Verifying Probabilistic Timed Automata Against Timed-Automata Specifications

Anonymous Author(s)

ABSTRACT

Probabilistic timed automata (PTAs) are timed automata 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 model-checking problem of linear *dense-time* temporal properties over PTAs. In particular, we consider linear dense-time properties that can be encoded by timed automata with both finite and infinite acceptance criteria. We first show that the problem of model-checking PTAs against deterministic-timed-automata specifications with infinite acceptance criterion can be solved through a product construction and is EXPTIME-complete. Then we show that when relaxed to general (nondeterministic) timed automata, the model-checking problem becomes undecidable. Finally, we investigate the situation where the acceptance criterion is restricted to be finite. We show that for deterministic timed automata, the model-checking problem can be solved using known zone-based algorithms for reachability probabilities on PTAs. Furthermore, for nondeterministic timed automata, we show an approximation algorithm for solving the problem whose correctness is based on a translation to infinite-state Markov decision processes.

ACM Reference Format:
Anonymous Author(s). 1997. Verifying Probabilistic Timed Automata Against Timed-Automata Specifications. In *Proceedings of 21st ACM International Conference on Hybrid Systems: Computation and Control (HSCC 2018).* ACM, New York, NY, USA, 11 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 [3], hence addressing fundamental issues such as safety and performance over these systems are important. *Probabilistic timed automata* (PTAs) [5, 20, 24] 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) [25] (for untimed probabilistic systems) with timing constraints.

Formal verification of PTAs has received much attention in recent years [24]. 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*) [5, 13, 20]. Advanced techniques for branching-time model checking of PTAs such as inverse method and symbolic method have been further explored in [2, 14, 17, 21]. Extension with *cost* or *reward*, resulting in *priced* PTAs, has also been well investigated. Jurdzinski *et al.* [15] and Kwiatkowska *et al.* [19] 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.* [6]. Most verification algorithms for PTAs have been implemented in the model checker PRISM [18]. Computational complexity of several verification problems for PTAs is studied in [15, 16, 22].

For linear-time model-checking, much less is known. As far as we know, the only relevant result is by ? [**?**] who proved that the problem of model-checking PTAs against linear *discrete-time* properties encoded by deterministic omega-regular automata can be solved by a product construction. In their paper, ? [**?**] first devised a production construction that produces a PTA out of the input PTA and the omega-regular automaton; then they 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. We focus on linear dense-time properties that can be encoded by timed automata [1]. Timed automata are normal 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 timed automata 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 timed automaton. The intuition is that a timed automaton 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.

Our Contributions. We distinguish between the subclass of *deterministic* timed automata (DTAs) and general *nondeterministic* timed automata. DTAs are the deterministic version of timed automata. 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. [11]). For infinite acceptance criterion, we show that the problem of model-checking PTAs against DTA specifications can be solved through a nontrivial product construction which tackles the integrated feature of timing constraints and

HSCC 2018, April 2018, Porto, Portugal
2016. ACM ISBN 123-4567-24-567/08/06...$15.00
https://doi.org/10.475/123_4

randomness. From the product construction, we further show that the problem is EXPTIME-complete. We also prove that the problem becomes undecidable when one considers general nondeterministic timed automata. For finite acceptance criterion, we show that the problem with DTA specifications can be solved by using efficient zone-based algorithms [5, 13, 20] through the same product construction. Moreover, we devised an approximation algorithm for the problem with general timed automata through translation to infinite-state MDPs.

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 $w = b_0 b_1 \dots$, we denote by $\mathit{inf}(w)$ 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 $\mathit{last}(w)$.

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 $\nu : \mathcal{X} \to [0, \infty)$. The set of clock valuations is denoted by $\mathit{Val}(\mathcal{X})$. 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 $x \in \mathcal{X}$. Moreover, we denote by $\mathbf{0}$ the clock valuation such that $\mathbf{0}(x) = 0$ for all $x \in \mathcal{X}$.

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 $c, d \in \mathbb{N}_0$. 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 $[\![\phi]\!]$ the set of all clock valuations that satisfy ϕ.

Clock Equivalence. Consider a nonnegative integer N such that values held by clocks are treated equivalent if they both exceed N. With such a threshold, the standard notion of clock equivalence [1] is an equivalence relation $\sim_N$ over $\mathit{Val}(\mathcal{X})$ as follows: for any two clock valuations ν, ν', $\nu \sim_N \nu'$ iff the following conditions hold:

- for all $x \in \mathcal{X}$, $\nu(x) > N$ iff $\nu'(x) > N$;
- for all $x \in \mathcal{X}$, if $\nu(x) \leq N$ then (i) $\lfloor \nu(x) \rfloor = \lfloor \nu'(x) \rfloor$ and (ii) $\mathit{frac}(\nu(x)) > 0$ iff $\mathit{frac}(\nu'(x)) > 0$;
- for all $x, y \in \mathcal{X}$, if $\nu(x), \nu(y) \leq N$ then it holds that $\mathit{frac}(\nu(x)) \bowtie \mathit{frac}(\nu(y))$ iff $\mathit{frac}(\nu'(x)) \bowtie \mathit{frac}(\nu'(y))$ for $\bowtie \in \{<, =, >\}$.

Equivalence classes of $\sim_N$ are called *regions*. The equivalence class that contains a given clock valuation ν is denoted by $[\nu]_{\sim_N}$. We simply write $[\nu]_\sim$ if N is clear from the context.

2.1 Probabilistic Timed Automata

A *discrete probability distribution* over a countable non-empty set U is a function $q : U \to [0, 1]$ such that $\sum_{z \in U} q(z) = 1$. The *support* of q is defined as $\mathit{supp}(q) := \{z \in U \mid q(z) > 0\}$. We denote the set of discrete probability distributions over U by $\mathcal{D}(U)$. For $u \in U$, we let μ_u be the *Dirac distribution at u* which assigns probability 1 to u.

Definition 2.1 (Probabilistic Timed Automata [24]). A *probabilistic timed automaton* (PTA) $\mathcal{C}$ is a tuple

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

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*;
- $\mathit{inv} : L \to CC(\mathcal{X})$ is an *invariant condition*;
- $\mathit{enab} : L \times \mathit{Act} \to CC(\mathcal{X})$ is an *enabling condition*;
- $\mathit{prob} : L \times \mathit{Act} \to \mathcal{D}(2^{\mathcal{X}} \times L)$ is a *probabilistic transition function*;
- AP is a finite set of *atomic propositions*;
- $\mathcal{L} : L \to 2^{AP}$ 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 $L \times \mathit{Val}(\mathcal{X})$ such that $\nu \models \mathit{inv}(\ell)$. The set of all states is denoted by $S_{\mathcal{C}}$. The *transition relation* $\to$ consists of all triples $((\ell, \nu), a, (\ell', \nu'))$ satisfying the following conditions:

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

By convention, we write $s \xrightarrow{a} s'$ instead of $(s, a, s') \in \to$. 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 $\mathbf{P} : S \times \mathit{Act} \times S \to [0, 1]$ such that

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

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 $a \in \mathit{Act}$ such that $\nu \models \mathit{enab}(\ell, a)$ and every $(X, \ell') \in \mathit{supp}(\mathit{prob}(\ell, a))$, one has that $\nu[X := 0] \models \mathit{inv}(\ell')$. 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 [21].

Paths. A *finite path* ρ (under $\mathcal{C}$) is a finite sequence

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

in $S \times ((Act \cup [0,\infty)) \times S)^*$ such that (i) $s_0 = (\ell^*, \mathbf{0})$, (ii) $a_{2k} \in [0,\infty)$ (resp. $a_{2k+1} \in Act$) 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 $(S \times (Act \cup [0,\infty)))^\omega$ 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 $Paths^*_{\mathcal{C}}$ (resp. $Paths^\omega_{\mathcal{C}}$).

Schedulers. A *scheduler* is a function σ from the set of finite paths into $Act \cup [0,\infty)$ such that for all finite paths $\rho = s_0 a_0 \dots s_n$, (i) $\sigma(\rho) \in Act$ (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 $Paths^*_{\mathcal{C},\sigma}$ (resp. $Paths^\omega_{\mathcal{C},\sigma}$). We note that the set $Paths^*_{\mathcal{C},\sigma}$ 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) $\Omega^{\mathcal{C},\sigma} := Paths^\omega_{\mathcal{C},\sigma}$, (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 $Cyl(\rho)$ of all infinite paths in $Paths^\omega_{\mathcal{C},\sigma}$ with 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 $Paths^*_{\mathcal{C},\sigma}$, $\mathbb{P}^{\mathcal{C},\sigma}(Cyl(\rho)) = \prod_{k=0}^{n-1} \mathbf{P}(s_k, \sigma(s_0 a_0 \dots a_{k-1} s_k), s_{k+1})$. For details see [**?**].

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).

Reachability. An infinite path $\pi = (\ell_0, \nu_0) a_0 (\ell_1, \nu_1) a_1 \dots$ is said to *visit* a subset $U \subseteq L$ of locations *eventually* if there exists $n \in \mathbb{N}_0$ such that $\ell_n \in U$. The set of infinite paths in $Paths^\omega_{\mathcal{C},\sigma}$ that visit U eventually is denoted by $Reach^U_{\mathcal{C},\sigma}$. From the fact that the set $Paths^*_{\mathcal{C},\sigma}$ is countably-infinite, $Reach^U_{\mathcal{C},\sigma}$ is measurable since it is a countable union of cylinder sets.

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

Example 2.2. In the PTA depicted in Figure 1, *WAIT*, $WORK_s$ and $DONE_s$ ($s \in \{\alpha, \beta\}$) 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, $inv(DONE_\alpha) = (x = 2)$ and

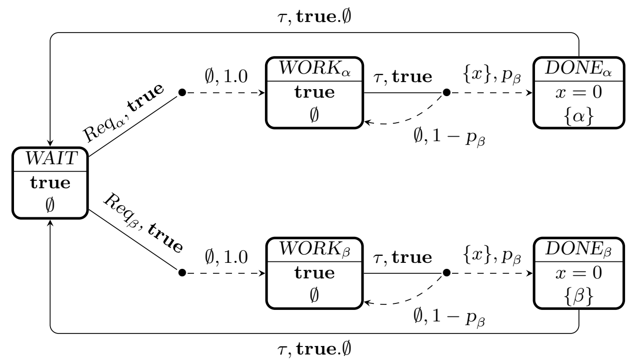

Figure 1: A Task-Handling Example

$\mathcal{L}(DONE_\alpha) = \{\alpha\}$. The four dot points together with corresponding arrows refer to four actions and their enabling conditions and probability transition functions. For example, the upper dot at the right of $WORK_\alpha$ refers to an action whose name is τ, the enabling condition for τ (from $WORK_\alpha$) is **true** (cf. the solid line emitting from $WORK_\alpha$), and the probability distribution for this action is to reset x and go to $DONE_\alpha$ with probability p_α and to reset x and go back to $DONE_\alpha$ with probability $1 - p_\alpha$. The PTA models a machine whiche deals with two different kinds of jobs.

2.2 Timed Automata

Definition 2.3 (Timed Automata [10–12]). A *timed automaton* (TA) $\mathcal{A}$ is a tuple

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

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 $[\![\phi_1]\!] \cap [\![\phi_2]\!] \neq \emptyset$ then $(\phi_1, X_1, q'_1) = (\phi_2, X_2, q'_2)$;
2. (*totality*) for all $(q, b) \in Q \times \Sigma$ and $\nu \in Val(\mathcal{X})$, 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 $\nu \in Val(\mathcal{Y})$. The *one-step transition relation*

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

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 $((q,\nu), a, (q',\nu')) \in \Rightarrow$.

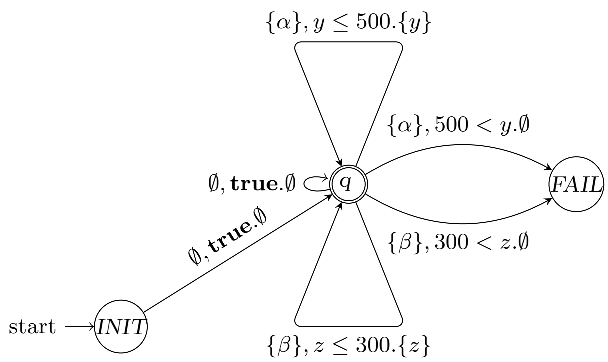

Figure 2: A DTA Specification

Infinite Time Words and Runs. An *infinite time word* is an infinite sequence $\{a_n\}_{n\in\mathbb{N}_0}$ such that $a_{2n} \in [0,\infty)$ and $a_{2n+1} \in \Sigma$ for all n. The *run* of $\mathcal{A}$ on an infinite timed word $w = \{a_n\}_{n\in\mathbb{N}_0}$ with *initial configuration* (q,ν), denoted by $\mathcal{A}_{q,\nu}(w)$, is the unique infinite sequence $\{(q_n,\nu_n,a_n)\}_{n\in\mathbb{N}_0}$ which satisfies 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* $\mathit{traj}(\mathcal{A}_{q,\nu}(w))$ of the run $\mathcal{A}_{q,\nu}(w)$ is defined as an infinite word over Q such that $\mathit{traj}(\mathcal{A}_{q,\nu}(w)) := q_0q_1\dots$.

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

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

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 $i < n$. A set $Q' \subseteq Q$ is *Rabin-accepting* by $\mathcal{F}$, written as the single predicate $\mathbf{ACC}(Q',\mathcal{F})$, if there exists $1 \le i \le n$ such that $Q' \cap H_i = \emptyset$ and $Q' \cap K_i \ne \emptyset$. An infinite timed word w is *Rabin-accepted* by $\mathcal{A}$ with *initial configuration* (q,ν) iff $\mathit{inf}(\mathit{traj}(\mathcal{A}_{(q,\nu)}(w)))$ is Rabin accepting by $\mathcal{F}$.

Example 2.5. Consider the DTA depicted in Figure 2 which works as a specification for the PTA in Example 2.2. *INIT*, q and *FAIL* are modes with $\mathcal{F} = \{\{FAIL\},\{q\}\}$, y,z are clocks and arrows between modes are rules. For example, there are five rules emitting from q , one is $(q\ ,\{\beta\},300 < z,\emptyset,FAIL)$ and another is $(q\ ,\emptyset,\mathbf{true},\emptyset,q\ )$. *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.

Definition 2.6 (Finite Acceptance Condition). A TA with *finite acceptance condition* (TFA) is a tuple

$$\mathcal{A} = (Q,\Sigma,\mathcal{Y},\Delta,F) \tag{4}$$

where $(Q,\Sigma,\mathcal{Y},\Delta)$ is a TA and F is a subset of Q representing the set of *final* modes. An infinite timed word w is *finitely-accepted* by $\mathcal{A}$ with *initial configuration* (q,ν) iff some final mode in F appear in the infinite word $\mathit{inf}(\mathit{traj}(\mathcal{A}_{(q,\nu)}(w)))$.

Note that finite acceptance condition is a special case of Rabin acceptance condition. Given a TFA in the form (4), one can transform it into an equivalent TRA by first removing all rules emitting from final modes, then making all final modes "absorbing" through adding self-loop rules $(q,b,\mathbf{true},\emptyset,q)$ for all final modes q and symbols b, and finally setting the Rabin condition $\mathcal{F}$ to be the singleton set $\{(\emptyset,F)\}$.

3 THE MODEL-CHECKING PROBLEM

In this part, we define the problem of model-checking PTAs against TA-specifications. The problem takes a PTA and a TRA/TFA as input, and computes the probability that infinite paths under the PTA are accepted by the TRA/TFA. Informally, the TRA/TFA 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/TFA. In practice, the TRA/TFA 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 (or TFA) $\mathcal{A}$ taking the form (3) (or (4))) . 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 $\mathit{Paths}^{\omega}_{\mathcal{C}}$ 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 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 $a_{2n+1} \in \mathit{Act}$ 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} $(n \ge 1)$ 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/TFA.*

Below we define the acceptance condition on paths as follows. For an infinite path $\pi = (\ell_0,\nu_0)a_0(\ell_1,\nu_1)a_1\dots$ under $\mathcal{C}$, we denote by $\mathit{init}(\pi)$ the initial location ℓ_0.

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 $((q', \nu'), \mathbf{0})$, Notice that $\mathbf{ACC}$ is already used but it is easy to distinguish the two different usage from the context.

In the definitions above, the initial location omitted in Definition 3.1 is preprocessed by specifying explicitly that the initial configuration is $(\kappa((q, \nu), \mathcal{L}(init(\pi))), \mathbf{0})$.

Now we define the notion of acceptance probabilities over infinite paths under $\mathcal{C}$.

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

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

where $AccPaths^{\mathcal{A},q}_{\mathcal{C},\sigma}$ is the set of paths in $\mathcal{C}$ that falls into the Rabin-accepted language of $\mathcal{A}$

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

Again, from the fact that the set $Paths^{*}_{\mathcal{C},\sigma}$ is countably-infinite, $AccPaths^{\mathcal{A},q}_{\mathcal{C},\sigma}$ is measurable since it can be represent int the form of a countable intersect and countable union of some cylinder sets.

Now the PTA-TRA problem is as follows.

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

We refer to the problem as PTA-DTA if $\mathcal{A}$ is deterministic.

4 THE PRODUCT CONSTRUCTION

In this section, we introduce the core part of our algorithms to solve the PTA-DTA problem and deterministic TRA is refered as DTA. The core part is a product construction which given a PTA $\mathcal{C}$ and a DTA $\mathcal{A}$, output a PTA which preserves the probability of the set of infinite paths of $\mathcal{C}$ accepted by $\mathcal{A}$. Below we fix a well-formed PTA $\mathcal{C}$ in the form (1) and a DTA $\mathcal{A}$ in the form (**??**) with the difference that the set of clocks for $\mathcal{C}$ (resp. for $\mathcal{A}$) is denoted by $\mathcal{X}_1$ (resp. $\mathcal{X}_2$). W.l.o.g., we assume that $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$ and $\Sigma = 2^{AP}$. We let $\mathcal{G}$ be the set of regions w.r.t $\sim_N$, where N is the maximal integer appearing in the clock constraints of $\mathcal{A}$.

The Main Idea. The intuition of the product construction 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 Δ into the enabling condition *enab* in $\mathcal{C}$. However, this simple solution does not work since a single rule in Δ fixes the labelling of a location in $\mathcal{C}$, while the probabilistic distribution given by *prob* can jump to locations with different labels. We solve this difficulty by integrating into the enabling condition *enab* enough information on clock valuations under $\mathcal{A}$ so that the rule used for the transition (in $\mathcal{A}$) is clear. In detail, we introduce two versions of the product construction, each having a computational advantage against the other.

Product Construction (First Version). The *product PTA* $\mathcal{C} \otimes \mathcal{A}_q$ between $\mathcal{C}$ and $\mathcal{A}$ with initial mode q is defined as the PTA

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

- $L_{\otimes} := L \times Q$;
- $\ell^{*}_{\otimes} := (\ell^{*}, q^{\star})$ where $q^{\star}$ is the unique mode such that $\kappa((q, \mathbf{0}), \mathcal{L}(\ell^{*})) = (q^{\star}, \mathbf{0})$;
- $\mathcal{X}_{\otimes} := \mathcal{X}_1 \cup \mathcal{X}_2$;
- $Act_{\otimes} := Act \times \mathcal{G}$;
- $inv_{\otimes}(\ell, q) := inv(\ell)$ for all $(\ell, q) \in L_{\otimes}$;
- $enab_{\otimes}((\ell, q), (a, R)) := enab(\ell, a) \wedge \phi_R$ for all $(\ell, q) \in L_{\otimes}$, where ϕ_R is any clock constraint such that $[\![\phi_R]\!] = R$;
- $\mathcal{L}_{\otimes}(\ell, q) := \{q\}$ for all $(\ell, q) \in L_{\otimes}$
- $prob_{\otimes}$ is given by

$$prob_{\otimes}((\ell, q), (a, R))(Y, (\ell', q')) := \begin{cases} prob(\ell, a)(Y \cap \mathcal{X}_1, \ell') & \text{if } (q, \mathcal{L}(\ell'), \phi^{q,\mathcal{L}(\ell')}_{R}, Y \cap \mathcal{X}_2, q') \in \Delta \\ 0 & \text{otherwise} \end{cases}.$$

where $(q, \mathcal{L}(\ell'), \phi^{q,\mathcal{L}(\ell')}_{R}, Y \cap \mathcal{X}_2, q')$ is the unique rule such that for all $\nu \in R$, $\nu \in [\![\phi^{q,\mathcal{L}(\ell')}_{R}]\!]$. The uniqueness follows from determinism and totality of DTAs.

Apart from standard constructions (e.g., the Cartesian product between L and Q), the product construction also has Cartesian product between Act and $\mathcal{G}$. Then for each extended action (a, R), the enabling condition for this action is just the conjunction between $enab(\ell, a)$ and R. This is to ensure that when the action (a, R) is taken, the clock valuation under $\mathcal{A}$ lies in R. Finally in the definition for $prob_{\otimes}$, upon the action (a, R) and the target location ℓ', the DTA $\mathcal{A}$ chooses the unique rule $(q, \mathcal{L}(\ell'), \phi^{q,\mathcal{L}(\ell')}_{R}, Y \cap \mathcal{X}_2, q')$ and then jump to q' with reset set $Y \cap \mathcal{X}_2$. By integrating regions into the enabling condition, the DTA $\mathcal{A}$ can know the status of the clock valuation under $\mathcal{A}$ through its region, hence can decide which rule to use for the transition. This version of product construction works well if the number of regions is not large. We note that the number of regions only depends on N, not on the size of $\mathcal{A}$. In the following, we introduce another version which depends directly on the size of $\mathcal{A}$. The second version has an advantage when the number of regions is large.

Product Construction (Second Version). For each $q \in Q$, we let

$$\mathcal{T}_q := \{h : \Sigma \to CC(\mathcal{X}_2) \mid \forall b \in \Sigma.\, (q, b, h(b), X, q') \in \Delta \text{ for some } X, q'\}\ .$$

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 almost the same as

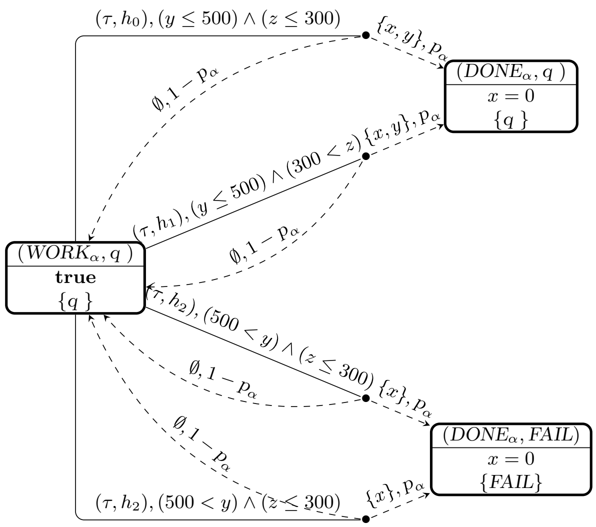

Figure 3: A Part of Product PTA

in the first version of the product construction, with the following differences:

- $Act_\otimes := Act \times \bigcup_q \mathcal{T}_q$;
- $enab_\otimes((\ell, q), (a, h)) := enab(\ell, a) \wedge \bigwedge_{b \in \Sigma} h(b)$ for all $(\ell, q) \in L_\otimes$ and $h \in \mathcal{T}_q$, and $enab_\otimes((\ell, q), (a, h)) :=$ **false** otherwise;
- $prob_\otimes$ is given by

$$prob_\otimes((\ell, q), (a, h))(Y, (\ell', q')) :=$$
$$\begin{cases} prob(\ell, a)(Y \cap \mathcal{X}_1, \ell') & \text{if } (q, \mathcal{L}(\ell'), h(\mathcal{L}(\ell')), Y \cap \mathcal{X}_2, q') \in \Delta \\ 0 & \text{otherwise} \end{cases} .$$

The intuition for the second version is that it is also possible to specify the information needed to identify the rule to be chosen by the DTA through a local conjunction of the rules emitting from a mode. For each mode, the local conjunction chooses one clock constraint from rules with the same symbol, and group them together through conjunction. From determinism and totality of DTAs, each conjunction constructed in this way determines which rule to use in the DTA for every symbol in a unique way. The advantage of the second version against the first one is that it is more suitable for DTAs with small size and large N (leading to a large number of reigons), as the size of the product PTA relies only the size of the DTA.

Example 4.1. Here we represent an running example to show how $\mathcal{T}_q$ works. Here for the accepting mod q in DTA, we have

$$\mathcal{T}_q = \{h_0 = \{\emptyset \mapsto \textbf{true}, \{\alpha\} \mapsto (x \leq 500), \{\beta\} \mapsto (y \leq 300), \{\alpha, \beta\} \mapsto \textbf{true}\},$$
$$h_1 = \{\emptyset \mapsto \textbf{true}, \{\alpha\} \mapsto (x \leq 500), \{\beta\} \mapsto (300 < y), \{\alpha, \beta\} \mapsto \textbf{true}\},$$
$$h_2 = \{\emptyset \mapsto \textbf{true}, \{\alpha\} \mapsto (500 < x), \{\beta\} \mapsto (y \leq 300), \{\alpha, \beta\} \mapsto \textbf{true}\},$$
$$h_3 = \{\emptyset \mapsto \textbf{true}, \{\alpha\} \mapsto (500 < x), \{\beta\} \mapsto (300 < y), \{\alpha, \beta\} \mapsto \textbf{true}\}\}.$$

And a part of the product of Example 2.2 and Example 2.5 is depicted in Figure 2.

Remark 2. *It is easy to see that the PTA $\mathcal{C} \otimes \mathcal{A}_q$ (in both versions) is well-formed as $\mathcal{C}$ is well-formed and the DTA $\mathcal{A}$ does not introduce extra invariant conditions.*

In the following, we clarify the relationship between $\mathcal{C}, \mathcal{A}$ and $\mathcal{C} \otimes \mathcal{A}_q$. We first show the relationship between paths under $\mathcal{C}$ and paths 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 $\mathcal{T}$ From Paths under $\mathcal{C}$ into Paths under $\mathcal{C} \otimes \mathcal{A}_q$. Since the two versions of product construction shares similarities, we illustrate the transformation in a unified fashion. The transformation is defined as the function $\mathcal{T} : Paths^*_{\mathcal{C}} \cup Paths^\omega_{\mathcal{C}} \to Paths^*_{\mathcal{C} \otimes \mathcal{A}_q} \cup Paths^\omega_{\mathcal{C} \otimes \mathcal{A}_q}$ which transform a finite or infinite path under $\mathcal{C}$ into one under $\mathcal{C} \otimes \mathcal{A}_q$ as follows. 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

$$\mathcal{T}(\rho) := ((\ell_0, q_0), \nu_0 \cup \mu_0) a'_0 \dots a'_{n-1} ((\ell_n, q_n), \nu_n \cup \mu_n) \quad (5)$$

under $\mathcal{C} \otimes \mathcal{A}_q$ such that (†)

- $\kappa((q, \mathbf{0}), \mathcal{L}(\ell^*)) = (q_0, \mu_0)$ (note that $\mu_0 = \mathbf{0}$), and
- for all $0 \leq k < n$, if $a_k \in [0, \infty)$ then $a'_k = a_k$ and $(q_k, \mu_k) \xRightarrow{a_k} (q_{k+1}, \mu_{k+1})$, and
- 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 either (i) the first version of the product construction is taken and ξ_k is the region $[\mu_k]_\sim$ or (ii) the second version is taken and 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

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

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

The following lemma shows that $\mathcal{T}$ is a bijection and preserves zenoness.

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

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

We also show the relationship on schedulers before and after product construction.

Transformation θ From Schedulers under $\mathcal{C}$ into Schedulers under $\mathcal{C} \otimes \mathcal{A}_q$. 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 $\mathcal{T}(\rho)$ is given as in (5),

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

where $\lambda(\rho)$ is either $[\mu_n]_\sim$ if the first version of the product construction is taken, or the unique function such that for each symbol $b \in \Sigma$, $\lambda(\rho)(b)$ is the unique clock constraint appearing in a 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. □

By Lemma 4.2, the product construction and the determinism and totality of DTAs, one can prove straightforwardly the following lemma.

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

Now we show the relationship between infinite paths accepted by a DTA before product construction and infinite paths visiting certain target locations after product construction. Below we lift the function $\mathcal{T}$ to all subsets of paths in the standard fashion: for all subsets $A \subseteq Paths^*_{\mathcal{C}} \cup Paths^\omega_{\mathcal{C}}$, $\mathcal{T}(A) := \{\mathcal{T}(\omega) \mid \omega \in A\}$.

Definition 4.4 (Traces). Let $\mathcal{T}(\pi) = ((\ell_0, q_0), \nu_0 \cup \mu_0)a'_0((\ell_1, q_1), \nu_1 \cup \mu_1)a'_1 \dots$ the trace of $\mathcal{T}(\pi)$ is defined by $trace(\mathcal{T}(\pi)) := q_0 q_1 \dots$.

Verifying Limit Rabin Properties. Paths in $\mathcal{C} \otimes \mathcal{A}_q$ that $\mathcal{C}$ is accepted by $\mathcal{A}$ is

$$RabinPaths_{\mathcal{C}\otimes\mathcal{A}_q,\sigma} = \left\{\pi \in Paths^\omega_{\mathcal{C}\otimes\mathcal{A}_q,\sigma} \mid \mathbf{ACC}\left(inf(trace(\pi)), \mathcal{F}\right)\right\}$$

and $RabinPaths_{\mathcal{C}\otimes\mathcal{A}_q,\sigma}$ is an limit LT Property [3, Notation10.121].

Proposition 4.5. *For any scheduler σ and any initial mode q on DTA $\mathcal{A}$,*

$$\mathcal{T}\left(AccPaths^{\mathcal{A},q}_{\mathcal{C},\sigma}\right) = RabinPaths_{\mathcal{C}\otimes\mathcal{A}_q,\theta(\sigma)}.$$

Proof. By definition we have

$$AccPaths^{\mathcal{A},q}_{\mathcal{C},\sigma} = \left\{\pi \in Paths^\omega_{\mathcal{C},\sigma} \mid \mathbf{ACC}\left(inf(traj(\mathcal{A}_{(q^*,\mathbf{0})}\left(\mathcal{L}(\pi)\right))), \mathcal{F}\right)\right\},$$

where $q^* = \kappa\left((q, \mathbf{0}), \mathcal{L}(init\,(\pi)\right)$. Let $\pi = (\ell_0, \nu_0)a_0(\ell_1, \nu_1)a_1 \dots$ be any infinite path. And by definition of $\mathcal{T}$ we have

$$\mathcal{T}(\pi) = ((\ell_0, q_0), \nu_0 \cup \mu_0)a'_0((\ell_1, q_1), \nu_1 \cup \mu_1)a'_1 \dots$$
$$\mathcal{A}_{(q^*,\mathbf{0})}\left(\mathcal{L}(\pi)\right) = \{(q_n, \mu_n, \mathcal{L}(\pi)_n)\}_{n\in\mathbb{N}_0}.$$

Then it's obvious that

$$trace(\mathcal{T}(\pi)) = q_0 q_1 \dots = traj(\mathcal{A}_{(q^*,\mathbf{0})}\left(\mathcal{L}(\pi)\right)).$$

Then we conclude that $inf(trace(\mathcal{T}(\pi)))$ is Rabin accepting by $\mathcal{F}$ iff $inf(traj(\mathcal{A}_{(q^*,\mathbf{0})}\left(\mathcal{L}(\pi)\right)))$ is Rabin accepting by $\mathcal{F}$. □

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.

Theorem 4.6. *For any scheduler σ and initial mode q,*

$$\mathfrak{p}^\sigma_q = \mathbb{P}^{\mathcal{C},\sigma}\left(AccPaths^{\mathcal{A},q}_{\mathcal{C},\sigma}\right) = \mathbb{P}^{\mathcal{C}\otimes\mathcal{A}_q,\theta(\sigma)}\left(RabinPaths_{\mathcal{C}\otimes\mathcal{A}_q,\theta(\sigma)}\right).$$

Moreover, $\mathbb{P}^{\mathcal{C},\sigma}\left(\{\pi \mid \pi \textit{ is zeno}\}\right) = \mathbb{P}^{\mathcal{C}\otimes\mathcal{A}_q,\theta(\sigma)}\left(\{\pi' \mid \pi' \textit{ is zeno}\}\right)$.

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 [7, Theorem 3.3], it suffices to consider cylinder sets as they form a pi-system (cf. [7, 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 4.5 and the second claim follows from Lemma 4.2. □

Note that a side result from Theorem 4.6 says that θ preserves time-divergence for schedulers before and after product construction. From Theorem 4.6 and Lemma 4.3, one immediately obtains the following result which transforms the PTA-DTA problem into computing reachability probabilities under the product PTA.

Corollary 4.7. *([27]) For any initial mode q,*

$$opt_\sigma \mathfrak{p}^\sigma_q = opt_{\sigma'} \mathbb{P}^{\mathcal{C}\otimes\mathcal{A}_q,\sigma'}\left(RabinPaths_{\mathcal{C}\otimes\mathcal{A}_q,\sigma'}\right)$$

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

The way [27] discards time-convergent path is making a copy of every location in PTA model and enforcing a transition from the original one to the copy happen when 1 time unit is passed. After transiting to the copy, A transition back to the original one will immediately happend with no delay. And we put a label *tick* in copy. We only deal with paths that satisfy $\Box\Diamond tick$ (i.e. *tick* is satisfied infinitely many times).

Then an MDP Reg$[\mathcal{C}\otimes\mathcal{A}_q]$ is obtained from the enlarged PTA of $\mathcal{C}\otimes\mathcal{A}_q$ through an region construction. Then we verify the limit rabin property on Reg$[\mathcal{C}\otimes\mathcal{A}_q]$ by using a standard MEC algorithm. First, We find all MECs satisfy the corresponding property of an Rabin acceptance condition. In order to guarantee time-divergence, we only pick up MECs with at least one location that has an *tick* label and let F_* be the union of those MECs. Then, we turn to resolve the probability reachability to F_*.

Lemma 4.8. *Time Complexity of Verifying Limit Rabin Properties([3, Theorem 10.127]) Let M be a finite MDP and P be a limit LT propery specified by a Rabin condition:*

$$\bigvee_{1\le i\le n} (\Diamond\Box\neg H_i \wedge \Box\Diamond K_i)$$

Then: the values $\mathrm{opt}_\sigma \mathbb{P}^{M,\sigma}\left(s \models P\right)$ *can be computed in time* $\mathcal{O}\left(poly\left(size\left(M\right)\right) \cdot k\right)$ *where opt refers to either* inf *(infimum) or* sup *(supremum).*

Noting that for $\mathcal{C}\otimes\mathcal{A}_q$, although the upper bound of $|Act_{\otimes}|$ is $|Act|\cdot|Q|\cdot|\Delta|^{|\Sigma|}$, $|L_{\otimes}|$ is polynomial to $|L|\cdot|Q|$, and $|\mathcal{X}_{\otimes}|=|\mathcal{X}_1|+|\mathcal{X}_2|$. The size of $\mathrm{Reg}[\mathcal{C}\otimes\mathcal{A}_q]$ is exponential to $|L|\cdot|Q|$ while the number of transitions is exponential, then $opt_{\sigma}\mathfrak{p}_q^{\sigma}$ can be calculated in exponential time follows from Lemma 4.8.

In [23], the authors proved that the reachablity problem for arbitary PTAs is *EXPTIME*-complete. Reduction from the PTA reachibility problem to the PTA-DTA model-checking problem can be easily constructed as follows.

For an arbitary PTA $\mathcal{C}=(L,l^*,\mathcal{X},Act,inv,enab,prob,\mathcal{L})$ and a set of final locations $L_F\subseteq L$. Let $\mathcal{C}'=(L,l^*,\mathcal{X},Act,inv,enab,prob,\mathcal{L}')$ where

$$\mathcal{L}'(l):L\to AP\cup\{\mathrm{acc}\}=\begin{cases}\mathcal{L}(l) & \text{if } l\notin L_F\\ \mathcal{L}(l)\cup\{\mathrm{acc}\} & l\in L_F\end{cases}$$

and DTA $\mathcal{A}'=(Q=\{q_0,q_1\},\Sigma=2^{AP\cup\{\mathrm{acc}\}},\mathcal{X}=\emptyset,\Delta,\mathcal{F}=\{(\emptyset,\{q_1\}\})$ where

- $\forall\sigma\in\Sigma,\mathrm{acc}\in\sigma\to(q_0,\sigma,true,\emptyset,q_1)\in Delta$,
- $\forall\sigma\in\Sigma,i=0,1,(q_i,\sigma,true,\emptyset,q_i)\in\Delta$.

It's clear that L_F is reachable in $\mathcal{C}$ iff. for $\mathcal{C}'$ and $\mathcal{A}'$, $sup_{\sigma}\mathfrak{p}_{q_0}^{\sigma}=1$.

Proposition 4.9. *The PTA-TRA problem is EXPTIME-complete.*

5 UNDECIDABILITY OF PTA-NTA PROBLEM

We show that the qualitative problem for minimum probabilities is already undecidable. We prove this by a reduction from the universality problem of timed automata, which is illustrated as follows.

Lemma 5.1. *A timed language is accepted by some timed Büchi automaton iff it is accepted by some timed Rabin automaton.*

Proof. The construction is similar to [28, Theorem 3.20.] □

Lemma 5.2. *([28, Theorem 5.2.]) Given a timed automaton over an alphabet Σ, the problem of deciding whether it accepts all time-divergent timed words over Σ is undecidable.*

The proof of lemma 5.2 is based on a construction of timed Büchi automata and it also holds for timed rabin automata since lemma 5.1.

Proposition 5.3. *Given a nonedeterministic timed rabin automaton $\mathcal{A}$ over an alphabet Σ, the qualitative problem of the minimal probability that $\mathcal{C}$* observes *$\mathcal{A}$ under initial mode $q_{start}\in Q$ is undecidable.*

Proof. For any nonedeterministic TRA $\mathcal{A}=(Q,\Sigma,\mathcal{X},\Delta)$, let $\Sigma=\{b_1,b_2,\cdots,b_k\}$.

we construct an $\mathcal{A}'=(Q',\Sigma',\mathcal{X},\Delta')$ where

$Q'=Q\cup\{q_{init}\},\Sigma'=\Sigma\cup\{b_0\},\Delta'=\Delta\cup\{\langle q_{init},b_0,\mathbf{true},\mathcal{X},q_{start}\rangle\}$.

We can choose an appropriate AP such that $k+1\le\left|2^{AP}\right|$ and assign each b_i to a different subset of AP. So we simplify the label of locations in $\mathcal{C}$ by single letters in Σ'.

Let 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 all $b_i\in L$,
- $enab(b_i,b_j):=\mathbf{true}$, for all $b_i\in L$ and all $b_j\in Act$,
- $prob(b_i,b_j):=\mu_{(\emptyset,b_j)}$, for all $b_i\in L$ and all $b_j\in Act$,
- $\mathcal{L}(b_i):=b_i$, for all $b_i\in L$.

It is natural to see, for any time word $w=\alpha_0\alpha_1\alpha_2\cdots$ there is a scheduler $\sigma_w(\rho):=\alpha_{|\rho|}$ such that $\mathbb{P}^{\mathcal{C},\sigma_w}(\{w\})=1$. σ_w is time-divergent since $Paths^{\omega}_{\mathcal{C},\sigma}=\{w\}$ and w is time-divergent. It's natural that

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

Then we have $\inf_{\sigma}\mathbb{P}^{\mathcal{C},\sigma}\left(AccPaths_{\mathcal{C},\sigma}^{\mathcal{A}',q_{init}}\right)=1$ iff $\mathcal{A}$ accepts all timewords w.r.t. $(q_{start},\mathbf{0})$. □

6 INFINITE-STATE-MDP CONSTRUCTION

Now we present the finite acceptance of nodeterministic timed automata for PTAs.

Definition 6.1 (Finite Acceptance Criterion). Let $F\subseteq Q$ be a set of *final* modes. An infinite word w is *finitely accepted* by $\mathcal{A}$ w.r.t the *initial configuration* (q,ν) and F if $\mathcal{A}_{q,\nu}(w)=\{(q_n,\nu_n,a_n)\}_{n\in\mathbb{N}_0}$ satisfies that $q_n\in F$ for some $n\in\mathbb{N}_0$.

Definition 6.2 (Path Acceptance). An infinite path π under $\mathcal{C}$ is *finitely accepted* by $\mathcal{A}$ w.r.t initial configuration (q,ν), if the infinite word $\mathcal{L}(\pi)$ is finitely accepted by $\mathcal{A}$ w.r.t $(\kappa((q,\nu),\mathcal{L}(init(\pi))),\mathbf{0})$.

Below we fix a well-formed PTA $\mathcal{C}$ taking the form (1) and a NTA $\mathcal{A}$ taking the form (2) with the difference that the set of clocks for $\mathcal{C}$ (resp. for $\mathcal{A}$) is denoted by $\mathcal{X}$ (resp. $\mathcal{Y}$). W.l.o.g., we assume that $\mathcal{X}\cap\mathcal{Y}=\emptyset$ and $\Sigma=2^{AP}$.

Let PTA be $\mathcal{C}$ with the set $\mathcal{X}$ and the NTA be $\mathcal{A}$ with the set $\mathcal{Y}$.

The transformation to MDP is as follows.

Let $\mathcal{Y}$ be a fixed finite set of clocks. We use integer Subscript denote a set of new clocks. Formally $\mathcal{Y}_k=\{(t,y)\in\mathbb{N}\times\mathcal{Y}\mid t=k\}$ for $k>0$. For convenience, we use $\mathcal{Y}_0$ denote $\mathcal{X}$.

And $R^{\mathcal{Y}_k}$ is a region for $\mathcal{Y}_k$.

Definition 6.3 (Product Construction (Infinite-State-MDP)). The *product MDP* $\mathcal{C}*\mathcal{A}_q$ between $\mathcal{C}$ and $\mathcal{A}$ with initial mode q is defined as the PTA

The transformation to MDP is follows. A state in $\mathcal{C}*\mathcal{A}_q$ is of the form

$$\left(\ell,(q_1,\cdots,q_n),\mathcal{X}\cup\left(\bigcup_{k=1}^{n}\mathcal{Y}_k\right),R\right)\tag{6}$$

where n is an unbounded natural number, ℓ (w.r.t q_i) is a location in $\mathcal{C}$ (w.r.t a mod in $\mathcal{A}$) and R is a region with clock names being $\mathcal{X} \cup \left(\bigcup_{k=1}^{n} \mathcal{Y}_k\right)$. The intuition is that $(\ell, R \downarrow \mathcal{X})$ reflects the region for $\mathcal{C}$, $((q_1, R \downarrow \mathcal{Y}_1) \cdots, (q_n, R \downarrow \mathcal{Y}_n))$ reflects a power set for $\mathcal{A}$.

Definition 6.4 (Rename function). Let $\mathcal{X}$ and $\mathcal{Y}$ be two sets of clocks, ν is a clock valuation on $\mathcal{X}$ and $f : \mathcal{X} \leftrightarrow \mathcal{Y}$ is a rename function then $\nu[f] = \nu \circ f^{-1}$.

Lemma 6.5. *Let R is a region with clock names $\mathcal{X}$ and $X \subseteq \mathcal{X}$, $R \downarrow X$ is a region with clock names X.*

Definition 6.6 (Time successor). A state

$$s' = \left(\ell, (q_1, \cdots, q_n), \mathcal{X} \cup \left(\bigcup_{k=1}^{n} \mathcal{Y}_k\right), R'\right)$$

is a time successor of s in the form of (6) where either

- $s = s'$ if $\forall \nu \in R, t \in \mathbb{R}_{>0} : \nu + t \in R'$ or
- R' is another unique region if there exist a $\nu \in R$ s.t.

$$\exists t \in \mathbb{R}_{\geq 0} : (\nu + t \in R' \wedge \forall t' \in [0, t] :$$
$$\left(\left(\nu + t' \in R \cup R'\right) \wedge \nu + t' \models inv(l)\right))$$

Definition 6.7 (Transition relation). The *transition relation* $\rightarrow$ is the smallest relation such that the following two inference rules are satisfied :

$$\text{(Delay)}\ \frac{s' \text{ is the time successor of } s}{s \xrightarrow{\tau} \mu_{s'}}$$

$$\text{(Jump)}\ \frac{\nu \in R \quad (\ell, \nu \downarrow \mathcal{X}) \xrightarrow{a} \mu \quad \nu \downarrow \mathcal{X} \models enab\,(\ell, a)}{s \xrightarrow{a} \mu^*}$$

where, let

$$s' = \left(\ell', \left(q_{1_0} \cdots, q_{i_0} \cdots, q_{i_{k_i}} \cdots, q_{n_{k_n}}\right), \mathcal{X} \cup \left(\bigcup_{i=1}^{n'} \bigcup_{j=0}^{k_i} \mathcal{Y}_{i_j}\right), R'\right)$$

$$\mu^*\left(s'\right) = \begin{cases} \mu(X, \ell') & \dagger \\ 0 & \text{otherwise} \end{cases}$$

The none zero case hold if $(R' \downarrow \mathcal{X}) = [\nu \downarrow \mathcal{X}[X := 0]]_{\sim}$, there exists $\left(q_i, \mathcal{L}\left(\ell'\right), \phi, Y_{i_j}, q_{i_j}\right) \in \Delta$ such that $R' \downarrow \mathcal{Y}_{i_j} = \left[\left(\nu \downarrow \mathcal{Y}_i\right)\left[Y_{i_j} := 0\right][y \mapsto < i_j, y >]\right]_{\sim}$ and $R \downarrow \mathcal{Y}_i \subseteq [\![\phi]\!]$, k_i is the number of successors of q_i.

7 CONCLUSION AND FUTURE WORK

In this paper, we studied the problem of model-checking PTAs against timed-automata specifications. We considered both Rabin and finite acceptance conditions. For Rabin acceptance condition, we first solved the problem with DTA specifications and Rabin acceptance condition through a product construction and prove 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. For finite acceptance condition, we demonstrated that the problem with DTA specifications can be solved through efficient zone-based algorithms on verifying reachability probability of PTAs [20, 24], while the problem with general timed-automata specifications can be solved by an approximation algorithm based on value iteration.

An interesting future direction is zone-based algorithms for Rabin acceptance condition. Another theoretical direction is to investigate timed-automata specifications with cost or reward. A more practical direction is to apply our approaches to industrial-level examples.

8 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.* [11] 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 [**?**] against one-clock DTAs. Then various results on verifying CTMPs are obtained for specifications through DTAs and general timed automata (cf. [4, 8–12]). 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 ?[**?**] who developed an approach for verifying PTAs against deterministic discrete-time omega-regular automata through a similar product construction. Our results extend theirs in two ways. First, our product construction extends theirs with extra ability to tackle timing constraints from both the PTA and the DTA. The extension is nontrivial since it needs to resolve the integration between randomness and timing constraints, while ensuring the EXPTIME-completeness of the problem, matching the computational complexity in the discrete-time case [**?**]. Second, our results also cover an undecidability result and an approximation algorithm in the case of general nondeterministic timed automata, extending [**?**] with nondeterminism.

REFERENCES

[1] Rajeev Alur and David L. Dill. 1994. A Theory of Timed Automata. *Theor. Comput. Sci.* 126, 2 (1994), 183–235. https://doi.org/10.1016/0304-3975(94)90010-8

[2] 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. https://doi.org/10.1007/s10703-012-0169-x

[3] Christel Baier and Joost-Pieter Katoen. 2008. *Principles of Model Checking.* MIT Press.

[4] Benoît Barbot, Taolue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre. 2011. Efficient CTMC Model Checking of Linear Real-Time Objectives. In *Tools and Algorithms for the Construction and Analysis of Systems - 17th International Conference, TACAS 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings (Lecture Notes in Computer Science)*, Parosh Aziz Abdulla and K. Rustan M. Leino (Eds.), Vol. 6605. Springer, 128–142. https://doi.org/10.1007/978-3-642-19835-9_12

[5] Danièle Beauquier. 2003. On probabilistic timed automata. *Theor. Comput. Sci.* 292, 1 (2003), 65–84. https://doi.org/10.1016/S0304-3975(01)00215-8

[6] Jasper Berendsen, Taolue Chen, and David N. Jansen. 2009. Undecidability of Cost-Bounded Reachability in Priced Probabilistic Timed Automata. In *Theory and Applications of Models of Computation, 6th Annual Conference, TAMC 2009, Changsha, China, May 18-22, 2009. Proceedings (Lecture Notes in Computer Science)*, Jianer Chen and S. Barry Cooper (Eds.), Vol. 5532. Springer, 128–137. https://doi.org/10.1007/978-3-642-02017-9_16

[7] Patrick Billingsley. 2012. *Probability and Measure* (anniversary edition ed.). Wiley.

[8] Luca Bortolussi and Roberta Lanciani. 2015. Fluid Model Checking of Timed Properties, See [26], 172–188. https://doi.org/10.1007/978-3-319-22975-1_12

[9] Tomás Brázdil, Jan Krcál, Jan Kretínský, Antonín Kucera, and Vojtech Rehák. 2011. Measuring performance of continuous-time stochastic processes using timed automata. In *Proceedings of the 14th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2011, Chicago, IL, USA, April 12-14, 2011*, Marco Caccamo, Emilio Frazzoli, and Radu Grosu (Eds.). ACM, 33–42. https://doi.org/10.1145/1967701.1967709

[10] 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). https://doi.org/10.2168/LMCS-7(1:12)2011

[11] Susanna Donatelli, Serge Haddad, and Jeremy Sproston. 2009. Model Checking Timed and Stochastic Properties with CSLˆ{TA}. *IEEE Trans. Software Eng.* 35, 2 (2009), 224–240. https://doi.org/10.1109/TSE.2008.108

[12] Hongfei Fu. 2013. Approximating acceptance probabilities of CTMC-paths on multi-clock deterministic timed automata. In *Proceedings of the 16th international conference on Hybrid systems: computation and control, HSCC 2013, April 8-11, 2013, Philadelphia, PA, USA*, Calin Belta and Franjo Ivancic (Eds.). ACM, 323–332. https://doi.org/10.1145/2461328.2461376

[13] 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.

[14] Aleksandra Jovanovic, Marta Z. Kwiatkowska, and Gethin Norman. 2015. Symbolic Minimum Expected Time Controller Synthesis for Probabilistic Timed Automata, See [26], 140–155. https://doi.org/10.1007/978-3-319-22975-1_10

[15] Marcin Jurdzinski, Marta Z. Kwiatkowska, Gethin Norman, and Ashutosh Trivedi. 2009. Concavely-Priced Probabilistic Timed Automata. In *CONCUR 2009 - Concurrency Theory, 20th International Conference, CONCUR 2009, Bologna, Italy, September 1-4, 2009. Proceedings (Lecture Notes in Computer Science)*, Mario Bravetti and Gianluigi Zavattaro (Eds.), Vol. 5710. Springer, 415–430. https://doi.org/10.1007/978-3-642-04081-8_28

[16] 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). https://doi.org/10.2168/LMCS-4(3:12)2008

[17] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. 2009. Stochastic Games for Verification of Probabilistic Timed Automata. In *Formal Modeling and Analysis of Timed Systems, 7th International Conference, FORMATS 2009, Budapest, Hungary, September 14-16, 2009. Proceedings (Lecture Notes in Computer Science)*, Joël Ouaknine and Frits W. Vaandrager (Eds.), Vol. 5813. Springer, 212–227. https://doi.org/10.1007/978-3-642-04368-0_17

[18] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. 2011. PRISM 4.0: Verification of Probabilistic Real-Time Systems. In *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings (Lecture Notes in Computer Science)*, Ganesh Gopalakrishnan and Shaz Qadeer (Eds.), Vol. 6806. Springer, 585–591. https://doi.org/10.1007/978-3-642-22110-1_47

[19] 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. https://doi.org/10.1007/s10703-006-0005-2

[20] 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. https://doi.org/10.1016/S0304-3975(01)00046-9

[21] 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. https://doi.org/10.1016/j.ic.2007.01.004

[22] François Laroussinie and Jeremy Sproston. 2007. State explosion in almost-sure probabilistic reachability. *Inf. Process. Lett.* 102, 6 (2007), 236–241. https://doi.org/10.1016/j.ipl.2007.01.003

[23] François Laroussinie and Jeremy Sproston. 2007. State explosion in almost-sure probabilistic reachability. *Inf. Process. Lett.* 102, 6 (2007), 236–241. https://doi.org/10.1016/j.ipl.2007.01.003

[24] Gethin Norman, David Parker, and Jeremy Sproston. 2013. Model checking for probabilistic timed automata. *Formal Methods in System Design* 43, 2 (2013), 164–190. https://doi.org/10.1007/s10703-012-0177-x

[25] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc.

[26] Sriram Sankaranarayanan and Enrico Vicario (Eds.). 2015. *Formal Modeling and Analysis of Timed Systems - 13th International Conference, FORMATS 2015, Madrid, Spain, September 2-4, 2015, Proceedings.* Lecture Notes in Computer Science, Vol. 9268. Springer. https://doi.org/10.1007/978-3-319-22975-1

[27] Jeremy Sproston. 2011. Discrete-Time Verification and Control for Probabilistic Rectangular Hybrid Automata. In *Eighth International Conference on Quantitative Evaluation of Systems, QEST 2011, Aachen, Germany, 5-8 September, 2011.* IEEE Computer Society, 79–88. https://doi.org/10.1109/QEST.2011.18

[28] Frits W. Vaandrager. 1997. A Theory of Testing for Timed Automata (Abstract). In *TAPSOFT'97: Theory and Practice of Software Development, 7th International Joint Conference CAAP/FASE, Lille, France, April 14-18, 1997, Proceedings (Lecture Notes in Computer Science)*, Michel Bidoit and Max Dauchet (Eds.), Vol. 1214. Springer, 39. https://doi.org/10.1007/BFb0030587

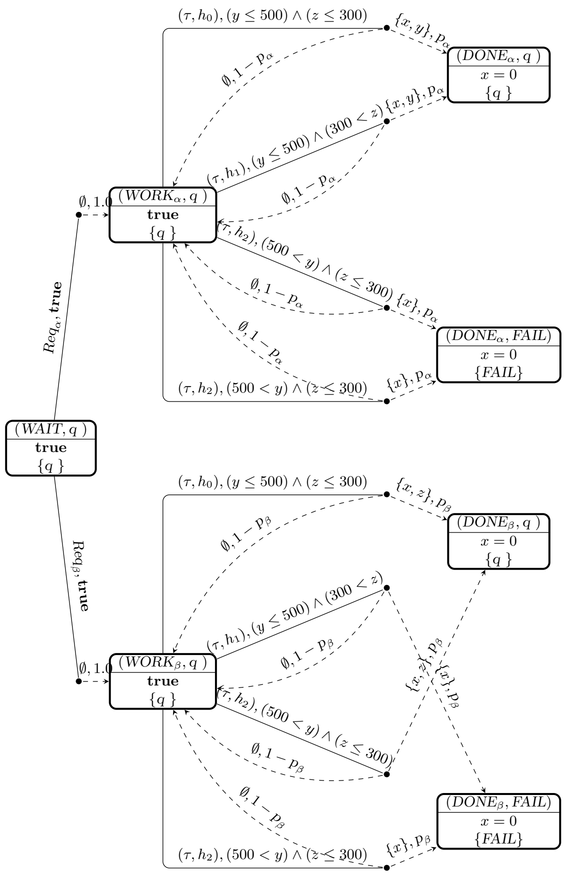

Figure 4: A Big Part of Product PTA

A APPENDIX