Monitoring CTMCs By Multi-Clock Timed Automata

Yijun Feng[1], Joost-Pieter Katoen[2], Haokun Li[1], Bican Xia[1], and Naijun Zhan[3,4] ⋆

[1] LMAM & School of Mathematical Sciences, Peking University, Beijing, China
[2] RWTH Aachen University, Aachen, Germany
[3] State Key Lab. of Comp. Sci., Institute of Software, Chinese Academy of Sciences, China
[4] University of Chinese Academy of Sciences, Beijing, China

Abstract. This paper presents a numerical algorithm to verify continuous-time Markov chains (CTMCs) against multi-clock deterministic timed automata (DTA). These DTA allow for specifying properties that cannot be expressed in CSL, the logic for CTMCs used by state-of-the-art probabilistic model checkers. The core problem is to compute the probability of timed runs by the CTMC $\mathcal{C}$ that are accepted by the DTA $\mathcal{A}$. These likelihoods equal reachability probabilities in an embedded piecewise deterministic Markov process (EPDP) obtained as product of $\mathcal{C}$ and $\mathcal{A}$'s region automaton. This paper provides a numerical algorithm to efficiently solve the PDEs describing these reachability probabilities. The key insight is to solve an ordinary differential equation (ODE) that exploits the specific characteristics of the product EPDP. We provide the numerical precision of our algorithm and present experimental results with a prototypical implementation.

1 Introduction

Continuous-time Markov chains (CTMCs) [16] are ubiquitous. They are used to model safety-critical systems like communicating networks and power management systems, are key to performance and dependability analysis, and naturally describe chemical reaction networks. The algorithmic verification of CTMCs has received quite some attention. Aziz *et al.* [3] proved that verifying CTMCs against CSL (Continuous Stochastic Logic) is decidable. CSL is a probabilistic and timed branching-time logic that allows for expressing properties like "is the probability of a given chemical reaction within 50 time units at least 10^{-3}?". Baier *et al.* [5] gave efficient numerical algorithms for CSL model checking that nowadays provide the basis of CTMC model checking in PRISM [23], MRMC [22] and Storm [14], as well as GreatSPN [2]. Extensions of CSL to cascaded timed-until operators [27], conditional probabilities [18], and (simple) timed regular expressions [4] have been considered.

This paper considers the verification of CTMCs against *linear-time* real-time properties. These include relevant properties in the design of a gas burner [28], like "the probability that the duration of leaking is more than one twentieth over an interval with a length more than 20 seconds is less than 10^{-6}". Such real-time properties can be conveniently expressed by deterministic timed automata (DTA) [1]. The core problem in the verification of CTMC $\mathcal{C}$ against DTA $\mathcal{A}$ is to compute the probability of $\mathcal{C}$'s timed runs that are accepted by $\mathcal{A}$, i.e. $\Pr(\mathcal{C} \models \mathcal{A})$. Chen *et al.* [10,11] showed that this

⋆ Joost-Pieter Katoen, Haokun Li, Bican Xia and Naijun Zhan are the corresponding authors.

quantity equals the reachability probability in a piecewise deterministic Markov process (PDP) [13]. This PDP is obtained by taking the product of CTMC $\mathcal{C}$ and the region automaton of $\mathcal{A}$. Computing reachability probabilities in PDPs is a challenge.

Practical implementations of verifying CTMCs against DTA specifications are rare. Barbot *et al.* [7] showed that for *single-clock* DTA, the PDP is in fact a Markov regenerative process. (This observation is also at the heart of model-checking CSL^{TA} [15].) This implies that for single-clock DTA, off-the-shelf CSL model-checking algorithms can be employed resulting in an efficient procedure [7]. Mikeev *et al.* [24] generalised these ideas to infinite-state CTMCs obtained from stoichiometric equations, whereas Chen *et al.* [12] showed the theory to generalize verifying single-clock DTA to continuous-time Markov decision processes.

Multi-clock DTA are however much harder to handle. The characterisation of PDP reachability probabilities as the unique solution of a set of partial differential equations (PDEs) [10,11] does not give insight into an efficient computational procedure. With the notable exception of [25], verifying PDPs has not been considered. Fu [17] provided an algorithm to approximate the probabilities using finite difference methods and gave an error bound. This method hampers scalability and therefore was never implemented. The same holds for model-checking using other linear-time real-time formalisms such as MTL and timed automata [9], linear duration invariants [8], and probabilistic duration calculus [20]. All these multi-clock approaches suffer from scalability issues due to the low efficiency of solving PDEs and/or integral equations on which they heavily depend.

This paper presents a numerical technique to approximate the reachability probability in the product PDP. The DTA $\mathcal{A}$ is approximated by DTA $\mathcal{A}[t_f]$ which extends $\mathcal{A}$ with an additional clock that is never reset and that needs to be at most t_f when accepting. By increasing the time-bound t_f, DTA $\mathcal{A}[t_f]$ approximates $\mathcal{A}$ arbitrarily closely. We show that the set of PDPs characterizing the reachability probability in the embedded PDP of $\mathcal{C}$ and $\mathcal{A}[t_f]$ can be reduced to solving an ordinary differential equation (ODE). The specific characteristics of the product EPDP, in particular the fact that all clocks run at the same pace, are key to obtain these ODEs. Our numerical algorithm to solve the ODEs is based on computing the approximations in a backward manner using t_f and the sum of all clocks. The complexity of the resulting procedure is linear in the EPDP size, and exponential in $\lceil \frac{t_f}{\delta} \rceil$ where δ is the discretization step size. We show the approximations converges to the real solution of the ODEs at a linear speed of δ. Using a prototypical tool implementation we present some results on a number of case studies such as robot navigation with varying number of clocks in their specification. The experimental results show promising results for checking CTMCs against multi-clock DTA.

Organization of the paper. Section 2 introduces basic notions including CTMCs, DTA, and PDPs. Section 3 presents the product of a CTMC and the region graph of a DTA and shows this is an embedded PDP. Section 4 derives the PDE (fixing some flaw in [10]), the reduction to the set of ODEs and presents the numerical algorithm to solve these ODEs. Section 5 presents the experimental results and Section 6 concludes.

2 Preliminaries

In this section, we introduce some basic notions which will be used later.

A probability space is denoted by a triple $(\Omega, \mathcal{F}, Pr)$, where Ω is a set of samples, $\mathcal{F}$ is a σ-algebra over Ω, and $Pr : \mathcal{F} \to [0,1]$ is a probability measure on $\mathcal{F}$ with $Pr(\Omega) = 1$. Let $\mathbb{P}_r(\Omega)$ denote the set of all probability measures over Ω. For a random variable X on the probability space, its expectation is denoted by $\mathbb{E}(X)$.

2.1 Continuous-time Markov Chain (CTMC)

Definition 1 (CTMC). *A CTMC is a tuple $\mathcal{C} = (S, \mathbf{P}, \alpha, AP, L, E)$, where*

- *S is a finite set of states;*
- *$\mathbf{P}: S \times S \to [0,1]$ is the transition probability function, which is identified with the matrix $\mathbf{P} \in [0,1]^{|S|\times|S|}$ such that $\sum_{t\in S} \mathbf{P}(s,t) = 1$, for all $s \in S$;*
- *$\alpha \in \mathbb{P}_r(S)$ is the initial distribution;*
- *AP is a finite set of atomic propositions;*
- *$L : S \to 2^{AP}$ is a labeling function; and*
- *$E : S \to \mathbb{R}_{>0}$ is the exit rate function.*

We denote by $s \xrightarrow{t} s'$ a transition from state s to state s' after residing in state s for t time units. The probability of the occurrence of this transition within t time units is $\mathbf{P}(s,s') \int_0^t E(s) \exp^{-E(s)x}\, dx$, where $\int_0^t E(s) \exp^{-E(s)x}\, dx$ stands for the probability to leave state s in t time units, and $\mathbf{P}(s,s')$ for the probability to select the transition to s' from all transitions outgoing from s. A state s is called *absorbing* if $\mathbf{P}(s,s) = 1$. Given a CTMC $\mathcal{C}$, removing the exit rate function E results in a discrete-time Markov chain (DMTC), which is called *embedded* DTMC of $\mathcal{C}$. A CTMC $\mathcal{C}$ is called *irreducible* if there exists a unique stationary distribution α, such that $\alpha(s) > 0$ for all $s \in S$, and *weakly irreducible* if $\alpha(s)$ may be zero for some $s \in S$.

Definition 2 (CTMC Path). *Let $\mathcal{C}$ be a CTMC, a path ρ of $\mathcal{C}$ starting form s_0 with length n is a sequence $\rho = s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{n-1}} s_n \in S \times (\mathbb{R}_{>0} \times S)^n$. The set of paths in $\mathcal{C}$ with length n is denoted by $Path_n^{\mathcal{C}}$; the set of all finite paths of $\mathcal{C}$ is $Path_{fin}^{\mathcal{C}} = \cup_n Path_n^{\mathcal{C}}$ and the set of infinite paths of $\mathcal{C}$ is $Path_{inf}^{\mathcal{C}} = (S \times \mathbb{R}_{>0})^\omega$. We use $Path^{\mathcal{C}} = Path_{fin}^{\mathcal{C}} \cup Path_{inf}^{\mathcal{C}}$ to denote all paths in $\mathcal{C}$. As a convention, ε stands for the empty path.*

Note that we assume the time to exit a state is strictly greater than 0. For an infinite path ρ, we use $Pref(\rho)$ to denote the set of its finite prefixes. For a (finite or infinite) path ρ with prefix $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \ldots$, the trace of the path is the sequence of states $trace(\rho) = s_0 s_1 \ldots$. Let $\rho(n) = s_n$ be the n-th state in the path and $\rho[n] = t_n$ be the corresponding exit time for s_n. For a finite path $\rho = s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{n-1}} s_n$, we use $T(\rho) = \sum_{i=0}^{n-1} t_i$ to denote the total time spent on this path if $n \geq 1$, otherwise $T(\rho) = 0$. For a time $t \leq T(\rho)$, $\rho(0 \ldots t)$ denotes the prefix of ρ within t time units, i.e., $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{m-1}} s_m$ if there exists some $m \leq n$ with $\sum_{i=0}^{m-1} \rho[m] \leq t \wedge \sum_{i=0}^{m} \rho[m] > t$, otherwise ε.

A basic cylinder set $C(s_0, I_0, \cdots, I_{n-1}, s_n)$ consists of all paths $\rho \in Path^{\mathcal{C}}$ such that $\rho(i) = s_i$ for $0 \leq i \leq n$, and $\rho[i] \in I_i$ for $0 \leq i < n$. Then the $\sigma-$algebra $\mathcal{F}_{s_0}(\mathcal{C})$ associated with CTMC $\mathcal{C}$ and initial state s_0 is the smallest $\sigma-$algebra that contains all cylinder sets $C(s_0, I_0, \cdots, I_{n-1}, s_n)$ with $\alpha(s_0) > 0$, and $\mathbf{P}(s_i, s_{i+1}) > 0$, for $1 \leq$

$i \leq n$, and $I_0, \ldots, I_{n-1}$ are non-empty intervals in $\mathbb{R}_{\geq 0}$. There is a unique probability measure $Pr^{\mathcal{C}}$ on the $\sigma-$algebra $\mathcal{F}_{s_0}(\mathcal{C})$, by which the probability for a cylinder set is given by

$$Pr^{\mathcal{C}}(C(s_0, I_0, \cdots, I_n, s_n)) = \alpha(s_0) \cdot \prod_{i=1}^{n} \int_{I_i} E(s_{i-1}) \exp^{-E(s_{i-1})x} dx \cdot \mathbf{P}(s_{i-1}, s_i)$$

Example 1. An example of CTMC is shown in Fig. 1, with $AP = \{a, b, c\}$ and initial state s_0. The exit rate $r_i,\ i = 0, 1, 2, 3$ and transition probability are shown in the figure.

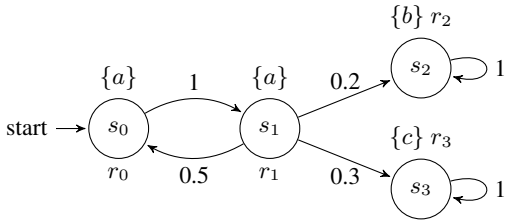


Fig. 1. An example CTMC

2.2 Deterministic Timed Automaton (DTA)

A timed automaton is a finite state graph equipped with a finite set of nonnegative real-valued clock variables, or clocks for short. Clocks can only be reset to zero, or proceed with rate 1 as time progresses independently. Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of clocks. $\eta(x) : \mathcal{X} \to \mathbb{R}_{\geq 0}$ is a $\mathcal{X}$-valuation which records the amount of time since its last reset. Let *Val*$(\mathcal{A})$ be the set of all clock valuations of $\mathcal{A}$. For a subset $X \subseteq \mathcal{X}$, the reset of X, denoted as $\eta[X := 0]$, is the valuation η' such that $\eta'(x) = 0, \forall x \in X$, and $\eta'(x) = \eta(x)$, otherwise. For $d \in \mathbb{R}_{>0}$, $(\eta + d)(x) = \eta(x) + d$ for any clock $x \in \mathcal{X}$.

A clock constraint over $\mathcal{X}$ is a formula with the following form

$$g := x < c \mid x \leq c \mid x > c \mid x \geq c \mid x - y \geq c \mid g \wedge g\,,$$

where x, y are clocks, $c \in \mathbb{N}$. Let *Con*$(\mathcal{X})$ denote the set of clock constraints over $\mathcal{X}$. A valuation η satisfies a guard g, denoted as $\eta \models g$, iff $\eta(x) \bowtie c$ when g is $x \bowtie c$, where $\bowtie \in \{<, \leq, >, \geq\}$; and $\eta \models g_1$ and $\eta \models g_2$ iff $g = g_1 \wedge g_2$.

Definition 3 (DTA). *A DTA is a tuple $\mathcal{A} = (\Sigma, \mathcal{X}, Q, q_0, Q_F, \hookrightarrow)$, where*

- *Σ is a finite set of actions;*
- *$\mathcal{X}$ is a finite set of clocks;*
- *Q is a finite set of locations;*
- *$q_0 \in Q$ is the initial location;*
- *$Q_F \subseteq Q$ is the set of accepting locations;*
- *$\hookrightarrow \in (Q \backslash Q_F) \times \Sigma \times$ Con$(\mathcal{X}) \times 2^{\mathcal{X}} \times Q$ is the transition relation, satisfying if $q \xhookrightarrow{a,g,X} q'$ and $q \xhookrightarrow{a,g',X'} q''$ with $q' \neq q''$ then $g \cap g' = \emptyset$.*

Each transition relation, or edge, $q \hookrightarrow q'$ in $\mathcal{A}$ is endowed with (a, g, X), where $a \in \Sigma$ is an action, $g \in \mathit{Con}(\mathcal{X})$ is the guard of the transition, and $X \subseteq \mathcal{X}$ is a set of clocks, which should be reset to 0 after the transition. An intuitive interpretation of the transition is that $\mathcal{A}$ can move from q to q' by taking action a and resetting all clocks in X to be 0 only if g is satisfied. There are no outgoing transitions from any accepting location in Q_F.

A finite timed path of $\mathcal{A}$ is of the form $\theta = q_0 \xhookrightarrow{a_0,t_0} q_1 \xhookrightarrow{a_1,t_1} \dots \xhookrightarrow{a_{n-1},t_{n-1}} q_n$, where $t_i \geq 0$, for $i = 0, \dots, n-1$. Moreover, there exists a sequence of transitions $q_j \xhookrightarrow{a_j,g_j,X_j} q_{j+1}$, for $0 \leq j \leq n-1$, such that $\eta_0 = \mathbf{0}$, $\eta_j + t_j \models g_j$ and $\eta_{j+1} = \eta_j[X_j := 0]$, where η_k denotes the clock valuation when entering q_k. θ is said to be *accepted by* $\mathcal{A}$ if there exists a state $q_i \in Q_F$ for some $0 \leq i \leq n$. As normal, it is assumed all DTA are non-Zeno [6], that is any circular transition sequence takes nonzero dwelling time.

A region is a set of valuations, usually represented by a set of clock constraints. Let $\mathit{Reg}(\mathcal{X})$ be the set of regions over $\mathcal{X}$. Given $\Theta, \Theta' \in \mathit{Reg}(\mathcal{X})$, Θ' is called a *successor* of Θ if for all $\eta \models \Theta$, there exists $t > 0$ such that $\eta + t \models \Theta'$ and $\forall t' < t, \eta + t' \models \Theta \vee \Theta'$. A region Θ satisfies a guard g, denoted as $\Theta \models g$, iff $\forall \eta \models \Theta$ implies $\eta \models g$. The reset operation on a region Θ is defined as $\Theta[X := 0] = \{\eta[X := 0] \mid \eta \models \Theta\}$. Then the region graph, viewed as a quotient transition system related to clock equivalence [6] can be defined as follows:

Definition 4 (Region Graph). *The region graph for DTA $\mathcal{A} = (\Sigma, \mathcal{X}, Q, q_0, Q_F, \hookrightarrow)$ is a tuple $\mathcal{G}(\mathcal{A}) = (\Sigma, \mathcal{X}, \overline{Q}, \overline{q_0}, \overline{Q_F}, \mapsto)$, where*

- *$\overline{Q} = Q \times \mathit{Reg}(\mathcal{X})$ is the set of states;*
- *$\overline{q_0} = (q_0, \mathbf{0}) \in \overline{Q}$ is the initial state;*
- *$\overline{Q_F} \subseteq Q_F \times \mathit{Reg}(\mathcal{X})$ is the set of final states;*
- *$\mapsto \subseteq \overline{Q} \times ((\Sigma \times 2^{\mathcal{X}}) \cup \{\lambda\}) \times \overline{Q}$ is the transition relation satisfying*
  - *$(q, \Theta) \overset{\lambda}{\mapsto} (q, \Theta')$ if Θ' is a successor of Θ;*
  - *$(q, \Theta) \overset{a,X}{\longmapsto} (q', \Theta'')$ if there exists $g \in \mathit{Con}(\mathcal{X})$ and transition $q \xhookrightarrow{a,g,X} q'$ such that $\Theta \models g$ and $\Theta'' = \Theta[X := 0]$.*

Example 2 (Adapted from [10]). Fig. 2 presents an example of DTA and Fig. 3 gives its region graph, in which double circle and double rectangle stand for final states, respectively.

2.3 Piecewise-deterministic Markov Process (PDP)

Piecewise-deterministic Markov Processes (PDPs for short) [13] cover a wide range of stochastic models in which the randomness appears as discrete events at fixed or random times, whose evolution is deterministically governed by an ODE system between these times. A PDP consists of a mixture of deterministic motion and random jumps between a finite set of locations. During staying in a location, a PDP evolves deterministically following a flow function, which is a solution to an ODE system. A PDP can jump between locations either randomly, in which case the residence time of a location is governed by an exponential distribution, or when the location invariant is violated. The successor state of the jump follows a probability measure depending on the current state. A PDP is right-continuous and has the strong Markov property [13].

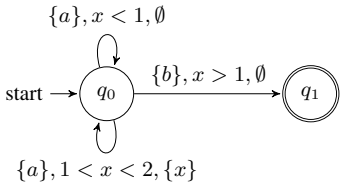


Fig. 2. A DTA $\mathcal{A}$

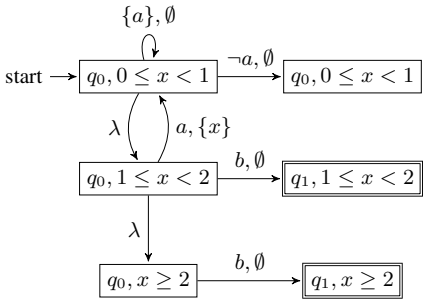


Fig. 3. The region graph of $\mathcal{A}$

Definition 5 (PDP [13]). *A PDP is a tuple* $\mathcal{Q} = (Z, \mathcal{X}, Inv, \phi, \Lambda, \mu)$ *with*

- Z *is a finite set of locations;*
- $\mathcal{X}$ *is a finite set of variables;*
- $Inv : Z \to 2^{\mathbb{R}^{|\mathcal{X}|}}$ *is an invariant function;*
- $\phi : Z \times \mathbb{R}^{|\mathcal{X}|} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{|\mathcal{X}|}$*, is a flow function, which is a solution of a system of ODEs with Lipschitz continuous vector fields;*
- $\Lambda : \mathbb{S} \to \mathbb{R}_{>0}$ *is an exit rate function;*
- $\overline{\mathbb{S}} \to \mathbb{P}_r(\mathbb{S})$*, is the transition probability function, where* $\mathbb{S} = \{\xi := (z, \eta) \mid z \in Z, \eta \models Inv(z)\}$ *is the state space for* $\mathcal{Q}$*,* $\overline{\mathbb{S}}$ *is the closure of* $\mathbb{S}$*,* $\mathbb{S}^o = \{(z, \eta) \mid z \in Z, \eta \models Inv(z)^o\}$ *is the interior of* $\mathbb{S}$*, in which* $Inv(z)^o$ *stands for the interior of* $\overline{Inv(z)}$*, and* $\partial\mathbb{S} = \cup_{z \in Z}\{z\} \times \partial Inv(z)$ *is the boundary of* $\mathbb{S}$*, in which* $\partial Inv(z) = \overline{Inv(z)} \backslash Inv^o$ *and* $\overline{Inv(z)}$ *is the closure of* $Inv(z)$*.*

For any $\xi = (z, \eta) \in \mathbb{S}$, there is an $\delta(\xi) > 0$ such that $\Lambda(z, \phi(z, \eta, t))$ is integrable on $[0, \delta(\xi))$. $\mu(\xi)(A)$ is measurable for any $A \in \mathcal{F}(\mathbb{S})$, where $\mathcal{F}(\mathbb{S})$ is the smallest $\sigma-$algebra generated by $\{\bigcup_{z \in Z} z \times A_z | A_z \in \mathcal{F}(Inv(z))\}$ and $\mu(\xi)(\{\xi\}) = 0$.

There are two ways to take transitions between locations in PDP $\mathcal{Q}$. A PDP $\mathcal{Q}$ is allowed to stay in a current location z only if $Inv(z)$ is satisfied. During its residence, the valuation η evolves time-dependently according to the flow function. Let $\xi \oplus t = (z, \phi(z, \eta, t))$ be the successor state of $\xi = (z, \eta)$ after residing t time units in z. Thus, $\mathcal{Q}$ is piecewise-deterministic since its behavior is determined by the flow function ϕ in each location. In a state $\xi = (z, \eta)$ with $\eta \models Inv(z)^o$, the PDP $\mathcal{Q}$ can either evolve to a state $\xi' = \xi \oplus t$ by delaying t time units, or take a Markovian jump to $\xi'' = (z'', \eta'') \in \mathbb{S}$ with probability $\mu(\xi)(\{\xi''\})$. When $\eta \models \partial Inv(z)$, $\mathcal{Q}$ is forced to take a boundary jump to $\xi'' = (z'', \eta'') \in \mathbb{S}$ with probability $\mu(\xi)(\{\xi''\})$.

3 Reduction to the Reachability Probability of EPDP

As proved in [10], model-checking of a given CTMC $\mathcal{C}$ against a linear real-time property expressed by a DTA $\mathcal{A}$, i.e., determining $Pr(\mathcal{C} \models \mathcal{A})$, can be reduced to computing the reachability probability of the product of $\mathcal{C}$ and $\mathcal{G}(\mathcal{A})$. This can be further reduced to computing the reachability probability of the embedded PDP (EPDP) of the product. But how to efficiently compute the reachability probability of the EPDP still remains challenging, as existing approaches [10,15,7] can only handle DTA with one clock.

We will attack this challenge in this paper. For self-containedness, we reformulate the reduction reported in [10] in this section.

A path $\rho = s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \dots$ of CTMC $\mathcal{C}$ is accepted by DTA $\mathcal{A}$ if $\hat{\rho} = q_0 \xrightarrow{L(s_0),t_0} q_1 \xrightarrow{L(s_1),t_1} \dots \xrightarrow{L(s_{n-1}),t_{n-1}} q_n$ induced by some ρ's prefix is an accepting path of $\mathcal{A}$. Then $Pr(\mathcal{C} \models \mathcal{A}) = Pr\{\rho \in Path^{\mathcal{C}} \mid \rho \text{ is accepted by } \mathcal{A}\}$.

Definition 6 (Product Region Graph [7]). *The product of CTMC $\mathcal{C} = (S, \mathbf{P}, \alpha, AP, L, E)$ and the region graph of DTA $\mathcal{G}(\mathcal{A}) = (\Sigma, \mathcal{X}, \overline{Q}, \overline{q_0}, \overline{Q_F}, \mapsto)$, denoted by $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$, is a tuple $(\mathcal{X}, V, \alpha', V_F, \rightharpoonup, \Lambda)$, where*

- *$V = S \times \overline{Q}$ is the state space;*
- *$\alpha'(s, \overline{q_0}) = \alpha(s)$ is the initial distribution;*
- *$V_F = S \times \overline{Q_F}$ is the set of accepting states;*
- *$\rightharpoonup \subseteq V \times (([0,1] \times 2^{\mathcal{X}}) \cup \{\lambda\}) \times V$ is the smallest relation satisfying*
  - *$(s, \overline{q}) \xrightarrow{\lambda} (s, \overline{q'})$ (called* delay transition*), if $\overline{q} \overset{\lambda}{\mapsto} \overline{q'}$;*
  - *$(s, \overline{q}) \xrightarrow{p,X} (s'', \overline{q''})$ (called* Markovian transition*), if $\mathbf{P}(s, s'') = p, p > 0$ and $\overline{q} \xrightarrow{L(s),X} \overline{q''}$;*
- *$\Lambda : V \to \mathbb{R}_{>0}$ is the exit rate function, where*

$$\Lambda(s, \overline{q}) = \begin{cases} E(s) & \text{if there exists a Markovian transition from } (s, \overline{q}) \\ 0 & \text{otherwise} \end{cases}$$

Remark 1. Note that the definition of region graph here is slightly different from the usual one in the sense that Markovian transitions starting from a boundary do not contribute to the reachability probability. Therefore we can merge the boundary into its unique delay successor.

Example 3 (Adapted from [10]). Fig. 4 shows the product region graph of CTMC $\mathcal{C}$ in Example 1 and DTA $\mathcal{A}$ in Example 2. The graph can be split into three subgraphs in a column-wise manner, where all transitions within a subgraph are probabilistic, all transitions evolve to the next subgraph are delay transitions, and transitions with reset lead to a state in the first subgraph. For conciseness, the location v_9 stands for all nodes that may be reached by a Markovian transition yet cannot reach an accepting node.

Proposition 1 ([10]). *For CTMC $\mathcal{C}$ and DTA $\mathcal{A}$, $Pr(\mathcal{C} \models \mathcal{A})$ is measurable and*

$$Pr(\mathcal{C} \models \mathcal{A}) = Pr^{\mathcal{C} \otimes \mathcal{G}(\mathcal{A})}\{Path^{\mathcal{C} \otimes \mathcal{G}(\mathcal{A})}(\Diamond \overline{Q_F})\}\,.$$

When treated as a stochastic process, $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$ can be interpreted as a PDP. In this way, computing the reachability probability of Q_F in $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$ can be reduced to computing the time-unbounded reachability probability in the EPDP of $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$.

Definition 7 (EPDP, [7]). *Given $\mathcal{C} \otimes \mathcal{G}(\mathcal{A}) = (\mathcal{X}, V, \alpha', V_F, \rightharpoonup, \Lambda)$, the EPDP $\mathcal{Q}^{\mathcal{C} \otimes \mathcal{A}}$ is a tuple $(\mathcal{X}, V, Inv, \phi, \Lambda, \mu)$ where for any $v = (s, (q, \Theta)) \in V$*

- *$Inv(v) = \Theta$, $\mathbb{S} = \{(v, \boldsymbol{\eta}) \mid v \in V, \boldsymbol{\eta} \models Inv(v)\}$ is the state space;*
- *$\phi(v, \boldsymbol{\eta}, t) = \boldsymbol{\eta} + t$ for $\boldsymbol{\eta} \models Inv(v)$;*

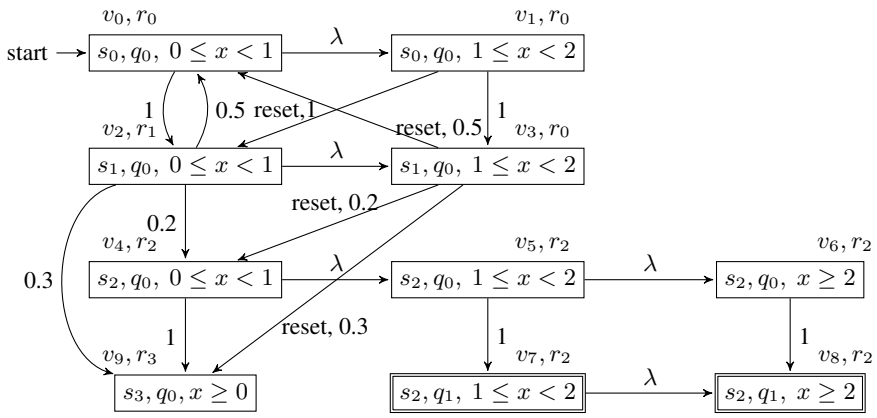


Fig. 4. Product region graph $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$ of CTMC $\mathcal{C}$ in Example 1 and DTA $\mathcal{A}$ in Example 2

- $\Lambda(v, \eta) = \Lambda(v)$ *is the exit rate of* (v, η)*;*
- Boundary jump: *for each delay transition* $v \xrightarrow{\lambda} v'$ *in* $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$, $\mu(\xi, \{\xi'\}) = 1$ *whenever* $\xi = (v, \boldsymbol{\eta})$, $\xi' = (v', \boldsymbol{\eta})$ *and* $\boldsymbol{\eta} \models \partial \mathit{Inv}(v)$*;*
- Markovian transition jump: *for each Markovian transition* $v \xrightarrow{p,X} v''$ *in* $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$, $\mu(\xi, \{\xi''\}) = p$ *whenever* $\xi = (v, \boldsymbol{\eta})$, $\boldsymbol{\eta} \models \mathit{Inv}(v)$ *and* $\xi'' = (v'', \boldsymbol{\eta}[X := 0])$.

The flow function here describes that all clocks increase with a uniform rate (i.e., $\dot{x_1} = 1, \ldots, \dot{x_n} = 1$, or simply $\dot{\mathcal{X}} = 1$) at all locations. The original reachability problem is then reduced to the reachability probability of the set $\{(v, \boldsymbol{\eta}) \mid v \in V_F, \boldsymbol{\eta} \models \mathit{Inv}(v)\}$, given the initial state $(v_0, \mathbf{0})$ and the EPDP $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}$. Let $Pr_v^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta})$ stand for the probability to reach the final states $(V_F \times *)$ from $(v, \boldsymbol{\eta})$ in $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}$. Thus, $Pr_v^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta})$ can be computed recursively by

$$Pr_v^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta}) = \begin{cases} Pr_{v,\lambda}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta}) + \sum_{v \xrightarrow{p,X} v'} Pr_{v,v'}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta}) & \text{if } v \notin V_F \\ 1, & v \in V_F \wedge \boldsymbol{\eta} \models \mathit{Inv}(v) \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Let $t_z^*(v, \boldsymbol{\eta})$ denote the minimal time for $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}$ to reach $\partial \mathit{Inv}(v)$ from $(v, \boldsymbol{\eta})$. More precisely,

$$t_z^*(v, \boldsymbol{\eta}) = \inf\{t \mid \phi(v, \boldsymbol{\eta}, t) \models \mathit{Inv}(v)\}.$$

$Pr_{v,\lambda}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta})$ is the probability from (v, η) with a delay and then a forced jump to $(v', \eta + t_z^*(v, \boldsymbol{\eta})$, onwards evolves to an accepting state, which can be recursively computed by

$$Pr_{v,\lambda}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta}) = exp(-\Lambda(v) t_z^*(v, \boldsymbol{\eta})) \cdot Pr_{v'}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta} + t_z^*(v, \boldsymbol{\eta})).$$

$Pr_{v,v'}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta})$ is the probability that a Markovian transition $v \xrightarrow{p,X} v'$ happens within $t_z^*(v, \boldsymbol{\eta})$ time units, onwards involves to an accepted state, which can be recursively

computed by

$$Pr_{v,v'}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta}) = \int_0^{t_z^*(v,\boldsymbol{\eta})} p \cdot \Lambda(v) \exp(-\Lambda(v)s) \cdot Pr_{v'}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\boldsymbol{\eta} + s[X := 0])\, ds.$$

$Pr(\mathcal{C} \models \mathcal{A})$ is reduced to compute $Pr_{v_0}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\mathbf{0})$, equivalent to computing the least fixed point of the equation (1). That is,

Theorem 1. *[10] For CTMC $\mathcal{C}$ and DTA $\mathcal{A}$, $Pr(\mathcal{C} \models \mathcal{A}) = Pr^{\mathcal{C}\otimes\mathcal{A}}\{Path^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\overline{Q_F})\}$ is the least fixed point of (1).*

Remark 2. Generally, it is difficult to solve a recursive equation like (1). As an alternative, we discuss the augmented EPDP of $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}$ by replacing $\mathcal{A}$ with a bounded DTA resulting from $\mathcal{A}$. As a consequence, using the extended generator of the augmented EPDP, we can induce a partial differential equation (PDE) whose solution is the reachability probability. We will elaborate the idea in the subsequent section.

4 Approximating the Reachability Probability of EPDP

In this section, we present a numerical method to approximate $Pr_{v_0}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\mathbf{0})$, as we discussed previously that exactly computing is impossible, at least too expensive, in general. We will first introduce the basic idea of our approach in detail, then discuss its time complexity and convergence property. A key point is that our approach exploits the observation that the flow function of $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}$ is linear, only related to time t, and remains the same at all locations. This enables to reduce computing $Pr_{v_0}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\mathbf{0})$ to solving an ODE system.

4.1 Reduction to a PDE System

In this subsection, we first show that $Pr_{v_0}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}(\mathbf{0})$ can be approximated by that of the EPDP of $\mathcal{C}$ and a bounded DTA derived from $\mathcal{A}$, i.e., the length of all its paths is bounded. Then show that the latter can be reduced to solving a PDE system.

Given a DTA $\mathcal{A}$, we construct a bounded DTA $\mathcal{A}[t_f]$ by introducing a new clock y, adding a timing constraint $y < t_f$ to the guard of each transition of $\mathcal{A}$ ingoing to an accepting state in Q_F, and never resetting y, where $t_f \in \mathbb{N}$ is a parameter. So, the length of all accepting paths of $\mathcal{A}[t_f]$ is time-bounded by t_f. Obviously, $Path^{\mathcal{C}}(\mathcal{A}[t_f])$ is a subset of $Path^{\mathcal{C}}(\mathcal{A})$. As $Pr(\mathcal{C} \models \mathcal{A})$ is measurable and $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}$ is Borel right continuous, we have the following proposition.

Proposition 2. *Given a CTMC $\mathcal{C}$, a DTA $\mathcal{A}$, and $t_f \in \mathbb{N}$,*

$$\lim_{t_f\to\infty} Pr(\mathcal{C} \models \mathcal{A}[t_f]) = Pr(\mathcal{C} \models \mathcal{A}). \tag{2}$$

Moreover, if $\mathcal{C}$ is weakly irreducible or satisfies some conditions (please refer to Chapter 4 of [26] for details), then there exist positive constants $K, K_0 \in \mathbb{R}_{\geq 0}$ such that

$$Pr(\mathcal{C} \models \mathcal{A}) - Pr(\mathcal{C} \models \mathcal{A}[t_f]) \leq K \exp\{-K_0 t_f\}. \tag{3}$$

Remark 3. (2) was first observed in [7], thereof the authors pointed out the feasibility of using a bounded system to approximate the original unbounded system in order to simplify a verification obligation. (3) further indicates that such approximation is exponentially convergent w.r.t. $-t_f$ if the CTMC is weakly irreducible.

For a path starting in a state $(v, \boldsymbol{\eta})$ at time y, we use $\mathit{Path}^{y}_{(v,\boldsymbol{\eta})}[t]$ to denote the set of its locations at time t, and $\hbar_v(y, \boldsymbol{\eta}) = \mathit{Pr}(\mathit{Path}^{y}_{(v,\boldsymbol{\eta})}[t_f] \in V_F) = \mathbb{E}(\mathbf{1}_{\mathit{Path}^{y}_{(v,\boldsymbol{\eta})}[t_f] \in V_F})$ as the probability of a path reaching V_F within t_f time units, where $\mathbf{1}_{\mathit{Path}^{y}_{(v,\boldsymbol{\eta})}[t_f] \in V_F}$ is the indicator function of $\mathit{Path}^{y}_{(v,\boldsymbol{\eta})}[t_f] \in V_F$. Then, $\hbar_{v_0}(0, \mathbf{0}) = \mathit{Pr}(\mathcal{C} \models \mathcal{A}[t_f])$ is the probability to reach the set of accepting states from the initial state $(0, \mathbf{0})$, which satisfies the following system of PDEs.

Theorem 2. *Given a CTMC $\mathcal{C}$, a bounded DTA $\mathcal{A}[t_f]$, and the EPDP $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{G}(\mathcal{A}[t_f])} = (\mathcal{X}, V, \mathit{Inv}, \phi, \Lambda, \mu)$, $\hbar_{v_0}(0, \mathbf{0})$ is the unique solution of the following system of PDEs:*

$$\frac{\partial \hbar_v(y, \boldsymbol{\eta})}{\partial y} + \sum_{i=1}^{|\mathcal{X}|} \frac{\partial \hbar_v(y, \boldsymbol{\eta})}{\partial \boldsymbol{\eta}^{(i)}} + \Lambda(v) \cdot \sum_{v \xrightarrow{p, X} v'} p \cdot (\hbar_{v'}(y, \boldsymbol{\eta}[X := 0]) - \hbar_v(y, \boldsymbol{\eta})) = 0, \quad (4)$$

where $v \in V \backslash V_F, \boldsymbol{\eta} \models \mathit{Inv}(v), \boldsymbol{\eta}^{(i)}$ is the i-th clock variable and $y \in [0, t_f)$. The boundary conditions are:

(i) $\hbar_v(y, \boldsymbol{\eta}) = \hbar_{v'}(y, \boldsymbol{\eta})$, *for every* $\boldsymbol{\eta} \models \partial \mathit{Inv}(v)$ *and transition* $v \xrightarrow{\lambda} v'$;
(ii) $\hbar_v(y, \boldsymbol{\eta}) = 1$, *for every vertex* $v \in V_F$, $\boldsymbol{\eta} \models \mathit{Inv}(v)$, *and* $y \in [0, t_f)$;
(iii) $\hbar_v(t_f, \boldsymbol{\eta}) = 0$, *for every vertex* $v \in V \backslash V_F$ *and* $\boldsymbol{\eta} \models \mathit{Inv}(v) \cup \partial \mathit{Inv}(v)$.

Remark 4. Note that the PDE system (4) in Theorem 2 is different from the one presented in [10] for reducing $\mathit{Pr}^{\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}}}_{v_0}(\mathbf{0})$. In particular, the boundary condition in [10] has been corrected here.

4.2 Reduction to an ODE System

There are several classical methods to solve PDEs. *Finite element method*, which is a numerical technique for solving PDEs as well as integral equations, is a prominent one, of which different versions have been established to solve different PDEs with specific properties. Other numerical methods include finite difference method and finite volume method and so on, the reader is referred to [21,19] for details. Thanks to the special form of the equation (4), we are able to obtain a numerical solution in a more efficient way.

The fact that the flow function (which is the solution to the ODE system $\bigwedge_{x \in \mathcal{X}} \dot{x} = 1 \wedge \dot{y} = 1$) is the same at all locations of the EPDP $\mathcal{Q}^{\mathcal{C}\otimes\mathcal{A}[t_f]}$ suggests that the partial derivatives of $\boldsymbol{\eta}$ and y in the left side of (4) evolve with the same pace. Thus, we can view all clocks as an array, and reformulate (4) as

$$\left[\frac{\partial \hbar_v(y, \boldsymbol{\eta})}{\partial y}, \frac{\partial \hbar_v(y, \boldsymbol{\eta})}{\partial \boldsymbol{\eta}^{(1)}}, \ldots, \frac{\partial \hbar_v(y, \boldsymbol{\eta})}{\partial \boldsymbol{\eta}^{(|\mathcal{X}|)}}\right] \bullet \mathbf{1} + \Lambda(v) \cdot \sum_{v \xrightarrow{p, X} v'} p \cdot (\hbar_{v'}(y, \boldsymbol{\eta}[X := 0]) - \hbar_v(y, \boldsymbol{\eta})) = 0, \quad (5)$$

where $\bullet$ stands for the inner product of two vectors of the same dimension, e.g., $(a_1, \ldots, a_n) \bullet (b_1, \ldots, b_n) = \sum_{i=1}^{n} a_i b_i$, and $\mathbf{1}$ for the vector $(\overbrace{1, \ldots, 1}^{n\text{times}})$.

By Theorem 2, there exist v_0, y_0 and η_0 such that $v_0 \in V_F$, $y_0 = t_f$, and $\boldsymbol{\eta}_0 \models \mathit{Inv}(v) \vee \partial \mathit{Inv}(v)$. Besides, by the definition of $\mathcal{Q}^{\mathcal{C} \otimes \mathcal{A}[t_f]}$, it follows $\frac{\partial z}{\partial t} = 1$, which implies $dz = dt$, for any $z \in \{y\} \cup \mathcal{X}$. Hence, we can simplify (5) as the following ODE system:

$$\frac{d\hbar_v((y_0, \boldsymbol{\eta}_0) + t)}{dt} + \Lambda(v) \cdot \sum_{v \xrightarrow{p, X} v'} p \cdot (\hbar_{v'}((y_0, \boldsymbol{\eta}_0) + t)[X := 0]) - \hbar_v(y_0, \boldsymbol{\eta}_0)) = 0, \quad (6)$$

with the initial condition $v_0 \in V_F$, $y_0 = t_f$, and $\boldsymbol{\eta}_0 \models \mathit{Inv}(v) \vee \partial \mathit{Inv}(v)$, where $v \in V \backslash V_F$. Note that we compute the reachability probability by (6) backwards.

4.3 Numerical Solution

Since $\hbar_v((y_0, \boldsymbol{\eta}_0) + t)$ satisfies an ODE equation, we can apply a discretization method to (6) and obtain an approximation efficiently. To this end, the remaining obstacle is how to deal with the reset part $\hbar_{v'}(y_0 + t, (\boldsymbol{\eta}_0 + t)[X := 0])$. Notice that $X \neq \emptyset \Rightarrow \text{sum}((\boldsymbol{\eta}_0 + t)[X := 0]) + (t_f - y_0 - t)) < \text{sum}(\boldsymbol{\eta_0} + t) + (t_f - t_0 - t)$, where $\text{sum}(\boldsymbol{\eta}) = \sum_{x \in \mathcal{X}} \boldsymbol{\eta}(x)$. So we just need to solve the ODE system starting from $(t_f, \boldsymbol{\eta}_0)$ using the descending order over $\text{sum}(\boldsymbol{\eta})$ in a backward manner. In this way, all of the reset values needed for the current iteration have been computed in the previous iterations. Therefore for each iteration, the derivation is fixed and easy to calculate.

We denote by δ the length of discretization step, the number of total discretization steps is $\lceil \frac{t_f}{\delta} \rceil \in \mathbb{N}$. An approximate solution to (4) can be computed efficiently by the following algorithm.

Algorithm 1 Finding numerical solution to (4)

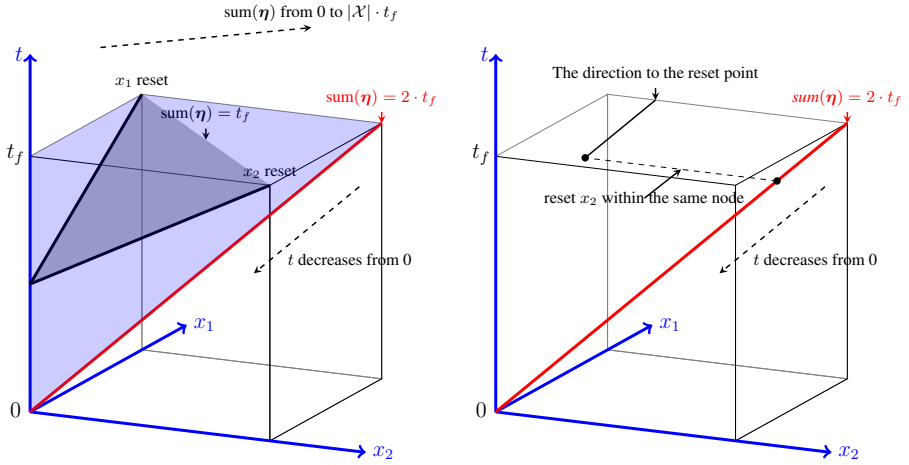
Input: $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$, the region graph of the product of CTMC $\mathcal{C}$ and DTA $\mathcal{A}$; t_f, the time bound
Output: A numerical solution for $\hbar_{v_0}(0, \mathbf{0})$, an approximation of $\mathit{Pr}(\mathcal{C} \models \mathcal{A}[t_f])$

1: **for** $n \leftarrow 0$ **to** $|\mathcal{X}| \cdot t_f$ **by** δ **do**
2: **for each** $\boldsymbol{\eta}$ in $\{\boldsymbol{\eta'} \mid \text{sum}(\boldsymbol{\eta'}) = n \wedge \forall i \in \{1, \ldots, |\mathcal{X}|\}\ 0 \leq \boldsymbol{\eta}^{(i)} \leq t_f\}$ **do**
3: **for** t **from** 0 **down to** $-\min(t_f, \boldsymbol{\eta})$ **do**
4: Compute numerical solution to (6) with $(y_0, \boldsymbol{\eta}_0) = (t_f, \boldsymbol{\eta})$ on $[t_f - t, t_f]$
5: **end for**
6: **end for**
7: **end for**
8: **return** numerical solution for $\hbar_{v_0}(0, \mathbf{0})$

Line 4 in Algorithm 1 computes a numerical solution to (6) on $[t_f - t, t_f]$ by discretizing $\frac{d\hbar_v((y_0, \boldsymbol{\eta}_0) + t)}{dt}$ with $\frac{1}{\delta}(\hbar_v((y_0, \boldsymbol{\eta}_0) + (t + \delta)) - \hbar_v((y_0, \boldsymbol{\eta}_0) + t))$. A pictorial illustration to Algorithm 1 for the two-dimensional setting is shown in Fig. 5. The blue polyhedron covers all the points we need to calculate. The algorithm starts from $(0, 0, t_f)$, where $\text{sum}(\boldsymbol{\eta}) = x_1 + x_2 = 0$. Then $\text{sum}(\boldsymbol{\eta})$ is incremented until $2t_f$ in a

stepwise manner. For each fixed sum($\boldsymbol{\eta}$), for example sum($\boldsymbol{\eta}$) $= t_f$, the algorithm calculates all discrete points in the gray plane following the direction $(-1,-1,-1)$, and finally reaches the two reset lines. The red line reaching the origin provides the final result.

Fig. 5. Illustrating Algorithm 1 (left) and Algorithm 2 (right) for the 2-dimensional setting



Example 4. Consider the product $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$ shown in Example 3 (in page 8). For state v_3 in which clock x is 1 and y is arbitrary, the corresponding PDE is

$$\frac{\partial \hbar_{v_3}(y,1)}{\partial y} + \frac{\partial \hbar_{v_3}(y,1)}{\partial x} + r_0[0.5 \cdot \hbar_{v_0}(y,0) + 0.2 \cdot \hbar_{v_4}(y,0) + 0.4 \cdot \hbar_{v_9}(y,0) - \hbar_{v_3}(y,0)] = 0.$$

Since $\text{sum}(y,0) = y < y+1 = \text{sum}(y,1)$, the value for $\hbar_{v_0}(y,0)$, $\hbar_{v_4}(y,0)$ and $\hbar_{v_3}(y,0)$ have been calculated in the previous iterations, thus the value for $\hbar_{v_3}(y,1)$ can be computed.

To optimize Algorithm 1 for multi-clock objects, we exploit the idea of "lazy computation". In Algorithm 1, in order to determine the reset part for (6), we calculate all discretized points generated by all ODEs. The efficiency is influenced since the amount of ODEs is quite large (the same as the number of states in product automaton). However in Algorithm 2, we only compute the reset part that we need for computing $\hbar_{v_0}(0,\mathbf{0})$. If we meet a reset part $\hbar_v(y, \boldsymbol{\eta}[X := 0])$ which has not been decided yet, we suspend the equation we are computing now and switch to compute the equation leading to the undecided point following the direction of $(-1,\ldots,-1)$. The algorithm terminates since the number of points it computes is no more than that of Algorithm 1. A pseudo-code is described in Algorithm 2.

4.4 Complexity Analysis

Let $|S|$ be the number of the states of the CTMC, and n the number of the clocks of the DTA. The worst-case time complexity of Algorithms 1 and 2 lies in $\mathcal{O}(|V| \cdot \lceil \frac{t_f}{\delta} \rceil^{(n+1)})$, where $|V|$ is the number of the equations in (4), i.e., the number of the locations in the

Algorithm 2 The lazy computation to find numerical solution to (4)

Input: $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$, the region graph of the product of CTMC $\mathcal{C}$ and DTA $\mathcal{A}$; t_f, the time bound
Output: A numerical solution for $\hbar_{v_0}(0, \mathbf{0})$, an approximation of $Pr(\mathcal{C} \models \mathcal{A}[t_f])$
Procedure dhv$(y, \boldsymbol{\eta})$ //Computes numerical solution for $(y, \boldsymbol{\eta})$

for t from 0 down to $-\min(t_f, \boldsymbol{\eta})$ by δ **do**
  for $v \in V$ **do**
    Check if $\boldsymbol{\eta}$ satisfies initial and boundary condition from Theorem 2
    for each Markovian transition $v \xrightarrow{p,X} v'$ **do**
      $up = (-t - \delta) \cdot \mathbf{1} + ((t + \delta) \cdot \mathbf{1})[X := 0]$
      if reset exists and $\boldsymbol{\eta}[X := 0] + up$ is undecided **then**
        call dhv$(t_f, \boldsymbol{\eta}[X := 0] + up)$
      end if
      comput h_v
    end for
  end for
  execute $\lambda-$transition according to Theorem 2
  compute $\hbar_v((y_0, \boldsymbol{\eta_0}) + t)$ by equation (6)
end for
mark $\boldsymbol{\eta}$ decided

End Procedure

Procedure dhv$(v_0, t_f, (\boldsymbol{t_f}))$
return numerical solution for $\hbar_{v_0}(0, \mathbf{0})$

product region graph, that are not accepting. The number of states in the region graph of the DTA is bounded by $n! \cdot 2^{n-1} \cdot \prod_{x \in \mathcal{X}}(c_x + 1)$, denoted by C_b, where c_x is the maximum constant occurring in the guards that constrain x. Note that C_b differs from the bound given in [1], since the boundaries of a region do not matter in our setting and hence can be merged into the region. Thus, the number of states in the product region graph, as well as the number of PDE equations in Theorem 2, is at most $C_b \cdot |S|$. So the total complexity is $\mathcal{O}(C_b \cdot |S| \cdot \lceil \frac{t_f}{\delta} \rceil^{(n+1)})$.

Let $\hbar_{v,n}(y_0, \boldsymbol{\eta}_0)$ denote the numerical solution of ODE (6) with $t = -n\delta$, and $\Lambda_{max} = \max\{\Lambda(v_i) \mid 0 \leq i \leq |S|\}$. Let $N = \lceil \frac{t_f}{\delta} \rceil$. By Proposition 2 , $\lim_{t_f \to +\infty} \hbar_v(0, \mathbf{0}) =$ $Pr(\mathcal{C} \models \mathcal{A})$ and $\hbar_v(0, \mathbf{0})$ is monotonically increasing for t_f. In the following proposition, for simplicity of discussion, we assume t_f equal to $N\delta$. Then, the error caused by discretization can be estimated as follows:

Proposition 3. *For $N \in \mathbb{N}^+$ and $\delta = \frac{t_f}{N}$,*

$$|\hbar_{v_0,N}(t_f, t_f \cdot \mathbf{1}) - \hbar_{v_0}(0, \mathbf{0})| = \mathcal{O}(\delta)$$

For function $f(\delta)$, f is of the magnitude $\mathcal{O}(\delta)$ if $\limsup_{\delta \to 0} \frac{f(\delta)}{\delta} = C$, where C is a constant. From Proposition 3, if we view $\Lambda_{\max}$ and t_f as constants, then the error is $\mathcal{O}(\delta)$ to the step length δ. By Proposition 2, the numerical solution generated by Algorithm 1 converges to the reachability probability of $\mathcal{C} \otimes \mathcal{A}$, and the error can be as small as we expect if we decrease the size of discretization δ, and increase the time bound t_f.

5 Experimental results

We implemented a prototype including Algorithm 1&2 in C and a tool taking a CTMC $\mathcal{C}$ and a DTA $\mathcal{A}$ as input and generating a .c file to store their product in Python, which is used as an input to Algorithm 1&2. The first two examples (Examples 5&6) come from [10] to show the feasibility of our tool. The last case study is an example of robot navigation from [7]. In order to demonstrate the scalability of our approach, we revise the example with different real-time requirements, which require DTA with different number of clocks. The examples are executed in Linux 16.04 LTS with Intel(R) Core(TM) i7-4710HQ 2.50GHz CPU and 16G RAM. The column "time" reports the running time for Algorithm 1, and "time (lazy)" reports the running time for Algorithm 2. All time is counted in seconds.

Example 5. Consider Example 3 with $r_i = 1$, $i = 0, \ldots 3$ and $\delta = 0.01$, experimental result is shown in Table 1. The relevant error when $t_f = 30$ and $t_f = 40$ is 5×10^{-7}.

Table 1. The experimental results for Example 5 and Example 6

t_f	Example 5			Example 6		
	$\hbar_{v_0}(0, \mathbf{0})$	time	time (lazy)	$\hbar_{v_0}(0, \mathbf{0})$	time	time (lazy)
20	0.110791	0.8070	0.7232	0.999999	0.1685	0.0002
30	0.110792	1.7246	1.6260	0.999999	0.3453	0.0003
40	0.110792	3.0344	2.8760	0.999999	0.6265	0.0003

Example 6. Consider the reachability probability for the product of a CTMC and a DTA as shown in Fig. 6. A part of its region graph is shown in Fig. 7. Set $r_0 = r_1 = 1$, $\delta = 0.1$, the experimental result is given in Table 1. The relevant error when $t_f = 30$ and t_f=40 is 1×10^{-7}. Note that even for this simple example, none of existing tools can handle it.

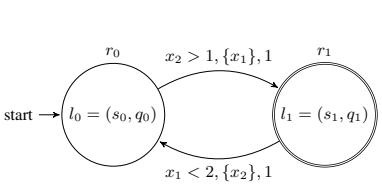


Fig. 6. The product automaton of Example 6

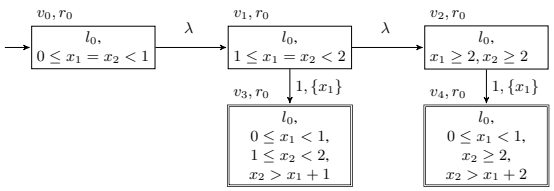


Fig. 7. The reachable product region graph of Figure 6.

Example 7. Consider a robot moves on a $N \times N$ grid as shown in Fig. 8 (adapted from [7]). It can move up, down, left and right. For each possible direction, the robot moves with the same probability. The cells are grouped with A, B, C and D. We consider the following real-time constraints:

P_1: The robot is allowed to stay in adjacent C-cells for at most T_1 time units, and D-cells for at most T_2 time units;

P_2: The total time of the robot continuously resides in adjacent C-cell and D-cell is no more than T_3 time units, with $T_1 \le T_3$ and $T_2 \le T_3$;

P_3: The total time of the robot continuously resides in adjacent A-cell and C-cell is no more than T_4 time units, with $T_1 \leq T_4$.

In this example, we are verifying whether the CTMC satisfies (i) P_1; (ii) $P_1 \wedge P_2$; (iii) $P_1 \wedge P_2 \wedge P_3$. Obviously, P_1 can be expressed by a DTA with one clock, see Fig. 9; to express $P_1 \wedge P_2$, a DTA with two clocks is necessary, see Fig. 10; to express $P_1 \wedge P_2 \wedge P_3$, A DTA with three clocks is necessary, see Fig. 11.

A	A	A	A	B
A	A	A	C	C
D	D	D	C	C
D	D	C	C	C
A	A	A	C	C

Fig. 8. An example grid

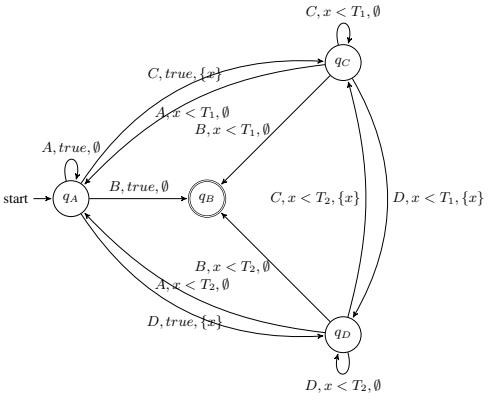
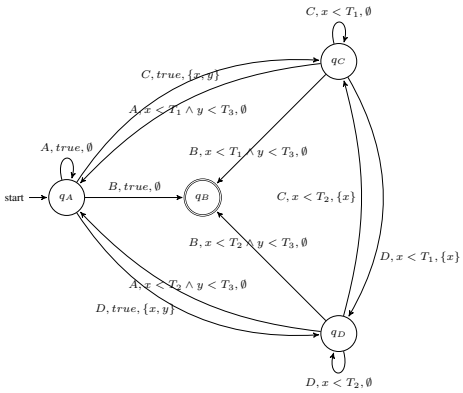


Fig. 9. A DTA with one clock for P_1



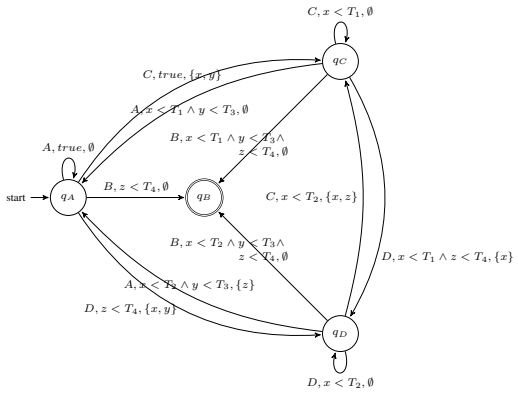


Fig. 10. A DTA with two clocks for $P_1 \wedge P_2$ **Fig. 11.** A DTA with three clocks for $P_1 \wedge P_2 \wedge P_3$

The experimental results are summarized in Table 2. The relevant error of $t_f = 20$ and $t_f = 21$ is smaller than 10^{-2}. As can be seen, the running time of our approach heavily depends on the number of clocks. Compared with the results reported in [7] for the case of one clock in this case study (when the precision is set to be 10^{-2}), our result is as fast as theirs, but their tool cannot handle the cases of multiple clocks. In contrast, our approach can handle DTA with multiple clocks as indicate in the verification of P_2 and P_3. Algorithm 2 is much more faster than Algorithm 1 when the number of clocks

grows up. To the best of our knowledge, this is the first prototypical tool verifying CTMCs against multi-clock DTA.

Table 2. Experimental results for the robot example with $\delta = 0.1$, running time longer than 2700s is denoted by 'TO' (timeout), the column "#(P)" counts the number of states in the product automaton $\mathcal{C} \otimes \mathcal{G}(\mathcal{A})$, "time([7])" is the running time of prototype in [7] when precision=0.01, $T_1 = T_2 = 3, T_3 = 5, T_4 = 7$

		one clock				two clocks			three clocks		
N	t_f	#(P)	time	time (lazy)	time([7])	#(P)	time	time (lazy)	#(P)	time	time (lazy)
4	10	39	0.027	0.027	0.011	139	2.583	1.746	733	525.7	141.4
	15		0.049	0.043			7.117	3.445		TO	257.35
	20		0.070	0.071			12.88	5.49		TO	583.76
10	10	232	0.167	0.164	0.087	968	39.41	25.92	5134	TO	1039.7
	15		0.278	0.278			108.48	53.28		TO	TO
	20		0.417	0.421			226.56	89.50		TO	TO
20	10	940	1.142	0.909	1.23	4000	250.1	180.7		TO	TO
	15		1.65	1.54			672.8	375.6		TO	TO
	20		2.54	2.41			1326.8	616.1		TO	TO
30	10	2125	2.38	2.45	6.84	9120	812.9	380.5		TO	TO
	15		4.45	5.42			2058.1	770.8		TO	TO
	20		7.45	7.28			TO	1283.4		TO	TO
40	10	3820	5.62	6.52	20.31	16395	1484.3	759.8		TO	TO
	15		11.97	11.02			TO	1619.9		TO	TO
	20		15.26	16.17			TO	2661.3		TO	TO

6 Concluding Remarks

In this paper, we present a practical approach to verify CTMCs against DTA objectives. First, the desired probability can be reduced to the reachability probability of the product region graph in the form of PDPs. Then we use the augmented PDP to approximate the reachability probability, in which the reachability probability coincides with the solution to a PDE system at the starting point. We further propose a numerical solution to the PDE system by reduction it to a ODE system. The experimental results indicate the efficiency and scalability compared with existing work, as it can handle DTA with multiple clocks.

As a future work, it deserves to investigate whether our approach also works in the verification of CTMCs against more complicated real-time properties, either expressed by timed automata and MTL as considered in [9], or by linear duration invariants as considered in [8].

Acknowledgements. This research is partly funded by the Sino-German Center for Research Promotion as part of the project CAP (GZ 1023), from Yijun Feng, Haokun Li and Bican Xia is partly funded by NSFC under grant No. 61732001 and 61532019, from Joost-Pieter Katoen is partly funded by the DFG Research Training Group 2236 UnRAVeL, from Naijun Zhan is funded partly by NSFC under grant No. 61625206 and 61732001, by "973 Program" under grant No. 2014CB340701 and by the CAS/SAFEA International Partnership Program for Creative Research Teams.

References

1. Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Science 126(2), 183–235 (1994)
2. Amparore, E.G., Beccuti, M., Donatelli, S.: (Stochastic) model checking in GreatSPN. In: Petri Nets. pp. 354–363 (2014)
3. Aziz, A., Sanwal, K., Singhal, V., Brayton, R.: Model-checking continous-time Markov chains. ACM Trans. on Comp. Logic 1(1), 162–170 (2000)
4. Baier, C., Cloth, L., Haverkort, B.R., Kuntz, M., Siegle, M.: Model checking Markov chains with actions and state labels. IEEE Trans. on Softw. Eng. 33(4) (2007)
5. Baier, C., Haverkort, B., Hermanns, H., Katoen, J.P.: Model-checking algorithms for continuous-time Markov chains. IEEE Trans. on Softw. Eng. 29(6), 524–541 (2003)
6. Baier, C., Katoen, J.P.: Principles of Model Checking. MIT press (2008)
7. Barbot, B., Chen, T., Han, T., Katoen, J.P., Mereacre, A.: Efficient CTMC model checking of linear real-time objectives. In: TACAS. LNCS, vol. 6605, pp. 128–142 (2011)
8. Chen, T., Diciolla, M., Kwiatkowska, M., Mereacre, A.: Verification of linear duration properties over continuous-time Markov chains. ACM Trans. on Comp. Logic 14(4), 33 (2013)
9. Chen, T., Diciolla, M., Kwiatkowska, M.Z., Mereacre, A.: Time-bounded verification of ctmcs against real-time specifications. In: FORMATS. LNCS, vol. 6919, pp. 26–42 (2011)
10. Chen, T., Han, T., Katoen, J.P., Mereacre, A.: Quantitative model checking of continuous-time Markov chains against timed automata specifications. In: LICS. pp. 309–318 (2009)
11. Chen, T., Han, T., Katoen, J., Mereacre, A.: Model checking of continuous-time Markov chains against timed automata specifications. Logical Methods in Computer Science 7(1) (2011)
12. Chen, T., Han, T., Katoen, J.P., Mereacre, A.: Observing continuous-time MDPs by 1-clock timed automata. In: RP 2011. LNCS, vol. 6945, pp. 2–25 (2011)
13. Davis, M.H.: Markov Models & Optimization, vol. 49. CRC Press (1993)
14. Dehnert, C., Junges, S., Katoen, J., Volk, M.: A storm is coming: A modern probabilistic model checker. In: CAV (2). LNCS, vol. 10427, pp. 592–600. Springer (2017)
15. Donatelli, S., Haddad, S., Sproston, J.: Model checking timed and stochastic properties with CSL^{TA}. IEEE Trans. on Softw. Eng. 35(2), 224–240 (2009)
16. Feller, W.: An Introduction to Probability Theory and Its Applications, vol. 3. John Wiley & Sons New York (1968)
17. Fu, H.: Approximating acceptance probabilities of CTMC-paths on multi-clock deterministic timed automata. In: HSCC. pp. 323–332. ACM (2013)
18. Gao, Y., Xu, M., Zhan, N., Zhang, L.: Model checking conditional CSL for continuous-time Markov chains. Information Processing Letters 113(1-2), 44–50 (2013)
19. Grossmann, C., Roos, H.G., Stynes, M.: Numerical Treatment of Partial Differential Equations, vol. 154. Springer (2007)
20. Hung, D.V., Chaochen, Z.: Probabilistic duration calculus for continuous time. Formal Asp. Comput. 11(1), 21–44 (1999)
21. Johnson, C.: Numerical Solution of Partial Differential Equations by the Finite Element Method. Courier Corporation (2012)
22. Katoen, J.P., Zapreev, I.S., Hahn, E.M., Hermanns, H., Jansen, D.N.: The ins and outs of the probabilistic model checker MRMC. Performance Evaluation 68(2), 90–104 (2011)
23. Kwiatkowska, M.Z., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: CAV. LNCS, vol. 6806, pp. 585–591. Springer (2011)
24. Mikeev, L., Neuhäußer, M.R., Spieler, D., Wolf, V.: On-the-fly verification and optimization of DTA-properties for large Markov chains. Formal Methods in System Design 43(2), 313–337 (2013)

25. Wisniewski, R., Sloth, C., Bujorianu, M.L., Piterman, N.: Safety verification of piecewise-deterministic Markov processes. In: HSCC. pp. 257–266. ACM (2016)
26. Yin, G.G., Zhang, Q.: Continuous-time Markov Chains and Applications: A Two-time-scale Approach, vol. 37. Springer Science & Business Media (2012)
27. Zhang, L., Jansen, D.N., Nielson, F., Hermanns, H.: Efficient CSL model checking using stratification. Logical Methods in Computer Science 8(2:17), 1–18 (2012)
28. Zhou, C., Hoare, C.A.R., Ravn, A.P.: A calculus of durations. Information Processing Letters 40(5), 269–276 (1991)