tree, qbd and (q)grid abstractions with similar sized state spaces for k = 0.

For initial state (2, 2, 2, 1, 1), goal level k = 4 and cut level 24, Fig. 17 shows upper and lower bounds for the refined grid abstraction with c = 0, 2, 4. Again, for matching goal and refinement level, the results differ only marginally. In the following, we investigate how the time for abstraction and for the computation of probability bounds scales with the cut level. The goal level is fixed to k = 0.

The results in Fig. 18 and in Table 2 show that for large enough cut level, the grid abstraction is capable of providing upper and lower bounds that differ only marginally and scale well with the size of the abstract model.

Note that the differences between upper and lower bounds of reachability probabilities listed in Table 2 cannot get smaller than $\epsilon = 10^{-6}$ because MRMC computes $\epsilon$-approximations of the probability bounds. In contrast, when using the uniformization method (cf. Section 2), a massive number of states would have to be considered (see Table 2, right part). Even with a terabyte of available memory, CTMCs with more than $10^{14}$ states cannot be dealt with. This shows evidently that abstraction is an attractive approach to computing time-bounded reachability probabilities in tree-structured QBDs.

6. Conclusions

In this paper we have shown that abstraction techniques allow us to quickly compute highly-accurate time-bounded reachability probabilities for a class of infinite-state CTMCs, namely tree-structured QBDs. In queueing theory this model class is widely used, however, mostly for steady-state analysis. Computing time-bounded reachability probabilities for tree-structured QBDs has to the best of our knowledge not been possible before. We present non-trivial partitionings for the infinite-state space of tree-structured QBDs that preserve different properties and discuss how they formally relate. In six experiments we thoroughly compare the efficiency of these partitionings, regarding the size of the resulting state space and...
Fig. 17. $E6$: Upper and lower bounds for refined grid abstractions with increasing $c$ for $k = 4$ and fixed cut level.

Fig. 18. $E6$: Upper and lower bounds for grid abstractions with increasing cut level and $k = 0$.

Table 2

$E6$: Differences between lower and upper bounds in grid abstractions vs. size of the state space for uniformization with error bound $\varepsilon = 10^{-6}$.

<table>
<thead>
<tr>
<th>grid abstraction</th>
<th>Diff</th>
<th>grid 12</th>
<th>grid 16</th>
<th>grid 20</th>
<th>grid 24</th>
<th>Uniformization</th>
<th>$\approx$states</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>trunc</td>
<td></td>
</tr>
<tr>
<td>2.5</td>
<td>0.0224</td>
<td>0.001</td>
<td>$10^{-5}$</td>
<td>$10^{-6}$</td>
<td>$10^{-6}$</td>
<td>185</td>
<td>$10^{129}$</td>
</tr>
<tr>
<td>7.5</td>
<td>0.3117</td>
<td>0.0580</td>
<td>0.0062</td>
<td>0.0004</td>
<td>$10^{-7}$</td>
<td>270</td>
<td>$10^{188}$</td>
</tr>
<tr>
<td>15</td>
<td>0.4054</td>
<td>0.1345</td>
<td>0.0376</td>
<td>0.0086</td>
<td>$10^{-7}$</td>
<td>398</td>
<td>$10^{278}$</td>
</tr>
<tr>
<td>States</td>
<td>6188</td>
<td>20349</td>
<td>53130</td>
<td>118755</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distributions</td>
<td>28666</td>
<td>96901</td>
<td>256796</td>
<td>579151</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time (h:m:s)</td>
<td>0:00:26</td>
<td>0:01:33</td>
<td>0:04:15</td>
<td>0:09:50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$t$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>trunc</td>
<td></td>
</tr>
<tr>
<td>2.5</td>
<td>$10^{-6}$</td>
<td>$10^{-6}$</td>
<td>$10^{-6}$</td>
<td>$10^{-6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.5</td>
<td>$10^{-5}$</td>
<td>$10^{-6}$</td>
<td>$10^{-6}$</td>
<td>$10^{-6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0.0015</td>
<td>0.0002</td>
<td>$2 \cdot 10^{-5}$</td>
<td>$3 \cdot 10^{-6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>States</td>
<td>237336</td>
<td>435894</td>
<td>749398</td>
<td>1221759</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distributions</td>
<td>1164206</td>
<td>2146761</td>
<td>3701296</td>
<td>6047091</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time (h:m:s)</td>
<td>0:20:14</td>
<td>0:38:13</td>
<td>1:07:57</td>
<td>2:06:04</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
the difference between upper and lower probability bounds, resulting from abstraction. Grid-like schemes that neglect the order of the jobs in the queue, have shown to perform best when the empty state is chosen as goal state. When the cut level is chosen according to the utilization and the choice of the initial and the goal state, we achieve differences between the upper and the lower probability bound of only $10^{-6}$. To reach the empty state, all jobs need to be processed, regardless of their ordering, hence grid abstraction contains all necessary information and hence produces very tight bounds. In case goal states belong to higher levels $k > 0$, we present a refinement of the grid abstraction, where the state space up to level $k$ is not abstracted and grid abstraction is used for all higher levels. This produces extremely tight bounds for this setting as well.

Even though the tree-structured QBDs in this paper result from M|PH|1 queues, our approach can also be used for the more general PH|PH|1 queues. Also, more complicated measures can be dealt with by labeling states of a tree-structured QBD and using CSL formulas for specification of measures, cf. [23]. Future work will include the automation of the parameterization, as well as finding partitionings for other highly structured, fast growing state spaces, such as for multi-class queueing networks.

References


Please cite this article in press as: D. Klink, et al., Time-bounded reachability in tree-structured QBDs by abstraction, Performance Evaluation (2010), doi:10.1016/j.peva.2010.04.002
Daniel Klink studied computer science at the RWTH Aachen University, Germany, where he received his diploma degree in 2006. From 2006 to 2010, he was a Ph.D. student at the chair for Software Modeling and Verification, and stipendiat of the DFG research training group 1298 (AlgoSyn). His research interests focus on model checking continuous-time Markovian models and abstraction techniques. Recently, he received his Ph.D. degree from the department of computer science, RWTH Aachen University.

Anne Remke is assistant professor in the group "Design and Analysis of Communication Systems" at the University of Twente. She received her Ph.D. degree (2008) from the Universiteit Twente and her Master degree (2004) from the RWTH Aachen University, both in Computer Science. In the context of the STU Centre for Dependable ICT Systems, her focus is now on modelling and analysis of critical infrastructures.

Boudewijn R. Haverkort is full professor for Design and Analysis of Communication Systems at the University of Twente. Since March 2009, he is also scientific director and chairman of the Embedded Systems Institute in Eindhoven, the Netherlands. He has written well over 100 papers on performance and dependability modelling and evaluation in computer and communication systems. He also published a textbook on this topic, as well as a number of conference proceedings. He has been host (general chair and PC chair) of a large number of international conferences. He is a Fellow of the IEEE, and member of the ACM and GI.

Joost-Pieter Katoen is a full professor at the RWTH Aachen University and is part-time associated with the University of Twente. His research interests include concurrency theory, model checking, timed and probabilistic systems, and semantics. He coauthored more than 130 journals and conferences papers, and co-authored a comprehensive book (with Christel Baier) on "Principles of Model Checking". Joost-Pieter is steering committee member of QEST, ETAPS, and FORMATS, and is senior member of the ACM.