Model Checking Probabilistic Timed Automata with One or Two Clocks ⋆

Marcin Jurdziński[1], François Laroussinie[2], and Jeremy Sproston[3]

[1]Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK
[2]Lab. Spécification & Verification, ENS Cachan – CNRS UMR 8643, France
[3]Dipartimento di Informatica, Università di Torino, 10149 Torino, Italy
mju@dcs.warwick.ac.uk, fl@lsv.ens-cachan.fr,
sproston@di.unito.it

Abstract. Probabilistic timed automata are an extension of timed automata with discrete probability distributions. We consider model-checking algorithms for the subclasses of probabilistic timed automata which have one or two clocks. Firstly, we show that PCTL probabilistic model-checking problems (such as determining whether a set of target states can be reached with probability at least 0.99 regardless of how nondeterminism is resolved) are PTIME-complete for one clock probabilistic timed automata, and are EXPTIME-complete for probabilistic timed automata with two clocks. Secondly, we show that the model-checking problem for the probabilistic timed temporal logic PTCTL is EXPTIME-complete for one clock probabilistic timed automata. However, the corresponding model-checking problem for the subclass of PTCTL which does not permit both (1) punctual timing bounds, which require the occurrence of an event at an exact time point, and (2) comparisons with probability bounds other than 0 or 1, is PTIME-complete.

1 Introduction

Model checking is an automatic method for guaranteeing that a mathematical model of a system satisfies a formally-described property [8]. Many real-life systems, such as multimedia equipment, communication protocols, networks and fault-tolerant systems, exhibit *probabilistic* behaviour. This leads to the study of *probabilistic model checking* of probabilistic models based on Markov chains or Markov decision processes [25, 12, 9, 7, 10, 6]. Similarly, it is common to observe complex *real-time* behaviour in systems. Model checking of (non-probabilistic) continuous-time systems against properties of timed temporal logics, which can refer to the time elapsed along system behaviours, has been studied extensively in, for example, the context of *timed automata* [3, 4], which are automata extended with *clocks* that progress synchronously with time. Finally, certain systems exhibit both probabilistic *and* timed behaviour, leading to the development of model-checking algorithms for such systems [2, 12, 10, 15, 5, 19].

In this paper, we aim to study model-checking algorithms for *probabilistic timed automata* [13, 15], a variant of timed automata extended with discrete probability distributions, or (equivalently) Markov decision processes extended with clocks. Probabilistic

⋆ Supported in part by EPSRC project EP/E022030/1, Miur project Firb-Perf, and EEC project Crutial.

Table 1. Complexity results for model checking probabilistic timed automata

	One clock	Two clocks
Reachability, PCTL	P-complete	EXPTIME-complete
PTCTL$^{0/1}[\leq,\geq]$	P-complete	EXPTIME-complete
PTCTL$^{0/1}$	EXPTIME-complete	EXPTIME-complete
PTCTL$[\leq,\geq]$	P-hard, in EXPTIME	EXPTIME-complete
PTCTL	EXPTIME-complete	EXPTIME-complete

timed automata have been used to model systems such as the IEEE 1394 root contention protocol, the backoff procedure in the IEEE 802.11 Wireless LANs, and the IPv4 link local address resolution protocol [14]. The temporal logic that we use to describe properties of probabilistic timed automata is PTCTL (Probabilistic Timed Computation Tree Logic) [15]. The logic PTCTL includes operators that can refer to bounds on exact time and on the probability of the occurrence of events. For example, the property "a request is followed by a response within 5 time units with probability 0.99 or greater" can be expressed by the PTCTL property $request \Rightarrow \mathbb{P}_{\geq 0.99}(\mathsf{F}_{\leq 5} response)$. The logic PTCTL extends the probabilistic temporal logic PCTL [12, 7], and the real-time temporal logic TCTL [3].

In the non-probabilistic setting, timed automata with one clock have recently been studied extensively [17, 21, 1]. In this paper we consider the subclasses of probabilistic timed automata with one or two clocks. While probabilistic timed automata with a restricted number of clocks are less expressive than their counterparts with an arbitrary number of clocks, they can be used to model systems with simple timing constraints, such as probabilistic systems in which the time of a transition depends only on the time elapsed since the last transition. Conversely, one clock probabilistic timed automata are more natural and expressive than Markov decision processes in which durations are associated with transitions (for example, in [11, 19]). We note that the IEEE 802.11 Wireless LAN case study has two clocks [14], and that an abstract model of the IEEE 1394 root contention protocol can be obtained with one clock [23].

After introducing probabilistic timed automata and PTCTL in Section 2 and Section 3, respectively, in Section 4 we show that model-checking properties of PCTL, such as the property $\mathbb{P}_{\geq 0.99}(\mathsf{F} target)$ ("a set of target states is reached with probability at least 0.99 regardless of how nondeterminism is resolved"), is PTIME-complete for one clock probabilistic timed automata, which is the same as for probabilistic reachability properties on (untimed) Markov decision processes [22]. We also show that, in general, model checking of PTCTL on one clock probabilistic timed automata is EXPTIME-complete. However, inspired by the efficient algorithms obtained for non-probabilistic one clock timed automata [17], we also show that, restricting the syntax of PTCTL to the sub-logic in which (1) punctual timing bounds and (2) comparisons with probability bounds other than 0 or 1, are disallowed, results in a PTIME-complete model-checking problem. In Section 5, we show that reachability properties with probability bounds of 0 or 1 are EXPTIME-complete for probabilistic timed automata with two or more

clocks, implying EXPTIME-completeness of all the model-checking problems that we consider for this class of models. Our results are summarized in Table 1, where $0/1$ denotes the sub-logics of PTCTL with probability bounds of 0 and 1 only, and $[\leq, \geq]$ denotes the sub-logics of PTCTL in which punctual timing bounds are disallowed. The EXPTIME-hardness results are based on the concept of *countdown games*, which are two-player games operating in discrete time in which one player wins if it is able to make a state transition after *exactly* c time units have elapsed, regardless of the strategy of the other player. We believe that countdown games may be of independent interest. Note that we restrict our attention to probabilistic timed automata in which positive durations elapse in all loops of the system.

2 Probabilistic Timed Automata

Preliminaries. We use $\mathbb{R}_{\geq 0}$ to denote the set of non-negative real numbers, $\mathbb{N}$ to denote the set of natural numbers, and AP to denote a set of atomic propositions. A (discrete) probability *distribution* over a countable set Q is a function $\mu : Q \to [0, 1]$ such that $\sum_{q \in Q} \mu(q) = 1$. For a function $\mu : Q \to \mathbb{R}_{\geq 0}$ we define $\mathsf{support}(\mu) = \{q \in Q \mid \mu(q) > 0\}$. Then for an uncountable set Q we define $\mathsf{Dist}(Q)$ to be the set of functions $\mu : Q \to [0, 1]$, such that $\mathsf{support}(\mu)$ is a countable set and μ restricted to $\mathsf{support}(\mu)$ is a (discrete) probability distribution.

We now introduce *timed Markov decision processes*, which are Markov decision processes in which rewards associated with transitions are interpreted as time durations.

Definition 1. *A* timed Markov decision process *(TMDP)* $\mathsf{T} = (S, s_{init}, \rightarrow, lab)$ *comprises a (possibly uncountable) set of* states S *with an initial state* $s_{init} \in S$*; a (possibly uncountable)* timed probabilistic, nondeterministic transition relation $\rightarrow \subseteq S \times \mathbb{R}_{\geq 0} \times \mathsf{Dist}(S)$ *such that, for each state* $s \in S$*, there exists at least one tuple* $(s, \_, \_) \in \rightarrow$*; and a* labelling function $lab : S \to 2^{AP}$.

The transitions from state to state of a TMDP are performed in two steps: given that the current state is s, the first step concerns a nondeterministic selection of $(s, d, \nu) \in \rightarrow$, where d corresponds to the duration of the transition; the second step comprises a probabilistic choice, made according to the distribution ν, as to which state to make the transition to (that is, we make a transition to a state $s' \in S$ with probability $\nu(s')$). We often denote such a transition by $s \xrightarrow{d,\nu} s'$.

An infinite or finite *path* of the TMDP T is defined as an infinite or finite sequence of transitions, respectively, such that the target state of one transition is the source state of the next. We use $Path_{fin}$ to denote the set of finite paths of T, and $Path_{ful}$ the set of infinite paths of T. If ω is a finite path, we denote by $last(\omega)$ the last state of ω. For any path ω, let $\omega(i)$ be its $(i+1)$th state. Let $Path_{ful}(s)$ refer to the set of infinite paths commencing in state $s \in S$.

In contrast to a path, which corresponds to a resolution of nondeterministic and probabilistic choice, an *adversary* represents a resolution of nondeterminism *only*. Formally, an adversary of a TMDP T is a function A mapping every finite path $\omega \in Path_{fin}$ to a transition $(last(\omega), d, \nu) \in \rightarrow$. Let Adv be the set of adversaries of T. For any adversary $A \in Adv$, let $Path^A_{ful}$ denote the set of infinite paths resulting from the choices

of distributions of A, and, for a state $s \in S$, let $Path^A_{ful}(s) = Path^A_{ful} \cap Path_{ful}(s)$. Then we can define the probability measure $Prob^A_s$ over $Path^A_{ful}(s)$ (for details, see, for example, [15]). Note that, by defining adversaries as functions from finite paths, we permit adversaries to be dependent on the history of the system. Hence, the choice made by an adversary at a certain point in system execution can depend on the sequence of states visited, the nondeterministic choices taken, and the time elapsed from each state, up to that point.

We distinguish the two classes of TMDP. *Discrete TMDPs* are TMDPs in which (1) the state space S is finite, and (2) the transition relation $\rightarrow$ is finite and of the form $\rightarrow \subseteq S \times \mathbb{N} \times \mathsf{Dist}(S)$. In discrete TMDPs, the delays are interpreted as discrete jumps, with no notion of a continuously changing state as time elapses. The size $|\mathsf{T}|$ of a discrete TMDP T is $|S| + |\rightarrow|$, where $|\rightarrow|$ includes the size of the encoding of the timing constants and probabilities used in $\rightarrow$: the timing constants are written in binary, and, for any $s, s' \in S$ and (s, d, ν), the probability $\nu(s')$ is expressed as a ratio between two natural numbers, each written in binary. We let T^u be the untimed Markov decision process (MDP) corresponding to the discrete TMDP T, in which each transition $(s, d, \nu) \in \rightarrow$ is represented by a transition (s, ν). We define the accumulated duration $\mathsf{DiscDur}(\omega, i)$ along the infinite path $\omega = s_0 \xrightarrow{d_0,\nu_0} s_1 \xrightarrow{d_1,\nu_1} \cdots$ of T until the $(i+1)$-th state to be the sum $\sum_{0 \leq k < i} d_k$. A discrete TMDP is *structurally non-Zeno* when any finite path of the form $s_0 \xrightarrow{d_0,\nu_0} s_1 \cdots \xrightarrow{d_n,\nu_n} s_{n+1}$, such that $s_{n+1} = s_0$, satisfies $\sum_{0 \leq i \leq n} d_i > 0$. *Continuous TMDPs* are infinite-state TMDPs in which any transition $s \xrightarrow{d,\nu} s'$ describes the continuous passage of time, and thus a path $\omega = s_0 \xrightarrow{d_0,\nu_0} s_1 \xrightarrow{d_1,\nu_1} \cdots$ describes implicitly an infinite set of visited states. In the sequel, we use continuous TMDPs to give the semantics of probabilistic timed automata.

Syntax of probabilistic timed automata. Let $\mathcal{X}$ be a finite set of real-valued variables called *clocks*, the values of which increase at the same rate as real-time. The set $\Psi_{\mathcal{X}}$ of *clock constraints* over $\mathcal{X}$ is defined as the set of conjunctions over atomic formulae of the form $x \sim c$, where $x, y \in \mathcal{X}$, $\sim \in \{<, \leq, >, \geq, =\}$, and $c \in \mathbb{N}$.

Definition 2. *A* probabilistic timed automaton *(PTA)* $\mathsf{P} = (L, \bar{l}, \mathcal{X}, inv, prob, \mathcal{L})$ *is a tuple consisting of a finite set L of* locations *with the initial location $\bar{l} \in L$; a finite set $\mathcal{X}$ of clocks; a function $inv : L \rightarrow \Psi_{\mathcal{X}}$ associating an* invariant condition *with each location; a finite set $prob \subseteq L \times \Psi_{\mathcal{X}} \times \mathsf{Dist}(2^{\mathcal{X}} \times L)$ of* probabilistic edges *such that, for each $l \in L$, there exists at least one $(l, \_, \_) \in prob$; and a* labelling function $\mathcal{L} : L \rightarrow 2^{AP}$.

A probabilistic edge $(l, g, p) \in prob$ is a triple containing (1) a source location l, (2) a clock constraint g, called a *guard*, and (3) a probability distribution p which assigns probability to pairs of the form (X, l') for some clock set X and target location l'. The behaviour of a probabilistic timed automaton takes a similar form to that of a timed automaton [4]: in any location time can advance as long as the invariant holds, and a probabilistic edge can be taken if its guard is satisfied by the current values of the clocks. However, probabilistic timed automata generalize timed automata in the sense

that, once a probabilistic edge is nondeterministically selected, then the choice of which clocks to reset and which target location to make the transition to is *probabilistic*.

The size $|\mathsf{P}|$ of the PTA P is $|L| + |\mathcal{X}| + |inv| + |prob|$, where $|inv|$ represents the size of the binary encoding of the constants used in the invariant condition, and $|prob|$ includes the size of the binary encoding of the constants used in guards and the probabilities used in probabilistic edges. As in the case of TMDPs, probabilities are expressed as a ratio between two natural numbers, each written in binary.

A PTA is *structurally non-Zeno* [24] if, for every sequence $X_0, (l_0, g_0, p_0), X_1, (l_1, g_1, p_1), \cdots, X_n, (l_n, g_n, p_n)$, such that $p_i(X_{i+1}, l_{i+1}) > 0$ for $0 \leq i < n$, and $p_n(X_0, l_0) > 0$, there exists a clock $x \in \mathcal{X}$ and $0 \leq i, j \leq n$ such that $x \in X_i$ and $g_j \Rightarrow x \geq 1$ (that is, g_j contains a conjunct of the form $x \geq c$ for some $c \geq 1$). We use 1C-PTA (resp. 2C-PTA) to denote the set of structurally non-Zeno PTA with only one (resp. two) clock(s).

Semantics of probabilistic timed automata. We refer to a mapping $v : \mathcal{X} \to \mathbb{R}_{\geq 0}$ as a *clock valuation*. Let $\mathbb{R}^{\mathcal{X}}_{\geq 0}$ denote the set of clock valuations. Let $\mathbf{0} \in \mathbb{R}^{\mathcal{X}}_{\geq 0}$ be the clock valuation which assigns 0 to all clocks in $\mathcal{X}$. For a clock valuation $v \in \mathbb{R}^{\mathcal{X}}_{\geq 0}$ and a value $d \in \mathbb{R}_{\geq 0}$, we use $v + d$ to denote the clock valuation obtained by letting $(v + d)(x) = v(x) + d$ for all clocks $x \in \mathcal{X}$. For a clock set $X \subseteq \mathcal{X}$, we let $v[X := 0]$ be the clock valuation obtained from v by resetting all clocks within X to 0; more precisely, we let $v[X := 0](x) = 0$ for all $x \in X$, and let $v[X := 0](x) = v(x)$ for all $x \in \mathcal{X} \setminus X$. The clock valuation v *satisfies* the clock constraint $\psi \in \Psi_{\mathcal{X}}$, written $v \models \psi$, if and only if ψ resolves to true after substituting each clock $x \in \mathcal{X}$ with the corresponding clock value $v(x)$.

Definition 3. *The semantics of the probabilistic timed automaton* $\mathsf{P} = (L, \bar{l}, \mathcal{X}, inv, prob, \mathcal{L})$ *is the continuous TMDP* $\mathsf{T}[\mathsf{P}] = (S, s_{init}, \rightarrow, lab)$ *where:*

- $S = \{(l, v) \mid l \in L$ *and* $v \in \mathbb{R}^{\mathcal{X}}_{\geq 0}$ *s.t.* $v \models inv(l)\}$ *and* $s_{init} = (\bar{l}, \mathbf{0})$*;*
- $\rightarrow$ *is the smallest set such that* $((l, v), d, \mu) \in \rightarrow$ *if there exist* $d \in \mathbb{R}_{\geq 0}$ *and a probabilistic edge* $(l, g, p) \in prob$ *such that:*
  1. $v + d \models g$*, and* $v + d' \models inv(l)$ *for all* $0 \leq d' \leq d$*;*
  2. *for any* $(X, l') \in 2^{\mathcal{X}} \times L$*, we have that* $p(X, l') > 0$ *implies* $(v + d)[X := 0] \models inv(l')$*;*
  3. *for any* $(l', v') \in S$*, we have that* $\mu(l', v') = \sum_{X \in \mathsf{Reset}(v, d, v')} p(X, l')$*, where* $\mathsf{Reset}(v, d, v') = \{X \subseteq \mathcal{X} \mid (v + d)[X := 0] = v'\}$*.*
- lab *is such that* $lab(l, v) = \mathcal{L}(l)$ *for each state* $(l, v) \in S$*.*

Given a path $\omega = (l_0, v_0) \xrightarrow{d_0, \nu_0} (l_1, v_1) \xrightarrow{d_1, \nu_1} \cdots$ of $\mathsf{T}[\mathsf{P}]$, for every i, we use $\omega(i, d)$, with $0 \leq d \leq d_i$, to denote the state $(l_i, v_i + d)$ reached from (l_i, v_i) after delaying d time units. Such a pair (i, d) is called a *position* of ω. We define a total order on positions: given two positions $(i, d), (j, d')$ of ω, the position (i, d) precedes (j, d') — denoted $(i, d) \prec_\omega (j, d')$ — if and only if either $i < j$, or $i = j$ and $d < d'$. Furthermore, we define the accumulated duration $\mathsf{CtsDur}(\omega, i, d)$ along the path ω until position (i, d) to be the sum $d + \sum_{0 \leq k < i} d_k$.

3 Probabilistic timed temporal logic

We now proceed to describe a *probabilistic, timed* temporal logic which can be used to specify properties of probabilistic timed automata [15].

Definition 4. *The formulae of* PTCTL *(Probabilistic Timed Computation Tree Logic) are given by the following grammar:*

$$\phi ::= a \mid \phi \wedge \phi \mid \neg\phi \mid \mathbb{P}_{\bowtie\zeta}(\phi \mathsf{U}_{\sim c}\phi)$$

where $a \in AP$ is an atomic proposition, $\bowtie \in \{<, \leq, \geq, >\}$, $\sim \in \{\leq, =, \geq\}$, $\zeta \in [0,1]$ is a probability, and $c \in \mathbb{N}$ is a natural number.

We use standard abbreviations such as $\mathtt{true}$, $\mathtt{false}$, $\phi_1 \vee \phi_2$, $\phi_1 \Rightarrow \phi_2$, and $\mathbb{P}_{\bowtie\zeta}(\mathsf{F}_{\sim c}\phi)$ (for $\mathbb{P}_{\bowtie\zeta}(\mathtt{true}\mathsf{U}_{\sim c}\phi)$). Formulae with "always" temporal operators $\mathsf{G}_{\sim c}$ can also be written; for example $\mathbb{P}_{\geq\zeta}(\mathsf{G}_{\sim c}\phi)$ can be expressed by $\mathbb{P}_{\leq 1-\zeta}(\mathsf{F}_{\sim c}\neg\phi)$). The modalities U, F and G without subscripts abbreviate $\mathsf{U}_{\geq 0}$, $\mathsf{F}_{\geq 0}$ and $\mathsf{G}_{\geq 0}$, respectively. We refer to PTCTL properties of the form $\mathbb{P}_{\bowtie\zeta}(\mathsf{F}a)$ or $\neg\mathbb{P}_{\bowtie\zeta}(\mathsf{F}a)$ as *(untimed) reachability properties*. When $\zeta \in \{0,1\}$, these properties are referred to as *qualitative* reachability properties.

We define PTCTL$[\leq,\geq]$ as the sub-logic of PTCTL in which subscripts of the form $= c$ are not allowed in modalities $\mathsf{U}_{\sim c}, \mathsf{F}_{\sim c}, \mathsf{G}_{\sim c}$. We define PTCTL$^{0/1}[\leq,\geq]$ and PTCTL$^{0/1}$ as the qualitative restrictions in which probability thresholds ζ belong to $\{0,1\}$. Furthermore PCTL is the sub-logic in which there is no timing subscript $\sim c$ associated with the modalities $\mathsf{U}, \mathsf{F}, \mathsf{G}$. The size $|\Phi|$ of Φ is defined in the standard way as the number of symbols in Φ, with each occurrence of the same subformula of Φ as a single symbol.

We now define the satisfaction relation of PTCTL for discrete and continuous TMDPs.

Definition 5. *Given a discrete TMDP $\mathsf{T} = (S, s_{init}, \rightarrow, lab)$ and a* PTCTL *formula Φ, we define the satisfaction relation $\models_{\mathsf{T}}$ of* PTCTL *as follows:*

$$\begin{array}{ll}
s \models_{\mathsf{T}} a & \textit{iff } a \in lab(s) \\
s \models_{\mathsf{T}} \Phi_1 \wedge \Phi_2 & \textit{iff } s \models_{\mathsf{T}} \Phi_1 \textit{ and } s \models_{\mathsf{T}} \Phi_2 \\
s \models_{\mathsf{T}} \neg\Phi & \textit{iff } s \not\models_{\mathsf{T}} \Phi \\
s \models_{\mathsf{T}} \mathbb{P}_{\bowtie\zeta}(\varphi) & \textit{iff } Prob_s^A\{\omega \in Path_{ful}^A(s) \mid \omega \models_{\mathsf{T}} \varphi\} \bowtie \zeta,\ \forall A \in Adv \\
\omega \models_{\mathsf{T}} \Phi_1 \mathsf{U}_{\sim c}\Phi_2 & \textit{iff } \exists i \in \mathbb{N} \textit{ s.t. } \omega(i) \models_{\mathsf{T}} \phi_2,\ \mathsf{DiscDur}(\omega, i) \sim c, \\
 & \quad \textit{and } \omega(j) \models_{\mathsf{T}} \phi_1,\ \forall j < i\ .
\end{array}$$

Definition 6. *Given a continuous TMDP $\mathsf{T} = (S, s_{init}, \rightarrow, lab)$ and a* PTCTL *formula Φ, we define the satisfaction relation $\models_{\mathsf{T}}$ of* PTCTL *as in Definition 5, except for the following rule for $\Phi_1 \mathsf{U}_{\sim c}\Phi_2$:*

$$\begin{array}{ll}
\omega \models_{\mathsf{T}} \Phi_1 \mathsf{U}_{\sim c}\Phi_2 \textit{ iff } & \exists \textit{ position } (i,\delta) \textit{ of } \omega \textit{ s.t. } \omega(i,\delta) \models_{\mathsf{T}} \phi_2,\ \mathsf{CtsDur}(\omega, i, \delta) \sim c, \\
 & \quad \textit{and } \omega(j,\delta') \models_{\mathsf{T}} \phi_1,\ \forall \textit{ positions } (j,\delta') \textit{ of } \omega \textit{ s.t. } (j,\delta') \prec_\omega (i,\delta)\ .
\end{array}$$

When clear from the context, we omit the T subscript from $\models_{\mathsf{T}}$. We say that the TMDP $\mathsf{T} = (S, s_{init}, \rightarrow, lab)$ satisfies the PTCTL formula Φ, denoted by $\mathsf{T} \models \Phi$,

if and only if $s_{init} \models \Phi$. Furthermore, the PTA P satisfies Φ, denoted by $\mathsf{P} \models \Phi$, if and only if $\mathsf{T}[\mathsf{P}] \models \Phi$. Given an arbitrary structurally non-Zeno PTA P, model checking PTCTL formulae is in EXPTIME [15] (the algorithm consists of executing a standard polynomial-time model-checking algorithm for finite-state probabilistic systems [7, 6] on the exponential-size region graph of P). Qualitative reachability problems are EXPTIME-complete for PTA with an arbitrary number of clocks [20].

4 Model Checking One Clock Probabilistic Timed Automata

In this section we consider the case of 1C-PTA. We will see that model checking PCTL and $\text{PTCTL}^{0/1}[\leq, \geq]$ over 1C-PTA is P-complete (where the lower bound follows from the fact that qualitative reachability properties are P-hard for MDPs [22]), but remains EXPTIME-complete for the logic $\text{PTCTL}^{0/1}$. First we have the following result about the model-checking of PCTL formulae.

Proposition 1. *The* PCTL *model-checking problem for 1C-PTA is P-complete.*

4.1 Model checking $\text{PTCTL}^{0/1}[\leq, \geq]$ on 1C-PTA

In this section, inspired by related work on timed concurrent game structures [16], we first show that model-checking $\text{PTCTL}^{0/1}[\leq, \geq]$ properties of discrete TMDPs can be done efficiently. Then, in Theorem 1, using ideas from the TMDP case, we show that model checking $\text{PTCTL}^{0/1}[\leq, \geq]$ on 1C-PTA can also be done in polynomial time.

Proposition 2. *Let $\mathsf{T} = (S, s_{init}, \rightarrow, lab)$ be a structurally non-Zeno discrete TMDP and Φ be a $\text{PTCTL}^{0/1}[\leq, \geq]$ formula. Deciding whether $\mathsf{T} \models \Phi$ can be done in time $O(|\Phi| \cdot |S| \cdot |\rightarrow|)$.*

Proof (sketch). The model-checking algorithm is based on several procedures to deal with each modality of $\text{PTCTL}^{0/1}[\leq, \geq]$. The boolean operators and the PCTL modalities (without timed subscripts) can be handled in the standard manner, with the PCTL properties verified on the untimed MDP T^u corresponding to T. For formulae $\mathbb{P}_{\bowtie \zeta}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2)$, we assume that the truth values of subformulae Φ_1 and Φ_2 are known for any states of T. First, given that the TMDP is structurally non-Zeno, we have the equivalences $\mathbb{P}_{\leq 0}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2) \equiv \neg \mathsf{E} \Phi_1 \mathsf{U}_{\sim c} \Phi_2$ and $\mathbb{P}_{\geq 1}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2) \equiv \mathsf{A} \Phi_1 \mathsf{U}_{\sim c} (\mathbb{P}_{\geq 1}(\Phi_1 \mathsf{U} \Phi_2))$, where E (resp. A) stands for the existential (resp. universal) quantification over paths which exist in the logic TCTL. Thus we can apply the procedure proposed for model checking TCTL formulae – running in time $O(|S| \cdot |\rightarrow|)$ – over weighted graphs [18] (in the case of $\mathbb{P}_{\geq 1}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2)$, by first obtaining the set of states satisfying $\mathbb{P}_{\geq 1}(\Phi_1 \mathsf{U} \Phi_2)$, which can be done on T^u in time $O(\sum_{(s,d,\nu) \in \rightarrow} |\mathsf{support}(\nu)|)$).

The problem of verifying the remaining temporal properties of $\text{PTCTL}^{0/1}[\leq, \geq]$ can be considered in terms of turn-based 2-player games. Such a game is played over the space $S \cup \rightarrow$, and play proceeds as follows: from a state $s \in S$, player P_n chooses a transition $(s, d, \nu) \in \rightarrow$; then, from the transition (s, d, ν), player P_p chooses a state $s' \in \mathsf{support}(\nu)$. The duration of the move from s to s' via (s, d, ν) is d. Notions of

strategy of each player, and winning with respect to (untimed) path formulae of the form $\Phi_1 \mathsf{U} \Phi_2$, are defined as usual for 2-player games.

For the four remaining formulae, namely $\mathbb{P}_{\bowtie\zeta}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2)$ for $\bowtie\zeta \in \{> 0, < 1\}$, and $\sim \in \{\leq, \geq\}$, we consider the functions $\alpha, \beta, \gamma, \delta : S \to \mathbb{N}$, for representing minimal and maximal durations of interest. Intuitively, for a state $s \in S$, the value $\alpha(s)$ (resp. $\gamma(s)$) is the minimal (resp. maximal) duration that player P_p can ensure, regardless of the counter-strategy of P_n, along a path prefix from s satisfying $\Phi_1 \mathsf{U} \Phi_2$ (resp. $\Phi_1 \mathsf{U}(\mathbb{P}_{>0}(\Phi_1 \mathsf{U} \Phi_2))$). Similarly, the value $\beta(s)$ (resp. $\delta(s)$) is the minimal (resp. maximal) duration that player P_n can ensure, regardless of the counter-strategy of P_p, along a path prefix from s satisfying $\Phi_1 \mathsf{U} \Phi_2$ (resp. $\Phi_1 \mathsf{U}(\neg \mathbb{P}_{<1}(\Phi_1 \mathsf{U} \Phi_2))$). [1]

Using the fact that the TMDP is structurally non-Zeno, for any state $s \in S$, we can obtain the following equivalences: $s \models \mathbb{P}_{>0}(\Phi_1 \mathsf{U}_{\leq c} \Phi_2)$ if and only if $\alpha(s) \leq c$; $s \models \mathbb{P}_{<1}(\Phi_1 \mathsf{U}_{\leq c} \Phi_2)$ if and only if $\beta(s) > c$; $s \models \mathbb{P}_{>0}(\Phi_1 \mathsf{U}_{\geq c} \Phi_2)$ if and only if $\gamma(s) \geq c$; $s \models \mathbb{P}_{<1}(\Phi_1 \mathsf{U}_{\geq c} \Phi_2)$ if and only if $\delta(s) < c$. The functions $\alpha, \beta, \gamma, \delta$ can be computed on the 2-player game by applying the results of [16] on timed concurrent game structures: for each temporal operator $\mathbb{P}_{\bowtie\zeta}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2)$, this computation runs in time $O(|S| \cdot | \to |)$. □

We use Proposition 2 to obtain an efficient model-checking algorithm for 1C-PTA.

Theorem 1. *Let $\mathsf{P} = (L, \bar{l}, \mathcal{X}, inv, prob, \mathcal{L})$ be a 1C-PTA and Φ be a* PTCTL$^{0/1}[\leq, \geq]$ *formula. Deciding whether $\mathsf{P} \models \Phi$ can be done in polynomial time.*

Proof (sketch). Our aim is to label every state (l, v) of $\mathsf{T}[\mathsf{P}]$ with the set of subformulae of Φ which it satisfies (as $|\mathcal{X}| = 1$, recall that v is a single real value). For each location $l \in L$ and subformula Ψ of Φ, we construct a set $\mathsf{Sat}[l, \Psi] \subseteq \mathbb{R}_{\geq 0}$ of intervals such that $v \in \mathsf{Sat}[l, \Psi]$ if and only if $(l, v) \models \Psi$. We write $\mathsf{Sat}[l, \Psi] = \bigcup_{j=1,\ldots,k} \langle c_j; c'_j \rangle$ with $\langle \in \{[, (\}$ and $\rangle \in \{],)\}$. We consider intervals which conform to the following rules: for $1 \leq j \leq k$, we have $c_j < c'_j$ and $c_j, c'_j \in \mathbb{N} \cup \{\infty\}$, and for $1 \leq j < k$, we have $c'_j < c_{j+1}$. We will see that $|\mathsf{Sat}[l, \Psi]|$ – *i.e.* the number of intervals corresponding to a particular location – is bounded by $|\Psi| \cdot 2 \cdot |prob|$.

The cases of obtaining the sets $\mathsf{Sat}[l, \Psi]$ for boolean operators and atomic propositions are straightforward, and therefore we concentrate on the verification of subformulae Ψ of the form $\mathbb{P}_{\bowtie\zeta}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2)$. Assume that we have already computed the sets $\mathsf{Sat}[\_, \_]$ for Φ_1 and Φ_2. Our aim is to compute $\mathsf{Sat}[l, \Psi]$ for each location $l \in L$.

There are several cases depending on the constraint "$\bowtie\ \zeta$". The equivalence $\mathbb{P}_{\leq 0}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2) \equiv \neg \mathsf{E} \Phi_1 \mathsf{U}_{\sim c} \Phi_2$ can be used to reduce the "≤ 0" case to the appropriate polynomial-time labeling procedure for $\neg \mathsf{E} \Phi_1 \mathsf{U}_{\sim c} \Phi_2$ on one clock timed automata [17]. In the "≥ 1" case, the equivalence $\mathbb{P}_{\geq 1}(\Phi_1 \mathsf{U}_{\sim c} \Phi_2) \equiv \mathsf{A} \Phi_1 \mathsf{U}_{\sim c}(\mathbb{P}_{\geq 1}(\Phi_1 \mathsf{U} \Phi_2))$ relies on first computing the state set satisfying $\mathbb{P}_{\geq 1}(\Phi_1 \mathsf{U} \Phi_2)$, which can be handled using a qualitative PCTL model-checking algorithm, applied to a discrete TMDP built from P,

[1] If there is no strategy for player P_p (resp. player P_n) to guarantee the satisfaction of $\Phi_1 \mathsf{U} \Phi_2$ along a path prefix from s, then we let $\alpha(s) = \infty$ (resp. $\beta(s) = \infty$). Similarly, if there is no strategy for player P_p (resp. player P_n) to guarantee the satisfaction of $\Phi_1 \mathsf{U}(\mathbb{P}_{>0}(\Phi_1 \mathsf{U} \Phi_2))$ (resp. $\Phi_1 \mathsf{U}(\neg \mathbb{P}_{<1}(\Phi_1 \mathsf{U} \Phi_2))$) along a path prefix from s, then we let $\gamma(s) = -\infty$ (resp. $\delta(s) = -\infty$).

$\mathsf{Sat}[l, \Phi_1]$ and $\mathsf{Sat}[l, \Phi_2]$, in time $O(|\mathsf{P}| \cdot |prob| \cdot (|\Phi_1| + |\Phi_2|))$, and second verifying the formula $\mathsf{A}\Phi_1\mathsf{U}_{\sim c}(\mathbb{P}_{\geq 1}(\Phi_1\mathsf{U}\Phi_2))$ using the aforementioned method for one clock timed automata.

For the remaining cases, our aim is to construct a (finite) discrete TMDP $\mathsf{T}^r = (S^r, \_, \rightarrow^r, lab^r)$, which represents partially the semantic TMDP $\mathsf{T}[\mathsf{P}]$, for which the values of the functions α, β, γ and δ of the proof of Proposition 2 can be computed, and then use these functions to obtain the required sets $\mathsf{Sat}[\_, \Psi]$ (the initial state of T^r is irrelevant for the model-checking procedure, and is therefore omitted). The TMDP T^r will take a similar form to the region graph MDP of PTA [15], but will be of reduced size (the size will be independent of the magnitude of the constants used in invariants and guards): this will ensure a procedure running in time polynomial in $|\mathsf{P}|$.

We now describe the construction of T^r. In the following we assume that the sets $\mathsf{Sat}[l, \Phi_i]$ contain only closed intervals and that the guards and invariant of the PTA contain non-strict comparisons (and possibly intervals of the form $[b; \infty)$). The general case is omitted for reasons of space. Formally we let $\mathbb{B} = \{0\} \cup \mathsf{Cst}(\mathsf{P}) \cup \bigcup_{i\in\{1,2\}} \bigcup_{l\in L} \mathsf{Cst}(\mathsf{Sat}[l, \Phi_i])$, where $\mathsf{Cst}(\mathsf{P})$ is the set of constants occurring in the clock constraints of P, and where $\mathsf{Cst}(\mathsf{Sat}[l, \Phi_i])$ is the set of constants occurring as endpoints of the intervals in $\mathsf{Sat}[l, \Phi_i]$. Moreover for any right-open interval $[b; \infty)$ occurring in some $\mathsf{Sat}[l, \_]$, we add the constant $b+c+1$ in $\mathbb{B}$. We enumerate $\mathbb{B}$ as $b_0, b_1, ...b_M$ with $b_0 = 0$ and $b_i < b_{i+1}$ for $i < |\mathbb{B}|$. Note that $|\mathbb{B}|$ is bounded by $4 \cdot |\Psi| \cdot |prob|$. For any interval $(b_i; b_{i+1})$ and clock constraint $\psi \in \Psi_{\mathcal{X}}$, we let $(b_i; b_{i+1}) \models \psi$ if $v \models \psi$ for all $v \in (b_i; b_{i+1})$.

Considering the discrete TMDP corresponding to $\mathsf{T}[\mathsf{P}]$ restricted to states (l, b_i), with $b_i \in \mathbb{B}$, is sufficient to compute the values of functions α, β, γ and δ in any state (l, b_i). However, this does not allows us to deduce the value for any intermediate states in $(b_i; b_{i+1})$: indeed some probabilistic edges enabled from b_i may be disabled inside the interval. Therefore, in T^r, we have to consider also (l, b_i^+) and (l, b_{i+1}^-) corresponding respectively to the leftmost and rightmost points in $(b_i; b_{i+1})$ (when $i < M$). Then S^r is defined as the pairs (l, b_i) with $b_i \in \mathbb{B}$ and $b_i \models inv(l)$, and (l, b_i^+) and (l, b_{i+1}^-) with $b_i \in \mathbb{B}$, $i < M$ and $(b_i; b_{i+1}) \models inv(l)$. Note that the truth value of any invariant is constant over such intervals $(b_i; b_{i+1})$. Moreover note that all $\mathsf{T}[\mathsf{P}]$ states of the form (l, v) with $v \in (b_i; b_{i+1})$ satisfy the same boolean combinations of Φ_1 and Φ_2, and *enable the same probabilistic edges*. For any $(l, g, p) \in prob$, we write $b_i^+ \models g$ (and $b_{i+1}^- \models g$) when $(b_i; b_{i+1}) \models g$. Similarly, we write $b_i^+ \models inv(l)$ (and $b_{i+1}^- \models inv(l)$) when $(b_i; b_{i+1}) \models inv(l)$. We also consider the following ordering $b_0 < b_0^+ < b_1^- < b_1 < b_1^+ < \cdots < b_M^- < b_M < b_M^+$. We now define the set $\rightarrow^r$ of transitions of T^r as the smallest set such that $((l, \lambda), d, \nu) \in \rightarrow^r$, where $\lambda \in \{b_i^-, b_i, b_i^+\}$ for some $b_i \in \mathbb{B}$, if there exists $\lambda' \geq \lambda$, where $\lambda' \in \{b_j^-, b_j, b_j^+\}$ for some $b_j \in \mathbb{B}$, and $(l, g, p) \in prob$ such that:

- $d = b_j - b_i$, $\lambda' \models g$, and $\lambda'' \models inv(l)$ for any $\lambda \leq \lambda'' \leq \lambda'$;
- for each $(X, l') \in \mathsf{support}(p)$, we have $0 \models inv(l')$ if $X = \{x\}$, and $\lambda' \models inv(l')$ if $X = \emptyset$;
- for each $(l', \lambda'') \in S^r$, we have $\nu(l', \lambda'') = \nu_0(l', \lambda'') + \nu_\lambda(l', \lambda'')$, where $\nu_0(l', \lambda'') = p(l', \{x\})$ if $\lambda'' = [0, 0]$ and $\nu_0(l', \lambda'') = 0$ otherwise, and $\nu_\lambda(l', \lambda'') = p(l', \emptyset)$ if $\lambda'' = \lambda'$ and $\nu_\lambda(l', \lambda'') = 0$ otherwise.

Finally, to define lab^r, for a state (l, b_i), we let $a_{\Phi_j} \in lab^r(l, b_i)$ if and only if $b_i \in \mathsf{Sat}[l, \Phi_j]$, for $j \in \{1, 2\}$. The states (l, b_i^+) and (l, b_{i+1}^-) are labeled depending on the truth value of the Φ_j's in the interval $(b_i; b_{i+1})$: if $(b_i; b_{i+1}) \subseteq \mathsf{Sat}[l, \Phi_j]$, then $a_{\Phi_j} \in lab^r(l, b_i^+)$ and $a_{\Phi_j} \in lab^r(l, b_{i+1}^-)$. Note that given the "closed intervals" assumption made on $\mathsf{Sat}[l, \Phi_j]$, we have $lab^r(l, b_i^+) \subseteq lab^r(l, b_i)$ and $lab^r(l, b_{i+1}^-) \subseteq lab^r(l, b_i)$. Note that the fact that P is structurally non-Zeno means that T^r is structurally non-Zeno. The size of T^r is in $O(|\mathsf{P}|^2 \cdot |\Psi|)$.

Now we can apply the algorithms defined in the proof of Proposition 2 and obtain the value of the coefficients α, β, γ or δ for the states of T^r. Our next task is to define functions $\overline{\alpha}, \overline{\beta}, \overline{\gamma}, \overline{\delta} : S \to \mathbb{R}_{\geq 0}$, where S is the set of states of $\mathsf{T}[\mathsf{P}]$, which are analogues of α, β, γ or δ defined on $\mathsf{T}[\mathsf{P}]$. Our intuition is that we are now considering an infinite-state 2-player game, with players P_n and P_p, as in the proof of Proposition 2, over the state space of $\mathsf{T}[\mathsf{P}]$. Consider location $l \in L$. For $b \in \mathbb{B}$, we have $\overline{\alpha}(l, b) = \alpha(l, b)$, $\overline{\beta}(l, b) = \beta(l, b)$, $\overline{\gamma}(l, b) = \gamma(l, b)$ and $\overline{\delta}(l, b) = \delta(l, b)$. For intervals of the form $(b_i; b_{i+1})$, the functions $\overline{\alpha}$ and $\overline{\delta}$ will be decreasing (with slope -1) throughout the interval, because, for all states of the interval, the optimal choice of player P_n is to delay as much as possible inside any interval. Hence, the value $\overline{\alpha}(l, v)$ for $v \in (b_i; b_{i+1})$ is defined entirely by $\alpha(l, b_{i+1}^-)$ as $\overline{\alpha}(l, v) = \alpha(l, b_{i+1}^-) - b_{i+1} + b_i + v$. Similarly, $\overline{\delta}(l, v) = \delta(l, b_{i+1}^-) - b_{i+1} + b_i + v$.

Next we consider the values of $\overline{\beta}$ and $\overline{\gamma}$ over intervals $(b_i; b_{i+1})$. In this case, the functions will be constant over a portion of the interval (possibly an empty portion, or possibly the entire interval), then decreasing with slope -1. The constant part corresponds to those states in which the optimal choice of player P_n is to take a probabilistic edge, whereas the decreasing part corresponds to those states in which it is optimal for player P_n to delay until the end of the interval. The value $\overline{\beta}(l, v)$ for $v \in (b_i; b_{i+1})$ is defined both by $\beta(l, b_i^+)$ and $\beta(l, b_{i+1}^-)$ as $\overline{\beta}(l, v) = \beta(l, b_i^+)$ if $b_i < v \leq b_{i+1} - (\beta(l, b_i^+) - \beta(l, \beta_{i+1}^-))$, and as $\overline{\beta}(l, v) = \beta(l, \beta_{i+1}^-) - (v - \beta(l, b_i^+))$ otherwise. An analogous definition holds also for $\overline{\gamma}$.

From the functions $\overline{\alpha}$, $\overline{\beta}$, $\overline{\gamma}$ and $\overline{\delta}$ defined above, it becomes possible to define $\mathsf{Sat}[l, \Psi]$ by keeping in this set of intervals only the parts satisfying the thresholds $\leq c$, $> c$, $\geq c$ and $< c$, respectively, as in the proof of Proposition 2. We can show that the number of intervals in $\mathsf{Sat}[l, \Psi]$ is bounded by $2 \cdot |\Psi| \cdot |prob|$. For the case in which a function $\overline{\alpha}$, $\overline{\beta}$, $\overline{\gamma}$ or $\overline{\delta}$ is decreasing throughout an interval, then an interval in $\mathsf{Sat}[l, \Phi_1]$ which corresponds to several consecutive intervals in T^r can provide at most one (sub)interval in $\mathsf{Sat}[l, \Psi]$, because the threshold can cross at most once the function in at most one interval. For the case in which a function $\overline{\beta}$ or $\overline{\gamma}$ combines a constant part and a part with slope -1 within an interval, the threshold can cross the function in several intervals $(b_i; b_{i+1})$ contained in a common interval of $\mathsf{Sat}[l, \Phi_1]$. However, such a cut is due to a guard $x \geq k$ of a given transition, and thus the number of cuts in bounded by $|prob|$. Moreover a guard $x \leq k$ may also add an interval. Thus the number of new intervals in $\mathsf{Sat}[q, \Psi]$ is bounded by $2 \cdot |prob|$.

In addition to these cuts, any interval in $\mathsf{Sat}[l, \Phi_2]$ may provide an interval in $\mathsf{Sat}[l, \Psi]$. This gives the $2 \cdot |\Psi| \cdot |prob|$ bound for the size of $\mathsf{Sat}[l, \Psi]$. □

Corollary 1. *The* $\text{PTCTL}^{0/1}[\leq, \geq]$ *model-checking problem for 1C-PTA is P-complete.*

4.2 Model checking PTCTL$^{0/1}$ on 1C-PTA

We now consider the problem of model-checking PTCTL$^{0/1}$ properties on 1C-PTA. An EXPTIME algorithm for this problem exists by the definition of a MDP analogous to the region graph used in non-probabilistic timed automata verification [15]. We now show that the problem is also EXPTIME-hard by the following three steps. First we introduce *countdown games*, which are a simple class of turn-based 2-player games with discrete timing, and show that the problem of deciding the winner in a countdown game is EXPTIME-complete. Secondly, we reduce the countdown game problem to the PTCTL$^{0/1}$ problem on TMDPs. Finally, we adapt the reduction to TMDPs to reduce also the countdown game problem to the PTCTL$^{0/1}$ problem on 1C-PTA.

A *countdown game* $\mathcal{C}$ consists of a weighted graph $(\mathtt{S}, \mathtt{T})$, where $\mathtt{S}$ is the set of *states* and $\mathtt{T} \subseteq \mathtt{S} \times \mathbb{N} \setminus \{0\} \times \mathtt{S}$ is the *transition relation*. If $\mathtt{t} = (\mathtt{s}, d, \mathtt{s}') \in \mathtt{T}$ then we say that the *duration* of the transition $\mathtt{t}$ is d. A configuration of a countdown game is a pair $(\mathtt{s}, c)$, where $\mathtt{s} \in \mathtt{S}$ is a state and $c \in \mathbb{N}$. A *move* of a countdown game from a configuration $(\mathtt{s}, c)$ is performed in the following way: first player 1 chooses a number d, such that $0 < d \leq c$ and $(\mathtt{s}, d, \mathtt{s}') \in \mathtt{T}$, for some state $\mathtt{s}' \in \mathtt{S}$; then player 2 chooses a transition $(\mathtt{s}, d, \mathtt{s}') \in \mathtt{T}$ of duration d. The resulting new configuration is $(\mathtt{s}', c - d)$. There are two types of *terminal* configurations, i.e., configurations $(\mathtt{s}, c)$ in which no moves are available. If $c = 0$ then the configuration $(\mathtt{s}, c)$ is terminal and is a *winning configuration for player 1*. If for all transitions $(\mathtt{s}, d, \mathtt{s}') \in \mathtt{T}$ from the state $\mathtt{s}$, we have that $d > c$, then the configuration $(\mathtt{s}, c)$ is terminal and it is a *winning configuration for player 2*. The algorithmic problem of *deciding the winner* in countdown games is, given a weighted graph $(\mathtt{S}, \mathtt{T})$ and a configuration $(\mathtt{s}, c)$, where all the durations of transitions in $\mathcal{C}$ and the number c are given in binary, to determine whether player 1 has a winning strategy from the configuration $(\mathtt{s}, c)$. If the state from which the game is started is clear from the context then we sometimes specify the initial configuration by giving the number c alone.

Theorem 2. *Deciding the winner in countdown games is* EXPTIME*-complete.*

Proof (sketch). Observe that every configuration of a countdown game played from a given initial configuration can be written down in polynomial space and every move can be computed in polynomial time; hence the winner in the game can be determined by a straightforward alternating PSPACE algorithm. Therefore the problem is in EXPTIME because APSPACE = EXPTIME.

We now prove EXPTIME-hardness by a reduction from the acceptance of a word by a linearly-bounded alternating Turing machine. Let $M = (\Sigma, Q, q_0, q_{acc}, Q_\exists, Q_\forall, \Delta)$ be an alternating Turing machine, where Σ is a finite alphabet, $Q = Q_\exists \cup Q_\forall$ is a finite set of states partitioned into existential states $Q_\exists$ and universal states $Q_\forall$, $q_0 \in Q$ is an initial state, $q_{acc} \in Q$ is an accepting state, and $\Delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{L, R\}$ is a transition relation. Let $B > 2 \cdot |Q \times \Sigma|$ be an integer constant and let $w \in \Sigma^n$ be an input word. W.l.o.g. we can assume that M uses exactly n tape cells when started on the word w, and hence a configuration of M is a word $\mathbf{b}_0 \mathbf{b}_1 \cdots \mathbf{b}_{n-1} \in (\Sigma \cup Q \times \Sigma)^n$. Let $\langle \cdot \rangle : (\Sigma \cup Q \times \Sigma) \to \{0, 1, \ldots, B-1\}$ be an injection. For every $\mathbf{a} \in \Sigma \cup Q \times \Sigma$, it is convenient to think of $\langle \mathbf{a} \rangle$ as a B-ary digit, and we can encode a configuration $u = \mathbf{b}_0 \mathbf{b}_1 \cdots \mathbf{b}_{n-1} \in (\Sigma \cup Q \times \Sigma)^n$ of M as the number $N(u) = \sum_{i=0}^{n-1} \langle \mathbf{b}_i \rangle \cdot B^i$.

Let $i \in \mathbb{N}$, $0 \leq i < n$, be a tape cell position, and let $\mathbf{a} \in \Sigma \cup Q \times \Sigma$. We define a countdown game $\mathrm{Check}^{i,\mathbf{a}}$, such that for every configuration $u = \mathbf{b}_0 \cdots \mathbf{b}_{n-1}$ of M, player 1 has a winning strategy from the configuration $(\mathtt{s}_0^{i,\mathbf{a}}, N(u))$ of the game $\mathrm{Check}^{i,\mathbf{a}}$ if and only if $\mathbf{b}_i = \mathbf{a}$. The game $\mathrm{Check}^{i,\mathbf{a}}$ has states $\mathtt{S} = \{\, \mathtt{s}_0^{i,\mathbf{a}}, \ldots, \mathtt{s}_n^{i,\mathbf{a}} \,\}$, and for every k, $0 \leq k < n$, we have a transition $(\mathtt{s}_k^{i,\mathbf{a}}, d, \mathtt{s}_{k+1}^{i,\mathbf{a}}) \in \mathtt{T}$, if:

$$d = \begin{cases} \langle \mathbf{a} \rangle \cdot B^k & \text{if } k = i, \\ \langle \mathbf{b} \rangle \cdot B^k & \text{if } k \neq i \text{ and } \mathbf{b} \in \Sigma \cup S \times \Sigma. \end{cases}$$

There are no transitions from the state $\mathtt{s}_n^{i,\mathbf{a}}$. Observe that if $\mathbf{b}_i = \mathbf{a}$ then the winning strategy for player 1 in game $\mathrm{Check}^{i,\mathbf{a}}$ from $N(u)$ is to choose the transitions $(\mathtt{s}_k^{i,\mathbf{a}}, \mathbf{b}_k \cdot B^k, \mathtt{s}_{k+1}^{i,\mathbf{a}})$, for all k, $0 \leq k < n$. If, however, $\mathbf{b}_i \neq \mathbf{a}$ then there is no way for player 1 to count down from $N(u)$ to 0 in the game $\mathrm{Check}^{i,\mathbf{a}}$.

Now we define a countdown game $\mathcal{C}_M$, such that M accepts $w = \sigma_0 \sigma_1 \ldots \sigma_{n-1}$ if and only if player 1 has a winning strategy in $\mathcal{C}_M$ from configuration $(q_0, N(u))$, where $u = (q_0, \sigma_0)\sigma_1 \ldots \sigma_{n-1}$ is the initial configuration of M with input w. The main part of the countdown game $\mathcal{C}_M$ is a gadget that allows the game to simulate one step of M. Note that one step of a Turing machine makes only local changes to the configuration of the machine: if the configuration is of the form $u = \mathbf{a}_0 \ldots \mathbf{a}_{n-1} = \sigma_0 \ldots \sigma_{i-1}(q, \sigma_i)\sigma_{i+1} \ldots \sigma_{n-1}$, then performing one step of M can only change entries in positions $i-1$, i, or $i+1$ of the tape. For every tape position i, $0 \leq i < n$, for every triple $\tau = (\sigma_{i-1}, (q, \sigma_i), \sigma_{i+1}) \in \Sigma \times (Q \times \Sigma) \times \Sigma$, and for every transition $t = (q, \sigma, q', \sigma', D) \in \Delta$ of machine M, we now define the number $d_t^{i,\tau}$, such that if $\sigma_i = \sigma$ and performing transition t at position i of configuration u yields configuration $u' = \mathbf{b}_0 \ldots \mathbf{b}_{n-1}$, then $N(u) - d_t^{i,\tau} = N(u')$. For example, assume that $i > 0$ and that $D = L$; we have that $\mathbf{b}_k = \mathbf{a}_k = \sigma_k$, for all $k \notin \{\, i-1, i, i+1 \,\}$ and $\mathbf{b}_{i+1} = \mathbf{a}_{i+1} = \sigma_{i+1}$. Moreover we have that $\mathbf{b}_{i-1} = (q', \sigma_{i-1})$, and $\mathbf{b}_i = \sigma'$. We define $d_t^{i,\tau}$ as follows:

$$\begin{aligned} d_t^{i,\tau} &= (\langle \mathbf{b}_{i-1} \rangle - \langle \mathbf{a}_{i-1} \rangle) \cdot B^{i-1} + (\langle \mathbf{b}_i \rangle - \langle \mathbf{a}_i \rangle) \cdot B^i \\ &= (\langle (q', \sigma_{i-1}) \rangle - \langle \sigma_{i-1} \rangle) \cdot B^{i-1} + (\langle \sigma' \rangle - \langle (q, \sigma_i) \rangle) \cdot B^i. \end{aligned}$$

The gadget for simulating one transition of M from a state $q \in Q \setminus \{\, q_{acc} \,\}$ has three layers. In the first layer, from a state $q \in Q \setminus \{\, q_{acc} \,\}$, player 1 chooses a pair (i, τ), where i, $0 \leq i < n$, is the position of the tape head, and $\tau = (\mathbf{a}, \mathbf{b}, \mathbf{c}) \in \Sigma \times (Q \times \Sigma) \times \Sigma$ is his guess for the contents of tape cells $i-1$, i, and $i+1$. In this way the state (q, i, τ) of the gadget is reached, where the duration of this transition is 0. Intuitively, in the first layer player 1 has to declare that he knows the position i of the head in the current configuration as well as the contents $\tau = (\mathbf{a}, \mathbf{b}, \mathbf{c})$ of the three tape cells in positions $i-1$, i, and $i+1$. In the second layer, in a state (q, i, τ) player 2 chooses between four successor states: the state $(q, i, \tau, *)$ and the three subgames $\mathrm{Check}^{i-1,\mathbf{a}}$, $\mathrm{Check}^{i,\mathbf{b}}$, and $\mathrm{Check}^{i+1,\mathbf{c}}$. The four transitions are of duration 0. Intuitively, in the second layer player 2 verifies that player 1 declared correctly the contents of the three tape cells in positions $i-1$, i, and $i+1$. Finally, in the third layer, if $q \in Q_\exists$ (resp., $q \in Q_\forall$), then from a state $(q, i, \tau, *)$ player 1 (resp., player 2) chooses a transition

$t = (q, \sigma, q', \sigma', D)$ of machine M, such that $\mathbf{b} = (q, \sigma)$, reaching the state $q' \in Q$ of the gadget, with a transition of duration $d_t^{i,\tau}$.

Note that the gadget described above violates some conventions that we have adopted for countdown games. Observe that durations of some transitions in the gadget are 0 and the duration $d_t^{i,\tau}$ may even be negative, while in the definition of countdown games we required that durations of all transitions are positive. In order to correct this we add the number B^n to the durations of all transitions described above. This change requires a minor modification to the subgames $\mathrm{Check}^{i,\mathbf{a}}$: we add an extra transition $(\mathtt{s}_n^{i,\mathbf{a}}, B^n, s_n^{i,\mathbf{a}})$. We need this extra transition because instead of starting from $(q_0, N(u))$ as the initial configuration of the game $\mathcal{C}_M$, where u is the initial configuration of M running on w, we are going to start from the configuration $(q_0, B^{3n} + N(u))$. In this way the countdown game can perform a simulation of at least B^n steps of M; note that B^n is an upper bound on the number of all configurations of M.

W.l.o.g., we can assume that whenever the alternating Turing machine M accepts an input word w then it finishes its computation with blanks in all tape cells, its head in position 0, and in the unique accepting state q_{acc}; we write u_{acc} for this unique accepting configuration of machine M. Moreover, assume that there are no transitions from q_{acc} in M. In order to complete the definition of the countdown game G_M, we add a transition of duration $N(u_{acc})$ from the state q_{acc} of game $\mathcal{C}_M$. □

Proposition 3. *The* PTCTL$^{0/1}$ *model-checking problem for structurally non-Zeno discrete TMDPs is* EXPTIME*-complete.*

Proof. An EXPTIME algorithm can be obtained by employing the algorithms of [19]. We now prove EXPTIME-hardness of PTCTL$^{0/1}$ model checking on discrete TMDPs by a reduction from countdown games. Let $\mathcal{C} = (\mathtt{S}, \mathtt{T})$ be a countdown game and $(\overline{\mathtt{s}}, c)$ be its initial configuration. We construct a TMDP $\mathsf{T}_{\mathcal{C},(\overline{\mathtt{s}},c)} = (S, s_{init}, \rightarrow, lab)$ such that player 1 wins $\mathcal{C}$ from $(\overline{\mathtt{s}}, c)$ if and only if $\mathsf{T}_{\mathcal{C},(\overline{\mathtt{s}},c)} \models \neg\mathbb{P}_{<1}(\mathsf{F}_{=c}\mathtt{true})$. Let $S = \mathtt{S}$ and $s_{init} = \overline{\mathtt{s}}$. We define $\rightarrow$ to be the smallest set satisfying the following: for each $\mathtt{s} \in \mathtt{S}$ and $d \in \mathbb{N}_{>0}$, if $(\mathtt{s}, d, \mathtt{s}') \in \mathtt{T}$ for some $\mathtt{s}' \in \mathtt{T}$, we have $(\mathtt{s}, d, \nu) \in \rightarrow$, where ν is an arbitrary distribution over $\mathtt{S}$ such that $\mathsf{support}(\nu) = \{\mathtt{s}' \mid (\mathtt{s}, d, \mathtt{s}') \in \mathtt{T}\}$. The labelling condition lab is arbitrary. Then we can show that player 1 wins from the configuration $(\overline{\mathtt{s}}, c)$ if and only if there exists an adversary of $\mathsf{T}_{\mathcal{C},(\overline{\mathtt{s}},c)}$ such that a state is reached from $s_{init} = \overline{\mathtt{s}}$ after exactly c time units with probability 1. The latter is equivalent to $s_{init} \models \neg\mathbb{P}_{<1}(\mathsf{F}_{=c}\mathtt{true})$. □

We now show that the proof of Proposition 3 can be adapted to show the EXPTIME-completeness of the analogous model-checking problem on 1C-PTA.

Theorem 3. *The* PTCTL$^{0/1}$ *model-checking problem for 1C-PTA is* EXPTIME*-complete.*

Proof. Recall that there exists an EXPTIME algorithm for model-checking PTCTL$^{0/1}$ properties on PTA; hence, it suffices to show EXPTIME-hardness for PTCTL$^{0/1}$ and 1C-PTA. Let $\mathcal{C}$ be a countdown game with an initial configuration $(\overline{\mathtt{s}}, c)$. We construct the 1C-PTA $\mathsf{P}^{1C}_{\mathcal{C},(\overline{\mathtt{s}},c)} = (L, \bar{l}, \{x\}, inv, prob, \mathcal{L})$ which simulates the behaviour of the TMDP $\mathsf{T}_{\mathcal{C},(\overline{\mathtt{s}},c)}$ of the proof of Proposition 3 in the following way. Each state $\mathtt{s} \in \mathtt{S}$ of $\mathsf{T}_{\mathcal{C},(\overline{\mathtt{s}},c)}$ corresponds to two distinct locations $l^1_{\mathtt{s}}$ and $l^2_{\mathtt{s}}$ of $\mathsf{P}^{1C}_{\mathcal{C},(\overline{\mathtt{s}},c)}$, and we let $L^i = \{l^i_{\mathtt{s}} \mid$

$\mathtt{s} \in \mathtt{S}\}$ for $i \in \{1,2\}$. Let $\bar{l} = l^1_{\bar{\mathtt{s}}}$. For every transition $(\mathtt{s}, d, \nu) \in \rightarrow$ of $\mathsf{T}_{\mathcal{C},(\bar{\mathtt{s}},c)}$, we have the probabilistic edges $(l^1_{\mathtt{s}}, x = 0, p^1), (l^2_{\mathtt{s}}, x = d, p^2) \in prob$, where $p^1(\{\ x\ \}, l^2_{\mathtt{s}}) = 1$, and $p^2(\{\ x\ \}, l^1_{\mathtt{s}'}) = \nu(\mathtt{s}')$ for each location $\mathtt{s}'$. For each state $\mathtt{s} \in \mathtt{S}$, let $inv(l^1_{\mathtt{s}}) = (x \leq 0)$ and $inv(l^2_{\mathtt{s}}) = (x \leq d)$. That is, the PTA $\mathsf{P}^{1C}_{\mathcal{C},(\bar{\mathtt{s}},c)}$ moves from the location $l^1_{\mathtt{s}}$ to $l^2_{\mathtt{s}}$ instantaneously. Locations in L^1 are labelled by the atomic proposition a, whereas locations in L^2 are labelled by $\emptyset$. Then we can observe that $\mathsf{P}^{1C}_{\mathcal{C},(\bar{\mathtt{s}},c)} \models \neg\mathbb{P}_{<1}(\mathsf{F}_{=c}a)$ if and only if $\mathsf{T}_{\mathcal{C},(\bar{\mathtt{s}},c)} \models \neg\mathbb{P}_{<1}(\mathsf{F}_{=c}\mathtt{true})$. As the latter problem has been shown to be EXPTIME-hard in the proof of Proposition 3, we conclude that model checking PTCTL$^{0/1}$ on 1C-PTA is also EXPTIME-hard. □

5 Model Checking Two Clocks Probabilistic Timed Automata

We now show EXPTIME-completeness of the simplest problems that we consider on 2C-PTA.

Theorem 4. *Qualitative probabilistic reachability problems for 2C-PTA are* EXPTIME*-complete.*

Proof. EXPTIME algorithms exist for probabilistic reachability problems on PTA, and therefore it suffices to show EXPTIME-hardness. We proceed by reduction from countdown games. Let $\mathcal{C}$ be a countdown game with initial configuration $(\bar{\mathtt{s}}, c)$, and let $\mathsf{P}^{1C}_{\mathcal{C},(\bar{\mathtt{s}},c)} = (L, \bar{l}, \{x\}, inv, prob, \mathcal{L})$ be the 1C-PTA constructed in the proof of Theorem 3. We define the 2C-PTA $\mathsf{P}^{2C}_{\mathcal{C},(\bar{\mathtt{s}},c)} = (L \cup \{l^\star\}, \bar{l}, \{x, y\}, inv', prob', \mathcal{L}')$ in the following way. The set of probabilistic edges $prob'$ is obtained by adding to $prob$ the following: for each location $l \in L$, we extend the set of outgoing probabilistic edges of l with $(l, y = c, p^{l^\star})$, where $p^{l^\star}(\emptyset, l^\star) = 1$; to make $prob'$ total, we also add $(l^\star, \mathtt{true}, p^{l^\star})$. For each $l \in L$, let $inv'(l) = inv(l)$, and let $inv'(l^\star) = \mathtt{true}$. Finally, we let $\mathcal{L}'(l^\star) = a$, and $\mathcal{L}(l) = \emptyset$ for all $l \in L$. Then $\mathsf{P}^{2C}_{\mathcal{C},(\bar{\mathtt{s}},c)} \models \neg\mathbb{P}_{<1}(\mathsf{F}a)$ if and only if $\mathsf{P}^{1C}_{\mathcal{C},(\bar{\mathtt{s}},c)} \models \neg\mathbb{P}_{<1}(\mathsf{F}_{=c}a)$. The EXPTIME-hardness of the latter problem has been shown in the proof of Theorem 3, and hence checking qualitative probabilistic reachability properties such as $\neg\mathbb{P}_{<1}(\mathsf{F}a)$ on 2C-PTA is EXPTIME-hard. □

Corollary 2. *The* PCTL, PTCTL$^{0/1}[\leq, \geq]$, PTCTL$^{0/1}$, PTCTL$[\leq, \geq]$ *and* PTCTL *model-checking problems for 2C-PTA are* EXPTIME*-complete.*

References

1. P. A. Abdulla, J. Deneux, J. Ouaknine, and J. Worrell. Decidability and complexity results for timed automata via channel machines. In *Proc. of the 32nd Int. Coll. on Aut., Lang. and Progr. (ICALP'05)*, volume 3580 of *LNCS*, pages 1089–1101. Springer, 2005.
2. R. Alur, C. Courcoubetis, and D. L. Dill. Model-checking for probabilistic real-time systems. In *Proc. of the 18th Int. Coll. on Aut., Lang. and Progr. (ICALP'91)*, volume 510 of *LNCS*, pages 115–136. Springer, 1991.
3. R. Alur, C. Courcoubetis, and D. L. Dill. Model-checking in dense real-time. *Inf. and Comp.*, 104(1):2–34, 1993.

4. R. Alur and D. L. Dill. A theory of timed automata. *Theo. Comp. Sci.*, 126(2):183–235, 1994.
5. C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Trans. on Soft. Enginee.*, 29(6):524–541, 2003.
6. C. Baier and M. Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11(3):125–155, 1998.
7. A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proc. of the 15th Conf. on Found. of Software Technol. and Theor. Comp. Sci. (FSTTCS'95)*, volume 1026 of *LNCS*, pages 499–513. Springer, 1995.
8. E. M. Clarke, O. Grumberg, and D. Peled. *Model checking*. MIT Press, 1999.
9. C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the ACM*, 42(4):857–907, 1995.
10. L. de Alfaro. *Formal verification of probabilistic systems*. PhD thesis, Stanford University, Department of Computer Science, 1997.
11. L. de Alfaro. Temporal logics for the specification of performance and reliability. In *Proc. of the 14th An. Symp. on Theor. Aspects of Comp. Sci. (STACS'97)*, volume 1200 of *LNCS*, pages 165–176. Springer, 1997.
12. H. A. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.
13. H. E. Jensen. Model checking probabilistic real time systems. In *Proc. of the 7th Nordic Work. on Progr. Theory*, pages 247–261. Chalmers Institute of Technology, 1996.
14. M. Kwiatkowska, G. Norman, D. Parker, and J. Sproston. Performance analysis of probabilistic timed automata using digital clocks. *Formal Meth. in Syst. Design*, 29:33–78, 2006.
15. M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. *Theo. Comp. Sci.*, 286:101–150, 2002.
16. F. Laroussinie, N. Markey, and G. Oreiby. Model checking timed ATL for durational concurrent game structures. In *Proc. of the 4th Int. Conf. on Formal Modelling and Analysis of Timed Systems (FORMATS'06)*, volume 4202 of *LNCS*, pages 245–259. Springer, 2006.
17. F. Laroussinie, N. Markey, and P. Schnoebelen. Model checking timed automata with one or two clocks. In *Proc. of the 15th Int. Conf. on Concurrency Theory (CONCUR'04)*, volume 3170 of *LNCS*, pages 387–401. Springer, 2004.
18. F. Laroussinie, N. Markey, and P. Schnoebelen. Efficient timed model checking for discrete-time systems. *Theo. Comp. Sci.*, 353(1–3):249–271, 2005.
19. F. Laroussinie and J. Sproston. Model checking durational probabilistic systems. In *Proc. of the 8th Int. Conf. on Foundations of Software Science and Computation Structures (FoSSaCS'05)*, volume 3441 of *LNCS*, pages 140–154. Springer, 2005.
20. F. Laroussinie and J. Sproston. State explosion in almost-sure probabilistic reachability. To appear in *IPL*, 2007.
21. S. Lasota and I. Walukiewicz. Alternating timed automata. In *Proc. of the 8th Int. Conf. on Foundations of Software Science and Computation Structures (FoSSaCS'05)*, volume 3441 of *LNCS*, pages 299–314. Springer, 2005.
22. C. Papadimitriou and J. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.
23. M. Stoelinga. *Alea Jacta est: Verification of probabilistic, real-time and parametric systems*. PhD thesis, Institute for Computing and Information Sciences, University of Nijmegen, 2002.
24. S. Tripakis, S. Yovine, and A. Bouajjani. Checking timed Büchi automata emptiness efficiently. *Formal Meth. in Syst. Design*, 26(3):267–292, 2005.
25. M. Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proc. of the 16th An. Symp. on Foundations of Computer Science (FOCS'85)*, pages 327–338. IEEE Computer Society Press, 1985.