Tweaking the Odds in Probabilistic Timed Automata

Arnd Hartmanns[1(✉)], Joost-Pieter Katoen[1,2], Bram Kohlen[1], and Jip Spel[2]

[1] University of Twente, Enschede, The Netherlands
a.hartmanns@utwente.nl
[2] RWTH Aachen University, Aachen, Germany

Abstract. We consider probabilistic timed automata (PTA) in which probabilities can be parameters, i.e. symbolic constants. They are useful to model randomised real-time systems where exact probabilities are unknown, or where the probability values should be optimised. We prove that existing techniques to transform probabilistic timed automata into equivalent finite-state Markov decision processes (MDPs) remain correct in the parametric setting, using a systematic proof pattern. We implemented two of these parameter-preserving transformations—using digital clocks and backwards reachability—in the Modest Toolset. Using Storm's parameter space partitioning approach, parameter values can be efficiently synthesized in the resulting parametric MDPs. We use several case studies from the literature of varying state and parameter space sizes to experimentally evaluate the performance and scalability of this novel analysis trajectory for parametric PTA.

1 Introduction

Probabilistic timed automata (PTA) [25,45] combine the features of timed automata (TA) [2], to capture hard continuous real-time behaviour with nondeterministic time and choices, with those of Markov decision processes (MDP) [50], to model discrete random decisions. PTA are well-equipped for the study of randomised algorithms interacting with an environment where actions with uncertain outcomes complete after (upper- and lower-bounded) delays. They have been fruitfully applied to verify performance and reliability aspects of communication protocols and networked systems, see e.g. [22–24,39].

Building a PTA model requires knowledge of the precise probabilities of all random events. While unproblematic for a randomised algorithm such as the binary exponential backoff procedure in a CSMA/CA wireless network by itself, uncertainty about the operating environment often means that we do not know, say, the precise probability p of message loss once we decide to send. In such cases, we may turn the verification question around: Instead of computing whether the probability of an eventual successful transmission is above the required threshold

This work was funded by DFG RTG 2236 "UnRAVeL", DFG grant 433044889 PASIWY, NWO grant OCENW.KLEIN.311, and NWO VENI grant 639.021.754.

A. Abate and A. Marin (Eds.): QEST 2021, LNCS 12846, pp. 39–58, 2021.
https://doi.org/10.1007/978-3-030-85172-9_3

given a concrete p, we determine the set of values of p for which the requirement is satisfied. We can then judge whether the network's setup or protocols are sufficiently robust for the intended environments.

In this paper, we focus on *parametric PTA* (pPTA), where the probabilities of some events are unknown and specified as polynomials over a finite set of parameters like p above. We consider the analysis of (time-bounded) probabilistic reachability properties, i.e. statements of the form "is the maximum/minimum probability of eventually/within t time units reaching a goal location above/below $v \in (0,1)$?", but instead of a yes/no answer we want to compute an (approximation of) the set of parameter valuations under which the property is satisfied.

Several approaches have been developed over the past two decades to verify PTA with known probabilities. Most approaches compute a finite abstraction of the continuous-time behaviour, turning the PTA into an equivalent MDP on which the property of interest can be verified using standard MDP model checking [10,11]. This includes using the region graph [45], backwards reachability [46], and digital clocks [34,44]. The latter two are implemented in the Prism [43] tool while the Modest Toolset's [28] mcsta model checker uses digital clocks. The stochastic games approach [42], implemented in Prism, employs games in place of MDP to iteratively refine the abstraction up to the desired precision. Uppaal smc [21] applies statistical model checking (SMC) [1] to possibly non-deterministic PTA by interpreting nondeterminism probabilistically. It thus delivers *some* probability between minimum and maximum. Using extensions of lightweight scheduler sampling [47] to PTA [20,31], the Modest Toolset's modes simulator [15] can deliver upper (lower) bounds on min. (max.) via SMC.

Our contribution is to **extend the reach of model checking to pPTA**: we lift the PTA-to-MDP abstraction techniques to pPTA, then apply *parameter space partitioning* [14,51] on the resulting MDP to approximate the set of satisfying valuations. We **prove** that the abstractions remain correct in the parametric setting (Sect. 3), provide an **implement**ation using digital clocks and backwards reachability in the Modest Toolset followed by parameter space partitioning in Storm [33] (Sect. 4), and use it to experimentally **evaluate** the performance and scalability of our approach (Sect. 5). For the evaluation, we extend all suitable PTA from the Quantitative Verification Benchmark Set (QVBS) [30] with parameters, and additionally study pPTA models of the AODV wireless routing protocol [39]. The latter solves a critical open problem the previous study of the protocol, which had to resort to "testing" by model checking a finite set of selected values for probabilities that are actually unknown.

Related Work. [41] provide a tool to model-check interval PTA. An interval PTA is a special case of pPTA in which no parameter occurs at multiple states. We are not aware of any other work tackling the problem of model-checking pPTA. Instead of parametrising the *probabilities*, however, one may parametrise the *delays* [3]. Where the guard of an edge in a TA with clock c may be the clock constraint $c \geq 3$, it may be $c \geq 2 \cdot p$ in such a *constraint-parametric* TA (cpTA) with p a parameter. For this model, even basic problems like the existence of a parameter valuation under which a goal state is reachable are undecidable [3,4],

except in restricted cases such as bounded integer values [37] or L/U TA [35]. Research on cpTA remains active to this day; recent work, for example, proposes a semi-algorithm for liveness [5]. The *inverse method* for cpTA [6] extends one parameter valuation v_0 to a set of valuations $V \supseteq \{ v_0 \}$ such that all $v \in V$ satisfy the same given reachability properties. In its adaptation to cpPTA [8], all $v \in V$ must result in the same reachability probabilities. Parametric interval probabilistic timed automata [7] combine cpPTA with interval Markov chains [36], i.e. compared to cpPTA, the concrete probabilities are replaced by *intervals* of possible probabilities. In contrast to parametric probabilities, intervals cannot express dependencies between the probabilities of different events (such as one event being twice as likely as another). For this model, research currently remains focused on the question of consistency [7].

2 Preliminaries

We now introduce TA, PTA, and pPTA in order, highlighting the differences.

2.1 Timed Automata

A timed automaton [2] is a labelled transition system (LTS) [12] extended with a finite set $\mathscr{X}$ of *clocks* taking non-negative real values and all increasing at rate 1 over time. A *clock valuation* is a function $v\colon \mathscr{X} \to \mathbb{R}_{\geq 0}$. We denote the set of all clock valuations by $\mathbb{R}_{\geq 0}^{\mathscr{X}}$. For $v \in \mathbb{R}_{\geq 0}^{\mathscr{X}}$ and $t \in \mathbb{R}_{\geq 0}$, the valuation $v + t$ is defined by $(v + t)(x) = v(x) + t$ for all $x \in \mathscr{X}$. For $X \subseteq \mathscr{X}$, $v[X := 0]$ is the clock valuation where $v[X := 0](x) = 0$ if $x \in X$ and $v[X := 0](x) = v(x)$ otherwise. Finally, $\mathbf{0}$ is the zero valuation, i.e. $\mathbf{0}(x) = 0$ for all $x \in \mathscr{X}$. The set $CC(\mathscr{X})$ of *clock constraints* over $\mathscr{X}$ contains all expressions defined by

$$\mathcal{X} ::= x < c \mid x \leq c \mid x > c \mid x \geq c \mid \mathcal{X} \wedge \mathcal{X}$$

where $x \in \mathscr{X}$ and $c \in \mathbb{N}$. To keep the presentation simple, w.l.o.g. we omit diagonal and disjunctive clock constraints. Valuation v satisfies clock constraint $\mathcal{X}$, denoted $v \models \mathcal{X}$, iff the expression $\mathcal{X}$ evaluates to true after replacing each $x \in \mathscr{X}$ with $v(x)$. The semantics of clock constraint $\mathcal{X}$ is the *zone* $\zeta_{\mathcal{X}} := \{v \in \mathbb{R}_{\geq 0}^{\mathscr{X}} \mid v \models \mathcal{X}\}$. Let $Zones(\mathscr{X})$ denote the set of zones over the constraints $\mathscr{X}$.

Definition 1. *A* ***timed automaton*** *(TA) is a tuple* $\mathcal{B} = (Loc, \ell_0, Act, \mathscr{X}, \hookrightarrow, inv)$ *where Loc, Act and* $\mathscr{X}$ *are finite sets of* locations, actions, *and clocks, respectively, with initial location* $\ell_0 \in Loc$*, the transition relation is*

$$\hookrightarrow \subseteq Loc \times CC(\mathscr{X}) \times Act \times 2^{\mathscr{X}} \times Loc,$$

and $inv\colon Loc \to CC(\mathscr{X})$ *assigns an* invariant *to each location.*

An edge from ℓ to ℓ' is a tuple $(\ell, g, a, X, \ell') \in \hookrightarrow$ where guard g must be satisfied in order to take the edge, and X contains the clocks to be reset. We assume that every edge is uniquely identified by its source location and action.

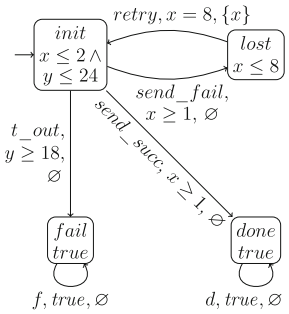

Fig. 1. TA $\mathcal{B}$

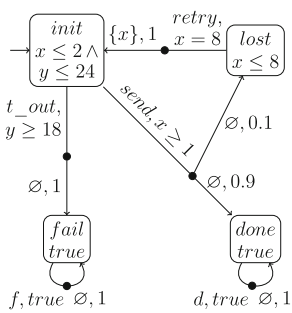

Fig. 2. PTA $\mathcal{A}$

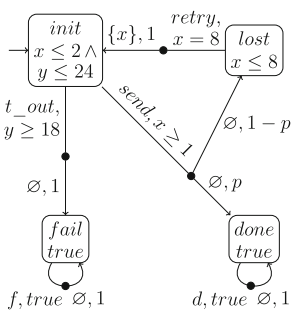

Fig. 3. pPTA $\mathcal{P}$

Example 1. Figure 1 shows TA $\mathcal{B}$ with four locations (labelled by the location name and its invariant) and two clocks x and y. Initially, the system is in location *init*. It can remain there as long as $x \leq 2 \wedge y \leq 24$; after one time unit, the edges to *done* and *lost* become enabled. In *lost*, the system remains as long as $x \leq 8$. The edge back to *init* is enabled when $x = 8$; when taken, clock x is reset. If 18 time units passed, and we are (still or again) in *init*, the edge to *fail* is enabled.

Formally, the semantics of a TA is defined in terms of a *timed* transition system where transitions are labelled with either an action or a time duration.

Definition 2. *The* ***semantics of a TA*** $\mathcal{B}$ *is the timed transition system* $[\![\mathcal{B}]\!]_{ts} = (S, s_0, Act', \rightarrow)$ *where* $S = \{(\ell, v) \in Loc \times \mathbb{R}_{\geq 0}^{\mathscr{X}} \mid v \models inv(\ell)\}$ *is the set of states with initial state* $s_0 = (\ell_0, \mathbf{0})$*, the action labels are in* $Act' = Act \cup \mathbb{R}_{\geq 0}$*, and the transition relation* $\rightarrow$ *is the smallest relation satisfying inference rules*

$$\frac{(\ell, g, a, X, \ell') \in \hookrightarrow \quad v \models g \quad v[X := 0] \models inv(\ell')}{(\ell, v) \xrightarrow{a} (\ell', v[X := 0])} \qquad \frac{v + d \models inv(\ell) \quad d \in \mathbb{R}_{\geq 0}}{(\ell, v) \xrightarrow{d} (\ell, v + d)}$$

We refer to transitions due to the left inference rule as *jumps* with action a and to the others as *delays* of duration d. Observe that $[\![\mathcal{B}]\!]_{ts}$ has uncountably many states in general. Infinite paths of $[\![\mathcal{B}]\!]_{ts}$ are of the form $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \cdots$ with the $s_i \in S$ and $a_i \in Act'$. Finite paths are defined similarly. Let $Paths^{\mathcal{B}}_{inf}$ ($Paths^{\mathcal{B}}_{fin}$) denote the set of all infinite (finite) paths of $[\![\mathcal{B}]\!]_{ts}$. W.l.o.g. we can assume that the first transition of a path is a delay and that delays and jumps alternate. $[\![\mathcal{B}]\!]_{ts}$ may contain non-divergent paths [45], i.e. paths along which infinitely many jumps happen in a finite amount of time.

Definition 3. *The path* $\pi \in Paths^{\mathcal{B}}_{inf}$ *is* ***divergent*** *if the sum of the durations of its delays—its* elapsed time*—is* ∞.

2.2 Probabilistic Timed Automata

A *probability distribution* over a countable set X is a function $\mu \colon X \to [0, 1] \subseteq \mathbb{R}$ with $\sum_{x \in X} \mu(x) = 1$. Let $Distr(X)$ denote the set of distributions on X. Probabilistic timed automata [25,45] extend TA with probabilistic transitions.

Definition 4. *A **probabilistic timed automaton** (PTA) is a tuple* $\mathcal{A} = (Loc, \ell_0, Act, \mathscr{X}, prob, inv)$ *as in Definition 1 except that* $prob\colon Loc \times CC(\mathscr{X}) \times Act \to Distr(2^{\mathscr{X}} \times Loc)$ *is the* probabilistic transition function.

If $prob(\ell, g, a)(X, \ell') > 0$ then (ℓ, g, a, X, ℓ') is an *edge* of the PTA.

Example 2. Consider TA $\mathcal{B}$ and PTA $\mathcal{A}$ from Fig. 2. Whereas $\mathcal{B}$ has two send edges (one for successful, one for failed transmissions), $\mathcal{A}$ has one probabilistic edge, where the probability of successful transmission is 0.9.

A TA is a PTA with probability 1 for all edges. We use MDP for their semantics:

Definition 5. *A **Markov decision process** (MDP)* $\mathcal{M}$ *is a timed transition system with probabilistic transition function* $prob\colon S \times Act' \to Distr(S)$.

If $prob(s, a, \mu)$ with $\mu(s') = p$, we write $prob(s, a, s') = p$. Finite and infinite paths for MDPs are defined as for timed transition systems, with the difference that for path $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \cdots$ we require that $prob(s_i, a_i, s_{i+1}) > 0$. To be able to reason about the probabilities of certain events in (timed) MDPs, we use *schedulers*. A scheduler maps finite paths to an available duration if the path has an even number of transitions and to an available action in Act otherwise.

Definition 6. *A (history-dependent) **scheduler** for a timed MDP* $\mathcal{M}$ *is a function* $\sigma\colon Paths^{\mathcal{M}}_{fin} \to (Act \uplus \mathbb{R}_{\geq 0})$ *where, for* $\widehat{\pi} = s_0 \xrightarrow{a_0} s_1 \ldots \xrightarrow{a_{n-1}} s_n$, *we have* $\sigma(\widehat{\pi}) \in Act$ *if* n *is odd and* $\sigma(\widehat{\pi}) \in \mathbb{R}_{\geq 0}$ *otherwise.*

Applying scheduler σ to MDP $\mathcal{M}$ yields the Markov chain $\mathcal{M}^\sigma$. For a fixed scheduler σ and state s, a probability measure $\Pr_s^{\mathcal{M}^\sigma}$ can be defined over the infinite paths starting in s induced by σ using the standard cylinder set construction (see e.g. [12, Ch. 10]). We restrict to *almost-sure divergent* schedulers, which are those schedulers σ where

$$\Pr_s^{\mathcal{M}^\sigma}\{\, \pi \in Paths^{\mathcal{M}\sigma}_{inf} \mid \pi \text{ is divergent} \,\} \;=\; 1.$$

Let $Sched_{div}(\mathcal{M})$ denote the set of almost-sure divergent schedulers of $\mathcal{M}$. The notions of (almost-sure divergent) schedulers can be lifted to PTA $\mathcal{A}$ in a straightforward manner by considering the corresponding notion on the *dense-time* semantics of a PTA, which is an uncountably large MDP:

Definition 7. *The **dense-time semantics of a PTA** $\mathcal{A}$ is the MDP*

$$[\![\mathcal{A}]\!]_{dense} = (S, s_0, Act', prob')$$

where S, s_0, *and* Act' *are as in Definition 2, and if* $prob(l, g, a, \cdot, \cdot)$ *is defined, then* $prob'((\ell, v), a, (\ell', v')) =$

$$\begin{cases} \displaystyle\sum_{\substack{X \subseteq \mathscr{X}\,:\\ v' = v[X:=0]}} prob(\ell, g, a, X, \ell') & \text{if } a \in Act \wedge v \models g \\ 1 & \text{if } a \in \mathbb{R}_{\geq 0} \wedge \ell = \ell' \wedge v' = v + a \models inv(\ell) \\ 0 & \text{otherwise.} \end{cases}$$

We assume PTA to be *well-formed*: if $prob'((\ell, v), a, (\ell', v')) > 0$ then $v' \models inv(\ell')$, for all reachable (ℓ, v). We lift the notion of a scheduler to PTA $\mathcal{A}$ using the above semantics, i.e. $Sched(\mathcal{A}) := Sched([\![\mathcal{A}]\!]_{dense})$ and $\mathcal{A}^\sigma := [\![\mathcal{A}]\!]^\sigma_{dense}$.

2.3 Parametric Probabilistic Timed Automata

Let V be a set of n real-valued *parameters* (or *variables*) $p_1, \ldots, p_n$. Let $\mathbb{Q}(V)$ denote the set of multivariate polynomials over V. We write $p \in f$ if parameter p occurs in polynomial f. A *parameter instantiation* is a function $u\colon V \to \mathbb{R}$. The *parameter space* for V is the hyper-rectangle over the lower/upper bounds for the parameter in V. A polynomial f can be interpreted as a function $f\colon \mathbb{R}^n \to \mathbb{R}$ where $f(u)$ is obtained by substitution, i.e. in $f(u)$ each occurrence of p_i in f is replaced by $u(p_i)$. To make it clear where substitution occurs, we write $f[u]$ instead of $f(u)$ from now on. Let $Distr_{\mathbb{Q}(V)}(X)$ denote the set of *parametric probability distributions* over X induced by the polynomials in $\mathbb{Q}(V)$, i.e. of functions $\mu\colon X \to \mathbb{Q}(V)$ such that $\sum_{x \in X} \mu(x)[u] = 1$ for all u within the parameter space. A *parametric* PTA is a PTA in which the transitions use parametric distributions to select the clocks to reset and the successor location.

Definition 8. *A **parametric PTA** (pPTA) is a tuple $\mathcal{P} = (Loc, \ell_0, Act, \mathscr{X}, V, prob, inv)$ where $Loc, \ell_0, Act, \mathscr{X}$ and inv are as in Definition 1, V is a finite set of parameters, and the parametric probabilistic transition function is*

$$prob\colon Loc \times CC(\mathscr{X}) \times Act \to Distr_{\mathbb{Q}(V)}(2^{\mathscr{X}} \times Loc).$$

Applying a parameter instantiation u to the pPTA $\mathcal{P}$ yields the PTA $\mathcal{P}[u]$ by replacing each $f \in \mathbb{Q}(V)$ in the function $prob$ of $\mathcal{P}$ by $f[u]$; we write the resulting function as $prob[u]$. Observe that $prob[u](\ell, g, a, X, \ell') = prob(\ell, g, a, X, \ell')[u]$ for all $\ell, \ell' \in Loc$, guard g, $a \in Act$, $X \subseteq \mathscr{X}$.

Example 3. Consider pPTA $\mathcal{P}$ in Fig. 3 where the probability of successful transmission is p. Applying instantiation $u = \{\, p \mapsto 0.9 \,\}$ to $\mathcal{P}$ yields PTA $\mathcal{A}$ of Fig. 2.

A *region* $R \subseteq \mathbb{R}^n$ is a fragment of the parameter space[1]. A region R is *graph-preserving* for pPTA $\mathcal{P}$ if for all $u \in R$ and polynomials f in $prob$ of $\mathcal{P}$ $f[u] > 0$, that is, none of the edges contains a probability 0. A region is thus graph-preserving if all its valuations induce the same topology. We define parametric MDPs as an extension of MDPs and use them for the semantics of pPTA.

Example 4. Consider pPTA $\mathcal{P}$ again. Region $R = [0.1, 0.9]$ is graph-preserving while $[0, 0.9]$ is not. For both, all probability distributions in $\mathcal{P}$ are well-defined.

Definition 9. *A **parametric MDP** (pMDP) $\mathcal{M}$ is an MDP over a finite set V of variables, and $prob\colon S \times Act' \to Distr_{\mathbb{Q}(V)}(S)$.*

[1] Parameter regions should not be confused with the regions of clock valuations as in the classic region graph construction for a (P)TA.

Definition 10. *The **semantics of a pPTA** $\mathcal{P}$ over V is the pMDP $[\![\mathcal{P}]\!]_{dense}$ over V defined analogously to Definition 7.*

The definition of $prob'$ includes sums of polynomials, which remain polynomials. Since the transition functions are equivalent, i.e. we have $prob'[u] = prob''$ for $[\![\mathcal{P}]\!]_{dense}[u] = (S, s_0, Act, V, prob'[u])$ and $[\![\mathcal{P}[u]]\!]_{dense} = (S, s_0, Act, V, prob'')$, parameter instantiation and the dense-time semantics commute.

Properties. We consider time-bounded and unbounded reachability properties on PTAs, i.e. the probability of eventually or within a time bound reaching a target set $T \subseteq Loc \times Zones(\mathscr{X})$. Specification $\varphi = \mathbb{P}_{\leq\lambda}(\Diamond T)$ asserts that the probability of eventually reaching T from the initial state $(\ell_0, \mathbf{0})$ is at most λ, where $\lambda \in \mathbb{Q} \cap [0, 1]$. That is, for PTA $\mathcal{A}$,

$$\mathcal{A} \models \mathbb{P}_{\sim\lambda}(\Diamond T) \quad \text{iff} \quad \text{for all } \sigma \in Sched(\mathcal{A}) \text{ we have } \mathsf{Pr}^{\mathcal{A}^\sigma}(\Diamond T) \sim \lambda,$$

where $\mathsf{Pr}^{\mathcal{A}^\sigma}(\Diamond T)$ is the probability mass of all infinite paths in $\mathcal{A}$ that start in $(\ell_0, \mathbf{0})$ and visit some pair $(\ell, \zeta) \in T$. To support time-bounded reachability, we assume that each PTA $\mathcal{A}$ has a clock $z \in \mathscr{X}$, which does not occur in any invariant or guard of $\mathcal{A}$ and is never reset. We then check that z does not exceed bound $n \in \mathbb{N}$. We reduce this to the unbounded case and focus on that in the remainder of this paper:

$$\mathcal{A} \models \mathbb{P}_{\sim\lambda}(\Diamond^{\leq n} T) \quad \text{iff} \quad \mathcal{A} \models \mathbb{P}_{\sim\lambda}(\Diamond T^{\leq n})$$

where $T^{\leq n} := \{(\ell, \zeta \cap \zeta_{z\leq n}) \mid (\ell, \zeta) \in T\}$. The definition for strict bounds $<$ is analogous. Negation is defined by

$$\mathcal{A} \models \neg\, \mathbb{P}_{\sim\lambda}(\Diamond T) \quad \text{iff} \quad \mathcal{A} \models \mathbb{P}_{(\neg\sim)\lambda}(\Diamond T)$$

and similar for $\mathbb{P}_{\sim\lambda}(\Diamond^{\leq n} T)$. These notions are lifted to pPTA by considering properties relative to ranges of parameter values, i.e. regions.

Definition 11. *Given region R and pPTA $\mathcal{P}$, let*

$$\mathcal{P}, R \models \mathbb{P}_{\sim\lambda}(\Diamond T) \quad \textit{iff} \quad \forall \sigma \in Sched(\mathcal{P}[u]),\ u \in R.\ \mathsf{Pr}^{\mathcal{P}[u]^\sigma}(\Diamond T) \sim \lambda \tag{1}$$

$$\mathcal{P}, R \models \neg\mathbb{P}_{\sim\lambda}(\Diamond T) \quad \textit{iff} \quad \forall \sigma \in Sched(\mathcal{P}[u]),\ u \in R.\ \mathsf{Pr}^{\mathcal{P}[u]^\sigma}(\Diamond T)\,(\neg\sim)\,\lambda. \tag{2}$$

We call a region accepting (rejecting), denoted R_a (R_r), if 1 (2) holds. If a region is neither accepting nor rejecting, it is called inconsistent (denoted R_i).

Example 5. Reconsider $\mathcal{P}$ from Fig. 3. Let $\varphi = \mathbb{P}_{\geq 0.75}(\Diamond\{(done, \zeta_{true})\})$. Region $R_r := [0.2, 0.4]$ is rejecting; $R_a := [0.6, 0.8]$ is accepting, and $R_i := [0.4, 0.6]$ is inconsistent, as property φ is violated for $p = 0.4$ but satisfied for $p = 0.6$.

Furthermore, we define the minimal probability in pPTA $\mathcal{P}$ of eventually reaching a state in T on region R as follows:

$$\mathsf{Pr}^{\mathcal{P},R}_{min}(\Diamond T) = \min\{\mathsf{Pr}^{\mathcal{P}[u]^\sigma}(\Diamond T) \mid \forall u \in R, \sigma \in Sched(\mathcal{P})\}.$$

The maximal probability is defined analogously. The definition can be applied to PTA $\mathcal{A}$ with any region, as there are no parameters; we then omit R: $\mathsf{Pr}^{\mathcal{A}}_{min}(\Diamond T)$.

2.4 Problem Statement

This paper focuses on parameter space partitioning [14,51] for pPTA. The key idea is to partition a graph-preserving parameter space into *accepting* and *rejecting* regions w.r.t. a property φ. As obtaining a complete partitioning is practically infeasible, the aim is to cover at least $c\,\%$ of the parameter space.

Approximate synthesis problem for pPTA. Given pPTA $\mathcal{P}$, specification φ, percentage c and region R, partition R into regions R_a, R_r, and R_i such that $\mathcal{P}, R_a \models \varphi$ and $\mathcal{P}, R_r \models \neg\varphi$ where $R_a \cup R_r$ covers at least $c\,\%$ of R.

To solve this problem, we consider techniques to obtain a finite-state pMDP for the semantics of the pPTA in the next section. We then solve the approximate synthesis on the resulting pMDP. Note that this is a computationally hard problem: finding parameter values for a pMDP that satisfy a reachability property is ETR-complete [38][2]. To check whether a region is accepting or rejecting, we apply parameter lifting [51] on this pMDP. Parameter lifting first drops all dependencies between parameters in a pMDP. It then transforms the pMDP into a 2-player stochastic game to obtain upper and lower bounds for the given property φ. It applies to finite-state pMDPs and graph-preserving regions. In Sect. 5 we experimentally show this approach's feasibility.

3 PPTA to pMDP Methods

The main question is now whether existing techniques that verify PTA by obtaining finite-state MDPs carry over to the parametric setting. We will answer this question affirmatively for the digital clocks and backwards reachability techniques. The correctness criterion is, as we will show, that they preserve reachability probabilities as defined on the dense-time semantics. The presentation below is along the lines of [49] adapted to the case with parameters.

3.1 Digital Clocks

The digital clocks approach for TA [9,34] and its adaptation to PTA [44] only consider integer clock valuations, i.e. valuations in $\mathbb{N}^{\mathscr{X}}$, and delays of 1 time unit. By capping the clock valuation for clock x to $\mathbf{k}_x + 1$, where $\mathbf{k}_x$ is the maximal constant to which x is compared in the PTA, digital clocks give rise to a *finite* MDP. To that end, let $(v \oplus t)(x) = \min\{v(x) + t, \mathbf{k}_x + 1\}$ for each $x \in \mathscr{X}$. The digital clock approach requires the PTA to be *closed*, i.e. all clock constraints must only contain non-strict comparisons such as $x \leq c$ and $x \geq c$.

Definition 12. *The **digital clocks semantics** of a closed pPTA $\mathcal{P}$ is the pMDP $[\![\mathcal{P}]\!]_{dc} = (S, s_0, Act', V, prob')$ with $S = \{\, (\ell, v) \in Loc \times \mathbb{N}^{\mathscr{X}} \mid v \models inv(\ell) \,\}$ and s_0, Act', and $prob'$ are as in Definition 7 (restricted to S), except that for time delays we use*

$$prob'((\ell, v), a, (\ell', v')) = 1 \quad \text{if } a = 1 \wedge \ell = \ell' \wedge v' = v \oplus 1 \models inv(\ell').$$

[2] Existential Theory of the Reals. ETR problems are between NP and PSPACE, and ETR-hard problems are as hard as finding the roots of a multi-variate polynomial.

Correctness. To prove the correctness of the digital clocks semantics for pPTA we first show that parameter instantiation and digital clocks semantics commute.

Lemma 1. *For pPTA $\mathcal{P}$ and parameter valuation u: $[\![\mathcal{P}[u]]\!]_{dc} = [\![\mathcal{P}]\!]_{dc}[u]$.*

Proof. By Definition 12, we need to prove that the transition functions are equivalent, i.e. $prob'((\ell, v), a, (\ell', v'))[u] = prob'[u]((\ell, v), a, (\ell', v'))$. From Definition 7 this follows for all cases except $a = 1 \wedge \ell = \ell' \wedge v' = v \oplus 1 \models inv(\ell')$. For this case, $prob'((\ell, v), a, (\ell', v')) = 1$. As no parameters occur in this case, equivalence follows trivially. Therefore the transition functions are equivalent. □

Furthermore, similar as for PTA, the following lemma holds:

Lemma 2. *For any closed pPTA $\mathcal{P}$, closed target T, and region R, we have*

$$\mathsf{Pr}^{\mathcal{P},R}_{min}(\Diamond T) \;=\; \mathsf{Pr}^{[\![\mathcal{P}]\!]_{dc},R}_{min}(\Diamond T) \quad \textit{and} \quad \mathsf{Pr}^{\mathcal{P},R}_{max}(\Diamond T) \;=\; \mathsf{Pr}^{[\![\mathcal{P}]\!]_{dc},R}_{max}(\Diamond T).$$

Proof. For minimal reachability, take an arbitrary but fixed $u \in R$. Then $\mathcal{P}[u]$ is a PTA. Thus, $\mathsf{Pr}^{\mathcal{P}[u]}_{min}(\Diamond T) = \mathsf{Pr}^{[\![\mathcal{P}[u]]\!]_{dc}}_{min}(\Diamond T)$ [44]. Using Lemma 1, we conclude $\mathsf{Pr}^{\mathcal{P}[u]}_{min}(\Diamond T) = \mathsf{Pr}^{[\![\mathcal{P}]\!]_{dc}[u]}_{min}(\Diamond T)$. Maximal reachability is proven analogously.

This yields the following result on preserving rejecting and accepting regions:

Theorem 1 (correctness of digital clocks). *For region R, closed target T and closed pPTA $\mathcal{P}$:*

$$\begin{aligned} \mathcal{P}, R \models \mathbb{P}_{\sim\lambda}(\Diamond T) \quad &\iff \quad [\![\mathcal{P}]\!]_{dc}, R \models \mathbb{P}_{\sim\lambda}(\Diamond T). \\ \mathcal{P}, R \models \neg\mathbb{P}_{\sim\lambda}(\Diamond T) \quad &\iff \quad [\![\mathcal{P}]\!]_{dc}, R \models \neg\mathbb{P}_{\sim\lambda}(\Diamond T) \end{aligned}$$

Proof. We show the case of preserving an accepting region for $\lesssim \in \{<, \leq\}$:

$$\begin{aligned} \mathcal{P}, R \models \mathbb{P}_{\lesssim\lambda}(\Diamond T) \overset{\text{Def. } 11}{\Leftrightarrow} \quad & \forall u \in R.\ \mathcal{P}[u] \models \mathbb{P}_{\lesssim\lambda}(\Diamond T) \\ \Leftrightarrow \quad & \mathsf{Pr}^{\mathcal{P},R}_{max}(\Diamond T) \lesssim \lambda \\ \overset{\text{Lem. } 2}{\Leftrightarrow} \quad & \mathsf{Pr}^{[\![\mathcal{P}]\!]_{dc},R}_{max}(\Diamond T) \lesssim \lambda \\ \Leftrightarrow \quad & \forall u \in R.\ [\![\mathcal{P}]\!]_{dc}[u] \models \mathbb{P}_{\lesssim\lambda}(\Diamond T) \\ \overset{\text{Def. } 11}{\Leftrightarrow} \quad & [\![\mathcal{P}]\!]_{dc}, R \models \mathbb{P}_{\lesssim\lambda}(\Diamond T). \end{aligned}$$

The proofs for preserving rejecting regions for $\lesssim$ and accepting/rejecting regions for $\gtrsim \in \{>, \geq\}$ are analogous using $\mathsf{Pr}^{\mathcal{P},R}_{min}(\Diamond T)$ rather than $\mathsf{Pr}^{\mathcal{P},R}_{max}(\Diamond T)$.

From Theorem 1 it follows that accepting/rejecting regions are preserved under the digital clocks semantics. Therefore, the inconsistent regions are preserved, too. Note that region R does not need to be closed. This is only necessary for clock constraints, and they are not influenced by parameters in our setting.

Complexity. An upper bound on the number of states in the digital clocks semantics is $|Loc| \cdot \prod_{x \in \mathscr{X}} (\mathbf{k}_x{+}1)$. This means that the runtime for parameter region verification as used in Theorem 1 is exponential in the number of clocks. In Sect. 5, we will report on the practical feasibility of digital clocks for pPTA.

3.2 Backwards Reachability

To tackle the state space explosion of digital clocks, we consider backwards reachability [46]. Instead of using explicit states (i.e. pairs of locations and valuations), it computes a finite set of *symbolic states*—all that can reach the target T.

As in [46], a *symbolic state* is a pair $(\ell, \zeta) \in Loc \times Zones(\mathscr{X})$. For a set of symbolic states $\mathtt{U} \subseteq Loc \times Zones(\mathscr{X})$, let $\zeta^\ell_\mathtt{U} = \bigcup\{\zeta \mid (\ell, \zeta) \in \mathtt{U}\}$ be all zones in $\mathtt{U}$ that are paired with location ℓ. To determine the reachable symbolic states, we use time ($tpre_\mathtt{U}$) and discrete ($dpre$) predecessor operations. Let $\mathtt{V}$ be the set of symbolic states explored so far; initially $\mathtt{V} = \{T\}$. Then $tpre_\mathtt{U}$ determines the symbolic states that can reach a state in $\mathtt{V}$ by delays, all the while staying in $\mathtt{U}$; $dpre$ are those that can do so via jumps.

Definition 13. *Given a pPTA $\mathcal{P} = (Loc, \ell_0, Act, \mathscr{X}, V, prob, inv)$, sets of symbolic states $\mathtt{U}$ and $\mathtt{V}$, let:*

$$tpre_\mathtt{U}(\mathtt{V}) := \{(\ell, \swarrow_{\zeta^\ell_\mathtt{U} \cap \zeta_{inv(\ell)}} (\zeta^\ell_\mathtt{V}) \cap \zeta_{inv(\ell)})\}$$

where $\swarrow_{\zeta'}(\zeta) := \{v \mid \exists t \geq 0.\ (v + t \models \zeta \wedge \forall t' \leq t.\ (v + t' \models \zeta \cup \zeta'))\}$ which denotes the zone that can reach ζ by delays while staying in ζ'.

The function $dpre$ is adopted from [46] and is omitted here. Backwards reachability iteratively applies $tpre_\mathtt{U}$ and $dpre$ to $\mathtt{V}$ until it reaches a fixed point. It returns an initial symbolic state $\mathtt{z}_0$, a set $\mathtt{Z}$ of symbolic states, and a probability function $prob'$ on symbolic states that is based on the probability function $prob$ of $\mathcal{P}$. For more details, we refer the interested reader to the $MaxU$ algorithm in [46]. Most important for us is the fact that we input a pPTA and a set of symbolic target states and that it returns a pMDP.

Definition 14. *For the initial symbolic state $\mathtt{z}_0$, set $\mathtt{Z}$ of symbolic states and probability function $prob'$, the **backwards reachability semantics** of $\mathcal{P} = (Loc, \ell_0, Act, \mathscr{X}, V, prob, inv)$ is the pMDP $[\![\mathcal{P}]\!]_{br(T)} = (\mathtt{Z}, \mathtt{z}_0, Act, V, prob')$.*

Correctness. To prove the correctness of the backwards reachability semantics of a pPTA, we first show that parameter instantiation and the semantics commute similarly to the case for digital clocks.

Lemma 3. *For pPTA $\mathcal{P}$, target T, and valuation u: $[\![\mathcal{P}]\!]_{br(T)}[u] = [\![\mathcal{P}[u]]\!]_{br(T)}$.*

The proof is analogous to the proof of Lemma 1.

Let $T^\mathcal{C} = Loc \times Zones(\mathscr{X}) \backslash T$ and let A_T be the set of symbolic states from which there exists a scheduler that almost surely never reaches T, then:

Lemma 4. *For pPTA $\mathcal{P}$, region R, and target T, with $[\![true]\!] = Loc \times Zones(\mathscr{X})$:*

$$\begin{aligned} \mathsf{Pr}^{\mathcal{P},R}_{max}(\Diamond T) &= \mathsf{Pr}^{[\![\mathcal{P}]\!]_{br(T)},R}_{max}(\Diamond tpre_{[\![true]\!]}(T)). \\ \mathsf{Pr}^{\mathcal{P},R}_{min}(\Diamond T) &= 1 - \mathsf{Pr}^{[\![\mathcal{P}]\!]_{br(T)},R}_{max}(\Diamond tpre_{T^c}(A_T)). \end{aligned}$$

The proof for maximal probabilities is analogous to that of Lemma 2, using Lemma 3 and that $\mathsf{Pr}^{\mathcal{A}}_{max}(\Diamond T) = \mathsf{Pr}^{[\![\mathcal{A}]\!]_{br(T)}}_{max}(\Diamond tpre_{[\![true]\!]}(T))$ for PTA $\mathcal{A}$ [46]. The proof cannot generally be applied to minimal probabilities since the backwards reachability semantics only preserves upper-bounded properties. Therefore, we have to convert lower-bounded properties to such. The idea behind the conversion is the following: Instead of calculating whether we reach T, we calculate the opposite, i.e. whether we never reach T. This is achieved by ending up in A_T. However, before reaching A_T, we must not visit T beforehand. This is encoded in the semantics by $MaxU$; paths through T cannot reach A_T. For the correctness of this conversion we refer to [40,46].

Theorem 2 (correctness of backwards reachability). *Given a region R, target T, and pPTA $\mathcal{P}$, we have*

$$\mathcal{P}, R \models \mathbb{P}_{\lesssim\lambda}(\Diamond T) \iff [\![\mathcal{P}]\!]_{br(T)}, R \models \mathbb{P}_{\lesssim\lambda}(\Diamond tpre_{[\![true]\!]}(T)).$$
$$\mathcal{P}, R \models \mathbb{P}_{\gtrsim\lambda}(\Diamond T) \iff [\![\mathcal{P}]\!]_{br(T)}, R \models \mathbb{P}_{\lesssim 1-\lambda}(\Diamond tpre_{T^c}(A_T))$$

The proof is analogous to that of Theorem 1 using Lemma 4 instead of Lemma 2.

From Theorem 2 it follows that accepting regions are preserved under the backwards reachability semantics. The proof for rejecting regions is analogous. Therefore, the inconsistent regions are preserved, too.

Complexity. In the worst case, running the algorithm on a PTA $\mathcal{P}$ generates an MDP in which the set of symbolic states is the set $Loc \times Zones(\mathscr{X})$, which is doubly exponential in the number of clocks for PTA [46]. This is the same for pPTA as parameters do not affect the size of the result. However, case studies have shown that for PTA the state space is significantly smaller than the worst case [46]. It is claimed that the algorithm is feasible for most practical applications. In Sect. 5, we will report on the practical feasibility of backwards reachability for pPTA.

3.3 Other Methods

Digital clocks are only compatible with a limited class of pPTAs and backwards reachability only calculates reachability properties. However, other methods are established for model checking PTAs that do not restrict the PTA and properties. We briefly discuss whether these techniques can be applied to pPTA.

Region Graph. In the *region graph* [45], clock equivalence classes are considered. All clock valuations that satisfy the same constraints are grouped and used to build a *clock region*. This is equivalent to the symbolic states of backwards reachability, where the clock regions are minimal in the most basic variant. This leads to a relatively large state space, although this problem is tackled by other variants, like probabilistic time-abstract bisimulation [17]. The algorithm is applicable to pPTA and the proof is similar to that of digital clocks and

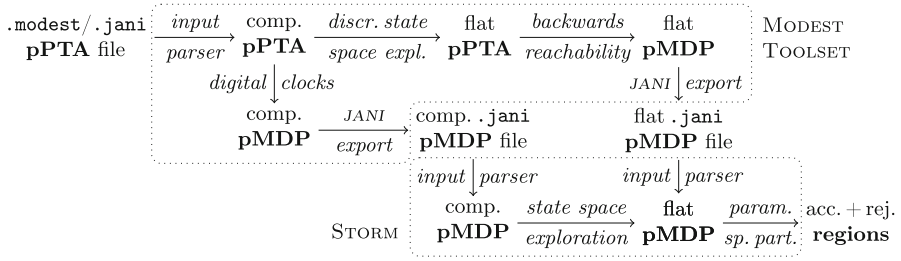

Fig. 4. Our toolchain for pPTA parameter space partitioning

backwards reachability: The substitution of parameters before/after obtaining the region graph semantics is equivalent and minimal/maximal reachability is preserved, which means accepting and rejecting regions are preserved.

Forwards Reachability. As for backwards reachability, *forwards reachability* [45] only considers relevant symbolic states. However, this method performs a forward search from the initial state to the target states. The complexity of forwards reachability is exponential in the number of clocks. However, the forwards algorithm is generally faster than its backwards equivalent [46], but it only provides upper (lower) bounds on the maximal (minimal) probability [22]. This makes forward reachability unsuited for parameter synthesis as regions may be falsely classified as accepting/rejecting. For example, we might have a region that is inconsistent for some upper bounded property. This means that there are both valuations that satisfy and violate the property. As forwards reachability gives bounds on the probability, it might push this probability beyond the bound of the property, resulting in a rejecting region where this is not the case.

Stochastic Games. The stochastic game abstraction [42] transforms the PTA into a 2-player stochastic game. It is usually faster than both digital clocks and backwards reachability in practice [42]. We conjecture that the method can be directly applied on pPTA, resulting in *parametric stochastic games*. However, as parameter synthesis is not implemented for this type of model in STORM or other tools, we would currently not be able to obtain an implementation in the same manner as for the MDP-based approaches.

4 Implementation

We implemented a parameter synthesis pipeline for pPTA by combining the MODEST TOOLSET and the STORM model checker as outlined in Fig. 4. The former has long had support for PTA model checking via digital clocks [27]. Since digital clocks are a syntax-level transformation, the toolset's MOCONV converter can turn closed PTA models specified in the MODEST modelling language [13,26] or the JANI model interchange format [16] into digital clocks MDP, in either of

these languages. The implementation is agnostic to the presence of parameters and thus readily works for pPTA, too, exporting pMDP models. Models are usually specified *compositionally* as a network of multiple pPTA extended with discrete (Boolean and bounded integer) variables. The conversion preserves this high-level structure without a blowup in (syntactical) model size. By exporting to Jani, the compositional pMDP can be read back by Storm, which implements parameter space partitioning to deliver the desired set of regions. The actual state space exploration—flattening the composition of variable-extended pMDP into one large pMDP—is performed by Storm in this setup.

For this work, we added an implementation of backwards reachability for maximum probabilities to the Modest Toolset for PTA and pPTA. As shown in the upper branch in Fig. 4, we first turn the compositional pPTA into a single automaton where only clock variables remain. To this we apply backwards reachability using our own implementation of difference-bound matrices. The resulting flat pMDP is then exported to a Jani file, which is usually much larger than the original input. Again, this file can be subsequently be read and regions computed via Storm. For minimum probabilities, backwards reachability would generate non-convex zones, for which we do not have an implementation yet.

5 Evaluation

To evaluate the feasibility of our approach as well as the relative scalability and performance of the two methods, we performed an experimental comparison using our implementation on parametric adaptations of existing PTA benchmarks as well as of the industrial AODV case study.

Benchmarks. As we provide the first tool support for pPTA, there are no existing pPTA benchmarks for us to use. We thus turned existing PTA models into pPTA. Among the 9 PTA models in QVBS [30], 5 could be parametrised in a sensible way: those of the bounded retransmission protocol (BRP) [27,32], the repudiation protocol with honest (RH) and malicious receiver (RM) [48], the Zeroconf protocol (ZC) [18], and the IEEE 1394 Firewire protocol (FW) [54]. We consider two variants of BRP: one with equal and one with different loss probabilities for the data and acknowledgement channels. Furthermore, we vary the constants used in the clock constraints to obtain one "small" (S) and one "large" (L) variant of BRP. In RH, we make the probability that a certain message is the last one parametric, and in RM additionally the probability to decode a message. The original repudiation pPTA are not closed and thus only suitable for backwards reachability. We created non-strict (nstr in Fig. 7) variants of these

Table 1. Model characteristics

model	params	clocks	max. $\mathbf{k}_x$
AODV(-n)	1-5	2	4
BRP	1-2	4	16-1657
RM	3	2	5
RH	1	2	5
ZC	1	2	20
FW	1	1	1670

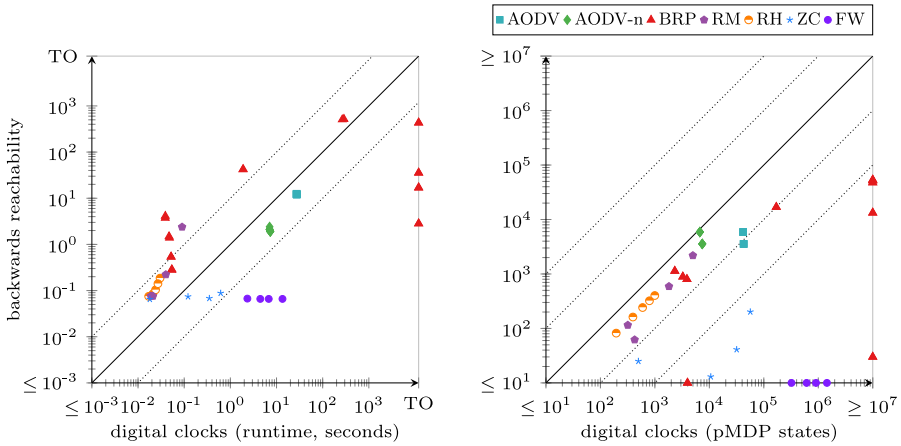

Fig. 5. Runtime up to state space exploration (left) and number of states (right)

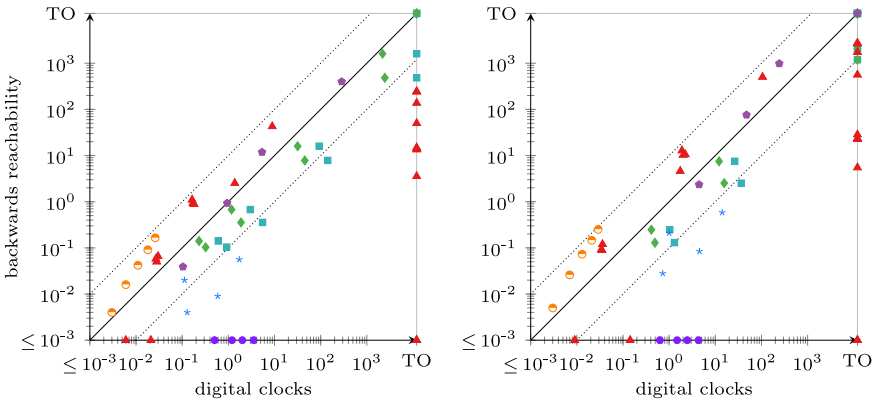

Fig. 6. Parameter space partitioning time (s), 90% (left), 99% (right) coverage

models to enable a comparison. We use two parameters in ZC: the probability to correctly receive a message and the probability that an occupied IP-address is selected. In FW, the parameter is the probability to select between the slow and the fast path.

Additionally, we consider the probabilistic version of the AODV routing protocol [39], which so far had been analysed for selected concrete message loss probabilities only. It comes in two variants, one with routing error (AODV) and one without (AODV-n). For each node, we make the probability to lose an incoming message parametric. By using the same parameter across different subsets of nodes, excluding symmetric cases, we obtain 5 models with 1 (all probabilities are p vs. $1-p$) to 5 (every node has its own parameter p_i) parameters. Table 1 summarises the characteristics of our benchmarks.

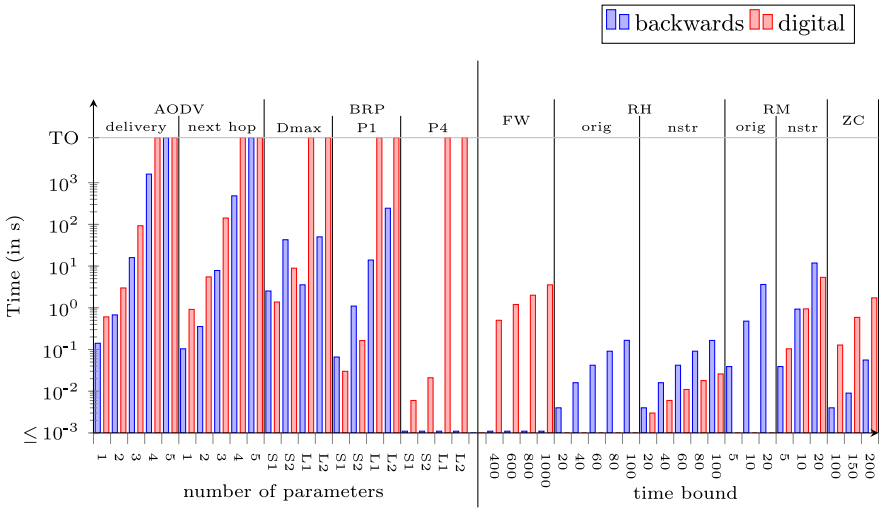

Fig. 7. Runtime for parameter space partitioning with 90% coverage

Setup. We did all experiments on an Intel Core i5-8300H (2.3–4.0 GHz) system with 8 GB of memory running 64-bit Ubuntu 20.04. The timeout was one hour.

Results. In Fig. 5, we compare digital clocks and backwards reachability in terms of the runtime (left) without the parameter space partitioning phase and the number of states of the pMDP (right). Note that for digital clocks, the runtime for the syntactical conversion is negligible. In these scatter plots, a point (x, y) indicates that digital clocks took x seconds or caused x states compared to y seconds/states for backwards reachability, for one specific combination of benchmark, variant (where applicable), and property to check. The dotted lines indicate differences of a factor of 10 and 100. Figure 6 similarly shows the runtime for parameter space partitioning for 90 and 99% coverage.

We observe that digital clocks generate larger state spaces than backwards reachability, just like it does for PTA [46]. We note that the number of transitions per state, however, is often larger with backwards reachability. Ultimately, performance also depends on the topology of the state space, and backwards reachability needs to perform a sometimes expensive symbolic reachability computation, explaining why digital clocks still often manage to be faster as far as obtaining the flat pMDP is concerned (Fig. 5 left). In Fig. 6, however, we can see that backwards reachability is mostly faster in the partitioning phase.

Figure 7 plots the partitioning runtime for those benchmarks where we can vary the number of parameters or the property time bound. On top, we indicate the respective benchmark and property being checked (in case multiple are available). We observe an exponential increase in runtime as we increase the number of parameters for AODV; AODV-n showed similar behaviour. A similar pattern occurs for BRP. Runtime increases mostly linearly with the time bound.

Another important observation is the difference between digital clocks and backwards reachability for property P4 of BRP with 2 parameters. Not only did backwards reachability produce a state space that is orders of magnitude smaller, but it was also able to completely *remove* a parameter by realising that it does not influence the probability *for this particular property*. This is due to the symbolic backwards exploration generating a property-dependent pMDP since it starts from the property's target set. Digital clocks, on the other hand, syntactically preserve all behaviour, and Storm then explores all states reachable *from the initial state*, including states that do not influence the probability for property P4. One parameter however happens to only occur on transitions out of such states in this very case. Consequently, on the backward reachability pMDP for P4, Storm generates a partitioning within a millisecond while it takes much longer on the digital clocks pMDP for the small BRP model (S). For the large BRP model (L), Storm generates a partitioning with backwards reachability within milliseconds while it runs out of memory when building the state space for digital clocks.

6 Conclusion

We presented an approach to tackle the approximate synthesis problem for parametric PTA, that is, to partition the parameter space into accepting and rejecting regions for a given property such that c% of the parameter space is covered. The idea is to first obtain a finite pMDP that is equivalent to the pPTA for the property at hand, and to then apply parameter space partitioning on the pMDP. In the application to AODV, a real-world case study [39], our experiments showed encouraging results, thereby highlighting the usefulness of parametric PTA, our approach, and its implementation.

Beyond this Paper. In this paper, we focused on unbounded and time-bounded probabilistic reachability. The overall approach, however, also works with expected accumulated reachability reward properties when we use digital clocks for the abstraction step [40]. In our case studies, we did not include lower-bounded reachability properties, as those refer to minimal probabilities and are thus affected by divergence. We did not take divergence into account in this paper, but solutions are available in the form of fairness and end-component analysis [40,52]. While backwards reachability calculates probabilities under divergence out of the box, it generates non-convex zones for minimum probabilities, for which we do not (yet) have an efficient implementation. The digital clock and backwards reachability approach can also be used to other parameter synthesis questions for pPTA such as feasibility checking ("does there exist a parameter valuation for which a specification holds?"). The resulting pMDPs can then be analysed using quadratic programming [19].

Outlook. Now that a toolchain for pPTA exists, we would like to study more pPTA case studies; the authors would be happy to assist application experts

(e.g. in wireless networks [39]) in modelling and by tuning the tools. Interesting future directions to extend our work are to combine parametrised probabilities with parameters in clock constraints (likely focusing on decidable subclasses as mentioned in Sect. 1), and to consider pPTA in which transition probabilities can depend on clocks [53]. Currently, the stochastic games abstraction is the most competitive technique for PTA. However, parameter space partitioning is not available for parametric stochastic games in Storm or related tools. Developing this would enable the use of the stochastic games abstraction for pPTA.

Data Availability. The tools used and data generated in our experimental evaluation are archived at DOI 10.4121/14910426 [29].

References

1. Agha, G., Palmskog, K.: A survey of statistical model checking. ACM Trans. Model. Comput. Simul. **28**(1), 6:1–6:39 (2018). https://doi.org/10.1145/3158668
2. Alur, R., Dill, D.L.: A theory of timed automata. Theor. Comput. Sci. **126**(2), 183–235 (1994)
3. Alur, R., Henzinger, T.A., Vardi, M.Y.: Parametric real-time reasoning. In: STOC, pp. 592–601. ACM (1993). https://doi.org/10.1145/167088.167242
4. André, É.: What's decidable about parametric timed automata? Int. J. Softw. Tools Technol. Transf. **21**(2), 203–219 (2019). https://doi.org/10.1007/s10009-017-0467-0
5. André, É., Arias, J., Petrucci, L., Pol, J.: Iterative bounded synthesis for efficient cycle detection in parametric timed automata. In: TACAS 2021. LNCS, vol. 12651, pp. 311–329. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72016-2_17
6. André, É., Chatain, T., Fribourg, L., Encrenaz, E.: An inverse method for parametric timed automata. Int. J. Found. Comput. Sci. **20**(5), 819–836 (2009). https://doi.org/10.1142/S0129054109006905
7. André, É., Delahaye, B., Fournier, P.: Consistency in parametric interval probabilistic timed automata. J. Log. Algebraic Methods Program. **110**, 100459 (2020). https://doi.org/10.1016/j.jlamp.2019.04.007
8. André, É., Fribourg, L., Sproston, J.: An extension of the inverse method to probabilistic timed automata. Formal Methods Syst. Des. **42**(2), 119–145 (2013). https://doi.org/10.1007/s10703-012-0169-x
9. Asarin, E., Maler, O., Pnueli, A.: On discretization of delays in timed automata and digital circuits. In: Sangiorgi, D., de Simone, R. (eds.) CONCUR 1998. LNCS, vol. 1466, pp. 470–484. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055642
10. Baier, C., de Alfaro, L., Forejt, V., Kwiatkowska, M.: Model checking probabilistic systems. In: Clarke, E., Henzinger, T., Veith, H., Bloem, R. (eds.) Handbook of Model Checking, pp. 963–999. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-10575-8_28
11. Baier, C., Hermanns, H., Katoen, J.-P.: The 10,000 facets of MDP model checking. In: Steffen, B., Woeginger, G. (eds.) Computing and Software Science. LNCS, vol. 10000, pp. 420–451. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-91908-9_21

12. Baier, C., Katoen, J.P.: Principles of Model Checking. MIT Press, Cambridge (2008)
13. Bohnenkamp, H.C., D'Argenio, P.R., Hermanns, H., Katoen, J.P.: MoDeST: a compositional modeling formalism for hard and softly timed systems. IEEE Trans. Software Eng. **32**(10), 812–830 (2006). https://doi.org/10.1109/TSE.2006.104
14. Brim, L., Češka, M., Dražan, S., Šafránek, D.: Exploring parameter space of stochastic biochemical systems using quantitative model checking. In: Sharygina, N., Veith, H. (eds.) CAV 2013. LNCS, vol. 8044, pp. 107–123. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39799-8_7
15. Budde, C.E., D'Argenio, P.R., Hartmanns, A., Sedwards, S.: An efficient statistical model checker for nondeterminism and rare events. Int. J. Softw. Tools Technol. Transf. **22**(6), 759–780 (2020). https://doi.org/10.1007/s10009-020-00563-2
16. Budde, C.E., Dehnert, C., Hahn, E.M., Hartmanns, A., Junges, S., Turrini, A.: JANI: quantitative model and tool interaction. In: Legay, A., Margaria, T. (eds.) TACAS 2017. LNCS, vol. 10206, pp. 151–168. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54580-5_9
17. Chen, T., Han, T., Katoen, J.: Time-abstracting bisimulation for probabilistic timed automata. In: Second IEEE/IFIP International Symposium on Theoretical Aspects of Software Engineering, TASE 2008, 17–19 June, 2008, Nanjing, China, pp. 177–184. IEEE Computer Society (2008). https://doi.org/10.1109/TASE.2008.29
18. Cheshire, S., Aboba, B., Guttman, E.: Dynamic configuration of ipv4 link-local addresses. RFC **3927**, 1–33 (2005)
19. Cubuktepe, M., Jansen, N., Junges, S., Katoen, J.-P., Topcu, U.: Synthesis in pMDPs: a tale of 1001 parameters. In: Lahiri, S.K., Wang, C. (eds.) ATVA 2018. LNCS, vol. 11138, pp. 160–176. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01090-4_10
20. D'Argenio, P.R., Hartmanns, A., Legay, A., Sedwards, S.: Statistical approximation of optimal schedulers for probabilistic timed automata. In: Ábrahám, E., Huisman, M. (eds.) IFM 2016. LNCS, vol. 9681, pp. 99–114. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-33693-0_7
21. David, A., Larsen, K.G., Legay, A., Mikucionis, M., Poulsen, D.B.: Uppaal SMC tutorial. Int. J. Softw. Tools Technol. Transf. **17**(4), 397–415 (2015). https://doi.org/10.1007/s10009-014-0361-y
22. Daws, C., Kwiatkowska, M.Z., Norman, G.: Automatic verification of the IEEE 1394 root contention protocol with KRONOS and PRISM. Int. J. Softw. Tools Technol. Transf. **5**(2–3), 221–236 (2004)
23. Dombrowski, C., Junges, S., Katoen, J., Gross, J.: Model-checking assisted protocol design for ultra-reliable low-latency wireless networks. In: SRDS, pp. 307–316. IEEE Computer Society (2016)
24. Fruth, M.: Probabilistic model checking of contention resolution in the IEEE 802.15.4 low-rate wireless personal area network protocol. In: ISoLA, pp. 290–297. IEEE Computer Society (2006)
25. Gregersen, H., Jensen, H.E.: Formal Design of Reliable Real Time Systems. Master's thesis, Department of Mathematics and Computer Science, Aalborg University (1995)
26. Hahn, E.M., Hartmanns, A., Hermanns, H., Katoen, J.P.: A compositional modelling and analysis framework for stochastic hybrid systems. Formal Methods Syst. Des. **43**(2), 191–232 (2013). https://doi.org/10.1007/s10703-012-0167-z
27. Hartmanns, A., Hermanns, H.: A Modest approach to checking probabilistic timed automata. In: QEST, pp. 187–196. IEEE (2009)

28. Hartmanns, A., Hermanns, H.: The Modest Toolset: an integrated environment for quantitative modelling and verification. In: Ábrahám, E., Havelund, K. (eds.) TACAS 2014. LNCS, vol. 8413, pp. 593–598. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54862-8_51
29. Hartmanns, A., Katoen, J.P., Kohlen, B., Spel, J.: Tweaking the odds in probabilistic timed automata (artifact). 4TU.Centre for Research Data (2021). https://doi.org/10.4121/14910426
30. Hartmanns, A., Klauck, M., Parker, D., Quatmann, T., Ruijters, E.: The quantitative verification benchmark set. In: Vojnar, T., Zhang, L. (eds.) TACAS 2019. LNCS, vol. 11427, pp. 344–350. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17462-0_20
31. Hartmanns, A., Sedwards, S., D'Argenio, P.R.: Efficient simulation-based verification of probabilistic timed automata. In: WSC, pp. 1419–1430. IEEE (2017). https://doi.org/10.1109/WSC.2017.8247885
32. Helmink, L., Sellink, M.P.A., Vaandrager, F.W.: Proof-checking a data link protocol. In: Barendregt, H., Nipkow, T. (eds.) TYPES 1993. LNCS, vol. 806, pp. 127–165. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-58085-9_75
33. Hensel, C., Junges, S., Katoen, J.P., Quatmann, T., Volk, M.: The probabilistic model checker storm. CoRR abs/2002.07080 (2020)
34. Henzinger, T.A., Manna, Z., Pnueli, A.: What good are digital clocks? In: Kuich, W. (ed.) ICALP 1992. LNCS, vol. 623, pp. 545–558. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-55719-9_103
35. Hune, T., Romijn, J., Stoelinga, M., Vaandrager, F.W.: Linear parametric model checking of timed automata. J. Log. Algebraic Methods Program. **52–53**, 183–220 (2002). https://doi.org/10.1016/S1567-8326(02)00037-1
36. Jonsson, B., Larsen, K.G.: Specification and refinement of probabilistic processes. In: LICS, pp. 266–277. IEEE Computer Society (1991). https://doi.org/10.1109/LICS.1991.151651
37. Jovanovic, A., Lime, D., Roux, O.H.: Integer parameter synthesis for real-time systems. IEEE Trans. Softw. Eng. **41**(5), 445–461 (2015). https://doi.org/10.1109/TSE.2014.2357445
38. Junges, S., Katoen, J., Pérez, G.A., Winkler, T.: The complexity of reachability in parametric Markov decision processes. J. Comput. Syst. Sci. **119**, 183–210 (2021)
39. Kamali, M., Katoen, J.-P.: Probabilistic model checking of AODV. In: Gribaudo, M., Jansen, D.N., Remke, A. (eds.) QEST 2020. LNCS, vol. 12289, pp. 54–73. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59854-9_6
40. Kohlen, B.: Parameter synthesis in probabilistic timed automata. Master's thesis, RWTH Aachen University, Aachen (2020). https://publications.rwth-aachen.de/record/811856
41. Krause, C., Giese, H.: Model checking probabilistic real-time properties for service-oriented systems with service level agreements. INFINITY. EPTCS, vol. 73, pp. 64–78 (2011)
42. Kwiatkowska, M., Norman, G., Parker, D.: Stochastic games for verification of probabilistic timed automata. In: Ouaknine, J., Vaandrager, F.W. (eds.) FORMATS 2009. LNCS, vol. 5813, pp. 212–227. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04368-0_17
43. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 585–591. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22110-1_47

44. Kwiatkowska, M.Z., Norman, G., Parker, D., Sproston, J.: Performance analysis of probabilistic timed automata using digital clocks. Formal Methods Syst. Des. **29**(1), 33–78 (2006). https://doi.org/10.1007/s10703-006-0005-2
45. Kwiatkowska, M.Z., Norman, G., Segala, R., Sproston, J.: Automatic verification of real-time systems with discrete probability distributions. Theor. Comput. Sci. **282**(1), 101–150 (2002). https://doi.org/10.1016/S0304-3975(01)00046-9
46. Kwiatkowska, M.Z., Norman, G., Sproston, J., Wang, F.: Symbolic model checking for probabilistic timed automata. Inf. Comput. **205**(7), 1027–1077 (2007)
47. Legay, A., Sedwards, S., Traonouez, L.-M.: Scalable verification of Markov decision processes. In: Canal, C., Idani, A. (eds.) SEFM 2014. LNCS, vol. 8938, pp. 350–362. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15201-1_23
48. Markowitch, O., Roggeman, Y.: Probabilistic non-repudiation without trusted third party. In: Proceedings 2nd Workshop on Security in Communication Networks (1999)
49. Norman, G., Parker, D., Sproston, J.: Model checking for probabilistic timed automata. Formal Methods Syst. Des. **43**(2), 164–190 (2013)
50. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics, John Wiley & Sons Inc., New York (1994)
51. Quatmann, T., Dehnert, C., Jansen, N., Junges, S., Katoen, J.-P.: Parameter synthesis for Markov models: faster than ever. In: Artho, C., Legay, A., Peled, D. (eds.) ATVA 2016. LNCS, vol. 9938, pp. 50–67. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46520-3_4
52. Sproston, J.: Strict divergence for probabilistic timed automata. In: Bravetti, M., Zavattaro, G. (eds.) CONCUR 2009. LNCS, vol. 5710, pp. 620–636. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04081-8_41
53. Sproston, J.: Probabilistic timed automata with clock-dependent probabilities. Fundam. Informaticae **178**(1–2), 101–138 (2021)
54. Stoelinga, M., Vaandrager, F.: Root contention in IEEE 1394. In: Katoen, J.-P. (ed.) ARTS 1999. LNCS, vol. 1601, pp. 53–74. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48778-6_4