Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties

Hongfei Fu
Shanghai Jiao Tong University
fuhf@ios.ac.cn

Jianlin Li
College of Computer Science and Technology
Nanjing University of Aeronautics and Astronautics
ljlin@nuaa.edu.cn

Yi Li
Department of Informatics, School of Mathematical Sciences, Peking University
liyi_math@pku.edu.cn

Lijun Zhang
State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences
zhanglj@ios.ac.cn

ABSTRACT
Probabilistic timed automata (PTAs) are timed automata (TAs) extended with discrete probability distributions. They serve as a mathematical model for a wide range of applications that involve both stochastic and timed behaviours. In this work, we consider the problem of model-checking linear dense-time properties over PTAs. In particular, we study linear dense-time properties that can be encoded by TAs with infinite acceptance criterion. First, we show that the problem of model-checking PTAs against deterministic-TA specifications can be solved through a product construction. Based on the product construction, we prove that the computational complexity of the problem with deterministic-TA specifications is EXPTIME-complete. Then we show that when relaxed to general (nondeterministic) TAs, the model-checking problem becomes undecidable. Our results substantially extend state of the art with both the dense-time feature and the nondeterminism in TAs.

ACM Reference Format:
Hongfei Fu, Yi Li, Jianlin Li, and Lijun Zhang. 1997. Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties. In Proceedings of HSCC 2018, Porto, Portugal, April 2018, 12 pages.
https://doi.org/10.475/123_4

1 INTRODUCTION
Stochastic timed systems are systems that exhibit both timed and stochastic behaviours. Such systems play a dominant role in many real-world applications [4], hence addressing fundamental issues such as safety and performance over these systems are important. Probabilistic timed automata (PTAs) [6, 22, 26] serve as a good mathematical model for these systems. They extend the well-known model of timed automata [1] (for nonprobabilistic timed systems) with discrete probability distributions, and Markov Decision Processes (MDPs) [28] (for untimed probabilistic systems) with timing constraints.

Formal verification of PTAs has received much attention in recent years [26]. For branching-time model-checking of PTAs, the problem is reduced to computation of reachability probabilities over MDPs through well-known finite abstractions for timed automata (namely regions and zones) [6, 14, 22]. Advanced techniques for branching-time model checking of PTAs such as inverse method and symbolic method have been further explored in [3, 15, 19, 23]. Extension with cost or reward, resulting in priced PTAs, has also been well investigated. Jurdzinski et al. [16] and Kwiatkowska et al. [21] proved that several notions of accumulated or discounted cost are computable over priced PTAs, while cost-bounded reachability probability over priced PTAs is shown to be undecidable by Berendsen et al. [7]. Most verification algorithms for PTAs have been implemented in the model checker PRISM [20]. Computational complexity of several verification problems for PTAs is studied in [16, 17, 24].

For linear-time model-checking, much less is known. As far as we know, the only relevant result is by Sproston [30] who proved that the problem of model-checking PTAs against linear discrete-time properties encoded by untimed deterministic omega-regular automata (e.g., Rabin automata) can be solved by a product construction. In his paper, Sproston first devised a production construction that produces a PTA out of the input PTA and the automaton; then he proved that the problem can be reduced to omega-regular verification of MDPs through maximal end components.

In this paper, we study the problem of model-checking linear dense-time properties over PTAs. Compared with discrete-time properties, dense-time properties take into account timing constraints and therefore is more expressive and applicable to time-critical systems. Simultaneously, verification of dense-time properties is more challenging since it requires
to track timing constraints. The extra feature of timing constraints also brings more theoretical difficulty, e.g., timed automata [1] (TAs) are generally not determinizable, which is in contrast to untimed automata (such as Rabin or Muller automata).

We focus on linear dense-time properties that can be encoded by TAs. TAs are finite automata extended with clocks and timing constraints. Due to the ability to model dense-time behaviours, they can be used to model real-time systems, while they can also act as language recognizers for timed omega-regular languages. Here we treat TAs as language recognizers for timed paths from a PTA, and study the problem to compute the probability that a timed path from the PTA is accepted by the TA. The intuition is that a TA can recognize the set of “good” (or “bad”) timed paths emitting from a PTA, so the problem is to compute the probability that the PTA behaves in a good (or bad) manner. The relationship between TAs and linear temporal logic (e.g., Metric Temporal Logic [18]) is studied in [2, 27].

Our Contributions. We distinguish between the subclass of deterministic TAs (DTAs) and general nondeterministic TAs. DTAs are the deterministic subclass of TAs. Although the class of DTAs is weaker than general timed automata, it can recognize a wide class of formal timed languages, and express interesting linear dense-time properties which cannot be expressed in branching-time logics (cf. [12]). We consider Rabin acceptance condition as the infinite acceptance criterion for TAs. We first show that the problem of model-checking PTAs against DTA specifications with Rabin acceptance condition can be solved through a nontrivial product construction which tackles the integrated feature of timing constraints and randomness. From the product construction, we further prove that the problem is EXPTIME-complete. Then we show that the problem becomes undecidable when one considers general TAs. Our results substantially extend previous ones (e.g. [30]) with both the dense-time feature and the nondeterminism in TAs.

Due to lack of space, detailed proofs of several results are put in the appendix.

2 PRELIMINARIES

We denote by $\mathbb{N}$, $\mathbb{N}_0$, $\mathbb{Z}$, and $\mathbb{R}$ the sets of all positive integers, non-negative integers, integers and real numbers, respectively. For any infinite word $w = b_0b_1 \ldots$ over an alphabet $\Sigma$, we denote by $\inf(w)$ the set of symbols (i.e., $b_i$’s) in $\Sigma$ that occur infinitely many times in $w$.

A clock is a variable for a nonnegative real number. Below we fix a finite set $X$ of clocks.

Clock Valuations. A clock valuation is a function $\nu : X \rightarrow [0, \infty)$. The set of clock valuations is denoted by $\text{Val}(X)$. Given a clock valuation $\nu$, a subset $X \subseteq X$ of clocks and a non-negative real number $t$, we let (i) $\nu[X := 0]$ be the clock valuation such that $\nu[X := 0](x) = 0$ for $x \in X$ and $\nu[X := 0](x) = \nu(x)$ otherwise, and (ii) $\nu + t$ be the clock valuation such that $(\nu + t)(x) = \nu(x) + t$ for all $x \in X$. We denote by $0$ the clock valuation such that $0(x) = 0$ for $x \in X$.

Clock Constraints. The set $\text{CC}(X)$ of clock constraints over $X$ is generated by the following grammar:

$$\phi ::= \text{true} \mid x \leq d \mid c \leq x \mid x + c \leq y + d \mid \neg \phi \mid \phi \land \phi$$

where $x, y \in X$ and $c, d \in \mathbb{N}_0$. We write $\text{false}$ for a short hand of $\neg \text{true}$. The satisfaction relation $|=\text{between valuations } \nu$ and clock constraints $\phi$ is defined through substituting every $x \in X$ appearing in $\phi$ by $\nu(x)$ and standard semantics for logical connectives. For a given clock constraint $\phi$, we denote by $[[\phi]]$ the set of all clock valuations that satisfy $\phi$.

2.1 Probabilistic Timed Automata

A discrete probability distribution over a countable non-empty set $U$ is a function $q : U \rightarrow [0, 1]$ such that $\sum_{z \in U} q(z) = 1$. The support of $q$ is defined as $\text{supp}(q) := \{z \in U \mid q(z) > 0\}$. We denote the set of discrete probability distributions over $U$ by $\mathcal{D}(U)$.

Definition 2.1 (Probabilistic Timed Automata [26]). A probabilistic timed automaton (PTA) $C$ is a tuple

$$C = (L, l^*, X, Act, inv, enab, prob, AP, L)$$

(1)

where:

- $L$ is a finite set of locations;
- $l^* \in L$ is the initial location;
- $X$ is a finite set of clocks;
- $Act$ is a finite set of actions;
- $inv : L \rightarrow \text{CC}(X)$ is an invariant condition;
- $enab : L \times Act \rightarrow \text{CC}(X)$ is an enabling condition;
- $prob : L \times Act \rightarrow \mathcal{D}(2^X \times L)$ is a probabilistic transition function;
- $AP$ is a finite set of atomic propositions;
- $L : L \rightarrow 2^{AP}$ is a labelling function.

W.l.o.g, we consider that both $Act$ and $AP$ is disjoint from $[0, \infty)$. Below we fix a PTA $C$ in the form (1). The semantics of PTAs is as follows.

States and Transition Relation. A state of $C$ is a pair $(l, \nu)$ in $L \times \text{Val}(X)$ such that $\nu \models inv(l)$. The set of all states is denoted by $S_C$. The transition relation $\rightarrow$ consists of all triples $((l, \nu), a, (l', \nu'))$ satisfying the following conditions:

- $(l, \nu), (l', \nu')$ are states and $a \in Act \cup [0, \infty);$
- if $a \in [0, \infty)$ then $\nu + \tau \models inv(l)$ for all $\tau \in [0, a]$ and $(l', \nu') = (l, \nu + a);$
- if $a \in Act$ then $\nu \models enab(l, a)$ and there exists a pair $(X, l''') \in \text{supp}(\text{prob}(l, a))$ such that $(l'', \nu') = (l''', \nu[X := 0]).$
Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties

By convention, we write \(s^a \rightarrow s'\) instead of \((s, a, s') \in \rightarrow\). We omit ‘\(C\)’ in ‘\(s_C\)’ if the underlying context is clear.

**Probability Transition Kernel.** The probability transition kernel \(P\) is the function \(P : S \times \text{Act} \times S \rightarrow [0, 1]\) such that

\[
P((\ell, v), a, (\ell', v')) =
\begin{cases}
1 & \text{if } (\ell, v)^a(\ell', v') \text{ and } a \in [0, \infty) \\
\sum_{Y \in B} \text{prob}(\ell, a)(Y, \ell') & \text{if } (\ell, v)^a(\ell', v') \text{ and } a \in \text{Act} \\
0 & \text{otherwise}
\end{cases}
\]

where \(B := \{X \subseteq X \mid v[X := 0]\}\).

**Well-formedness.** We say that \(C\) is well-formed if for every state \((\ell, v)\) and action \(a \in \text{Act}\) such that \(v \models \text{enab}(\ell, a)\) and every \((X, \ell') \in \supp(\text{prob}(\ell, a))\), one has that \(v[X := 0] \models \text{inv}(\ell')\). The well-formedness is to ensure that when an action is enabled, the next state after taking this action will always be legal. In the following, we always assume that the underlying PTA is well-formed. Non-well-formed PTAs can be repaired into well-formed PTAs [23].

**Paths.** A finite path \(\rho\) (under \(C\)) is a finite sequence

\[
(s_0, a_0, s_1, \ldots, a_{n-1}, s_n) \quad (n \geq 0)
\]

in \((S \times ((\text{Act} \cup [0, \infty)) \times S))^\omega\) such that (i) \(s_0 = (\ell', 0)\), (ii) \(a_2k \in [0, \infty)\) (resp. \(a_{2k+1} \in \text{Act}\)) for all integers \(0 \leq k \leq \frac{n}{2}\) (resp. \(0 \leq k < \frac{n}{2}\)) and (iii) for all \(0 \leq k \leq n - 1\), \(s_k \xrightarrow{a_k} s_{k+1}\). The length \(|\rho|\) of \(\rho\) is defined by \(|\rho| := n\). An infinite path (under \(C\)) is an infinite sequence

\[
(s_0, a_0, s_1, a_1, \ldots)
\]

in \((S \times ((\text{Act} \cup [0, \infty)) \times S))^\omega\) such that for all \(n \in \mathbb{N}_0\), the prefix \(s_0, a_0, s_1, a_1, \ldots, s_n\) is a finite path. The set of finite (resp. infinite) paths under \(C\) is denoted by \(\text{Paths}_C\) (resp. \(\text{Paths}_C^\omega\)).

**Schedulers.** A scheduler is a function \(\sigma\) from the set of finite paths into \(\text{Act} \cup [0, \infty]\) such that for all finite paths \(\rho = s_0a_0 \ldots s_n\), (i) \(\sigma(\rho) \in \text{Act}\) (resp. \(\sigma(\rho) \in [0, \infty)\)) if \(n\) is odd (resp. even) and (ii) there exists a state \(s'\) such that \(s_n \xrightarrow{\sigma(\rho)} s'\).

**Paths under Schedulers.** A finite path \(s_0a_0 \ldots s_n\) follows a scheduler \(\sigma\) if for all \(0 \leq m < n\), \(a_m = \sigma(s_0a_0 \ldots s_m)\). An infinite path \(s_0a_0s_1a_1 \ldots\) follows \(\sigma\) if for all \(n \in \mathbb{N}_0\), \(a_n = \sigma(s_0a_0 \ldots s_n)\). The set of finite (resp. infinite) paths following a scheduler \(\sigma\) is denoted by \(\text{Paths}_{C, \sigma}^\omega\) (resp. \(\text{Paths}_{C, \sigma}\)). We note that the set \(\text{Paths}_{C, \sigma}\) is countably infinite from definition.

**Probability Spaces under Schedulers.** Let \(\sigma\) be any scheduler. The probability space w.r.t \(\sigma\) is defined as

\[
(\Omega_{C, \sigma}^\omega, \mathcal{F}_{C, \sigma}^\omega, \mathbb{P}_{C, \sigma}^\omega)
\]

where (i) \(\Omega_{C, \sigma}^\omega := \text{Paths}_{C, \sigma}^\omega\), (ii) \(\mathcal{F}_{C, \sigma}^\omega\) is the smallest sigma-algebra generated by all cylinder sets induced by finite paths for which a finite path \(\rho\) induces the cylinder set \(\text{Cyl}(\rho)\) of all infinite paths in \(\text{Paths}_{C, \sigma}^\omega\) with \(\rho\) being their (common) prefix, and (iii) \(\mathbb{P}_{C, \sigma}^\omega\) is the unique probability measure such that for all finite paths \(\rho = s_0a_0 \ldots s_n\) in \(\text{Paths}_{C, \sigma}^\omega\),

\[
\mathbb{P}_{C, \sigma}^\omega(\text{Cyl}(\rho)) = \prod_{k=0}^{n-1} \mathbb{P}(s_k, \sigma(s_0a_0 \ldots s_{k-1}s_k), s_{k+1}).
\]

For details see [22].

**Zenoness and Time-Divergent Schedulers.** An infinite path \(\pi = s_0a_0s_1a_1 \ldots\) is zenoness if \(\sum_{n=0}^{\infty} d_n < \infty\), where \(d_n := a_n\) if \(a_n \in [0, \infty)\) and \(a_n := 0\) otherwise. Then a scheduler \(\sigma\) is time divergent if \(\mathbb{P}_{C, \sigma}^\omega(\{\pi \mid \pi\text{ is zenoness}\}) = 0\). In the following, we only consider time-divergent schedulers. The purpose is to eliminate non-realistic zenoness behaviours (i.e., performing infinitely many actions within a finite amount of time).

In the following example, we illustrate a PTA which models a simple task-processing example.

**Example 2.2.** Consider the PTA depicted in Figure 1. \(\text{WORK}_\alpha\), \(\text{WORK}_\beta\) are locations and \(x\) is the only clock. Below each location first comes (vertically) its invariant condition and then the set of labels assigned to the location. For example, \(\text{inv}(\text{WORK}_\alpha) = x \leq 10 \text{ and } L(\text{WORK}_\alpha) = \{a\}\). The two dot points together with corresponding arrows refer to two actions \(\tau_\alpha, \tau_\beta\) with their enabling conditions and transition probabilities given by the probabilistic transition function. For example, the upper dot at the right of \(\text{WORK}_\alpha\) refers to the action \(\tau_\alpha\) for which

- \(\text{enab}(\text{WORK}_\alpha, \tau_\alpha) = \text{true}\), and
- \(\text{prob}(\text{WORK}_\alpha, \tau_\alpha)(\{x\}, \text{WORK}_\alpha) = 0.1\), and
- \(\text{prob}(\text{WORK}_\alpha, \tau_\beta)(\{x\}, \text{WORK}_\beta) = 0.9\).

The PTA models a faulty machine which processes two different kinds of jobs (i.e., \(\alpha, \beta\)) in an alternating fashion. If the machine fails to complete the current job, then it will repeat processing the job until it completes the job. For job \(\alpha\), the machine always processes the job within 10 time units (cf. the invariant condition \(x \leq 10\), but may fail to complete the job with probability 0.1; Analogously, the machine always processes the job \(\beta\) within 15 time units (cf. the invariant condition \(x \leq 15\) with probability 0.9).
condition $x \leq 15$, but may fail to complete the job with probability 0.2. Note that we omit the initial location in this example.

### 2.2 Timed Automata

**Definition 2.3 (Timed Automata [11–13]).** A timed automaton (TA) $\mathcal{A}$ is a tuple

$$\mathcal{A} = (Q, \Sigma, \mathcal{Y}, \Delta)$$

where

- $Q$ is a finite set of modes;
- $\Sigma$ is a finite alphabet of symbols disjoint from $[0, \infty)$;
- $\mathcal{Y}$ is a finite set of clocks;
- $\Delta \subseteq Q \times \Sigma \times \mathcal{X}(\mathcal{Y}) \times 2^\mathcal{Y} \times Q$ is a finite set of rules.

$\mathcal{A}$ is a deterministic TA (DTA) if the following holds:

1. (determinism) for $(q_1, b_1, \phi_1, X_1, q_1') \in \Delta$ ($i \in \{1, 2\}$), if $(q_1, b_1) = (q_2, b_2)$ and $[\phi_1] \cap [\phi_2] \neq \emptyset$ then $(q_1, X_1, q_1') = (q_2, X_2, q_2')$;
2. (totality) for all $(q, b) \in Q \times \Sigma$ and $\nu \in \mathcal{X}(\mathcal{Y})$, there exists $(q, b, \phi, X, q') \in \Delta$ such that $\nu \models \phi$.

Below we illustrate the semantics of TAs. We fix a TA $\mathcal{A}$ in the form (2).

**Configurations and One-Step Transition Relation.** A configuration is a pair $(q, \nu)$, where $q \in Q$ and $\nu \in \mathcal{X}(\mathcal{Y})$. The one-step transition relation

$$\Rightarrow \subseteq (Q \times \mathcal{X}(\mathcal{Y})) \times (Q \cup \{0, \infty\}) \times (Q \times \mathcal{X}(\mathcal{Y}))$$

is defined by: $((q, \nu), a, (q', \nu')) \Rightarrow$ iff either (i) $a \in [0, \infty)$ and $(q', \nu') = (q, \nu + a)$ or (ii) $a \in \Sigma$ and there exists a rule $(q, a, \phi, X, q') \in \Delta$ such that $\nu \models \phi$ and $\nu' = \nu[X := 0]$. For the sake of convenience, we write $((q, \nu), a, (q', \nu')) \Rightarrow ((q, \nu), a, (q', \nu'))$. Note that if $\mathcal{A}$ is deterministic, then there is a unique $(q', \nu')$ such that $(q, \nu) \Rightarrow (q', \nu')$ given any $(q, \nu), a$.

**Infinite Timed Words and Runs.** An infinite timed word is an infinite sequence $w = \{a_n\}_{n \in \mathbb{N}}$ such that $a_{n+1} \in [0, \infty)$ and $a_{n+1}$ for all $n$; the infinite timed word $w$ is time-divergent if $\sum_{n \in \mathbb{N}} a_n = \infty$. A run of $\mathcal{A}$ on an infinite timed word $w = \{a_n\}_{n \in \mathbb{N}}$ with initial configuration $(q, \nu)$, is an infinite sequence $\xi = \{q_n, \nu_n, a_n\}_{n \in \mathbb{N}}$ satisfying that $(q_0, \nu_0) = (q, \nu)$ and $(q_n, \nu_n) \Rightarrow (q_{n+1}, \nu_{n+1})$ for all $n \in \mathbb{N}$. The trajectory $\text{tra}(\xi)$ of the run $\xi$ is defined as an infinite word over $\mathcal{Y}$ such that $\text{tra}(\xi) := q_0 \nu_1 \ldots$. Note that if $\mathcal{A}$ is deterministic, then there is a unique run on every infinite timed word.

**Example 2.5.** Consider the DTRA depicted in Figure 2. In the figure, INIT, $q_\alpha$, $q_\beta$ and FAIL are modes with the Rabin condition $\mathcal{F} = \{(\text{FAIL}), \{q_\alpha, q_\beta\}\}$, $y$ is a clock and arrows between modes are rules. $C_\gamma, W_\gamma$ ($y \in \{\alpha, \beta\}$) are undetermined integer constants. For example, there are four rules emitting from $q_\alpha$: $$(q_\alpha, \{\alpha\}, y \leq C_\alpha, \emptyset), (q_\alpha, \{\beta\}, y \leq W_\beta, \{y\}, q_\beta), (q_\alpha, \{\alpha\}, C_\alpha < y, \emptyset, \text{FAIL}), (q_\alpha, \{\beta\}, W_\beta < y, \emptyset, \text{FAIL}).$$

**INIT** is the initial mode to read the first symbol upon which transitioning to either $q_\alpha$ or $q_\beta$. **FAIL** is a deadlock mode from which all rules go to itself. Note that the rules of the DTRA does not satisfy the totality condition. However, we assume that all missing rules lead to the mode **FAIL** and does not affect the Rabin acceptance condition. The mode $q_\alpha$ does not
Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties

reset the clock \( y \) until it reads \( \beta \). Moreover, \( q_a \) does not transit to \textit{FAIL} only if the time spent within a maximal consecutive segment of \( \alpha \)'s (in an infinite timed word) is no greater than \( C_\alpha \) time units (cf. the rule \( (q_a, \{ \alpha \}, y \leq C_\alpha, 0, q_a) \)) and the total time from the start of the segment until \( \beta \) is read (the time within a maximal consecutive segment of \( \alpha \)'s plus the time spent on the last \( \alpha \) in the segment) is no greater than \( W_\beta \) (cf. the rule \( (q_a, \{ \beta \}, y \leq W_\beta, \{ y \}, q_\beta) \)). The behaviour of the mode \( q_\beta \) can be argued similar to that of \( q_a \) with the only difference to flip \( \sigma \) and \( \beta \). From Rabin acceptance condition, the DTRA specifies a property on infinite timed words that the time spent within a maximal consecutive segment of \( \alpha \)'s (resp. \( \beta \)'s) and the total time until \( \beta \) (resp. \( \alpha \)) is read always satisfy the conditions specified by \( q_a \) (resp. \( q_\beta \)).

3 PROBLEM STATEMENT

In this part, we define the PTA-TRA problem of model-checking PTAs against TA-specifications. The problem takes a PTA and a TRA as input, and computes the probability that infinite paths under the PTA are accepted by the TRA. Informally, the TRA enforces that the TRA accepts any path under the PTA that satisfies the TRA's acceptance condition. In practice, the TRA can be used to capture all good (or bad) behaviours, so the problem can be treated as a task to evaluate its external behaviour, then the problem is to compute the acceptance probabilities over infinite paths under \( C \).

Definition 3.2 (Path Acceptance). An infinite path \( \pi \) under \( C \) is accepted by \( \mathcal{A} \) w.r.t initial configuration \( (q, 0) \), written as the single predicate \( \text{ACC} (\mathcal{A} (q, 0), \pi) \), if there is a configuration \( (q', 0) \) such that \( (q, 0) === (q', v') \) and the infinite word \( L(\pi) \) is Rabin-accepted by \( \mathcal{A} \) with \( (q', v') \).

The initial location omitted in Definition 3.1 is preprocessed by specifying explicitly that the first label \( L(\ell') \) is read by the initial configuration \( (q, 0) \). Below we define acceptance probabilities over infinite paths under \( C \).

Definition 3.3 (Acceptance Probabilities). The probability that \( C \) observes \( \mathcal{A} \) under scheduler \( \sigma \) and initial mode \( q \in Q \), denoted by \( v_\sigma^q \), is defined by:

\[
v_\sigma^q := \mathbb{P}_{C, \sigma} (\text{AccPaths}^C_{C, \sigma})
\]

where \( \text{AccPaths}^C_{C, \sigma} \) is the set of infinite paths under \( C \) that are accepted by the TRA \( \mathcal{A} \) w.r.t. \( (q, 0) \):

\[
\text{AccPaths}^C_{C, \sigma} = \{ \pi \in \text{Paths}^\omega_{C, \sigma} \mid \text{ACC} ((\mathcal{A} (q, 0), \pi)) \}.
\]

Since the set \( \text{Paths}^\omega_{C, \sigma} \) is countably-infinite, \( \text{AccPaths}^C_{C, \sigma} \) is measurable since it can be represented as a countable intersection of certain countable unions of some cylinder sets (cf. [4, Remark 10.24] for details).

Now we introduce the PTA-TRA problem.

- \textbf{Input:} a well-formed PTA \( C \), a TRA \( \mathcal{A} \) and an initial mode \( q \) in \( \mathcal{A} \);
- \textbf{Output:} if \( v_\sigma^q \) and \( \sup_\sigma v_\sigma^q \), where \( \sigma \) ranges over all time-divergent schedulers for \( C \).

We refer to the problem as PTA-DTRA if \( \mathcal{A} \) is deterministic.

4 THE PTA-DTRA PROBLEM

In this section, we solve the PTA-DTRA problem through a product construction. Based on the product construction, we also settle the complexity of the problem. Below we fix a well-formed PTA \( C \) in the form (1) and a DTRA \( \mathcal{A} \) in the form (3). W.l.o.g, we consider that \( X \cap Y = \emptyset \) and \( \Sigma = 2^{AP} \).

The Main Idea. The core part of the product construction is a PTA which preserves the probability of the set of infinite paths accepted by \( \mathcal{A} \). The intuition is to let \( \mathcal{A} \) reads external actions of \( C \) while \( C \) evolves along the time axis. The major difficulty is that when \( C \) performs actions in \( Act \), there is a probabilistic choice between the target locations. Then \( \mathcal{A} \) needs to know the labelling of the target location and the rule (in \( \Delta \)) used for the transition. A naive solution is to integrate each single rule in \( \Delta \) into the enabling condition \( \text{enab} \) in \( C \). However, this simple solution does not work since a single rule fixes the labelling of a location in \( C \), while...
the probability distribution (given by \( \text{prob} \)) can jump to locations with different labels. We solve this difficulty by integrating into the enabling condition enough information on clock valuations under \( \mathcal{A} \) so that the rule used for the transition is clear.

**The Product Construction.** For each \( q \in Q \), we let

\[
\mathcal{T}_q := \{ h : \Sigma \to CC(\mathcal{Y}) \mid \\
\forall b \in \Sigma. (q, b, h(b), X, q') \in \Delta \text{ for some } X, q' \}.
\]

The totality of \( \Delta \) ensures that \( \mathcal{T}_q \) is non-empty. Intuitively, every element of \( \mathcal{T}_q \) is a tuple of clock constraints \( \{ \phi_b \}_{b \in \Sigma} \), where each clock constraint \( \phi_b \) is chosen from the rules emitting from \( q \) and \( b \).

**The product construction** of \( \mathcal{C} \otimes \mathcal{A}_q \) (between \( \mathcal{C} \) and \( \mathcal{A} \) with initial mode \( q \)) is defined as

\[
(L_q, \ell_q^*, \nu_q, \alpha_q, \beta_q, \mu_q, \mathcal{A}_q)
\]

where:

- \( L_q := L \times Q \);
- \( \ell_q^* := (\ell^*, q^*) \) where \( q^* \) is the unique mode such that \((q, 0) \xrightarrow{\ell}(q^*, 0)\);
- \( \nu_q := X \cup \mathcal{Y} \);
- \( \alpha_q := \text{Act}_q \times \bigcup_q \mathcal{T}_q \);
- \( \beta_q := \text{inv}_q(f, q) := \text{inv}(f) \) for all \( (f, q) \in \mathcal{L}_q \);
- \( \mu_q := \text{prob}_q ((\ell, q), (a, h)) = \text{enab}(f, a) \wedge \bigwedge_{b \in \Sigma} h(b) \) for all \((\ell, q) \in \mathcal{L}_q \) and \( h \in \mathcal{T}_q \), and \( \text{enab}_q ((\ell, q), (a, h)) := \text{false otherwise} \);
- \( L_q(\ell, q) := \{ q \} \) for all \((\ell, q) \in \mathcal{L}_q \);
- \( \text{prob}_q ((\ell, q), (a, h))((\ell', q')) := \begin{cases} \text{prob}(\ell, a)(Y \cap X, \ell') & \text{if } (q, \mathcal{L}(\ell'), h(\mathcal{L}(\ell')), Y \cap \mathcal{Y}, q') \text{ is a unique rule in } \Delta \\ 0 & \text{otherwise} \end{cases} \)

Besides standard constructions (e.g., the Cartesian product between \( L \) and \( Q \)), the product construction also has Cartesian product between \( \text{Act} \) and \( \bigcup_q \mathcal{T}_q \). For each extended action \((a, h)\), the enabling condition for this action is the conjunction between \( \text{enab}(\ell, a) \) and all clock constraints from \( h \). This is to ensure that when the action \((a, h)\) is taken, the clock valuation under \( \mathcal{A} \) satisfies every clock constraint in \( h \). Then in the definition for \( \text{prob} \), upon the action \((a, h)\), the product PTA first perform probabilistic jump from \( C \) with the target location \( \ell' \), then chooses the unique rule \((q, \mathcal{L}(\ell'), h(\mathcal{L}(\ell')))) \) from the emitting mode \( q \) and the label \( \mathcal{L}(\ell') \) for which the uniqueness comes from the determinism of \( \mathcal{A} \), then perform the discrete transition from \( \mathcal{A} \). Finally, we label each \((\ell, q)\) by \( q \) to meet the Rabin acceptance condition.

It is easy to see that the PTA \( C \otimes \mathcal{A}_q \) is well-formed as \( C \) is well-formed and \( \mathcal{A} \) does not introduce extra invariant conditions.

**Example 4.1.** The product PTA between the PTA in Example 2.2 and the DTRA in Example 2.5 is depicted in Figure 3. In the figure, \((\text{WORK}_\alpha, \nu_q), (\text{WORK}_\beta, \nu_q)\) and \((\text{WORK}_\alpha, \text{FAIL}), (\text{WORK}_\beta, \text{FAIL})\) are product locations. We omit the initial location and unreachable locations in the product construction. From the construction of \( \mathcal{T}_q \)'s, the functions \( h_i \)'s are as follows (we omit redundant labels such as \( \emptyset \) and \( \{ \alpha, \beta \} \) which never appear in the PTA):

- \( h_0 = \{ \{ \alpha \} \to y \leq C_{\alpha}, \{ \beta \} \to y < W_{\beta} \} \);
- \( h_1 = \{ \{ \alpha \} \to y \leq C_{\alpha}, \{ \beta \} \to y < W_{\beta} \} \);
- \( h_2 = \{ \{ \alpha \} \to C_{\alpha} < y, \{ \beta \} \to y \leq W_{\beta} \} \);
- \( h_3 = \{ \{ \alpha \} \to C_{\alpha} < y, \{ \beta \} \to W_{\beta} < y \} \);
- \( h_4 = \{ \{ \beta \} \to y \leq C_{\beta}, \{ \alpha \} \to y \leq W_{\alpha} \} \);
- \( h_5 = \{ \{ \beta \} \to y < C_{\beta}, \{ \alpha \} \to y \leq W_{\alpha} \} \);
- \( h_6 = \{ \{ \beta \} \to C_{\beta} < y, \{ \alpha \} \to y \leq W_{\alpha} \} \);
- \( h_7 = \{ \{ \beta \} \to C_{\beta} < y, \{ \alpha \} \to W_{\alpha} < y \} \).

The intuition is that the DTA accepts all infinite paths under the PTA such that the failing time for job \( y \in (\{ \alpha, \beta \}) \) (the time within the consequent \( y \)’s) should be no greater than \( C_y \) and the waiting time for job \( y \) (the failing time plus the time spent on the last \( y \)) should be no greater than \( W_y \).

Below we clarify the correspondence between \( \mathcal{C}, \mathcal{A} \) and \( C \otimes \mathcal{A}_q \). We first show the relationship between paths under \( \mathcal{C} \) and those under \( C \otimes \mathcal{A}_q \). Informally, paths under \( C \otimes \mathcal{A}_q \) are just paths under \( C \) extended with runs of \( \mathcal{A} \).

**Transformation \( \mathcal{T} \) for Paths from \( \mathcal{C} \) into \( C \otimes \mathcal{A}_q \).** The transformation is defined as the function \( \mathcal{T} : \text{Paths}_{C} \cup \text{Paths}_{C}^\infty \to \text{Paths}_{C \otimes \mathcal{A}_q} \cup \text{Paths}_{C \otimes \mathcal{A}_q}^\infty \) which transform a finite or infinite path under \( C \) into one under \( C \otimes \mathcal{A}_q \). For a finite path \( \rho = (\ell_0, v_0) a_0 \ldots a_{n-1}(\ell_n, v_n) \) under \( C \) (note that \((\ell_0, v_0) = (\ell', \emptyset)\) by definition), we define \( \mathcal{T}(\rho) \) to be the unique infinite path

\[
\mathcal{T}(\rho) := (((\ell_0, q_0), v_0 \cup \mu_0) a_0 \ldots a_{n-1}(\ell_n, q_n), v_n \cup \mu_n) \tag{4}
\]

under \( C \otimes \mathcal{A}_q \) such that the following conditions (\( \uparrow \)) hold:

- \((q, 0) \xrightarrow{\ell}(q_0, \mu_0)\) (note that \( \mu_0 = \emptyset \))
- for all \( 0 \leq k < n \), if \( a_k \in [0, \infty) \) then \( a_k' = a_k \) and \((q_k, \mu_k) \xrightarrow{\ell(a_k, \xi_k)} (q_{k+1}, \mu_{k+1})\)
- for all \( 0 \leq k < n \), if \( a_k \in \text{Act} \) then \( a_k' = (a_k, \xi_k) \) and \((q_k, \mu_k) \xrightarrow{\ell(a_k, \xi_k)} (q_{k+1}, \mu_{k+1})\) where \( \xi_k \) is the unique function such that for each symbol \( b \in \Sigma \), \( \xi(b) \) is the unique clock constraint appearing in a rule emitting from \( q_b \) and with symbol \( b \) such that \( \mu_k \models \xi(b) \).

Likewise, for an infinite path \( \pi = (\ell_0, v_0) a_0(\ell_1, v_1) a_1 \ldots \) under \( C \), we define \( \mathcal{T}(\pi) \) to be the unique infinite path

\[
\mathcal{T}(\pi) := (((\ell_0, q_0), v_0 \cup \mu_0) a_0'(\ell_1, q_1), v_1 \cup \mu_1) a_1' \ldots \tag{5}
\]
under \( C \odot A_q \) such that the three conditions in \( (\dagger) \) hold for all \( k \in \mathbb{N}_0 \) instead of all \( 0 \leq k < n \). From the determinism and totality of \( \mathcal{A} \), it is straightforward to prove the following result.

**Lemma 4.2.** The function \( T \) is a bijection. Moreover, for any infinite path \( \pi \) under \( C \), \( \pi \) is non-zero if \( T(\pi) \) is non-zero.

Below we also show the correspondence on schedulers before and after the product construction.

**Transformation \( \theta \) for Schedulers from \( C \) into \( C \odot A_q \).**

We define the function \( \theta \) from the set of schedulers under \( C \) into the set of schedulers under \( C \odot A_q \) as follows: for any scheduler \( \sigma \) for \( C \), \( \theta(\sigma) \) (for \( C \odot A_q \)) is defined such that for any finite path \( \rho \) under \( C \) where \( \rho = (\ell_0, v_0)a_0 \ldots a_{n-1}(\ell_n, v_n) \) and \( T(\rho) \) given as in \( (4) \).

\[
\theta(\sigma)(T(\rho)) = \begin{cases} 
\sigma(\rho) & \text{if } n \text{ is even} \\
(\sigma(\rho), \lambda(\rho)) & \text{if } n \text{ is odd}
\end{cases}
\]

where \( \lambda(\rho) \) is the unique function such that for each symbol \( b \in \Sigma \), \( \lambda(\rho)(b) \) is the clock constraint in the unique rule emitting from \( q_a \) and with symbol \( b \) such that \( \mu_n \models \lambda(\rho)(b) \).

Note that the well-definedness of \( \theta \) follows from Lemma 4.2.

From Lemma 4.2, the product construction, the determinism and totality of \( \Delta \), one can prove directly the following lemma.

**Lemma 4.3.** The function \( \theta \) is a bijection.

Now we prove the relationship between infinite paths accepted by a DTRA before product construction and infinite paths satisfying certain Rabin condition.

We introduce more notations. First, we lift the function \( T \) to all subsets of paths in the standard fashion: for all subsets \( A \subseteq \text{Paths}_{C}^\omega \cup \text{Paths}_{C}^\omega \), \( T(A) := \{ T(\omega) \mid \omega \in A \} \). Then for an infinite path \( \pi \) under \( C \odot A_q \) in the form \( (5) \), we define the trace of \( \pi \) as an infinite word over \( Q \) by \( \text{trace}(\pi) := q_0q_1 \ldots \).

Finally, for any scheduler \( \sigma \) for \( C \odot A_q \), we define the set \( RPaths_\sigma \) by

\[
RPaths_\sigma := \left\{ \pi \in \text{Paths}_{C \odot A_q}^\omega \mid \text{ACC (inf(trace(\pi)), } T) \right\}.
\]

Intuitively, \( RPaths_\sigma \) is the set of infinite paths under \( C \odot A_q \) that meet the Rabin condition \( T \) from \( A \). The following proposition clarifies the role of \( RPaths_\sigma \).

**Proposition 4.4.** For any scheduler \( \sigma \) for \( C \) and any initial mode \( q \) for \( \mathcal{A} \), we have \( T \left( \text{AccPaths}_{C \odot A_q}^{\omega, \sigma} \right) = RPaths_\theta(\sigma) \).
Finally, we demonstrate the relationship between acceptance probabilities before product construction and Rabin probabilities after product construction. We also clarify the probability of zenoness before and after the product construction. The proof follows standard argument from measure theory.

**Proposition 4.5.** For any scheduler σ for C and mode q, the followings hold:

- \( v^C_q = \sup_{\sigma} (\text{AccPaths}_{C, \sigma}^{\mathcal{A}, q} \cap \text{Paths}_{q}(\sigma)) = \sup_{\sigma} (\text{AccPaths}_{\sigma}^{\mathcal{A}, q}) \)
- \( \sup_{\sigma} (\{ \pi \mid \pi \text{ is zeno} \}) = \sup_{\sigma} (\{\pi' \mid \pi' \text{ is zeno} \}) \)

A side result from Proposition 4.5 says that \( \theta \) preserves time-divergence for schedulers before and after product construction. From Proposition 4.5 and Lemma 4.3, one immediately obtains the following result which transforms the PTA-DTRA problem into Rabin(-accepting) probabilities under the product PTA.

**Corollary 4.6.** For any initial mode q,

\[ \sup_{\sigma} v_q^C = \sup_{\sigma} (\text{AccPaths}_{C, \sigma}^{\mathcal{A}, q} \cap \text{Paths}_{q}(\sigma)) \]

where \( \sup \) refers to either \( \inf \) (infimum) or \( \sup \) (supremum), \( \sigma \) (resp. \( \sigma' \)) range over all time-divergent schedulers for C (resp. \( C \otimes \mathcal{A} \)).

**Solving Rabin Probabilities.** We follow the approach in [30] to solve Rabin probabilities over PTAs. Below we briefly describe the approach. The approach can be divided into two steps. The first step is to ensure time-divergence. This is achieved by (i) making a copy for every location in the PTA, (ii) enforcing a transition from every location to its copy to happen after 1 time-unit elapses, (iii) enforcing a transition from every copy location back to the original one immediately with no time-delay, and (iv) putting a special label \( \text{tick} \) in every copy. Then time-divergence is guaranteed by adding the label \( \text{tick} \) to the Rabin condition. The second step is to transform the problem into limit Rabin properties over MDPs [4, Theorem 10.127]. This step constructs an MDP Reg(C \( \otimes \mathcal{A} \)) from the PTA C \( \otimes \mathcal{A} \) through a region-graph construction so that the problem is reduced to solving limit Rabin properties over Reg(C \( \otimes \mathcal{A} \)).

**5 THE PTA-TRA PROBLEM**

In this section, we study the PTA-TRA problem where the input timed automaton needs not to be deterministic. In contrast to the deterministic case (which is shown to be decidable and EXPTIME-complete in the previous section), we show that the problem is undecidable.

**The Main Idea.** The main idea for the undecidability result is to reduce the universality problem of timed automata to the PTA-TRA problem. The universality problem over timed automata is well-known to be undecidable, as follows.

**Lemma 5.1.** ([1, Theorem 5.2]) Given a timed automaton over an alphabet \( \Sigma \) and an initial mode, the problem of deciding whether it accepts all time-divergent timed words w.r.t Büchi acceptance condition over \( \Sigma \) is undecidable.

Although Lemma 5.1 is on Büchi acceptance condition, it holds also for Rabin acceptance condition since Rabin acceptance condition extends Büchi acceptance condition. Actually the two acceptance conditions are equivalent over timed automata (cf. [1, Theorem 3.20]). We also remark that Lemma 5.1 was originally for multiple initial modes, which can be mimicked by a single initial mode through aggregating all rules emitting from some initial mode as rules emitting from one initial mode.
Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties

Now we prove the undecidability result as follows. The proof idea is that we construct a PTA that can generate every time-divergent timed words with probability 1 by some time-divergent scheduler. Then the TRA accepts all time-divergent timed words iff the minimal probability that the PTA observes the TRA equals 1.

**Theorem 5.2.** Given a PTA $C$ and a TRA $A$, the problem to decide whether the minimal probability that $C$ observes $A$ (under a given initial mode) is equal to 1 is undecidable.

**Proof Sketch.** Let $A = (Q, \Sigma, \mathcal{Y}, \Delta, T)$ be any TRA where the alphabet $\Sigma = \{b_1, b_2, \ldots, b_k\}$ and the initial mode is $q_{\text{start}}$. W.l.o.g., we consider that $\Sigma \subseteq 2^{\text{AP}}$ for some finite set AP. This assumption is not restrictive since what $b_i$’s concretely are is irrelevant, while the only thing that matters is that $\Sigma$ has $k$ different symbols. We first construct the TRA $A' = (Q', \Sigma', \mathcal{Y'}, \Delta', T')$ where:

- $Q' = Q \cup \{q_{\text{init}}\}$ for which $q_{\text{init}}$ is a fresh mode;
- $\Sigma' = \Sigma \cup \{b_0\}$ for which $b_0$ is a fresh symbol;
- $\Delta' = \Delta \cup \{q_{\text{init}}, b_0, \text{true}, \mathcal{Y}, q_{\text{start}}\}$.

Then we construct the PTA

$$C' = (L, \ell', X, Act, inv, \text{enab}, \text{prob}, AP, L)$$

where:

- $L := \Sigma'$, $\ell' := b_0$, $X := \emptyset$ and $Act := \Sigma$;
- $\text{inv}(b_i) := \text{true}$ for $b_i \in L$;
- $\text{enab}(b_i, b_j) := \text{true}$ for $b_i \in L$ and $b_j \in Act$;
- $\text{prob}(b_i, b_j)$ is the Dirac distribution at $(\emptyset, b_j)$ (i.e., $\text{prob}(b_i, b_j)(\emptyset, b_j) = 1$ and $\text{prob}(b_i, b_j)(X, b_j) = 0$ whenever $(X, b_j) \neq (\emptyset, b_j)$), for $b_i \in L$ and $b_j \in Act$;
- $L(b_i) := b_i$ for $b_i \in L$.

Note that we allow no clocks in the construction since clocks are irrelevant for our result. Since we omit clocks, we also treat states (of $C'$) as single locations. One can prove that $A$ accepts all time-divergent timed words over $\Sigma$ with initial mode $q_{\text{start}}$ iff the minimal probability that $C'$ observes $A'$ with initial mode $q_{\text{init}}$ equals 1. For details see Appendix C.

**Remark 3.** Theorem 5.2 shows that the problem to qualitatively decide the minimal probability is undecidable. On the other hand, the decidability of the problem to decide maximum acceptance probabilities is left open.

6 CONCLUSION

In this paper, we studied the problem of model-checking PTAs against timed-automata specifications. We considered Rabin acceptance condition as the acceptance criterion. We first solved the problem with deterministic-timed-automata specifications through a product construction and proved that its computational complexity is EXPTIME-complete. Then we proved that the problem with general timed-automata specifications is undecidable through a reduction from the universality problem of timed automata.

A future direction is zone-based algorithms for Rabin acceptance condition. Another direction is to investigate timed-automata specifications with cost or reward. Besides, it is also interesting to explore model-checking PTAs against Metric Temporal Logic [18].

7 RELATED WORKS

Model-checking TAs or MDPs against omega-regular (dense-time) properties is well-studied (cf. [4, 27, 32], etc.). PTAs extend both TAs and MDPs with either probability or timing constraints, hence require new techniques for verification problems. On one hand, our technique extends techniques for MDPs (e.g. [32]) with timing constraints. On the other hand, our technique is incomparable to techniques for TAs since linear-time model checking of TAs focus mostly on proving decidability of temporal logic formulas (e.g. Metric Temporal logic [2, 18, 27]), while we prove that model-checking PTAs against TA-specifications is undecidable.

Model-checking probabilistic timed models against linear dense-time properties are mostly considered for continuous-time Markov processes (CTMPs). First, Donatelli et al. [12] proved an expressibility result that the class of linear dense-time properties encoded by DTMs is not subsumed by branching-time properties. They also demonstrated an efficient algorithm for verifying continuous-time Markov chains [12] against one-clock DTAs. Then various results on verifying CTMPs are obtained for specifications through DTAs and general timed automata (cf. e.g. [5, 9–13]). The fundamental difference between CTMPs and PTAs is that the former assign probability distributions to time elapses, while the latter treat time-elapses as pure nondeterminism. As a consequence, the techniques for CTMPs cannot be applied to PTAs.

For PTAs, the only relevant result is by Sproston [30] who developed an approach for verifying PTAs against deterministic discrete-time omega-regular automata by a similar product construction. Our results extend his result in two ways. First, our product construction has the extra ability to tackle timing constraints from the DTA. The extension is nontrivial since it needs to resolve the integration between randomness (from the PTA) and timing constraints (from the DTA), and still ensures the EXPTIME-completeness of the problem, matching the computational complexity in the discrete-time case [30]. Second, we have proved an undecidability result in the case of general nondeterministic timed automata, thus extending [30] with nondeterminism.

REFERENCES

[1] Rajeek Alur and David L. Dill. 1994. A Theory of Timed Automata. Theor. Comput. Sci. 126, 2 (1994), 183–235.
[2] Rajeev Alur, Tomáš Feder, and Thomas A. Henzinger. 1996. The Benefits of Relaxing Punctuality. J. ACM 43, 1 (1996), 116–146.
[3] Étienne André, Laurent Fribourg, and Jeremy Sproston. 2013. An extension of the inverse method to probabilistic timed automata. Formal Methods in System Design 42, 2 (2013), 119–145.
[4] Christel Baier and Joost-Pieter Katoen. 2008. Principles of Model Checking. MIT Press.
[5] Benoît Barbot, Taolue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandra Mereacre. 2011. Efficient CTMC Model Checking of Linear Real-Time Objectives. In TACAS (LNCS), Parosh Aziz Abdulla and K. Rustan M. Leino (Eds.), Vol. 6605. Springer, 128–142.
[6] Danièle Beauquier. 2003. On probabilistic timed automata. Theor. Comput. Sci. 292, 1 (2003), 65–84.
[7] Jasper Berendes, Taolue Chen, and David N. Jansen. 2009. Undecidability of Cost-Bounded Reachability in Priced Probabilistic Timed Automata. In TAMC (LNCS), Jianer Chen and S. Barry Cooper (Eds.), Vol. 5352. Springer, 128–137.
[8] Patrick Billingsley. 2012. Probability and Measure (anniversary edition ed.). Wiley.
[9] Luca Bortolussi and Roberta Lanciani. 2015. Fluid Model Checking of Timed Properties, See [29], 172–188.
[10] Tomás Brázdil, Jan Kral, Jan Kretinsky, Antonin Kucera, and Vojtech Rueh. 2011. Measuring Performance of Continuous-Time Stochastic Processes using timed Automata. In HSCC (LNCS), Calin Belta and Franjo Ivancic (Eds.). ACM, 247–261. Report 86.
[11] Taoxue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandra Mereacre. 2011. Model Checking of Continuous-Time Markov Chains Against Timed Automata Specifications. Logical Methods in Computer Science 7, 1 (2011).
[12] Susanna Donatelli, Serge Haddad, and Jeremy Sproston. 2009. Model Checking Timed and Stochastic Properties with CSL* (TA). IEEE Trans. Software Eng. 35, 2 (2009), 224–240.
[13] Hongfei Fu. 2013. Approximating acceptance probabilities of CTMC-paths on multi-clock deterministic timed automata. In HSCC, Calin Belta and Franjo Ivancic (Eds.). ACM, 323–332.
[14] Henrik Ejerbo Jensen. 1996. Model Checking Probabilistic Real Time Systems. In 7th Nordic Workshop on Programming Theory. Chalmers University of Technology, 247–261. Report 86.
[15] Aleksandra Jovanovic, Marta Z. Kwiatkowska, and Gethin Norman. 2015. Symbolic Minimum Expected Time Controller Synthesis for Probabilistic Timed Automata. See [29], 140–155.
[16] Marcin Jurdzinski, Marta Z. Kwiatkowska, and Ashutosh Trivedi. 2009. Concavely-Priced Probabilistic Timed Automata. In CONCUR (LNCS), Mario Bravetti and Gianluigi Zavattaro (Eds.), Vol. 5710. Springer, 415–430.
[17] Marcin Jurdzinski, Jeremy Sproston, and François Laroussinie. 2008. Model Checking Probabilistic Timed Automata with One or Two Clocks. Logical Methods in Computer Science 4, 3 (2008).
[18] Ron Koymans. 1990. Specifying Real-Time Properties with Metric Temporal Logic. Real-Time Systems 2, 4 (1990), 255–299.
[19] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. 2009. Stochastic Games for Verification of Probabilistic Timed Automata. In FORMATS (LNCS), Joel Ouaknine and Frits W. Vaandrager (Eds.), Vol. 5813. Springer, 212–227.
[20] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. 2011. PRISM 4.0: Verification of Probabilistic Real-Time Systems. In CAV (LNCS), Ganesh Gopalakrishnan and Shaz Qadeer (Eds.), Vol. 6806. Springer, 585–591.
[21] Marta Z. Kwiatkowska, Gethin Norman, David Parker, and Jeremy Sproston. 2006. Performance analysis of probabilistic timed automata using digital clocks. Formal Methods in System Design 29, 1 (2006), 78–91.
Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties

A PROOFS FOR TECHNICAL RESULTS

Lemma 4.2. The function $T$ is a bijection. Moreover, for any infinite path $\pi$ under $C$, $\pi$ is non-zeno if $T(\pi)$ is non-zeno.

Proof. The first claim follows directly from the determinism and totality of DTAs. The second claim follows from the fact that $T$ preserves time elapses in the transformation. □

Proposition 4.4. For any scheduler $\sigma$ for $C$ and any initial mode $q$ for $A$, we have $T\left(\text{AccPaths}_{C,\sigma}^{A,q}\right) = \text{RPaths}_{\emptyset}(\sigma)$.\ 

Proof. By definition, the set $\text{AccPaths}_{C,\sigma}^{A,q}$ equals

$$\left\{ \pi \in \text{Paths}_{C,\sigma}^{A,q} \mid \text{ACC}(\inf(\text{traj}(\xi_\pi)), T) \right\}$$

where $\xi_\sigma$ is the unique run of $A$ on $L(\pi)$ with initial configuration $(q', 0)$ for which $q'$ is the unique location such that $(q, 0) \in \Delta^0(\xi)$, for any initial path under $C$. By the definition of $T$ we have

$$T(\pi) = ((\ell_0, v_0), v_0 \cup \mu_0, a_0(\ell_1, v_1), a_1 \cdots)$$

The claim follows from the definition. Hence $\text{traj}(T(\pi)) = q_0a_1 \cdots = \text{traj}(\xi_\sigma)$. It follows that $\inf(\text{traj}(T(\pi)))$ is Rabin accepting by $T$ if $\inf(\text{traj}(\xi_\sigma))$ is Rabin accepting by $T$. Hence the result follows from Lemma 4.2. □

Proposition 4.5. For any scheduler $\sigma$ for $C$ and mode $q$, the followings hold:

- $P_{q}^{C,\sigma} = P_{q}^{C,\sigma}(\text{AccPaths}_{C,\sigma}^{A,q}) = P_{C \otimes A, \sigma}^{\emptyset}(\text{RPaths}_{\emptyset}(\sigma))$;
- $P_{C,\sigma}^{C,\sigma}(\left\{ \pi \mid \text{ACC}(\pi \mid \text{ACC}(\inf(\text{traj}(\xi_\pi)), T) \right\}) = P_{C \otimes A, \sigma}^{\emptyset}(\text{RPaths}_{\emptyset}(\sigma))$.

Proof. Define the probability measure $P'$ by: $P'(A) = P_{C \otimes A, \sigma}^{\emptyset}(\text{RPaths}_{\emptyset}(\sigma))$. We show that $P' = P_{C,\sigma}^{C,\sigma}$. By [8, Theorem 3.3], it suffices to consider cylinder sets as they form a pi-system (cf. [8, Page 43]). Let $\rho = (\ell_0, v_0) a_0 \cdots a_{n-1}(\ell_n, v_n) \ell_n$ be any finite path under $C$. By definition, we have that

$$P_{C,\sigma}^{C,\sigma}(\text{Cyl}(\rho)) = P_{C \otimes A, \sigma}^{\emptyset}(\text{Cyl}(\tau(\rho))) = P_{C \otimes A, \sigma}^{\emptyset}(\Delta(\tau(\rho))) = P'(\text{Cyl}(\rho)).$$

The first equality comes from the fact that the product construction preserves transition probabilities. The second equality is due to $\text{Cyl}(\tau(\rho)) = \text{Cyl}(\rho)$. The final equality follows from the definition. Hence $P_{C,\sigma}^{C,\sigma} = P'$. Then the first claim follows from Proposition 4.4 and the second claim follows from Lemma 4.2. □

B THE HARDNESS RESULT

Below we prove the hardness of the PTA-DTRA problem. It is proved in [25] that the reachability-probability problem for arbitrary PTAs is EXPSPACE-complete. We show a polynomial-time reduction from the PTA reachability problem to the PTA-DTRA problem as follows. For an arbitrary PTA $C$ in the form (1) and a set $F \subseteq L$ of final locations, let $C' = (L, \ell^*, X, \text{Act}, \text{inv, enab, prob, AP, L'})$ where $AP' := AP \cup \{\text{acc}\}$ and $L'$ is defined by

$$L'(l) := \begin{cases} L(l), & \text{if } l \notin F \\ L(l) \cup \{\text{acc}\}, & \text{if } l \in F \end{cases}$$

for which acc is a fresh atomic proposition. We also construct the DTRA $A'$ by

$$A' := ((\{q_0, q_1\}, \Sigma, \emptyset, \Delta, \{\emptyset, \{q_1\}\})$$

where $\Sigma := \{L'(l) \mid l \in L\}$ and $\Delta$ contains exactly the following rules:

- $q_0, U, \text{true}, 0, q_1 \in \Delta$ for all $U \in \Sigma$ such that $\text{acc} \in U$;
- $q_0, U, \text{true}, 0, q_0 \in \Delta$ for all $U \in \Sigma$ such that $\text{acc} \notin U$;
- $\{q_1, U, \text{true}, 0, q_1\} \in \Delta$ for all $U \in \Sigma$.

It is then straightforward from definition that an infinite path under $C$ visits some location in $F$ if and only if the infinite path (under $C'$) is accepted by $A'$ under initial mode $q_0$. Hence, under any scheduler (for both $C$ and $C'$), the probability to reach $F$ in $C$ equals the probability that $C'$ observes $A'$ under initial mode $q_0$. It follows that the problem to compute the maximum/minimum probability to reach $F$ can be polynomially reduced to the PTA-DTRA problem. Hence the problem PTA-DTRA is EXPSPACE-hard.

C PROOF FOR PTA-TRA UNDECIDABILITY

Theorem 5.2. Given a PTA $C$ and a TRA $A$, the problem to decide whether the minimal probability that $C$ observes $A$ (under a given initial mode) is equal to 1 is undecidable.

Proof. Let $A = (Q, \Sigma, \mathcal{Y}, \Delta, F)$ be any TRA where the alphabet $\Sigma = \{b_1, b_2, \cdots, b_k\}$ and the initial mode is $q_{\text{start}}$. For example, we consider that $\Sigma \subseteq 2^{AP}$ for some finite set $AP$. This assumption is not restrictive since what $b_i$'s concretely are is irrelevant, while the only things that matters is that $\Sigma$ has $k$ different symbols. We first construct the TRA $A' = (Q', \mathcal{Y}, \Delta', F')$ where

- $Q' = Q \cup \{q_{\text{init}}\}$ for which $q_{\text{init}}$ is a fresh mode;
- $\Sigma' = \Sigma \cup \{b_0\}$ for which $b_0$ is a fresh symbol;
- $\Delta' = \Delta \cup \{q_{\text{init}}, b_0, \text{true}, \mathcal{Y}, q_{\text{start}}\}$.

Then we construct the PTA $C' = (L, \ell^*, X, \text{Act}, \text{inv, enab, prob, AP, L})$ where:

- $L := \Sigma', \ell^* := b_0, X := \emptyset$ and $\text{Act} := \Sigma$;
\begin{itemize}
  \item \( \text{inv}(b_i) := \text{true} \) for \( b_i \in L \);
  \item \( \text{enab}(b_i, b_j) := \text{true} \) for \( b_i \in L \) and \( b_j \in \text{Act} \);
  \item \( \text{prob}(b_i, b_j) \) is the Dirac distribution at \((\emptyset, b_j)\) (i.e., \( \text{prob}(b_i, b_j)(\emptyset, b_j) = 1 \) and \( \text{prob}(b_i, b_j)(X, b) = 0 \) whenever \((X, b) \neq (\emptyset, b_j)\)), for \( b_i \in L \) and \( b_j \in \text{Act} \);
  \item \( \mathcal{L}(b_i) := b_i \) for \( b_i \in L \).
\end{itemize}

Note that we allow no clocks in the construction since clocks are irrelevant for our result. Since we omit clocks, we also treat states (of \( C' \)) as single locations. Below we prove that \( \mathcal{A} \) accepts all time-divergent timed words over \( \Sigma \) with initial mode \( q_{\text{start}} \) iff the minimal probability that \( C' \) observes \( \mathcal{A}' \) with initial mode \( q_{\text{init}} \) equals 1.

Consider any time-divergent infinite timed word \( w = t_0 b'_0 t_1 b'_1 \cdots \) over \( \Sigma \) (where \( t_i \in \mathbb{R} \) and \( b'_i \in \Sigma \)). We define an infinite sequence \( \{\rho_n\}_{n \in \mathbb{N}_0} \) of finite paths (of \( C' \)) inductively as follows:

\begin{itemize}
  \item \( \rho_0 := b_0 (= \ell^{\ast}) \); (Note that we treat states as locations since clocks are irrelevant.)
  \item for \( m \geq 0 \), \( \rho_{2m+1} := (s_0, a_0, s_1, \ldots, a_{k-1}, s_k, t_m, s_k) \) if \( \rho_{2m} = (s_0, a_0, s_1, \ldots, a_{k-1}, s_k) \);
  \item for \( m \geq 0 \), \( \rho_{2m+2} := (s_0, a_0, s_1, \ldots, a_{k-1}, s_k, b'_m, b'_m) \) if \( \rho_{2m+1} = (s_0, a_0, s_1, \ldots, a_{k-1}, s_k) \).
\end{itemize}

Intuitively, the sequence \( \{\rho_n\}_{n \in \mathbb{N}_0} \) is constructed by letting the PTA \( C' \) read the timed word \( w \) in a stepwise fashion, while adjusting the next location upon reading a symbol (as an action) from \( \Sigma \). Then one can define a scheduler \( \sigma_w \) by:

\begin{itemize}
  \item \( \sigma_w(\rho_{2m}) := t_m \) for \( m \geq 0 \);
  \item \( \sigma_w(\rho_{2m+1}) := b'_m \) for \( m \geq 0 \);
  \item \( \sigma_w(\rho) \) is arbitrarily defined if \( \rho \) is not from the sequence \( \{\rho_n\}_{n \in \mathbb{N}_0} \).
\end{itemize}

Intuitively, \( \sigma_w \) always chooses time-delays and actions from \( w \). From definition, \( P_{C',\sigma_w}^\mathcal{A}(\{\pi | \mathcal{L}(\pi) = w\}) = 1 \). Note that \( \sigma_w \) is time divergent since \( w \) is time divergent. Hence

\[ P_{\mathcal{A}_{\text{init}}}^{\sigma_w} = \begin{cases} 1 & \text{if } \mathcal{A} \text{ accepts } w \text{ with } (q_{\text{start}}, 0), \\ 0 & \text{if } \mathcal{A} \text{ rejects } w \text{ with } (q_{\text{start}}, 0). \end{cases} \]

where the underlying PTA (resp. TRA) is \( C' \) (resp. \( \mathcal{A}' \)). Since all those \( \sigma_w \)'s correspond to all time divergent schedulers for \( C' \), we have that \( \inf_{\sigma} P_{C',\sigma}^\mathcal{A}(\text{AccPaths}_{C',\sigma}^\mathcal{A}_{\text{init}}) = 1 \) iff \( \mathcal{A} \) accepts all time-divergent timed words w.r.t. \( (q_{\text{start}}, 0) \). \( \square \)
This figure "product.png" is available in "png" format from:

http://arxiv.org/ps/1712.00275v1