Computing Laboratory

STOCHASTIC GAMES FOR VERIFICATION OF PROBABILISTIC TIMED AUTOMATA

Marta Kwiatkowska
Gethin Norman
David Parker

CL-RR-09-05

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD

Abstract

Probabilistic timed automata (PTAs) are used for formal modelling and verification of systems with probabilistic, nondeterministic and real-time behaviour. For non-probabilistic timed automata, forwards reachability is the analysis method of choice, since it can be implemented extremely efficiently. However, for PTAs, such techniques are only able to compute upper bounds on maximum reachability probabilities. In this paper, we propose a new approach to the analysis of PTAs using abstraction and stochastic games. We show how efficient forwards reachability techniques can be extended to yield both lower and upper bounds on maximum (and minimum) reachability probabilities. We also present abstraction-refinement techniques that are guaranteed to improve the precision of these probability bounds, providing a fully automatic method for computing the exact values. We have implemented these techniques and applied them to a set of large case studies. We show that, in comparison to alternative approaches to verifying PTAs, such as backwards reachability and digital clocks, our techniques exhibit superior performance and scalability.

1 Introduction

Probabilistic behaviour occurs naturally in many real-time systems, either due to the use of randomisation, or because of the presence of unreliable components. Prominent examples include communication protocols such as Bluetooth, IEEE 802.11 and FireWire, which use randomised back-off schemes and are designed to function over faulty communication channels. Another important class are security protocols, such as for non-repudiation, anonymity and non-interference, where randomisation and timing are both essential ingredients.

Probabilistic timed automata (PTAs) [10, 2, 17], which are finite state automata extended with real-valued clocks and discrete probabilistic choice, are a natural formalism for modelling and analysing such systems. Formal verification techniques for PTAs can help to identify anomalies resulting from the subtle interplay between probabilistic, real-time and nondeterministic aspects of these systems. A fundamental property of a PTA is the minimum or maximum probability of reaching a particular class of states in the model. This allows the expression of a wide range of useful properties, for example, "the minimum probability that a data packet is correctly delivered with t seconds".

There are three main existing algorithmic approaches to the verification of PTAs: (i) *forwards reachability* [17, 6]; (ii) *backwards reachability* [18]; and (iii) *digital clocks* [16]. Forwards reachability is based on a symbolic forwards exploration, similar to the techniques implemented in state-of-the art tools for non-probabilistic timed automata [7, 19]. This approach is appealing because it can be implemented extremely efficiently with data structures such as difference-bound matrices (DBMs). However, in the context of *probabilistic* timed automata, these techniques yield only an *upper bound* on *maximum* reachability probabilities.

Backwards reachability [18] performs a state-space exploration in the opposite direction, from target to initial states. This computes exact values for both minimum and maximum reachability probabilities; however, the operations required to implement it are expensive, limiting its applicability. The digital clocks technique of [16] uses an efficient language-level translation to a probabilistic model with finite state semantics. This also gives precise values for minimum and maximum probabilities, but is only applicable to a restricted class of PTAs.

PTAs are, because of their real-valued model of time, inherently *infinite-state*. The three PTA verification techniques described above work by constructing a finite-state Markov decision process (MDP) that can be analysed with existing tools and techniques. This MDP can be viewed as an *abstraction* of the infinite-state semantics of the PTA. In this paper, we take a new approach, using the ideas of [14] to represent PTA abstractions as *stochastic two-player games*.

We first show how the forwards reachability technique of [17] can be generalised to produce a stochastic game that yields *lower* and *upper* bounds on either *minimum* or *maximum* reachability probabilities of PTAs. Then, using *abstraction-refinement* methods, we show how the stochastic game can be iteratively refined in order to tighten these bounds. This gives a fully automatic technique to compute exact reachability probabilities within a finite number of steps. Finally, we present a prototype tool implementing these techniques that exhibits significantly better performance than other PTA verification approaches. This paper is a full version of [15], including an appendix of proofs.

Related work. Existing PTA verification techniques are discussed above and a detailed experimental comparison is included in Section 6. Also relevant is [4], which presents an algorithm for computing time-abstracting bisimulation quotients of PTAs. Abstraction-refinement approaches have been proposed for *non-probabilistic* timed automata, e.g. [8] which uses bounded model checking and SAT-based techniques, [22] which is based on the region graph construction, and [13] for verifying PLC automata using UPPAAL [19].

2 Markov decision processes and stochastic games

Markov decision processes (MDPs) are a widely used formalism for modelling systems that exhibit both nondeterministic and probabilistic behaviour.

Definition 1 *An MDP* M *is a tuple* $(S, \overline{S}, Act, Steps_{\mathsf{M}})$ *where* S *is a set of* states*,* $\overline{S} \subseteq S$ *is the set of* initial states*,* Act *is a set of* actions *and* $Steps_{\mathsf{M}} : S \times Act \to \mathsf{Dist}(S)$ *is the* probabilistic transition function.

In each state $s \in S$ of an MDP M, there is a nondeterministic choice between one or more *available* actions $a \in Act$ (those for which $Steps_{\mathsf{M}}(s, a)$ is defined). After the choice of an action a, a successor state is selected at random according to the probability distribution $Steps_{\mathsf{M}}(s, a)$. A *path* through M is a sequence of states selected in this fashion.

To reason about the MDP M, we use the notion of an *adversary*, which is a possible resolution of all nondeterministic choices in M (formally, an adversary is a function from finite paths to actions). For a fixed adversary A, we can define a probability measure over the set of paths from a state s and, in particular, the probability $p_s^A(F)$ of reaching a *target* $F \subseteq S$ from s under A. We are typically interested in the *minimum and maximum reachability probabilities* for F:

$$p_{\mathsf{M}}^{\min}(F) \stackrel{\text{def}}{=} \inf_{s \in \overline{S}} \inf_A p_s^A(F) \quad \text{and} \quad p_{\mathsf{M}}^{\max}(F) \stackrel{\text{def}}{=} \sup_{s \in \overline{S}} \sup_A p_s^A(F)\,.$$

These values, and an adversary of M which produces them, can be computed with a simple numerical computation called *value iteration* [20].

Stochastic two-player games [21, 5] extend MDPs by allowing two types of nondeterministic choice, controlled by separate *players*. We use stochastic games in the manner proposed in [14] to represent an *abstraction* of an MDP.

Definition 2 *A* stochastic game G *is a tuple* $(S, \overline{S}, Act, Steps_{\mathsf{G}})$ *where:* S *is a set of* states*,* $\overline{S} \subseteq S$ *is the set of* initial states Act *is a set of* actions *and* $Steps_{\mathsf{G}} : S \times Act \to 2^{\mathsf{Dist}(S)}$ *is the* probabilistic transition function*.*

Each transition of a stochastic game G comprises three choices: first, like for an MDP, player 1 picks an available action $a \in Act$; next, player 2 selects a distribution λ from the set $Steps_{\mathsf{G}}(s, a)$; finally, a successor state is chosen at random according to λ. A resolution of the nondeterminism in G (the analogue of an MDP adversary) is a pair of *strategies* σ_1, σ_2 for the players, under which we can define the probability $p_s^{\sigma_1,\sigma_2}(F)$ of reaching a target $F \subseteq S$ from a state s.

Intuitively, the idea of [14] is that, in a stochastic game G, representing an abstraction of an MDP M, player 2 choices represent nondeterminism present in M and player 1 choices represent additional nondeterminism introduced through abstraction. By quantifying over strategies for players 1 and 2, we can obtain both lower bounds (*lb*) and upper bounds (*ub*) on the minimum and maximum reachability probabilities of M. If G is constructed from M using the approach of [14], then, in the case of maximum probabilities, for example:

$$p_{\mathsf{G}}^{lb,\max}(F) \;\leqslant\; p_{\mathsf{M}}^{\max}(F) \;\leqslant\; p_{\mathsf{G}}^{ub,\max}(F)$$

where, in the stochastic game G:

$$\begin{array}{rcl} p_{\mathsf{G}}^{lb,\max}(F) & \stackrel{\text{def}}{=} & \sup_{s \in \overline{S}} \inf_{\sigma_1} \sup_{\sigma_2} p_s^{\sigma_1,\sigma_2}(F) \\ p_{\mathsf{G}}^{ub,\max}(F) & \stackrel{\text{def}}{=} & \sup_{s \in \overline{S}} \sup_{\sigma_1} \sup_{\sigma_2} p_s^{\sigma_1,\sigma_2}(F) \end{array}$$

Using similar techniques as those for MDPs, we can efficiently compute these values and strategies for players 1 and 2 that result in them [5].

3 Probabilistic Timed Automata

Time, clocks and zones. Probabilistic timed automata model time using *clocks*, variables over the set $\mathbb{R}$ of non-negative reals. We assume a finite set $\mathcal{X}$ of clocks. A function $v : \mathcal{X} \to \mathbb{R}$ is referred to as a *clock valuation* and the set of all clock valuations is denoted by $\mathbb{R}^{\mathcal{X}}$. For any $v \in \mathbb{R}^{\mathcal{X}}$, $t \in \mathbb{R}$ and $X \subseteq \mathcal{X}$, we use $v{+}t$ to denote the clock valuation which increments all clock values in v by t and $v[X{:=}0]$ for the valuation in which clocks in X are reset to 0.

The set of *zones* of $\mathcal{X}$, written $Zones(\mathcal{X})$, is defined by the syntax:

$$\zeta ::= \mathtt{true} \mid x \leqslant d \mid c \leqslant x \mid x{+}c \leqslant y{+}d \mid \neg\zeta \mid \zeta \vee \zeta$$

where $x, y \in \mathcal{X}$ and $c, d \in \mathbb{N}$. A zone ζ represents the set of clock valuations v which *satisfy* ζ, denoted $v \triangleleft \zeta$, i.e. those for which ζ resolves to `true` by substituting each clock x with $v(x)$.

We will use several classical operations on zones [9, 23]. The zone $\nearrow\zeta$ contains all clock valuations that can be reached from a valuation in ζ by letting time pass. Conversely, $\swarrow\zeta$ contains those that can reach ζ by letting time pass. For $X \subseteq \mathcal{X}$, the zone $[X{:=}0]\zeta$ contains the clock valuations which result in a valuation in ζ when the clocks in X are reset to 0, while $\zeta[X{:=}0]$ contains the valuations obtained from those in ζ by resetting these clocks to 0.

Syntax and semantics of PTAs. We now present the formal syntax and semantics of probabilistic timed automata.

Definition 3 *A PTA is a tuple* $\mathsf{P}{=}(L, \bar{l}, Act, inv, enab, prob)$ *where:*

- L *is a finite set of* locations *and* $\bar{l} \in L$ *is the* initial location*;*
- Act *is a finite set of actions;*
- $inv : L \to Zones(\mathcal{X})$ *is the* invariant condition*;*
- $enab : L{\times}Act \to Zones(\mathcal{X})$ *is the* enabling condition*;*
- $prob : L{\times}Act \to \mathsf{Dist}(2^{\mathcal{X}}{\times}L)$ *is the* probabilistic transition function*.*

A *state* of a PTA is a pair $(l, v) \in L{\times}\mathbb{R}^{\mathcal{X}}$ such that $v \triangleleft inv(l)$. In any state (l, v), a certain amount of time $t \in \mathbb{R}$ can elapse, after which an action $a \in Act$ is performed. The choice of t requires that, while time passes, the invariant $inv(l)$ remains continuously satisfied. Each action a can be only chosen if it is *enabled*, that is, the zone $enab(l, a)$ is satisfied by $v{+}t$. Once action a is chosen, a set of clocks to reset and successor location are selected at random, according to the distribution $prob(l, a)$. We call each element $(X, l') \in 2^{\mathcal{X}}{\times}L$ in the support of $prob(l, a)$ an *edge* and, for convenience, assume that the set of such edges, denoted $edges(l, a)$, is an ordered list $\langle e_1, \ldots, e_n \rangle$.

Definition 4 *Let* $\mathsf{P}{=}(L, \bar{l}, Act, inv, enab, prob)$ *be a PTA. The semantics of* P *is defined as the (infinite-state) MDP* $[\![\mathsf{P}]\!] = (S, \overline{S}, \mathbb{R}{\times}Act, Steps_{\mathsf{P}})$ *where:*

- $S = \{(l, v) \in L \times \mathbb{R}^{\mathcal{X}} \mid v \triangleleft inv(l)\}$ *and* $\overline{S} = \{(\bar{l}, \mathbf{0})\}$*;*
- $Steps_{\mathsf{P}}((l, v), (t, a)) = \lambda$ *if and only if* $v{+}t' \triangleleft inv(l)$ *for all* $0 \leqslant t' \leqslant t$*,*
  $v{+}t \triangleleft enab(l, a)$ *and, for any* $(l', v') \in S$*:*

$$\lambda(l', v') = \sum \left\{\!\left|\ prob(l, a)(X, l') \mid X \in 2^{\mathcal{X}} \wedge v' = (v{+}t)[X{:=}0] \ \right|\!\right\} .$$

Each transition of the semantics of the PTA is a time-action pair (t, a), representing time passing for t time units, followed by a discrete a-labelled transition. If $Steps_{\mathsf{P}}((l, v), (t, a))$ is defined and $edges(l, a) = \langle (l_1, X_1), \ldots, (l_n, X_n) \rangle$, we write $(l, v) \xrightarrow{t,a} \langle s_1, \ldots, s_n \rangle$ where $s_i = (l_i, (v{+}t)[X_i{:=}0])$ for all $1 \leqslant i \leqslant n$.

We make several standard assumptions about probabilistic timed automata. Firstly, we restrict our attention to *structurally non-Zeno* automata [24]. This class of models, which can be identified syntactically and in a compositional fashion [25], guarantees time-divergent behaviour. Secondly, for technical reasons, we assume all zones appearing in a PTA are diagonal-free [3].

Probabilistic Reachability. The minimum and maximum probabilities of reaching, from the initial state of a PTA P, a certain target $F \subseteq L$ are:

$$p_{\mathsf{P}}^{\min}(F) = p_{[\![\mathsf{P}]\!]}^{\min}(S_F) \qquad \text{and} \qquad p_{\mathsf{P}}^{\max}(F) = p_{[\![\mathsf{P}]\!]}^{\max}(S_F)$$

where $S_F = \{(l,v) \mid v \triangleleft inv(l) \wedge l \in F\}$. We can easily consider more expressive targets, that refer to both locations and clock values, through a simple syntactic modification of the PTA [17].

Symbolic states and operations. In order to represent sets of PTA states, we use the concept of a *symbolic state*: a pair $\mathsf{z} = (l, \zeta)$, comprising a location l and a zone ζ over $\mathcal{X}$, representing the set of PTA states $\{(l,v) \mid v \triangleleft \zeta\}$. We use the notation $(l,v) \in (l,\zeta)$ to denote inclusion of a PTA state in a symbolic state.

We will use the *time successor* and *discrete successor* operations of [9, 23]. For a symbolic state (l,ζ), action a, and edge $e = (X, l') \in edges(l,a)$, we define:

- $\mathsf{tsuc}(l,\zeta) \stackrel{\text{def}}{=} (l, inv(l) \wedge \nearrow\zeta)$ is the *time successor* of (l,ζ);
- $\mathsf{dsuc}[a,e](l,\zeta) \stackrel{\text{def}}{=} (l', (\zeta \wedge enab(l,a))[X{:=}0] \wedge inv(l'))$ is the *discrete successor* of (l,ζ) with respect to e;
- $\mathsf{post}[a,e](l,\zeta) \stackrel{\text{def}}{=} \mathsf{tsuc}(\mathsf{dsuc}[a,e](l,\zeta))$ is the *post* of (l,ζ) with respect to e.

The *c-closure* of a zone ζ is obtained by removing any constraint that refers to integers greater than c. For a given c, there are only a finite number of c-closed zones. For the remainder of this paper, we assume that all zones are c-closed where c is the largest constant appearing in the PTA under study.

4 Forwards Reachability for PTAs

In this section, we begin by describing the approach of [17], which we will refer to as *MDP-based forwards reachability*. This computes only *upper* bounds on *maximum* reachability probabilities of a PTA. Subsequently, we will propose a new algorithm, based on stochastic games, which addresses these limitations.

4.1 MDP-based forwards reachability

The MDP-based forwards reachability approach of [17] works by building an *abstraction* of a PTA P. This abstraction is represented by an MDP M whose state space is a set Z of symbolic states, i.e. each state of M represents a set of states of the infinite-state MDP semantics $[\![\mathsf{P}]\!]$. The algorithm of [17] is shown in Figure 1. For the purposes of

BuildReachGraph(P, F)

$\mathtt{Z} := \emptyset$
$\mathtt{Y} := \{\mathsf{tsuc}(\bar{l}, \mathbf{0})\}$
while $\mathtt{Y} \neq \emptyset$
  choose $(l, \zeta) \in \mathtt{Y}$
  $\mathtt{Y} := \mathtt{Y} \setminus \{(l, \zeta)\}$
  $\mathtt{Z} := \mathtt{Z} \cup \{(l, \zeta)\}$
  for $a \in Act$ **such that** $enab(l, a) \wedge \zeta \neq \emptyset$
    for $e_i \in edges(l, a) = \langle e_1, \ldots, e_n \rangle$
      $(l'_i, \zeta'_i) := \mathsf{post}[(l, a), e_i](l, \zeta)$
      if $(l'_i, \zeta'_i) \notin \mathtt{Z}$ **and** $l'_i \notin F$ **then** $\mathtt{Y} := \mathtt{Y} \cup \{(l'_i, \zeta'_i)\}$
    $\mathtt{R} := \mathtt{R} \cup \{((l, \zeta), a, \langle (l'_1, \zeta'_1), \ldots, (l'_n, \zeta'_n) \rangle)\}$
return $(\mathtt{Z}, \mathtt{R})$

BuildMDP($\mathtt{Z}, \mathtt{R}$)

$\overline{\mathtt{Z}} := \{(l, \zeta) \in \mathtt{Z} \mid l = \bar{l}\}$
for $(l, \zeta) \in \mathtt{Z}$ **and** $\theta \in \mathtt{R}(l, \zeta)$
  $Steps_{\mathsf{M}}((l, \zeta), \theta) := \lambda_\theta$
return $\mathsf{M} = (\mathtt{Z}, \overline{\mathtt{Z}}, \mathtt{R}, Steps_{\mathsf{M}})$

Figure 1: Algorithm for MDP-based forwards reachability, based on [17]

this presentation, we have reformulated the algorithm into: (i) the construction of a *reachability graph* over the set of symbolic states $\mathtt{Z}$; and (ii) the construction of an MDP M from this graph.

The algorithm to build this reachability graph is based on the well-known forwards reachability algorithm for non-probabilistic timed automata [7, 19]. It performs a forwards exploration through the automata, successively computing symbolic states using the post operation. One important difference is that, in the probabilistic setting, *on-the-fly* techniques cannot be used: the state-space exploration is exhaustive. This is because the aim is to determine, not just the *existence* of a path to the target, but the *probability* of reaching the target. For this, an MDP containing all such paths is constructed and analysed.

A reachability graph captures information about the transitions in a PTA. It comprises a multiset[1] $\mathtt{Z}$ of symbolic states and a set $\mathtt{R} \subseteq \mathtt{Z} \times Act \times \mathtt{Z}^+$ of *symbolic transitions*. Each symbolic transition $\theta \in \mathtt{R}$ takes the form:

$$\theta = \big((l, \zeta), a, \langle (l_1, \zeta_1), \ldots, (l_n, \zeta_n) \rangle\big)$$

where $n = |edges(l, a)|$. Intuitively, θ represents the possibility of taking action a from a PTA state in (l, ζ) and, for each edge $(X_i, l_i) \in edges(l, a)$, reaching a state in (l_i, ζ_i). A key property of symbolic transitions is the notion of *validity*:

$$valid(\theta) \stackrel{\text{def}}{=} \zeta \wedge \swarrow\!\big(enab(l, a) \wedge (\wedge_{i=1}^{n} ([X_i{:=}0]\zeta_i))\big)$$

[1] The use of a multiset is a technical requirement, later used for abstraction refinement.

which gives precisely the set of clock valuations satisfying ζ from which it is possible to let time pass and perform the action a such that taking the ith edge (X_i, l_i) gives a state in (l_i, ζ_i). A symbolic transition θ is *valid* if the zone $\mathit{valid}(\theta)$ is non-empty. This leads to the following formal definition of a reachability graph.

Definition 5 *A* reachability graph *for a PTA* $\mathsf{P}{=}(L, \bar{l}, \mathit{Act}, \mathit{inv}, \mathit{enab}, \mathit{prob})$ *and target* F*, is a pair* $(\mathtt{Z}, \mathtt{R})$ *where:*

- $\mathtt{Z} \subseteq L{\times}\mathit{Zones}(\mathcal{X})$ *is a multiset of symbolic states where* $\{s \in \mathtt{z} \,|\, \mathtt{z} \in \mathtt{Z}\} = S$*;*
- $\mathtt{R} \subseteq \mathtt{Z}{\times}\mathit{Act}{\times}\mathtt{Z}^{+}$ *is a set of* valid *symbolic transitions;*

and, if $\mathtt{z} = (l, \zeta) \in \mathtt{Z}$*,* $l \notin F$*,* $s \in \mathtt{z}$ *and* $s \xrightarrow{t,a} \langle s_1, \ldots, s_n \rangle$*, then* $\mathtt{R}$ *contains a symbolic transition* $(\mathtt{z}, a, \langle \mathtt{z}_1, \ldots, \mathtt{z}_n \rangle)$ *such that* $s_i \in \mathtt{z}_i$ *for all* $1 \leqslant i \leqslant n$*.*

For any PTA P and target F, it follows from the definition of post that algorithm $\mathsf{BuildReachGraph}(\mathsf{P}, F)$ in Figure 1 returns a (unique) reachability graph for (P, F). However, the above conditions do not imply the uniqueness of reachability graphs, and there may exist many other such graphs for (P, F).

Given a reachability graph $(\mathtt{Z}, \mathtt{R})$ we can construct an MDP M with state space $\mathtt{Z}$ using the symbolic transitions in $\mathtt{R}$ to build the transitions of M. More precisely, a symbolic transition $\theta = ((l, \zeta), a, \langle (l_1, \zeta_1), \ldots, (l_n, \zeta_n) \rangle)$ induces a probability distribution λ_θ over symbolic states $\mathtt{Z}$ where for any $(l', \zeta') \in \mathtt{Z}$:

$$\lambda_\theta(l', \zeta') \stackrel{\text{def}}{=} \sum \left\{\!\!\left|\, \mathit{prob}(l, a)(e_i) \,|\, e_i \in \mathit{edges}(l, a) \wedge \zeta_i{=}\zeta' \,\right|\!\!\right\} .$$

Using these distributions, the algorithm $\mathsf{BuildMDP}(\mathtt{Z}, \mathtt{R})$ in Figure 1 constructs an MDP M, analysis of which yields bounds on the behaviour of P.

Theorem 4.1 *Let* P *be a PTA with target* F*. If* $(\mathtt{Z}, \mathtt{R})$ *is a reachability graph for* (P, F) *and* M *is the MDP returned by* $\mathsf{BuildMDP}(\mathtt{Z}, \mathtt{R})$ *(see Figure 1), then* $p_{\mathsf{M}}^{\min}(\mathtt{Z}_F) \leqslant p_{\mathsf{P}}^{\min}(F)$ *and* $p_{\mathsf{P}}^{\max}(F) \leqslant p_{\mathsf{M}}^{\max}(\mathtt{Z}_F)$ *where* $\mathtt{Z}_F = F{\times}\mathit{Zones}(\mathcal{X})$*.*

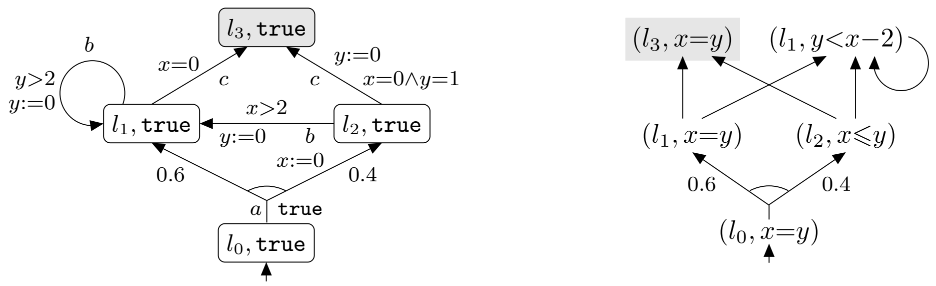
This theorem extends [17], by establishing the result for *any* reachability graph, not just that returned by $\mathsf{BuildReachGraph}$ and, by restricting to structurally non-Zeno PTAs, also yields *lower* bounds on *minimum* reachability probabilities.

Example 4.2 *We illustrate these ideas using the simple PTA* P *in Figure 2(a). We use the standard graphical notation for PTAs and omit probability 1 labels. Applying* $\mathsf{BuildReachGraph}(\mathsf{P}, \{l_3\})$ *(see Figure 1) yields the symbolic states:*

$$\mathtt{Z} = \{(l_0, x{=}y), (l_1, x{=}y), (l_1, y{<}x{-}2), (l_2, x{\leqslant}y), (l_3, x{=}y)\}$$

and the set of symbolic transitions $\mathtt{R}$*. From the first two symbolic states, for example, we have* $\mathtt{R}(l_0, x{=}y) = \{\theta_a\}$ *and* $\mathtt{R}(l_1, x{=}y) = \{\theta_b, \theta_c\}$ *where:*

$$\begin{array}{rcl} \theta_a & = & \big((l_0, x{=}y), a, \langle (l_1, x{=}y), (l_2, x{\leqslant}y) \rangle\big) \\ \theta_b & = & \big((l_1, x{=}y), b, \langle (l_1, x{=}y) \rangle\big), \ \theta_c = \big((l_1, x{=}y), c, \langle (l_3, x{=}y) \rangle\big) \end{array}$$



(a) PTA (b) MDP

Figure 2: Analysis of a PTA through MDP-based forwards reachability

$\mathsf{BuildGame}(\mathtt{Z}, \mathtt{R})$

$\overline{\mathtt{Z}} = \{(l,\zeta) \in \mathtt{Z} \mid l = \bar{l}\}$
for $(l,\zeta) \in \mathtt{Z}$
  for $\Theta \subseteq \mathtt{R}(l,\zeta)$ **such that** $\Theta \neq \emptyset$ **and** $valid(\Theta)$
    $Steps_{\mathsf{G}}((l,\zeta),\Theta) := \{\lambda_\theta \mid \theta \in \Theta\}$
return $\mathsf{G} = (\mathtt{Z}, \overline{\mathtt{Z}}, 2^{\mathtt{R}}, Steps_{\mathsf{G}})$

Figure 3: Algorithm to construct a stochastic game from a reachability graph

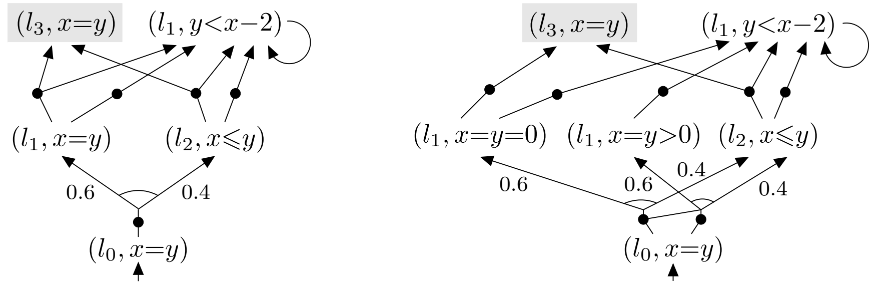
The resulting MDP is shown in Figure 2(b). The maximum probability of reaching location l_3 *in the PTA is* 0.6*, which results from taking action* a *in* l_0 *immediately and, if* l_1 *is reached, proceeding straight to* l_3*. An alternative is to wait for 1 time unit in* l_0 *and then take* a*, reaching* l_3 *via* l_2*, however, this results in a lower probability of 0.4. An upper bound on the maximum probability for the PTA is obtained from the maximum probability of reaching* $(l_3, x{=}y)$ *in the MDP. The resulting value is* 1*. This is because the symbolic states for locations* l_1 *and* l_2 *are too coarse to preserve the precise time that action* a *is taken.*

4.2 Game-based forwards reachability

The main limitation of the MDP-based forwards reachability algorithm is that it only provides *lower* bounds for minimum and *upper* bounds for maximum reachability probabilities. We now describe how to construct, from a reachability graph, a stochastic game G that yields both *lower* and *upper* bounds. The game G is, like the MDP in the previous section, an abstraction of the infinite-state MDP semantics of the PTA, whose state space is the symbolic states $\mathtt{Z}$.

We utilise the approach of [14] to represent an abstraction of an MDP as a stochastic two-player game. The basic idea is that the two players in the game represent nondeterminism introduced by the abstraction and nondeterminism from the original model. In a symbolic state (l,ζ) of the game abstraction of a PTA, player 1 first picks a PTA state $(l,v) \in (l,\zeta)$ and then player 2 makes a choice over the actions that become enabled after letting time pass from (l,v).

In order to construct such a game from a reachability graph $(\mathtt{Z}, \mathtt{R})$, we first extend the



(a) From reachability graph (b) After one refinement

Figure 4: Stochastic games for the PTA example of Figure 2

notion of *validity* to sets of symbolic transitions with the same source. For any symbolic state $(l,\zeta) \in \mathtt{Z}$ and set of symbolic transitions $\Theta \subseteq \mathtt{R}(l,\zeta)$, let:

$$valid(\Theta) \stackrel{\text{def}}{=} (\wedge_{\theta\in\Theta} valid(\theta)) \wedge \big(\wedge_{\theta\in\mathtt{R}(l,\zeta)\setminus\Theta} \neg valid(\theta)\big) \ .$$

By construction, $valid(\Theta)$ identifies precisely the clock valuations $v \triangleleft \zeta$ such that, from (l,v), it is possible to perform a transition encoded by any symbolic transition $\theta \in \Theta$, but it is not possible to perform a transition encoded by any other symbolic transition of $\mathtt{R}(l,\zeta)$.

The algorithm $\mathsf{BuildGame}$ in Figure 3 describes how to construct, from a reachability graph $\mathtt{R}$, a stochastic game with symbolic states $\mathtt{Z}$. In a state $\mathtt{z}$ of the game, player 1 chooses between any non-empty *valid* set of symbolic transitions $\Theta \subseteq \mathtt{R}(\mathtt{z})$. Player 2 then selects a symbolic transition $\theta \in \Theta$. As the following result demonstrates, this game yields lower and upper bounds on either minimum or maximum reachability probabilities of the PTA.

Theorem 4.3 *Let P be a PTA with target F. If $(\mathtt{Z},\mathtt{R})$ is a reachability graph for (P,F) and G is the stochastic game returned by $\mathsf{BuildGame}(\mathtt{Z},\mathtt{R})$ (see Figure 3), then $p^{lb,\star}_{\mathsf{G}}(\mathtt{Z}_F) \leqslant p^{\star}_{\mathsf{P}}(F) \leqslant p^{ub,\star}_{\mathsf{G}}(\mathtt{Z}_F)$ for $\star \in \{\min,\max\}$.*

Example 4.4 *We return to the PTA from Figure 2 and the reachability graph constructed in Example 4.2. The corresponding stochastic game is shown in Figure 4(a). As for PTAs and MDPs, we draw probability distributions as arrows grouped by an arc, omitting the labelling of probability 1 transitions. A set of distributions emanating from a black circle indicates a player 2 choice; the outgoing edges from each symbolic state represent a player 1 choice.*

Consider, the symbolic state $(l_1, x{=}y)$, for which there are two symbolic transitions θ_b and θ_c (see Example 4.2). Since $valid(\theta_b){=}(x{=}y)$ and $valid(\theta_c){=}(x{=}y{=}0)$, we have $valid(\{\theta_b\}){=}(x{=}y{>}0)$, $valid(\{\theta_c\}){=}\emptyset$ and $valid(\{\theta_b,\theta_c\}){=}(x{=}y{=}0)$. This tells us that there are two classes of PTA states in $(l_1, x{=}y)$: those in which both actions b and c become enabled, and those in which only b becomes enabled. Thus, in the game state (see Figure 4(a)), we see that player 1 chooses between these two classes and then player 2 chooses an available action.

$\mathsf{Refine}(\mathtt{Z}, \mathtt{R}, (l,\zeta), \Theta_{lb}, \Theta_{ub})$

$\zeta_{lb} := \mathit{valid}(\Theta_{lb})$
$\zeta_{ub} := \mathit{valid}(\Theta_{ub})$
$\mathtt{Z}^{new} := \{(l,\zeta_{lb}), (l,\zeta_{ub}), (l, \zeta \wedge \neg(\zeta_{lb} \vee \zeta_{ub}))\} \setminus \{\emptyset\}$
$\mathtt{Z}^{ref} := (\mathtt{Z} \setminus \{(l,\zeta)\}) \uplus \mathtt{Z}^{new}$
$\mathtt{R}^{ref} := \emptyset$
for $\theta = (\mathtt{z}_0, a, \langle \mathtt{z}_1, \ldots, \mathtt{z}_n \rangle) \in \mathtt{R}$
  if $(l,\zeta) \notin \{\mathtt{z}_0, \mathtt{z}_1, \ldots, \mathtt{z}_n\}$ **then**
    $\mathtt{R}^{ref} := \mathtt{R}^{ref} \cup \{\theta\}$
  else
    $\Theta^{new} := \{(\mathtt{z}'_0, a, \langle \mathtt{z}'_1, \ldots, \mathtt{z}'_n \rangle) \mid \mathtt{z}'_i \in \mathtt{Z}^{new}$ if $\mathtt{z}_i = (l,\zeta)$ and $\mathtt{z}'_i = \mathtt{z}_i$ o/wise$\}$
    for $\theta^{new} \in \Theta^{new}$ **such that** $\mathit{valid}(\theta^{new}) \neq \emptyset$
      $\mathtt{R}^{ref} := \mathtt{R}^{ref} \cup \{\theta^{new}\}$
return $(\mathtt{Z}^{ref}, \mathtt{R}^{ref})$

Figure 5: Algorithm to refine symbolic state (l,ζ) in reachability graph $(\mathtt{Z}, \mathtt{R})$

Using Theorem 4.3, the stochastic game in Figure 4(a) gives bounds on the maximum probability of reaching l_3 in the PTA. The upper bound (as for the MDP) is 1 as, after either branch of the initial probabilistic choice, player 1 can make a choice which allows l_3 to be reached with probability 1. The lower bound, however, is 0 because player 1 can also, in both cases, make l_3 unreachable.

As the above example illustrates, it is possible that the difference between the lower and upper bounds from the game is too great to provide useful information. In the next section, we will address this issue by introducing a way to refine the abstraction to reduce the difference between the bounds.

5 Abstraction Refinement

The game-based abstraction approach of [14] has been extended with *refinement* techniques in [11, 12]. Inspired by non-probabilistic counterexample-guided abstraction refinement, the idea is that an initially coarse abstraction is iteratively refined until it is precise enough to yield useful verification results. Crucial to this approach is the use of the lower and upper bounds provided by a stochastic game abstraction as a *quantitative* measure of the preciseness of the abstraction.

The refinement algorithm. Our refinement algorithm takes a reachability graph $(\mathtt{Z}, \mathtt{R})$, splits one or more of the symbolic states in $\mathtt{Z}$ and then modifies the symbolic transitions of $\mathtt{R}$ accordingly. This process is guided by the analysis of the stochastic game constructed from $(\mathtt{Z}, \mathtt{R})$, i.e. the bounds for the probability of reaching the target and player 1 strategies that attain these bounds.

We now outline the refinement of a single symbolic state (l,ζ) for which the bounds

$\mathsf{AbstractRefine}(\mathsf{P}, F, \star, \varepsilon)$

$(\mathtt{Z}, \mathtt{R}) := \mathsf{BuildReachGraph}(\mathsf{P}, F)$
$\mathsf{G} := \mathsf{BuildGame}(\mathtt{Z}, \mathtt{R})$
$(p_{\mathsf{G}}^{lb,\star}, p_{\mathsf{G}}^{ub,\star}, \sigma_1^{lb}, \sigma_1^{ub}) := \mathsf{AnalyseGame}(\mathsf{G}, F, \star)$
while $p_{\mathsf{G}}^{ub,\star} - p_{\mathsf{G}}^{lb,\star} > \varepsilon$
  choose $(l, \zeta) \in \mathtt{Z}$
  $(\mathtt{Z}, \mathtt{R}) := \mathsf{Refine}(\mathtt{Z}, \mathtt{R}, (l, \zeta), \sigma_1^{lb}(l, \zeta), \sigma_1^{ub}(l, \zeta))$
  $\mathsf{G} := \mathsf{BuildGame}(\mathtt{Z}, \mathtt{R})$
  $(p_{\mathsf{G}}^{lb,\star}, p_{\mathsf{G}}^{ub,\star}, \sigma_1^{lb}, \sigma_1^{ub}) := \mathsf{AnalyseGame}(\mathsf{G}, F, \star)$
return $[p_{\mathsf{G}}^{lb,\star}, p_{\mathsf{G}}^{ub,\star}]$

Figure 6: Abstraction-refinement loop to compute reachability probabilities

differ and for which distinct player 1 strategies yield each bound.[2] A player 1 strategy chooses, for any state in the stochastic game, an action available in the state. By construction, an available action in (l, ζ) is a valid set of symbolic transitions from $\mathtt{R}(l, \zeta)$. We let $\Theta_{lb}, \Theta_{ub} \subseteq \mathtt{R}(l, \zeta)$ denote the distinct player 1 strategy choices for the lower and upper bound respectively. Since the validity conditions for Θ_{lb} and Θ_{ub} identify precisely the clock valuations in ζ for which the corresponding transitions of $[\![\mathsf{P}]\!]$ are possible, we split (l, ζ) into:

$$\big(l, \mathit{valid}(\Theta_{lb})\big), \quad \big(l, \mathit{valid}(\Theta_{ub})\big) \text{ and } \big(l, \zeta \wedge \neg(\mathit{valid}(\Theta_{lb}) \vee \mathit{valid}(\Theta_{ub}))\big)\,.$$

By construction, $\mathit{valid}(\Theta_{lb})$ and $\mathit{valid}(\Theta_{ub})$ are both non-empty. Furthermore, since $\Theta_{lb} \neq \Theta_{ub}$, from the definition of validity, we have $\mathit{valid}(\Theta) \wedge \mathit{valid}(\Theta') = \emptyset$, and hence the split of (l, ζ) produces a strict refinement of $\mathtt{Z}$.

The complete refinement algorithm is shown in Figure 5. Lines 1–4 refine $\mathtt{Z}$, as just described, and lines 5–12 update the set of symbolic transitions $\mathtt{R}$. The result is a new reachability graph, for which the corresponding stochastic game is a refined abstraction of the PTA, satisfying the following properties.

Theorem 5.1 *Let* P *be a PTA with target* F *and* $(\mathtt{Z}, \mathtt{R})$ *be a reachability graph for* (P, F)*. If* $(\mathtt{Z}^{ref}, \mathtt{R}^{ref})$ *is the result of applying algorithm* Refine *(see Figure 5) to* $(\mathtt{Z}, \mathtt{R})$*,* $\mathsf{G} = \mathsf{BuildGame}(\mathtt{Z}, \mathtt{R})$ *and* $\mathsf{G}^{ref} = \mathsf{BuildGame}(\mathtt{Z}^{ref}, \mathtt{R}^{ref})$*, then:*

(i) $(\mathtt{Z}^{ref}, \mathtt{R}^{ref})$ *is a reachability graph for* (P, F)*;*

(ii) $p_{\mathsf{G}}^{lb,\star}(\mathtt{Z}_F) \leqslant p_{\mathsf{G}^{ref}}^{lb,\star}(\mathtt{Z}_F)$ *and* $p_{\mathsf{G}^{ref}}^{ub,\star}(\mathtt{Z}_F) \leqslant p_{\mathsf{G}}^{ub,\star}(\mathtt{Z}_F)$ *for* $\star \in \{\min, \max\}$*.*

This refinement scheme, applied in a iterative manner, provides a way of computing exact values for minimum or maximum reachability probabilities of a PTA. This algorithm, outlined in Figure 6, starts with the reachability graph constructed through forwards reachability and then repeatedly: (i) builds a stochastic game; (ii) solves the game to

[2] From the results of [14] such a state exists when the bounds differ in some state.

obtain lower and upper bounds; and (iii) refines the reachability graph, based on an analysis of the game. The iterative process terminates when the difference between the bounds falls below a given level of precision ε. In fact, as the following result states, this process is guaranteed to terminate, in a finite number of steps, with the precise answer.

Theorem 5.2 *Let* P *be a PTA with target* F *and* $\star \in \{\min, \max\}$. *The algorithm* $\mathsf{AbstractRefine}(\mathsf{P}, F, \star, 0)$ *(see Figure 6) terminates after a finite number of steps and returns* $[p_{\mathsf{G}}^{lb,\star}, p_{\mathsf{G}}^{ub,\star}]$ *where* $p_{\mathsf{G}}^{lb,\star} = p_{\mathsf{P}}^{\star}(F) = p_{\mathsf{G}}^{ub,\star}$.

Example 5.3 *We return to our running example (see Figures 2 and 4) and consider the refinement of* $(l_1, x{=}y)$, *from which the lower and upper bounds on the maximum probability of reaching location* l_3 *are 0 and 1. The player 1 strategies (see Example 4.4) to achieve these bounds select* $\Theta_{lb} = \{\theta_b\}$ *and* $\Theta_{ub} = \{\theta_b, \theta_c\}$, *respectively. The validity conditions for these choices are* $(x{=}y{>}0)$ *and* $(x{=}y{=}0)$, *and hence* $(l_1, x{=}y)$ *is divided into* $\mathsf{z}_1 = (l_1, x{=}y{>}0)$ *and* $\mathsf{z}_2 = (l_1, x{=}y)$.

We then update the set $\mathtt{R}$, *as described in Figure 5, splitting symbolic transitions whose source or target is* $(l_1, x{=}y)$. *For example,* θ_a, θ_b *and* θ_c *(see Example 4.2) are split into, for* $i = 1, 2$:

$$\theta_a^i{=}\big((l_0, x{=}y), a, \langle \mathsf{z}_i, (l_2, x{\leqslant}y)\rangle\big),\ \theta_b^i{=}\big(\mathsf{z}_i, b, \langle \mathsf{z}_i\rangle\big) \text{ and } \theta_c^i{=}\big(\mathsf{z}_i, c, \langle (l_3, x{=}y{=}0)\rangle\big).$$

After removing θ_c^2, *which is not valid, the resulting stochastic game is shown in Figure 4(b). While this still yields bounds of* $[0,1]$ *for the initial state, two subsequent refinement tighten this to* $[0.6, 1.0]$ *and then* $[0.6, 0.6]$.

6 Experimental Results

Implementation. We have implemented a prototype PTA model checker based on the techniques in this paper. It uses difference-bound matrices (DBMs) to represent zones. Since refinement can introduce non-convex zones, we also employ lists of DBMs. Our tool takes a textual description of a PTA (or the parallel composition of several PTAs) and a set of target locations. It then executes the abstraction-refinement loop described in Section 5 to compute either the minimum or maximum reachability probability.

Several aspects of the abstraction-refinement implementation merit further discussion. In particular, the refinement process presented in Section 5 discusses the refinement of a single symbolic state. Because each refinement requires a potentially expensive numerical solution phase, an efficient scheme to select which state (or states) are to be split is essential. In fact, we found it possible to obtain very good performance with relatively simple heuristics. In the results presented here, we simply refine all states for which the lower and upper bounds differ.

Our implementation includes several useful optimisations. Firstly, we modify the BuildGame algorithm so that it only rebuilds states of a stochastic game that have actually been modified during refinement. Secondly, we use the techniques described in [11] to re-use numerical results between refinement iterations, reducing the amount of numerical solution required.

Case study (parameters) [min / max]		Game-based verification			Backwards reachability [18]		Digital clocks [16]		Min/Max reachability probability
		Iters	States	Time (s)	States	Time (s)	States	Time (s)	
csma	2 4	10	6,476	**3.9**	243	20.7	n/a	n/a	0.143555
(*max_backoff*	2 8	10	18,196	**8.9**	575	77.8	n/a	n/a	0.005259
collisions)	4 4	10	34,826	**20.5**	303	1443.7	n/a	n/a	0.076904
[max]	4 8	10	239,298	**431.4**	time out	time out	n/a	n/a	1.65e-5
csma	∞	0	117	**0.2**	0	8.7	5240	21.2	1.0
abst	1000	0	6,392	**1.9**	366	68.2	1,876,105	71.2	0.0
(*deadline*)	2000	37	24,173	**20.7**	722	367.8	6,570,692	651.8	0.869791
[min]	3000	76	79,608	**448.0**	1,736	1436.3	11,780,692	1951.9	0.999820
firewire	∞	0	257	**0.7**	127	26.4	212,268	39.7	1.0
(*deadline*)	25	0	1,369	**2.0**	1,004	839.5	14,089,691	324.6	0.5
[min]	50	17	4,215	**10.6**	3,096	3149.9	time out	time out	0.78125
	75	34	10,252	**83.4**	time out	time out	mem out	mem out	0.931641
firewire	∞	0	10	**0.03**	0	1.0	776	0.3	1.0
abst	50	7	205	**0.25**	63	2.4	298,010	14.5	0.78125
(*deadline*)	100	19	1,023	**1.76**	180	3.8	686,008	36.4	0.974731
[min]	200	40	9,059	**26.1**	640	26.4	1,462,010	149.2	0.999630
zeroconf	∞	0	26	**0.17**	19	0.22	357	1.69	0.001302
(*deadline*)	100	0	132	**0.16**	15	0.32	8,423	0.93	6.52e-4
[max]	150	13	380	**0.44**	101	0.72	23,888	1.71	0.001073
	200	17	670	**0.73**	274	4.77	41,713	2.92	0.001222
nrp	∞	0	5	**0.04**	0	0.70	n/a	n/a	1.0
honest	40	19	428	**1.80**	33	5.25	n/a	n/a	0.612580
(*deadline*)	80	39	1,448	**3.56**	63	6.18	n/a	n/a	0.864915
[min]	100	49	2,183	**5.35**	78	6.97	n/a	n/a	0.920234
nrp	∞	11	351	**1.3**	62	1.5	n/a	n/a	0.105658
malicious	5	3	1,663	**1.5**	75	2.9	n/a	n/a	0.1
(*deadline*)	10	15	8,080	**11.1**	408	117.3	n/a	n/a	0.105444
[max]	20	7	49,622	**218.1**	1,108	1606.5	n/a	n/a	0.105657

Table 1: Performance statistics and comparisons for game-based PTA verification

Experimental results. We evaluate our implementation on 7 large PTA case studies from the literature: (i) *csma* and *csma abst*, two models of the IEEE 802.3 CSMA/CD protocol; (ii) *firewire* and *firewire abst*, two models of the IEEE 1394 FireWire root contention protocol; (iii) *zeroconf*, the Zeroconf network configuration protocol; and (iv) *nrp honest* and *nrp malicious*, two model of Markowitch & Roggeman's non-repudiation protocol. Full details of all these case studies, their parameters, and the properties checked are available.[3]

We present a comparison of our implementation with the two other existing techniques for reachability analysis of PTAs: *backwards reachability* [18] and *digital clocks* [16]. For the former, we use the implementation of [18] which uses PRISM as a back-end to analyse MDP. For the latter, we use a simple language-level translation. We do not consider the MDP-based forwards reachability algorithm [17, 6] since this does compute exact probability values and is thus not directly comparable. All experiments were run on a 2GHz PC with 2GB RAM. Any run exceeding a time-limit of 1 hour was disregarded.

Table 1 summarises the experimental results. We give, for each PTA and each applicable analysis technique,[4] the total time required and the size of the probabilistic model

[3] http://www.prismmodelchecker.org/files/formats09/

[4] The digital clocks approach is not applicable to several of the case studies since the PTAs contain zones with strict constraints.

constructed. For backwards reachability and digital clocks, this model is an MDP; for our approach, it is a stochastic game (we give the size of the final game constructed during abstraction-refinement). For backwards reachability, the time given includes both generation of an MDP and its solution in PRISM; for digital clocks, the value is just the solution time in PRISM. For our game-based verification approach, we give the total time for all steps: reachability graph generation and multiple iterations of game construction, solution and analysis. The number of refinement steps required is also shown; in all cases, we refine until precise values are obtained (i.e. $\varepsilon=0$). Finally, Table 1 also gives the actual reachability probability for each model checking query and whether this a minimum or maximum value.

Analysis of the results. Our game-based approach to PTA verification performs extremely well. In all cases, it is faster than both backwards reachability and digital clocks, often by several orders of magnitude. We are also able to analyse PTAs too large to be verified using the digital clocks approach.

In terms of the size of the probabilistic models generated by the three techniques, we find that backwards reachability usually yields the smallest state spaces. This is because it only considers symbolic states for which the required probability is greater than 0. Thanks to the fact that our approach avoids some of the complex zone operations required for backwards reachability, we are able to consistently outperform it, despite this fact. On PTAs with a very small number of clocks (e.g. *firewire abst* has only 2), the overhead of these complex operations is reduced and backwards reachability performs better. By contrast, for PTAs with more clocks (*firewire* has 7 and *csma* has 5), the opposite is true.

The reason that our game-based technique outperforms the digital clocks approach is that the latter generates models with much larger state spaces, which are slow to analyse, even with the efficient symbolic techniques of PRISM.

7 Conclusions

We have presented a novel technique for the verification of probabilistic automata, based on the use of two-player stochastic games to represent abstractions of their semantics. Our approach generates lower and upper bounds for either minimum or maximum reachability probabilities and then iteratively refines the game to compute the exact values in a finite number of steps. We have implemented this process and shown that it outperforms existing PTA verification techniques on a wide range of large case studies.

Our approach can easily be extended to compute expected-reward properties for the case where rewards are associated with transitions of a PTA. Furthermore, we plan to adapt our techniques to compute lower and upper bounds on more general classes of rewards properties. Another direction of future work is the investigation of improved abstraction-refinement schemes. The simple approach adopted in this paper works very well but we anticipate that there is considerable scope for improving performance further in this way. Finally, we also plan to apply this approach to the verification of real-time properties of software.

Acknowledgments

The authors are supported in part by EPSRC grants EP/D07956X and EP/D076625.

References

[1] R. Alur, C. Courcoubetis, and D. Dill. Model-checking in dense real-time. *Inf. and Comp.*, 104(1):2–34, 1993.

[2] D. Beauquier. Probabilistic timed automata. *Theoretical Computer Science*, 292(1):65–84, 2003.

[3] P. Bouyer. Untameable timed automata! In *Proc. STACS'03*, volume 2607 of *LNCS*, pages 620–631. Springer, 2003.

[4] T. Chen, T. Han, and J.-P. Katoen. Time-abstracting bisimulation for probabilistic timed automata. In *Proc. TASE'08*, pages 177–184. IEEE CS Press, 2008.

[5] A. Condon. The complexity of stochastic games. *Inf. and Comp.*, 96(2):203–224, 1992.

[6] C. Daws, M. Kwiatkowska, and G. Norman. Automatic verification of the IEEE 1394 root contention protocol with KRONOS and PRISM. *International Journal on Software Tools for Technology Transfer (STTT)*, 5(2–3):221–236, 2004.

[7] C. Daws, A. Olivero, S. Tripakis, and S. Yovine. The tool Kronos. In *Hybrid Systems III*, volume 1066 of *LNCS*, pages 208–219. Springer, 1996.

[8] H. Dierks, S. Kupferschmid, and K. Larsen. Automatic abstraction refinement for timed automata. In *Proc. FORMATS'07*, volume 4763 of *LNCS*, pages 114–129. Springer, 2007.

[9] T. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Inf. and Comp.*, 111(2):193–244, 1994.

[10] H. Jensen. Model checking probabilistic real time systems. In *Proc. 7th Nordic Workshop on Programming Theory*, pages 247–261, 1996.

[11] M. Kattenbelt, M. Kwiatkowska, G. Norman, and D. Parker. A game-based abstraction-refinement framework for Markov decision processes. Technical Report RR-08-06, Oxford University Computing Laboratory, February 2008.

[12] M. Kattenbelt, M. Kwiatkowska, G. Norman, and D. Parker. Abstraction refinement for probabilistic software. In *Proc. VMCAI'09*, volume 5403 of *LNCS*, pages 182–197. Springer, 2009.

[13] S. Kemper and A. Platzer. SAT-based abstraction refinement for real-time systems. In *Proc. FACS 2006*, volume 182 of *ENTCS*, pages 107–122, 2007.

[14] M. Kwiatkowska, G. Norman, and D. Parker. Game-based abstraction for Markov decision processes. In *Proc. QEST'06*, pages 157–166. IEEE CS Press, 2006.

[15] M. Kwiatkowska, G. Norman, and D. Parker. Stochastic games for verication of probabilistic timed automata. In *Proc. 7th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'09)*, LNCS. Springer, 2009. To appear.

[16] M. Kwiatkowska, G. Norman, D. Parker, and J. Sproston. Performance analysis of probabilistic timed automata using digital clocks. *Formal Methods in System Design*, 29:33–78, 2006.

[17] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. *Theoretical Computer Science*, 282:101–150, 2002.

[18] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang. Symbolic model checking for probabilistic timed automata. *Inf. and Comp.*, 205(7):1027–1077, 2007.

[19] K. Larsen, P. Pettersson, and W. Yi. UPPAAL in a nutshell. *International Journal on Software Tools for Technology Transfer*, 1(1-2):134–152, 1997.

[20] M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley and Sons, 1994.

[21] L. Shapley. Stochastic games. In *Proc. National Academy of Science*, volume 39, pages 1095–1100, 1953.

[22] M. Sorea. Lazy approximation for dense real-time systems. In *Proc. FORMATS/FTRTFT'04*, volume 3253 of *LNCS*, pages 363–378. Springer, 2004.

[23] S. Tripakis. *The formal analysis of timed systems in practice.* PhD thesis, Université Joseph Fourier, 1998.

[24] S. Tripakis. Verifying progress in timed systems. In *Proc. ARTS'99*, volume 1601 of *LNCS*, pages 299–314. Springer, 1999.

[25] S. Tripakis, S. Yovine, and A. Bouajjani. Checking timed Büchi automata emptiness efficiently. *Formal Methods in System Design*, 26(3):267–292, 2005.

Appendix

This appendix contains provides proofs of the four theorems stated in the paper. Throughout, we fix a PTA $\mathsf{P} = (L, \bar{l}, Act, inv, enab, prob)$ with MDP semantics $[\![\mathsf{P}]\!] = (S, \overline{S}, \mathbb{R}{\times}Act, Steps_{\mathsf{P}})$ and target $F \subseteq L$.

A Proof of Theorem 4.1

Let $(\mathtt{Z}, \mathtt{R})$ be a reachability graph of (P, F) and $\mathsf{M} = (\mathtt{Z}, \overline{\mathtt{Z}}, \mathtt{R}, Steps_{\mathsf{M}})$ be the MDP constructed from $\mathsf{BuildMDP}(\mathtt{Z}, \mathtt{R})$. The proof of Theorem 4.1 follows similarly to [17] and relies on proving that for any $s \in S$ and $\mathtt{z} \in \mathtt{Z}$ such that $s \in \mathtt{z}$:

$$p^{\min}_{\mathtt{z}}(\mathtt{Z}_F) \leqslant p^{\min}_s(S_F) \;\text{ and }\; p^{\max}_s(S_F) \leqslant p^{\max}_{\mathtt{z}}(\mathtt{Z}_F)\,.$$

This is a direct result of the following lemma.

Lemma A.1 *For any adversary A of $[\![\mathsf{P}]\!]$ and $s \in S$, there exists an adversary B_A of M where $p^A_s(S_F) = p^{B_A}_{\mathtt{z}}(\mathtt{Z}_F)$ for all $\mathtt{z} \in \mathtt{Z}$ such that $s \in \mathtt{z}$.*

Proof A.2 *Consider any adversary A of $[\![\mathsf{P}]\!]$, $s \in S$ and $\mathtt{z} \in \mathtt{Z}$ such that $s \in \mathtt{z}$. We construct the adversary B_A by matching the transitions in $[\![\mathsf{P}]\!]$ with transitions in M such that the targets of $[\![\mathsf{P}]\!]$ are elements of the targets of M. Supposing, in state s, under A the action $(t, a) \in \mathbb{R}{\times}Act$ is chosen, that is the transition*

$$s \xrightarrow{(t,a)} \langle s_1, \ldots, s_n \rangle$$

is performed, then since $(\mathtt{Z}, \mathtt{R})$ is a reachability graph (see Definition 5) there exists a symbolic transition $\theta = (\mathtt{z}, a, \langle \mathtt{z}_1, \ldots, \mathtt{z}_n \rangle) \in \mathtt{R}$ such that $s_i \in \mathtt{z}_i$ for all $1{\leqslant}i{\leqslant}n$ and we let B_A choose θ in state $\mathtt{z}$ of M. The fact that the reachability probabilities are the same for A and B_A then follows from the fact that the corresponding transitions are constructed from the same distribution, namely $prob(l, a)$ when s is of the form (l, v) for some $v \in \mathbb{R}^{\mathcal{X}}$. □

B Proof of Theorem 4.3

Let $(\mathtt{Z}, \mathtt{R})$ be a reachability graph of (P, F) and $\mathsf{G} = (\mathtt{Z}, \overline{\mathtt{Z}}, 2^{\mathtt{R}}, Steps_{\mathsf{G}})$ be the stochastic game constructed from $\mathsf{BuildGame}(\mathtt{Z}, \mathtt{R})$. Before we give the proof of Theorem 4.3 we require the following lemmas.

Lemma B.1 *For any adversary A of $[\![\mathsf{P}]\!]$ and $s \in S$ there exists a strategy pair (σ_1, σ_2) of G where $p^A_s(S_F) = p^{\sigma_1,\sigma_2}_{\mathtt{z}}(\mathtt{Z}_F)$ for all $\mathtt{z} \in \mathtt{Z}$ such that $s \in \mathtt{z}$.*

Proof B.2 *Consider any adversary A of $[\![\mathsf{P}]\!]$, $s = (l, v) \in S$ and $\mathtt{z} \in \mathtt{Z}$ such that $s \in \mathtt{z}$. The proof follows similarly to Lemma A.1, except that we construct a strategy pair of the game G which mimics the choices made by B_A. More precisely, if B_A chooses θ, then we let σ_1 choose any Θ such that $\theta \in \Theta$ and let σ_2 choose θ from Θ. The existence of such* *a Θ follows from the fact that we keep only valid symbolic transitions. Now, for such a strategy pair, since they mimic the choices of B_A we have:*

$$p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) = p_{\mathtt{z}}^{B_A}(\mathtt{Z}_F)$$

which, combined with Lemma A.1, completes the proof. □

Lemma B.3 *for any abstract state $\mathtt{z} \in \mathtt{Z}$ and player 2 strategy σ_2 of G there exists an adversary A of $[\![\mathsf{P}]\!]$ where:*

$$\inf_{\sigma_1} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) \leqslant p_s^A(S_F) \text{ and } \sup_{\sigma_1} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) \geqslant p_s^A(S_F)$$

for all $s \in S$ such that $s \in \mathtt{z}$.

Proof B.4 *Consider any player 2 strategy σ_2 of G, $\mathtt{z} \in \mathtt{Z}$ and $s = (l, v) \in \mathtt{z}$. By the assumptions we make on PTAs, $s \xrightarrow{(t,a)} \langle s_1, \ldots, s_n \rangle$ for some $(t, a) \in \mathbb{R} \times Act$. Now, since $(\mathtt{Z}, \mathtt{R})$ is a reachability graph of (P, F), there exists $\theta = (\mathtt{z}, a, \langle \mathtt{z}_1, \ldots, \mathtt{z}_n \rangle) \in \mathtt{R}$ such that $s_i \in \mathtt{z}_i$ for all $1 \leqslant i \leqslant n$. It then follows by definition that $v \lhd valid(\theta)$, and hence there exists Θ such that $v \lhd valid(\Theta)$. Now we let σ_1 be the player 1 strategy which chooses Θ. Supposing σ_2 chooses some $\theta' = (\mathtt{z}, a', \langle \mathtt{z}'_1, \ldots, \mathtt{z}'_m \rangle) \in \Theta$ then, since $v \lhd valid(\Theta)$, it follows by definition that $v \lhd valid(\theta')$. Furthermore, since $v \in valid(\theta')$, there exists $t' \in \mathbb{R}$ such that $s \xrightarrow{(t',a')} \langle s'_1, \ldots, s'_m \rangle$ and $s'_i \in \mathtt{z}'_i$ for all $1 \leqslant i \leqslant m$. Now we construct A to choose (a', t') in state s. Repeating this process inductively on the path of the game we arrive at a player 1 strategy σ_1 and adversary A of $[\![\mathsf{P}]\!]$ such that*

$$p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) = p_s^A(S_F)$$

which is sufficient to complete the proof. □

Proof B.5 (of Theorem 4.3) *From Lemma B.1 it follows that for any $s \in S$:*

$$\inf_{\sigma_1,\sigma_2} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) \leqslant \inf_A P_s^A(S_F)$$
$$\sup_A p_s^A(S_F) \leqslant \sup_{\sigma_1,\sigma_2} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F)$$

for all $\mathtt{z} \in \mathtt{Z}$ such that $s \in \mathtt{z}$, and hence $p_{\mathsf{G}}^{lb,\min}(\mathtt{Z}_F) \leqslant p_{\mathsf{P}}^{\min}(S_F)$ and $p_{\mathsf{P}}^{\max}(S_F) \leqslant p_{\mathsf{G}}^{ub,\max}(\mathtt{Z}_F)$. On the other hand, using Lemma B.3, we have for any $s \in S$ and $\mathtt{z} \in \mathtt{Z}$ such that $s \in z$:

$$\inf_A p_s^A(S_F) \leqslant \inf_{\sigma_2} \sup_{\sigma_1} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) = \sup_{\sigma_1} \inf_{\sigma_2} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F)$$

where the second step follows from properties of stochastic games [5]. Similarly, we can show that:

$$\inf_{\sigma_1} \sup_{\sigma_2} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) \leqslant \sup_A p_s^A(S_F)$$

and therefore $p_{\mathsf{P}}^{\min}(\mathtt{Z}_F) \leqslant p_{\mathsf{G}}^{ub,\min}(\mathtt{Z}_F)$ and $p_{\mathsf{G}}^{lb,\max}(\mathtt{Z}_F) \leqslant p_{\mathsf{P}}^{\max}(S_F)$ which completes the proof.

C Proof of Theorem 5.1

Let $(\mathtt{Z},\mathtt{R})$ and $\mathsf{G} = (\mathtt{Z},\overline{\mathtt{Z}},2^{\mathtt{R}},\mathit{Steps}_{\mathsf{G}})$ be the reachability graph and game before refinement, $(\mathtt{Z}^{ref},\mathtt{R}^{ref})$ be the result of applying algorithm Refine to $(\mathtt{Z},\mathtt{R})$ and $\mathsf{G}^{ref} = (\mathtt{Z}^{ref},\overline{\mathtt{Z}}^{ref},2^{\mathtt{R}^{ref}},\mathit{Steps}_{\mathsf{G}}^{ref})$ be the game returned by BuildGame$(\mathtt{Z}^{ref},\mathtt{R}^{ref})$. Before we give the proof we require the following lemmas.

Lemma C.1 *If $\mathtt{z}^{ref} \in \mathtt{Z}^{ref}$, $(\mathtt{z}^{ref},a,\langle \mathtt{z}_1^{ref},\dots,\mathtt{z}_n^{ref}\rangle) \in \mathtt{R}(\mathtt{z}^{ref})$ and $\mathtt{z} \in \mathtt{Z}$ such that $\mathtt{z}^{ref}\subseteq\mathtt{z}$, then there exists $(\mathtt{z},a,\langle \mathtt{z}_1,\dots,\mathtt{z}_n\rangle) \in \mathtt{R}$ such that $\mathtt{z}_i^{ref}\subseteq\mathtt{z}_i$ for all $1\leqslant i\leqslant n$.*

Proof C.2 *Consider any $\mathtt{z}^{ref} \in \mathtt{Z}^{ref}$, $(\mathtt{z}^{ref},a,\langle \mathtt{z}_1^{ref},\dots,\mathtt{z}_n^{ref}\rangle) \in \mathtt{R}(\mathtt{z}^{ref})$ and $\mathtt{z} \in \mathtt{Z}$ such that $\mathtt{z}^{ref}\subseteq\mathtt{z}$. We split the proof into two cases.*

- *If $\mathtt{z}^{ref} \in \mathtt{Z}$, then by construction $\mathtt{z}^{ref}{=}\mathtt{z}$, and therefore we have that either $(\mathtt{z}^{ref},a,\langle \mathtt{z}_1^{ref},\dots,\mathtt{z}_n^{ref}\rangle) \in \mathtt{R}(\mathtt{z})$ in which case the lemma holds, or there exists $(\mathtt{z},a,\langle \mathtt{z}_1,\dots,\mathtt{z}_n\rangle \in \mathtt{R}(\mathtt{z})$ from which $(\mathtt{z}^{ref},a,\langle \mathtt{z}_1^{ref},\dots,\mathtt{z}_n^{ref}\rangle)$ was constructed. In the second case, it follows from* Refine *(see Figure 5) that $\mathtt{z}_i^{ref} \in \mathtt{z}_i$ for all $1\leqslant i\leqslant n$ as required.*
- *If $\mathtt{z}^{ref} \notin \mathtt{Z}$, then for $\mathtt{z}^{ref}\subseteq\mathtt{z}$ it follows that $\mathtt{z}^{ref}$ was formed by splitting $\mathtt{z}$. Hence, there exists a symbolic transition $(\mathtt{z},a,\langle \mathtt{z}_1,\dots,\mathtt{z}_n\rangle \in \mathtt{R}(\mathtt{z})$ which was used to construct $(\mathtt{z}^{ref},a,\langle \mathtt{z}_1^{ref},\dots,\mathtt{z}_n^{ref}\rangle)$. It follows from this construction that $\mathtt{z}_i^{ref} \in \mathtt{z}_i$ for all $1\leqslant i\leqslant n$ as required.*

Since these are the only cases to consider, the lemma holds. □

Lemma C.3 *For any strategy pair $(\sigma_1^{ref},\sigma_2^{ref})$ of G^{ref} and $\mathtt{z}^{ref} \in \mathtt{Z}^{ref}$ there exists a strategy pair (σ_1,σ_2) of G where $p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) = p_{\mathtt{z}^{ref}}^{\sigma_1^{ref},\sigma_2^{ref}}(\mathtt{Z}_F)$ for all $\mathtt{z} \in \mathtt{Z}$ such that $\mathtt{z}^{ref}\subseteq\mathtt{z}$.*

Proof C.4 *Consider any any strategy pair $(\sigma_1^{ref},\sigma_2^{ref})$ of G^{ref}, $\mathtt{z}^{ref} \in \mathtt{Z}^{ref}$ and $\mathtt{z} \in \mathtt{Z}$ such that $\mathtt{z}^{ref}\subseteq\mathtt{z}$. We construct the strategy pair (σ_1,σ_2) of G so that in state $\mathtt{z}$ they match the choice made by the pair $(\sigma_1^{ref},\sigma_2^{ref})$ in $\mathtt{z}^{ref}$. If in $\mathtt{z}^{ref}$ the choice of $(\sigma_1^{ref},\sigma_2^{ref})$ corresponds to the symbolic transition $(\mathtt{z}^{ref},a,\langle \mathtt{z}_1^{ref},\dots,\mathtt{z}_n^{ref}\rangle)$, then, using Lemma C.1, there exists $(\mathtt{z},a,\langle \mathtt{z}_1,\dots,\mathtt{z}_n\rangle)$ of G such that $\mathtt{z}_i^{ref}\subseteq\mathtt{z}_i$ for all $1\leqslant i\leqslant n$ and we construct (σ_1,σ_2) such that their choice corresponds to this symbolic transition. The remainder of the proof then follows in an identical fashion to Lemma B.1.* □

Lemma C.5 *For any $\mathtt{z} \in \mathtt{Z}$ and player 2 strategy σ_2 of G there exists a strategy pair $(\sigma_1^{ref},\sigma_2^{ref})$ of G^{ref} where*

$$\inf_{\sigma_1} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) \leqslant p_{\mathtt{z}^{ref}}^{\sigma_1^{ref},\sigma_2^{ref}}(\mathtt{Z}_F) \ \ \text{and} \ \ p_{\mathtt{z}^{ref}}^{\sigma_1^{ref},\sigma_2^{ref}}(\mathtt{Z}_F) \leqslant \sup_{\sigma_1} p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F)$$

for all $\mathtt{z}^{ref}$ such that $\mathtt{z}^{ref}\subseteq\mathtt{z}$.

Proof C.6 *Given a player 2 strategy σ_2^{ref} of G^{ref} the proof follows by constructing a player 1 strategy σ_1^{ref} of G^{ref} and strategy pair (σ_1, σ_2) of G such that:*

$$p_{\mathtt{z}}^{\sigma_1,\sigma_2}(\mathtt{Z}_F) = p_{\mathtt{z}^{ref}}^{\sigma_1^{ref},\sigma_2^{ref}}(\mathtt{Z}_F)$$

for all $\mathtt{z}^{ref}$ such that $\mathtt{z}^{ref} \subseteq \mathtt{z}$. This follows similarly to Lemma B.3 using Lemma C.1 to construct the choices of σ_1^{ref} and (σ_1, σ_2). □

Proof C.7 (of Theorem 5.1) *The fact that $(\mathtt{Z}^{ref}, \mathtt{R}^{ref})$ is a reachability graph follows from* Refine *since we only split symbolic states and remove symbolic transitions which are not valid. The second part of the proof follows similarly to the proof of Theorem 4.3 using Lemmas C.3 and C.5 instead of Lemmas B.1 and B.3.* □

D Proof of Theorem 5.2

Proof D.1 (of Theorem 5.2) *The proof is based on the correctness of the region graph construction for timed automata [1]. More precisely, the proof follows by combining the following results:*

- *the refinement scheme always divides zones into zones which are proper subsets (see Section 5);*
- *any zone is a union of regions;*
- *the refinement scheme cannot split regions;*
- *there are only finitely many (c-closed) regions;*
- *if $(\mathtt{Z}, \mathtt{R})$ is a reachability graph where all zones appearing in $\mathtt{Z}$ are regions, then for any $(l, \zeta) \in \mathtt{Z}$ we have valid$(\theta) = \zeta$ for all $\theta \in \mathtt{R}(l, \zeta)$;*
- *if $\mathsf{G} = (\mathtt{Z}, \overline{\mathtt{Z}}, 2^{\mathtt{R}}, Steps_{\mathsf{G}})$ is the stochastic game returned by* BuildGame$(\mathtt{Z}, \mathtt{R})$*, then $Steps_{\mathsf{G}}(\mathtt{z}, \Theta)$ is a singleton set for all $\mathtt{z} \in \mathtt{Z}$ and available actions Θ;*
- *if $Steps_{\mathsf{G}}(\mathtt{z}, \Theta)$ is always a singleton set, then $p_{\mathsf{G}}^{lb,\star}(\mathtt{Z}_F) = p_{\mathsf{G}}^{ub,\star}(\mathtt{Z}_F)$;*
- *from Theorem 5.1 we have $p_{\mathsf{G}}^{lb,\star}(\mathtt{Z}_F) \leqslant p_{\mathsf{P}}^{\star}(S_F) \leqslant p_{\mathsf{G}}^{ub,\star}(\mathtt{Z}_F)$.* □