

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

Anonymous Author(s)

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 paper, we study the problem of model-checking linear *dense-time* temporal properties over PTAs. In particular, we consider 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:

Anonymous Author(s). 1997. Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties. In *Proceedings of HSCC 2018*. ACM, New York, NY, USA, 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 abstraction 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 DTA 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.

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 $w = b_0b_1\dots$, we denote by $\text{inf}(w)$ the set of symbols (i.e., b_i 's) that occur infinitely many times in w . Given a finite word $w = b_0\dots b_n$ ($n \geq 0$), the last symbol b_n is denoted by $\text{last}(w)$.

A *clock* is a variable for a nonnegative real number. Below we fix a finite set \mathcal{X} of clocks.

Clock Valuations. A *clock valuation* is a function $\nu : \mathcal{X} \rightarrow [0, \infty)$. The set of clock valuations is denoted by $\text{Val}(\mathcal{X})$. Given a clock valuation ν , a subset $X \subseteq \mathcal{X}$ 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 $x \in \mathcal{X}$. We denote by $\mathbf{0}$ the clock valuation such that $\mathbf{0}(x) = 0$ for $x \in \mathcal{X}$.

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

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

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

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) \mathcal{C} is a tuple

$$\mathcal{C} = (L, \ell^*, \mathcal{X}, \text{Act}, \text{inv}, \text{enab}, \text{prob}, AP, \mathcal{L}) \quad (1)$$

where:

- L is a finite set of *locations*;
- $\ell^* \in L$ is the *initial* location;
- \mathcal{X} is a finite set of *clocks*;
- Act is a finite set of *actions*;
- $\text{inv} : L \rightarrow CC(\mathcal{X})$ is an *invariant condition*;
- $\text{enab} : L \times \text{Act} \rightarrow CC(\mathcal{X})$ is an *enabling condition*;
- $\text{prob} : L \times \text{Act} \rightarrow \mathcal{D}(2^{\mathcal{X}} \times L)$ is a *probabilistic transition function*;
- AP is a finite set of *atomic propositions*;
- $\mathcal{L} : L \rightarrow 2^{AP}$ 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 \mathcal{C} in the form (1). The semantics of PTAs is as follows.

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

- $(\ell, \nu), (\ell', \nu')$ are states and $a \in \text{Act} \cup [0, \infty)$;
- if $a \in [0, \infty)$ then $\nu + \tau \models \text{inv}(\ell)$ for all $\tau \in [0, a]$ and $(\ell', \nu') = (\ell, \nu + a)$;
- if $a \in \text{Act}$ then $\nu \models \text{enab}(\ell, a)$ and there exists a pair $(X, \ell'') \in \text{supp}(\text{prob}(\ell, a))$ such that $(\ell', \nu') = (\ell'', \nu[X := 0])$.

By convention, we write $s \xrightarrow{a} s'$ instead of $(s, a, s') \in \rightarrow$. We omit ‘ \mathcal{C} ’ in ‘ $S_{\mathcal{C}}$ ’ if the underlying context is clear.

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

that

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

where $B := \{X \subseteq \mathcal{X} \mid \nu' = \nu[X := 0]\}$.

Well-formedness. We say that \mathcal{C} is *well-formed* if for every state (ℓ, ν) and action $a \in \text{Act}$ such that $\nu \models \text{enab}(\ell, a)$ and every $(X, \ell') \in \text{supp}(\text{prob}(\ell, a))$, one has that $\nu[X := 0] \models \text{inv}(\ell')$. 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* ρ (under \mathcal{C}) is a finite sequence

$$\langle s_0, a_0, s_1, \dots, a_{n-1}, s_n \rangle \quad (n \geq 0)$$

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

$$\langle s_0, a_0, s_1, a_1, \dots \rangle$$

in $(S \times (\text{Act} \cup [0, \infty)))^\omega$ such that for all $n \in \mathbb{N}_0$, the prefix $\langle s_0, a_0, \dots, a_{n-1}, s_n \rangle$ is a finite path. The set of finite (resp. infinite) paths under \mathcal{C} is denoted by $\text{Paths}_{\mathcal{C}}^*$ (resp. $\text{Paths}_{\mathcal{C}}^\omega$).

Schedulers. A *scheduler* is a function σ from the set of finite paths into $\text{Act} \cup [0, \infty)$ such that for all finite paths $\rho = s_0 a_0 \dots s_n$, (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 $s_0 a_0 \dots s_n$ follows a scheduler σ if for all $0 \leq m < n$, $a_m = \sigma(s_0 a_0 \dots s_m)$. An infinite path $s_0 a_0 s_1 a_1 \dots$ follows σ if for all $n \in \mathbb{N}_0$, $a_n = \sigma(s_0 a_0 \dots s_n)$. The set of finite (resp. infinite) paths following a scheduler σ is denoted by $\text{Paths}_{\mathcal{C}, \sigma}^*$ (resp. $\text{Paths}_{\mathcal{C}, \sigma}^\omega$). We note that the set $\text{Paths}_{\mathcal{C}, \sigma}^*$ is countably infinite from definition.

Probability Spaces under Schedulers. Let σ be any scheduler. The probability space w.r.t σ is defined as

$$(\Omega^{\mathcal{C}, \sigma}, \mathcal{F}^{\mathcal{C}, \sigma}, \mathbb{P}^{\mathcal{C}, \sigma})$$

where (i) $\Omega^{\mathcal{C}, \sigma} := \text{Paths}_{\mathcal{C}, \sigma}^\omega$, (ii) $\mathcal{F}^{\mathcal{C}, \sigma}$ is the smallest sigma-algebra generated by all cylinder sets induced by finite paths for which a finite path ρ induces the cylinder set $\text{Cyl}(\rho)$ of all infinite paths in $\text{Paths}_{\mathcal{C}, \sigma}^\omega$ with

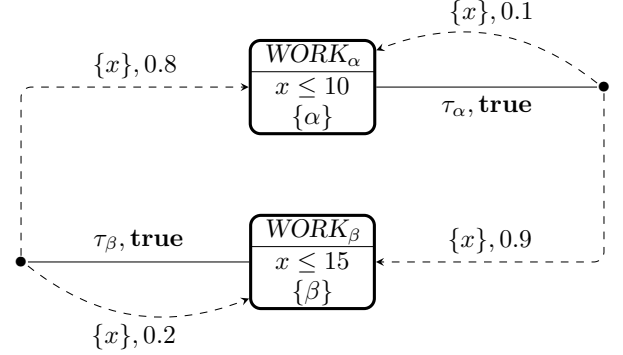


Figure 1: A Simple Task-Processing Example

ρ being their (common) prefix, and (iii) $\mathbb{P}^{\mathcal{C}, \sigma}$ is the unique probability measure such that for all finite paths $\rho = s_0 a_0 \dots a_{n-1} s_n$ in $\text{Paths}_{\mathcal{C}, \sigma}^*$,

$$\mathbb{P}^{\mathcal{C}, \sigma}(\text{Cyl}(\rho)) = \prod_{k=0}^{n-1} \mathbf{P}(s_k, \sigma(s_0 a_0 \dots a_{k-1} s_k), s_{k+1}).$$

For details see [22].

Zenoness and Time-Divergent Schedulers. An infinite path $\pi = s_0 a_0 s_1 a_1 \dots$ is *zeno* if $\sum_{n=0}^{\infty} d_n < \infty$, where $d_n := a_n$ if $a_n \in [0, \infty)$ and $d_n := 0$ otherwise. Then a scheduler σ is *time divergent* if $\mathbb{P}^{\mathcal{C}, \sigma}(\{\pi \mid \pi \text{ is zeno}\}) = 0$. In the following, we only consider time-divergent schedulers. The purpose is to eliminate non-realistic zeno 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, $\text{inv}(\text{WORK}_\alpha) = x \leq 10$ and $\mathcal{L}(\text{WORK}_\alpha) = \{\alpha\}$. The two dot points together with corresponding arrows refer to two actions τ_α, τ_β with their enabling conditions and transition probabilities given by the probabilistic transition function. For example, the upper dot at the right of WORK_α refers to the action τ_α 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
- $\text{prob}(\text{WORK}_\alpha, \tau_\alpha)(\{x\}, \text{WORK}_\beta) = 0.9$.

The PTA models a faulty machine which processes two different kinds of jobs (i.e., α, β) 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 α , 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 β within 15 time units (cf. the invariant 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

$$\mathcal{A} = (Q, \Sigma, \mathcal{Y}, \Delta) \quad (2)$$

where

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

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

- (1) (*determinism*) for $(q_i, b_i, \phi_i, X_i, q'_i) \in \Delta$ ($i \in \{1, 2\}$), if $(q_1, b_1) = (q_2, b_2)$ and $\llbracket \phi_1 \rrbracket \cap \llbracket \phi_2 \rrbracket \neq \emptyset$ then $(\phi_1, X_1, q'_1) = (\phi_2, X_2, q'_2)$;
- (2) (*totality*) for all $(q, b) \in Q \times \Sigma$ and $\nu \in \text{Val}(\mathcal{X})$, 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, ν) , where $q \in Q$ and $\nu \in \text{Val}(\mathcal{Y})$. The *one-step transition relation*

$$\Rightarrow \subseteq (Q \times \text{Val}(\mathcal{Y})) \times (\Sigma \cup [0, \infty)) \times (Q \times \text{Val}(\mathcal{Y}))$$

is defined by: $((q, \nu), a, (q', \nu')) \in \Rightarrow$ 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 $(q, \nu) \xrightarrow{a} (q', \nu')$ instead of $((q, \nu), a, (q', \nu')) \in \Rightarrow$. Note that if \mathcal{A} is deterministic, then there is a unique (q', ν') such that $(q, \nu) \xrightarrow{a} (q', \nu')$ given any $(q, \nu), a$.

Infinite Time Words and Runs. An *infinite time word* is an infinite sequence $w = \{a_n\}_{n \in \mathbb{N}_0}$ such that $a_{2n} \in [0, \infty)$ and $a_{2n+1} \in \Sigma$ for all n ; the infinite timed word w is *time-divergent* if $\sum_{n \in \mathbb{N}_0} a_{2n} = \infty$. A *run* of \mathcal{A} on an infinite timed word $w = \{a_n\}_{n \in \mathbb{N}_0}$ with *initial configuration* (q, ν) , is an infinite sequence $\xi = \{(q_n, \nu_n, a_n)\}_{n \in \mathbb{N}_0}$ satisfying that $(q_0, \nu_0) = (q, \nu)$ and $(q_n, \nu_n) \xrightarrow{a_n} (q_{n+1}, \nu_{n+1})$ for all $n \in \mathbb{N}_0$; the *trajectory* $\text{traj}(\xi)$ of the run ξ is defined as an infinite word over Q such that $\text{traj}(\xi) := q_0 q_1 \dots$. Note that if \mathcal{A} is deterministic, then there is a unique run on every infinite timed word.

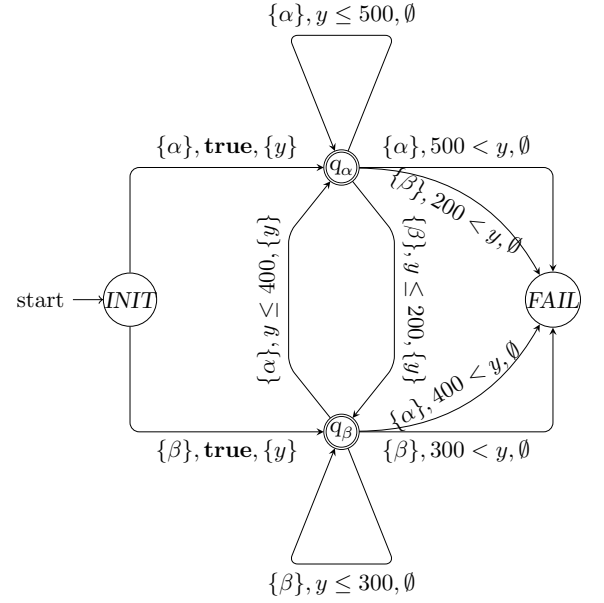


Figure 2: A DTRA Specification

Below we illustrate the acceptance conditions for TAs. We first illustrate infinite acceptance condition for which we consider Rabin acceptance condition.

Definition 2.4 (Rabin Acceptance Condition). A TA with *Rabin acceptance condition* (TRA) is a tuple

$$\mathcal{A} = (Q, \Sigma, \mathcal{Y}, \Delta, \mathcal{F}) \quad (3)$$

where $(Q, \Sigma, \mathcal{Y}, \Delta)$ is a TA and \mathcal{F} is a finite set of pairs $\mathcal{F} = \{(H_1, K_1), \dots, (H_n, K_n)\}$ representing a Rabin condition for which H_i and K_i are subsets of Q for all $i \leq n$. \mathcal{A} is a *deterministic* TRA (DTRA) if $(Q, \Sigma, \mathcal{Y}, \Delta)$ is a DTA. A set $Q' \subseteq Q$ is *Rabin-accepting* by \mathcal{F} , written as the predicate $\mathbf{ACC}(Q', \mathcal{F})$, if there is $1 \leq i \leq n$ such that $Q' \cap H_i = \emptyset$ and $Q' \cap K_i \neq \emptyset$. An infinite timed word w is *Rabin-accepted* by \mathcal{A} with *initial configuration* (q, ν) iff there exists a run ξ of $(Q, \Sigma, \mathcal{Y}, \Delta)$ on w with (q, ν) such that $\text{inf}(\xi)$ is Rabin-accepting by \mathcal{F} .

Example 2.5. Consider the DTRA depicted in Figure 2 which works as a specification for the PTA in Example 2.2. In the figure, $INIT, q_\alpha, q_\beta$ and $FAIL$ are modes with the Rabin condition $\mathcal{F} = \{\{q_\alpha, q_\beta\}\}$, y is a clock and arrows between modes are rules. For example, there are four rules emitting from q_α :

$$(q_\alpha, \{\alpha\}, y \leq 500, \emptyset, q_\alpha), (q_\alpha, \{\beta\}, y \leq 200, \{y\}, q_\beta)$$

$INIT$ is the initial mode to read the label of the initial location of a PTA in the product construction, and $FAIL$

is a trap mode. Note that this DTA does not satisfy the totality condition. However, this can be remedied by adding rules leading to a deadlock mode without changing the acceptance behaviour of the DTA. This DTA specified the property that every α job should be done within 500 units of time after last α job done and every β job should be done within 300 units of time after last β job done.

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 encodes the linear dense-time property by judging whether an infinite path is accepted or not through its external behaviour, then the problem is to compute the probability that an infinite path is accepted by the TRA. 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 to what extent the PTA behaves in a good (or bad) way.

Below we fix a well-formed PTA \mathcal{C} taking the form (1) and a TRA \mathcal{A} taking the form (3). W.l.o.g., we assume that $\mathcal{X} \cap \mathcal{Y} = \emptyset$ and $\Sigma = 2^{AP}$. We first show how an infinite path in $Paths_{\mathcal{C}}^{\omega}$ can be interpreted as an infinite timed word.

Definition 3.1 (Infinite Paths as Infinite Timed Words). Given an infinite path

$$\pi = (\ell_0, \nu_0) a_0 (\ell_1, \nu_1) a_1 (\ell_2, \nu_2) a_2 \dots$$

under \mathcal{C} , the infinite timed word $\mathcal{L}(\pi)$ is defined as

$$\mathcal{L}(\pi) := a_0 \mathcal{L}(\ell_2) a_2 \mathcal{L}(\ell_4) \dots a_{2n} \mathcal{L}(\ell_{2n+2}) \dots$$

Recall that $\nu_0 = \mathbf{0}$, $a_{2n} \in [0, \infty)$ and $a_{2n+1} \in Act$ for $n \in \mathbb{N}_0$.

REMARK 1. Informally, the interpretation in Definition 3.1 works by (i) dropping (a) the initial location ℓ_0 , (b) all clock valuations ν_n 's, (c) all locations ℓ_{2n+1} 's following a time-elapse, (d) all internal actions a_{2n+1} 's of \mathcal{C} and (ii) replacing every ℓ_{2n} ($n \geq 1$) by $\mathcal{L}(\ell_{2n})$. The interpretation captures only external behaviours including time-elapses and labels of locations upon state-change, and discards internal behaviours such as the concrete locations, clock valuations and actions. Although the interpretation ignores the initial location, we deal with it in our acceptance condition where the initial location is preprocessed by the TRA.

Definition 3.2 (Path Acceptance). An infinite path π under \mathcal{C} is accepted by \mathcal{A} w.r.t initial configuration

(q, ν) , written as the single predicate $\mathbf{ACC}(\mathcal{A}, (q, \nu), \pi)$, if there exists a mode $q \in Q$ such that $(q, \nu) \xrightarrow{\mathcal{L}(\ell^*)} (q', \nu')$ and the infinite word $\mathcal{L}(\pi)$ is Rabin- (or finitely-) accepted by \mathcal{A} w.r.t $((q', \nu'), \mathbf{0})$.

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

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

$$\mathbf{p}_q^{\sigma} := \mathbb{P}^{\mathcal{C}, \sigma} \left(\text{AccPaths}_{\mathcal{C}, \sigma}^{\mathcal{A}, q} \right)$$

where $\text{AccPaths}_{\mathcal{C}, \sigma}^{\mathcal{A}, q}$ is the set of infinite paths under \mathcal{C} that are accepted by the TRA \mathcal{A}

$$\text{AccPaths}_{\mathcal{C}, \sigma}^{\mathcal{A}, q} = \{ \pi \in \text{Paths}_{\mathcal{C}, \sigma}^{\omega} \mid \mathbf{ACC}(\mathcal{A}, (q, \mathbf{0}), \pi) \}.$$

Since the set $\text{Paths}_{\mathcal{C}, \sigma}^*$ is countably-infinite, $\text{AccPaths}_{\mathcal{C}, \sigma}^{\mathcal{A}, q}$ 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.

- **Input:** a well-formed PTA \mathcal{C} , a TRA \mathcal{A} and an initial mode q in \mathcal{A} ;
- **Output:** $\inf_{\sigma} \mathbf{p}_q^{\sigma}$ and $\sup_{\sigma} \mathbf{p}_q^{\sigma}$, where σ ranges over all time-divergent schedulers.

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 \mathcal{C} in the form (1) and a DTRA \mathcal{A} in the form (3). W.l.o.g, we consider that $\mathcal{X} \cap \mathcal{Y} = \emptyset$ 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 \mathcal{C} while \mathcal{C} evolves along the time axis. The major difficulty is that when \mathcal{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 Δ) used for the transition. A naive solution is to integrate each single rule in Δ into the enabling condition *enab* in \mathcal{C} . However, this simple solution does not work since a single rule fixes the labelling of a location in \mathcal{C} , while the probability distribution (given by *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

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

The totality of Δ 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 ϕ_b is chosen from the rules emitting from q and b . The *product PTA* $\mathcal{C} \otimes \mathcal{A}_q$ (between \mathcal{C} and \mathcal{A} with initial mode q) is defined as

$$(L_\otimes, \ell_\otimes^*, \mathcal{X}_\otimes, Act_\otimes, inv_\otimes, enab_\otimes, prob_\otimes, \mathcal{L}_\otimes)$$

where:

- $L_\otimes := L \times Q$;
- $\ell_\otimes^* := (\ell^*, q^*)$ where q^* is the unique mode such that $(q, \mathbf{0}) \xrightarrow{\mathcal{L}(\ell^*)} (q^*, \mathbf{0})$;
- $\mathcal{X}_\otimes := \mathcal{X} \cup \mathcal{Y}$;
- $Act_\otimes := Act \times \bigcup_q \mathcal{T}_q$;
- $inv_\otimes(\ell, q) := inv(\ell)$ for all $(\ell, q) \in L_\otimes$;
- $enab_\otimes((\ell, q), (a, h)) := enab(\ell, a) \wedge \bigwedge_{b \in \Sigma} h(b)$ for all $(\ell, q) \in L_\otimes$ and $h \in \mathcal{T}_q$, and $enab_\otimes((\ell, q), (a, h)) := \text{false}$ otherwise;
- $\mathcal{L}_\otimes(\ell, q) := \{q\}$ for all $(\ell, q) \in L_\otimes$;
- $prob_\otimes$ is given by

$$prob_\otimes((\ell, q), (a, h))(Y, (\ell', q')) := \begin{cases} prob(\ell, a)(Y \cap \mathcal{X}, \ell') & \text{if } (q, \mathcal{L}(\ell'), h(\mathcal{L}(\ell')), Y \cap \mathcal{Y}, q') \\ & \text{is a (unique) rule in } \Delta \\ 0 & \text{otherwise} \end{cases}$$

Besides standard constructions (e.g., the Cartesian product between L and Q), the product construction also has Cartesian product between Act and $\bigcup_q \mathcal{T}_q$. For each extended action (a, h) , the enabling condition for this action is the conjunction between $enab(\ell, a)$ 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 $prob_\otimes$, upon the action (a, h) and the target location ℓ' , the product PTA first chooses the unique rule $(q, \mathcal{L}(\ell'), h(\mathcal{L}(\ell')), Y \cap \mathcal{Y}, q')$ from the emitting mode q and the label $\mathcal{L}(\ell')$ for which the uniqueness comes from the determinism of Δ , then perform probabilistic jump from \mathcal{C} and the discrete transition from \mathcal{A} . Finally, we label each (ℓ, q) by q to meet the Rabin or finite acceptance condition. \square

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

REMARK 2 (OUR NOVELTY). *The novelty for our product construction is that by adopting extended actions and integrating them into enabling condition and probabilistic transition function, the DTA \mathcal{A} can know which rule to use upon any symbol to be read. This solves the problem that probabilistic jumps can lead to different locations.*

Example 4.1. Here we represent an running example to show how our product construction works. Here for the accepting mod in DTA, we have

$$\begin{aligned} \mathcal{T}_{q_\alpha} &= \{h_0 = \{\emptyset \mapsto \text{true}, \{\alpha\} \mapsto (y \leq 500), \{\beta\} \mapsto (y \leq 200), \{\alpha, \beta\} \mapsto \text{true}\}, \\ h_1 &= \{\emptyset \mapsto \text{true}, \{\alpha\} \mapsto (y \leq 500), \{\beta\} \mapsto (200 < y), \{\alpha, \beta\} \mapsto \text{true}\}, \\ h_2 &= \{\emptyset \mapsto \text{true}, \{\alpha\} \mapsto (500 < y), \{\beta\} \mapsto (y \leq 200), \{\alpha, \beta\} \mapsto \text{true}\}, \\ h_3 &= \{\emptyset \mapsto \text{true}, \{\alpha\} \mapsto (500 < y), \{\beta\} \mapsto (200 < y), \{\alpha, \beta\} \mapsto \text{true}\} \end{aligned}$$

And a part of the product of Example 2.2 and Example 2.5 is depicted in Figure 2. $(WORK_\alpha, q_\alpha)$, $(WORK_\beta, q_\beta)$, $(WORK_\gamma, q_\gamma)$ are locations.

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

Transformation \mathcal{T} for Paths from \mathcal{C} into $\mathcal{C} \otimes \mathcal{A}_q$. The transformation is defined as the function $\mathcal{T} : Paths_{\mathcal{C}}^* \cup Paths_{\mathcal{C}}^\omega \rightarrow Paths_{\mathcal{C} \otimes \mathcal{A}_q}^* \cup Paths_{\mathcal{C} \otimes \mathcal{A}_q}^\omega$ which transform a finite or infinite path under \mathcal{C} into one under $\mathcal{C} \otimes \mathcal{A}_q$. For a finite path $\rho = (\ell_0, \nu_0)a_0 \dots a_{n-1}(\ell_n, \nu_n)$ under \mathcal{C} (note that $(\ell_0, \nu_0) = (\ell^*, \mathbf{0})$ by definition), we define $\mathcal{T}(\rho)$ to be the unique finite path

$$\mathcal{T}(\rho) := ((\ell_0, q_0), \nu_0 \cup \mu_0)a'_0 \dots a'_{n-1}((\ell_n, q_n), \nu_n \cup \mu_n) \quad (4)$$

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

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

Likewise, for an infinite path $\pi = (\ell_0, \nu_0)a_0(\ell_1, \nu_1)a_1 \dots$ under \mathcal{C} , we define $\mathcal{T}(\pi)$ to be the unique infinite path

$$\mathcal{T}(\pi) := ((\ell_0, q_0), \nu_0 \cup \mu_0)a'_0((\ell_1, q_1), \nu_1 \cup \mu_1)a'_1 \dots \quad (5)$$

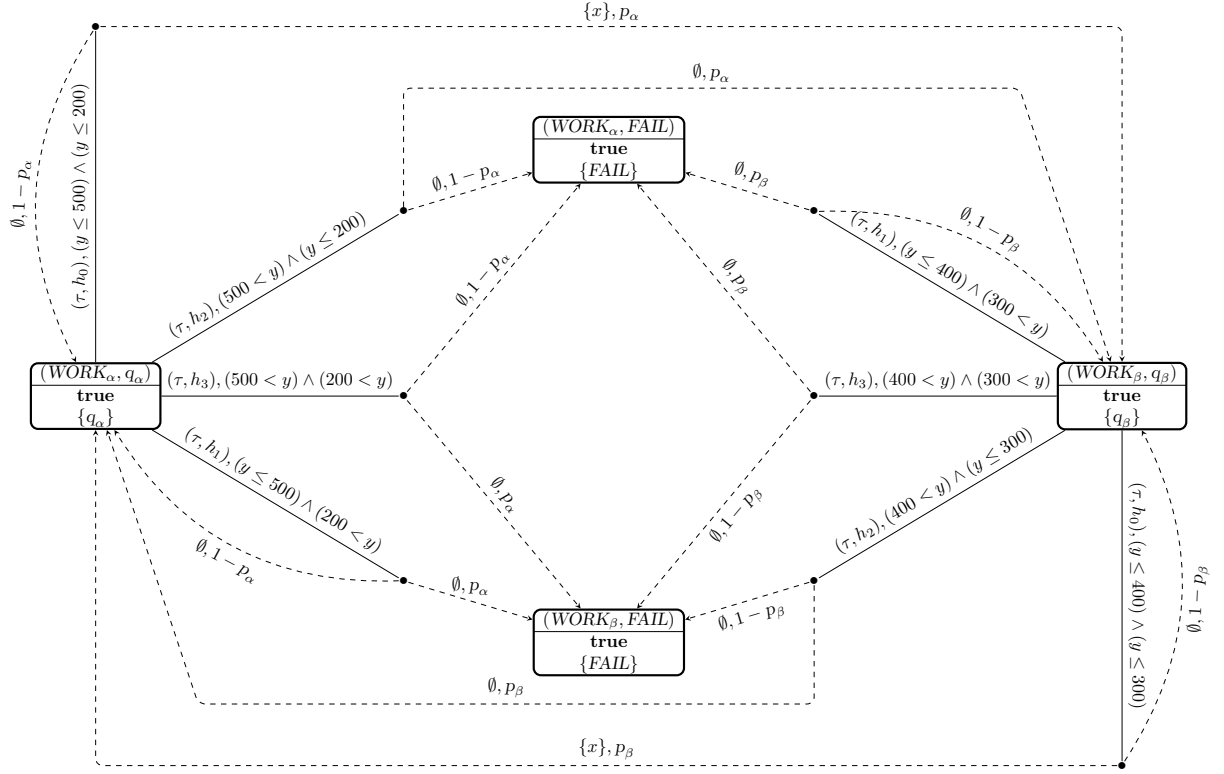


Figure 3: A Part of Product PTA

under $\mathcal{C} \otimes \mathcal{A}_q$ such that the three conditions in (†) hold for all $k \in \mathbb{N}_0$ instead of all $0 \leq k < n$.

LEMMA 4.2. *The function \mathcal{T} is a bijection. Moreover, for any infinite path π , π is non-zero iff $\mathcal{T}(\pi)$ is non-zero.*

PROOF. The first claim follows directly from the determinism and totality of DTAs. The second claim follows from the fact that \mathcal{T} preserves time elapses in the transformation. \square

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

Transformation θ for Schedulers from \mathcal{C} into $\mathcal{C} \otimes \mathcal{A}_q$.

We define the function θ from the set of schedulers under \mathcal{C} into the set of schedulers under $\mathcal{C} \otimes \mathcal{A}_q$ as follows: for any scheduler σ for \mathcal{C} , $\theta(\sigma)$ (for $\mathcal{C} \otimes \mathcal{A}_q$) is defined such that for any finite path ρ under \mathcal{C} where $\rho = (\ell_0, \nu_0)a_0 \dots a_{n-1}(\ell_n, \nu_n)$ and $\mathcal{T}(\rho)$ given as in (4),

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

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_k and with symbol b such that $\mu_n \models \lambda(\rho)(b)$. Note that the well-definedness of θ follows from Lemma 4.2.

From Lemma 4.2, the product construction, the determinism and totality of Δ , one can prove directly the following lemma.

LEMMA 4.3. *The function θ is a bijection.*

Now we prove the relationship between infinite paths accepted by a DTRA/DTFa before product construction and infinite paths satisfying certain Rabin/reachability condition. We first consider that \mathcal{A} is a DTRA in the form (3).

We introduce more notations. First, we lift the function \mathcal{T} to all subsets of paths in the standard fashion: for all subsets $A \subseteq \text{Paths}_{\mathcal{C}}^* \cup \text{Paths}_{\mathcal{C}}^\omega$, $\mathcal{T}(A) := \{\mathcal{T}(\omega) \mid \omega \in A\}$. Then for an infinite path $\pi = (\ell_0, \nu_0)a_0(\ell_1, \nu_1)a_1 \dots$ with $\mathcal{T}(\pi)$ taking the form (5), we define the *trace* of $\mathcal{T}(\pi)$ as an infinite word over Q defined by $\text{trace}(\mathcal{T}(\pi)) := q_0q_1 \dots$. Finally, for any scheduler σ for $\mathcal{C} \otimes \mathcal{A}_q$, we define the set $R\text{Paths}_\sigma$ by

$$R\text{Paths}_\sigma := \left\{ \pi \in \text{Paths}_{\mathcal{C} \otimes \mathcal{A}_q, \sigma}^\omega \mid \mathbf{ACC}(\inf(\text{trace}(\pi)), \mathcal{F}) \right\}.$$

Intuitively, $RPaths_\sigma$ is the set of infinite paths that meet the Rabin condition \mathcal{F} from \mathcal{A} .

PROPOSITION 4.4. *For any scheduler σ for \mathcal{C} and any initial mode q for \mathcal{A} , we have $\mathcal{T}(AccPaths_{\mathcal{C},\sigma}^{\mathcal{A},q}) = RPaths_{\theta(\sigma)}$.*

PROOF. By definition, the set $AccPaths_{\mathcal{C},\sigma}^{\mathcal{A},q}$ equals

$$\{\pi \in Paths_{\mathcal{C},\sigma}^\omega \mid \mathbf{ACC}(\inf(traj(\xi_\pi)), \mathcal{F})\}$$

where ξ_π is the unique run of \mathcal{A} on $\mathcal{L}(\pi)$ with initial configuration $(q^*, \mathbf{0})$ for which q^* is the unique location such that $(q, \mathbf{0}) \xrightarrow{\mathcal{L}(\ell^*)} (q^*, \mathbf{0})$. Let $\pi = (\ell_0, \nu_0)a_0(\ell_1, \nu_1)a_1 \dots$ be any infinite path. By the definition of \mathcal{T} we have

$$\mathcal{T}(\pi) = ((\ell_0, q_0), \nu_0 \cup \mu_0)a'_0((\ell_1, q_1), \nu_1 \cup \mu_1)a'_1 \dots$$

in the form (5) with $\xi_\pi = \{(q_n, \mu_n, a_n)\}_{n \in \mathbb{N}_0}$ being the unique run on $\mathcal{L}(\pi) = a_0a_1 \dots$. Then it is obvious that

$$trace(\mathcal{T}(\pi)) = q_0q_1 \dots = traj(\mathcal{A}_{(q^*, \mathbf{0})}(\mathcal{L}(\pi))).$$

It follows that $\inf(trace(\mathcal{T}(\pi)))$ is Rabin accepting by \mathcal{F} iff $\inf(traj(\mathcal{A}_{(q^*, \mathbf{0})}(\mathcal{L}(\pi))))$ is Rabin accepting by \mathcal{F} . \square

Finally, we demonstrate the relationship between acceptance probabilities before product construction and reachability probabilities after product construction. We also clarify the probability of zenoness before and after the product construction.

PROPOSITION 4.5. *For any scheduler σ and mode q , the followings hold:*

- $\mathbf{p}_q^\sigma = \mathbb{P}^{\mathcal{C},\sigma}(AccPaths_{\mathcal{C},\sigma}^{\mathcal{A},q}) = \mathbb{P}^{\mathcal{C} \otimes \mathcal{A}_q, \theta(\sigma)}(RPaths_{\theta(\sigma)})$;
- $\mathbb{P}^{\mathcal{C},\sigma}(\{\pi \mid \pi \text{ is zeno}\}) = \mathbb{P}^{\mathcal{C} \otimes \mathcal{A}_q, \theta(\sigma)}(\{\pi' \mid \pi' \text{ is zeno}\})$.

PROOF. Define the probability measure \mathbb{P}' by: $\mathbb{P}'(A) = \mathbb{P}^{\mathcal{C} \otimes \mathcal{A}_q, \theta(\sigma)}(\mathcal{T}(A))$ for $A \in \mathcal{F}^{\mathcal{C},\sigma}$. We show that $\mathbb{P}' = \mathbb{P}^{\mathcal{C},\sigma}$. By [8, Theorem 3.3], it suffices to consider cylinder sets as they form a pi-system (cf. [8, Page 43]). Let $\rho = (\ell_0, \nu_0)a_0 \dots a_{n-1}(\ell_n, \nu_n)$ be any finite path under \mathcal{C} . By definition, we have that

$$\begin{aligned} \mathbb{P}^{\mathcal{C},\sigma}(Cyl(\rho)) &= \mathbb{P}^{\mathcal{C} \otimes \mathcal{A}_q, \theta(\sigma)}(Cyl(\mathcal{T}(\rho))) \\ &= \mathbb{P}^{\mathcal{C} \otimes \mathcal{A}_q, \theta(\sigma)}(\mathcal{T}(Cyl(\rho))) \\ &= \mathbb{P}'(Cyl(\rho)) . \end{aligned}$$

The first equality comes from the fact that both versions of product construction preserves transition probabilities. The second equality is due to $Cyl(\mathcal{T}(\rho)) = \mathcal{T}(Cyl(\rho))$. The final equality follows from the definition. Hence $\mathbb{P}^{\mathcal{C},\sigma} = \mathbb{P}'$. Then the first claim follows from Proposition 4.4 and the second claim follows from Lemma 4.2. \square

A side result from Proposition 4.5 says that θ 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-DTA problem into Rabin(-accepting) probabilities under the product PTA.

COROLLARY 4.6. *For any initial mode q ,*

$$opt_\sigma \mathbf{p}_q^\sigma = opt_{\sigma'} \mathbb{P}^{\mathcal{C} \otimes \mathcal{A}_q, \sigma'}(RPaths_{\sigma'})$$

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

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 *tick* in every copy. Then time-divergence is guaranteed by adding the label *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 $\text{Reg}[\mathcal{C} \otimes \mathcal{A}_q]$ from the PTA $\mathcal{C} \otimes \mathcal{A}_q$ through a *region-graph* construction so that the problem is reduced to solving limit Rabin properties over $\text{Reg}[\mathcal{C} \otimes \mathcal{A}_q]$. *Regions* are finitely-many equivalence classes of clock valuations that serve as a finite abstraction which capture exactly reachability behaviours over timed transitions (cf. [1]). Then standard methods based on *maximal end components* (MECs) are applied to $\text{Reg}[\mathcal{C} \otimes \mathcal{A}_q]$. In detail, the algorithm computes the reachability probability to MECs that satisfy the Rabin acceptance condition; In order to guarantee time-divergence, the algorithm only picks up MECs with at least one location that has a *tick* label. Based on this approach, our result leads to an algorithm for solving the problem PTA-DTA.

Note that in $\mathcal{C} \otimes \mathcal{A}_q$, although the size of Act_\otimes may be exponential due to possible exponential blow-up from \mathcal{T}_q , one easily sees that $|L_\otimes|$ is $|L| \cdot |Q|$ and $|\mathcal{X}_\otimes| = |\mathcal{X}| + |\mathcal{Y}|$. Hence, the size of $\text{Reg}[\mathcal{C} \otimes \mathcal{A}_q]$ is still exponential in the sizes of \mathcal{C} and \mathcal{A} . It follows that $opt_\sigma \mathbf{p}_q^\sigma$ can be calculated in exponential time from the MEC-based algorithm illustrated in [4, Theorem 10.127], as is demonstrated by the following proposition.

PROPOSITION 4.7. *The problem PTA-DTA is in EXPTIME in the size of the input PTA and DTA.*

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

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

for which *acc* is a fresh atomic proposition. We also construct the DTRA \mathcal{A}' by

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

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

- $(q_0, U, true, \emptyset, q_1) \in \Delta$ for all $U \in \Sigma$ such that $acc \in U$;
- $(q_i, U, true, \emptyset, q_i) \in \Delta$ for all $U \in \Sigma$ and $i \in \{0, 1\}$.

It is then straightforward from definition that an infinite path under \mathcal{C} visits some location in F iff the infinite path (under \mathcal{C}') is accepted by \mathcal{A}' . Hence, under any scheduler (for both \mathcal{C} and \mathcal{C}'), the probability to reach F in \mathcal{C} equals the probability that \mathcal{C}' observes \mathcal{A}' under initial mode q_0 . It follows that the maximum/minimum probability to reach F can be solved by the algorithm demonstrated in Proposition 4.7 for Rabin acceptance condition. By the reduction, we obtain the main result of this section which settles the computational complexity of the problem PTA-DTA.

THEOREM 4.8. *The PTA-DTA problem is EXPTIME-complete.*

5 THE PTA-TRA PROBLEM

In this section, we study the PTA-NTA problem where the input timed automaton needs not to be deterministic. We show that for Rabin acceptance condition, the problem is undecidable, while the problem can be solved approximately through an infinite-state MDP construction when we restrict ourselves to finite acceptance condition.

We first consider Rabin acceptance condition. The main idea for the undecidability result is to reduce the universality problem of timed automata to the PTA-NTA problem. The universality problem over timed automata is well-known to be undecidable, as follows.

LEMMA 5.1. ([31, Theorem 5.2]) *Given a timed automaton over an alphabet Σ and an initial mode, the problem of deciding whether it accepts all time-divergent timed words w.r.t Büchi acceptance condition over Σ 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. [31, Theorem 3.20]).

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 \mathcal{C} and a TRA \mathcal{A} , the problem to decide whether the minimal probability that \mathcal{C} observes \mathcal{A} (under a given initial mode) is equal to 1 is undecidable.*

PROOF. Let $\mathcal{A} = (Q, \Sigma, \mathcal{Y}, \Delta, \mathcal{F})$ be any TRA where the alphabet $\Sigma = \{b_1, b_2, \dots, b_k\}$ and the initial mode is q_{start} . W.l.o.g, 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 thing that matters is that Σ has k different symbols. We first construct the TRA $\mathcal{A}' = (Q', \Sigma', \mathcal{Y}, \Delta', \mathcal{F})$ where:

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

Then we construct the PTA

$$\mathcal{C}' = (L, \ell^*, \mathcal{X}, Act, inv, enab, prob, \mathcal{L})$$

where:

- $L := \Sigma'$;
- $\ell^* := b_0$;
- $\mathcal{X} := \emptyset$;
- $Act := \Sigma$;
- $inv(b_i) := \mathbf{true}$ for $b_i \in L$;
- $enab(b_i, b_j) := \mathbf{true}$ for $b_i \in L$ and $b_j \in Act$;
- $prob(b_i, b_j)$ is the Dirac distribution at (\emptyset, b_j) (i.e., $\mu(\emptyset, b_j) = 1$ and $\mu(X, b) = 0$ whenever $(X, b) \neq (\emptyset, b_j)$), for $b_i \in L$ and $b_j \in Act$;
- $\mathcal{L}(b_i) := b_i$ 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 \mathcal{C}') as single locations. Below we prove that \mathcal{A} accepts all time-divergent timed words over Σ with initial mode q_{start} iff the minimal probability that \mathcal{C}' observes \mathcal{A}' with initial mode q_{init} equals 1.

Consider any time-divergent infinite timed word $w = t_0 b'_0 t_1 b'_1 \dots$ over Σ (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 \mathcal{C}') inductively as follows:

- $\rho_0 := b_0 (= \ell^*)$; (Note that we treat states as locations since clocks are irrelevant.)

- for $m \geq 0$, $\rho_{2m+1} := \langle s_0, a_0, s_1, \dots, a_{k-1}, s_k, t_m, s_k \rangle$
if $\rho_{2m} = \langle s_0, a_0, s_1, \dots, a_{k-1}, s_k \rangle$;
- for $m \geq 0$, $\rho_{2m+2} := \langle s_0, a_0, s_1, \dots, a_{k-1}, s_k, b'_m, b'_m \rangle$
if $\rho_{2m+1} = \langle s_0, a_0, s_1, \dots, a_{k-1}, s_k \rangle$.

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

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

Intuitively, σ_w always chooses time-delays and actions from w . Note that σ_w is time divergent since w is time divergent. Moreover, from definition we have that $\mathbb{P}^{\mathcal{C}, \sigma_w}(\{\pi \mid \mathcal{L}(\pi) = w\}) = 1$. Hence

$$\mathbf{p}_{q_{\text{init}}}^{\sigma_w} = \begin{cases} 1 & \text{if } \mathcal{A} \text{ accepts } w \text{ w.r.t. } (q_{\text{start}}, \mathbf{0}), \\ 0 & \text{if } \mathcal{A} \text{ rejects } w \text{ w.r.t. } (q_{\text{start}}, \mathbf{0}). \end{cases}$$

Then we have that $\inf_{\sigma} \mathbb{P}^{\mathcal{C}, \sigma}(\text{AccPaths}_{\mathcal{C}, \sigma}^{\mathcal{A}', q_{\text{init}}}) = 1$ iff \mathcal{A} accepts all time-divergent timed words w.r.t. $(q_{\text{start}}, \mathbf{0})$. \square

REMARK 3. *In the statement of Theorem 5.2, one can also see that the problem to decide whether the minimal probability is equal to 0 is undecidable. This is because in the proof for the theorem, $\inf_{\sigma_w} \mathbf{p}_{q_{\text{init}}}^{\sigma_w}$ is either 0 or 1, depending on whether \mathcal{A} accepts all time-divergent timed words or not. Hence, the PTA-TRA problem is undecidable even in the qualitative sense. On the other hand, we did not consider the case of maximum acceptance probabilities. Whether the problem to maximum acceptance probabilities is decidable or not 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 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 DTAs 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. Because of this difference, the techniques for CTMPs cannot be applied to PTAs.

For PTAs, the only relevant result is by Sprosten [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 extends his with 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.

It is also worth noting that 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 requires 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.

REFERENCES

- [1] Rajeev Alur and David L. Dill. 1994. A Theory of Timed Automata. *Theor. Comput. Sci.* 126, 2 (1994), 183–235.
- [2] Rajeev Alur, Tomás 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 Alexandru 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 Berendsen, 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. 5532. 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] Tomáš Brázdil, Jan Krcál, Jan Kretínský, Antonín Kucera, and Vojtech Rehák. 2011. Measuring performance of continuous-time stochastic processes using timed automata. In *HSCC*, Marco Caccamo, Emilio Frazzoli, and Radu Grosu (Eds.). ACM, 33–42.
- [11] Taolue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandru 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 C-SL^{*}{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 Ejersbo 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, Gethin Norman, 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)*, Joël 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), 33–78.
- [22] Marta Z. Kwiatkowska, Gethin Norman, Roberto Segala, and Jeremy Sproston. 2002. Automatic verification of real-time systems with discrete probability distributions. *Theor. Comput. Sci.* 282, 1 (2002), 101–150.
- [23] Marta Z. Kwiatkowska, Gethin Norman, Jeremy Sproston, and Fuzhi Wang. 2007. Symbolic model checking for probabilistic timed automata. *Inf. Comput.* 205, 7 (2007), 1027–1077.
- [24] François Laroussinie and Jeremy Sproston. 2007. State explosion in almost-sure probabilistic reachability. *Inf. Process. Lett.* 102, 6 (2007), 236–241.
- [25] François Laroussinie and Jeremy Sproston. 2007. State explosion in almost-sure probabilistic reachability. *Inf. Process. Lett.* 102, 6 (2007), 236–241.
- [26] Gethin Norman, David Parker, and Jeremy Sproston. 2013. Model checking for probabilistic timed automata. *Formal Methods in System Design* 43, 2 (2013), 164–190.
- [27] Joël Ouaknine and James Worrell. 2005. On the Decidability of Metric Temporal Logic. In *LICS*. IEEE Computer Society, 188–197.
- [28] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.
- [29] Sriram Sankaranarayanan and Enrico Vicario (Eds.). 2015. *FORMATS*. LNCS, Vol. 9268. Springer.
- [30] Jeremy Sproston. 2011. Discrete-Time Verification and Control for Probabilistic Rectangular Hybrid Automata. In *QEST*. IEEE Computer Society, 79–88.
- [31] Frits W. Vaandrager. 1997. A Theory of Testing for Timed Automata (Abstract). In *TAPSOFT (LNCS)*, Michel Bidoit and Max Dauchet (Eds.), Vol. 1214. Springer, 39.
- [32] Moshe Y. Vardi. 1999. Probabilistic Linear-Time Model Checking: An Overview of the Automata-Theoretic Approach. In *AMAST (LNCS)*, Joost-Pieter Katoen (Ed.), Vol. 1601. Springer, 265–276.

A APPENDIX