Verification and Control of Partially Observable Probabilistic Real-Time Systems

Gethin Norman[1], David Parker[2], and Xueyi Zou[3]

[1] School of Computing Science, University of Glasgow, UK
[2] School of Computer Science, University of Birmingham, UK
[3] Department of Computer Science, University of York, UK

Abstract. We propose automated techniques for the verification and control of probabilistic real-time systems that are only partially observable. To formally model such systems, we define an extension of probabilistic timed automata in which local states are partially visible to an observer or controller. We give a probabilistic temporal logic that can express a range of quantitative properties of these models, relating to the probability of an event's occurrence or the expected value of a reward measure. We then propose techniques to either verify that such a property holds or to synthesise a controller for the model which makes it true. Our approach is based on an integer discretisation of the model's dense-time behaviour and a grid-based abstraction of the uncountable belief space induced by partial observability. The latter is necessarily approximate since the underlying problem is undecidable, however we show how both lower and upper bounds on numerical results can be generated. We illustrate the effectiveness of the approach by implementing it in the PRISM model checker and applying it to several case studies, from the domains of computer security and task scheduling.

1 Introduction

Guaranteeing the correctness of complex computerised systems often needs to take into account quantitative aspects of system behaviour. This includes the modelling of *probabilistic* phenomena, such as failure rates for physical components, uncertainty arising from unreliable sensing of a continuous environment, or the explicit use of randomisation to break symmetry. It also includes *real-time* characteristics, such as time-outs or delays in communication or security protocols. To further complicate matters, such systems are often *nondeterministic* because their behaviour depends on inputs or instructions from some external entity such as a controller or scheduler.

Automated verification techniques such as probabilistic model checking have been successfully used to analyse quantitative properties of probabilistic, real-time systems across a variety of application domains, including wireless communication protocols, computer security and task scheduling. These systems are commonly modelled using *Markov decision processes* (MDPs), if assuming a discrete notion of time, or *probabilistic timed automata* (PTAs), if using a dense model of time. On these models, we can consider two problems: *verification* that it satisfies some formally specified property for any possible resolution of nondeterminism; or, dually, *synthesis* of a controller (i.e., a means to resolve nondeterminism) under which a property is guaranteed. For either case, an important consideration is the extent to which the system's state is *observable* to the entity controlling it. For example, to verify that a security protocol is functioning correctly, it may be essential to model the fact that some data held by a participant is not externally visible, or, when synthesising a controller for a robot, the controller may not be implementable in practice if it bases its decisions on information that cannot be physically observed.

Partially observable MDPs (POMDPs) are a natural way to extend MDPs in order to tackle this problem. However, the analysis of POMDPs is considerably more difficult than MDPs since key problems are undecidable [24]. A variety of verification problems have been studied for these models (see e.g., [1,3,11]) and the use of POMDPs is common in fields such as AI and planning [8], but there is limited progress in the development of practical techniques for probabilistic verification in this area, or exploration of their applicability.

In this paper, we present novel techniques for verification and control of probabilistic real-time systems under partial observability. We propose a model called *partially observable probabilistic timed automata* (POPTAs), which extends the existing model of PTAs with a notion of partial observability. The semantics of a POPTA is an infinite-state POMDP. We then define a temporal logic, based on [27], to express properties of POPTAs relating to the probability of an event or the expected value of various reward measures. Nondeterminism in a POPTA is resolved by a *strategy* that decides which actions to take and when to take them, based only on the history of observations (not states). The core problems we address are how to *verify* that a temporal logic property holds for all possible strategies, and how to *synthesise* a strategy under which the property holds.

In order to achieve this, we use a combination of techniques. First, we develop a *digital clocks* discretisation for POPTAs, which extends the existing notion for PTAs [20], and reduces the analysis to a *finite* POMDP. We define the conditions under which properties in our temporal logic are preserved and prove the correctness of the reduction. To analyse the resulting POMDP, we use grid-based techniques [23,29], which transform it to a fully observable but continuous-space MDP and then approximate its solution based on a finite set of grid points. We use this to construct and solve a strategy for the POMDP. The result is a pair of lower and upper bounds on the property of interest for the original POPTA. If the results are not precise enough, we can refine the grid and repeat.

We implemented these methods in a prototype tool based on PRISM [19], and investigated their applicability by developing three case studies: a non-repudiation protocol, a task scheduling problem and a covert channel prevention device (the NRL pump). Despite the complexity of POMDP solution methods, we show that useful results can be obtained, often with precise bounds. In each case study, nondeterminism, probability, real-time behaviour *and* partial observability are all crucial ingredients to the analysis, a combination not supported by any existing techniques or tools.

Related work. POMDPs are common in fields such as AI and planning, and have many applications [8]. They have also been studied in the verification community, e.g. [1,3,11], establishing undecidability and complexity results for various qualitative and quantitative verification problems. Work in this area often also studies related models such as Rabin's probabilistic automata [3], which can be seen as a special case of POMDPs, and partially observable stochastic games (POSGs) [12], which generalise them. More practically oriented work includes: [15], which proposes a counterexample-driven refinement method to approximately solve MDPs in which components have partial observability of each other; and [10], which synthesises concurrent program constructs, using a search over memoryless strategies in a POSG. Theoretical results [6] and algorithms [9,14] have been developed for synthesis of partially observable timed games. In [6], it is shown that the synthesis problem is undecidable and, if the resources of the controller are fixed, decidable but prohibitively expensive. The algorithms require constraints on controllers: in [9], controllers only respond to changes made by the environment and, in [14], their structure must be fixed in advance. We are not aware of any work for probabilistic real-time models.

An extended version of this paper, with proofs, is available as [26].

2 Partially Observable Markov Decision Processes

We begin with background material on MDPs and POMDPs. Let $Dist(X)$ denote the set of discrete probability distributions over a set X, δ_x the distribution that selects $x \in X$ with probability 1, and $\mathbb{R}$ the set of non-negative real numbers.

Definition 1 (MDP). *An MDP is a tuple* $\mathsf{M}=(S, \bar{s}, A, P, R)$ *where:* S *is a set of states;* $\bar{s} \in S$ *an initial state;* A *a set of actions;* $P : S \times A \to Dist(S)$ *a (partial) probabilistic transition function; and* $R : S \times A \to \mathbb{R}$ *a reward function.*

Each state s of an MDP M has a set $A(s) \overset{\text{def}}{=} \{a \in A \mid P(s,a) \text{ is defined}\}$ of *enabled* actions. If action $a \in A(s)$ is selected, then the probability of moving to state s' is $P(s,a)(s')$ and a reward of $R(s,a)$ is accumulated in doing so. A *path* of M is a finite or infinite sequence $\omega = s_0 a_0 s_1 a_1 \cdots$, where $s_i \in S$, $a_i \in A(s_i)$ and $P(s_i, a_i)(s_{i+1}) > 0$ for all $i \in \mathbb{N}$. We write $FPaths_{\mathsf{M}}$ and $IPaths_{\mathsf{M}}$, respectively, for the set of all finite and infinite paths of M starting in the initial state $\bar{s}$.

A *strategy* of M (also called a *policy* or *scheduler*) is a way of resolving the choice of action in each state, based on the MDP's execution so far.

Definition 2 (Strategy). *A* strategy *of an MDP* $\mathsf{M}=(S, \bar{s}, A, P, R)$ *is a function* $\sigma : FPaths_{\mathsf{M}} \to Dist(A)$ *such that* $\sigma(s_0 a_0 s_1 \ldots s_n)(a) > 0$ *only if* $a \in A(s_n)$.

A strategy is *memoryless* if its choices only depend on the current state, *finite-memory* if it suffices to switch between a finite set of modes and *deterministic* if it always selects an action with probability 1. The set of strategies of M is denoted by Σ_{M}.

When M is under the control of a strategy σ, the resulting behaviour is captured by a probability measure Pr^{σ}_{M} over the infinite paths of M [18].

POMDPs. POMDPs extend MDPs by restricting the extent to which their current state can be observed, in particular by strategies that control them. In this paper (as in, e.g., [3,11]), we adopt the following notion of observability.

Definition 3 (POMDP). *A POMDP is a tuple* $\mathsf{M}{=}(S, \bar{s}, A, P, R, \mathcal{O}, obs)$ *where:* $(S, \bar{s}, A, P, R)$ *is an MDP;* $\mathcal{O}$ *is a finite set of* observations*; and* $obs : S \to \mathcal{O}$ *is a labelling of states with observations. For any states* $s, s' \in S$ *with* $obs(s){=}obs(s')$*, their enabled actions must be identical, i.e.,* $A(s){=}A(s')$.

The current state s of a POMDP cannot be directly determined, only the corresponding observation $obs(s) \in \mathcal{O}$. More general notions of observations are sometime used, e.g., that depend also on the previous action taken or are probabilistic. Our analysis of probabilistic verification case studies where partial observation is needed (see, e.g., Sec. 5) suggests that this simpler notion of observability will often suffice in practice. To ease presentation, we assume the initial state is observable, i.e., there exists $\bar{o} \in \mathcal{O}$ such that $obs(s){=}\bar{o}$ if and only if $s{=}\bar{s}$.

The notions of paths, strategies and probability measures given above for MDPs transfer directly to POMDPs. However, the set Σ_{M} of all strategies for a POMDP M only includes *observation-based strategies*, that is, strategies σ such that, for any paths $\pi = s_0 a_0 s_1 \dots s_n$ and $\pi' = s'_0 a_0{}' s'_1 \dots s'_n$ satisfying $obs(s_i) = obs(s'_i)$ and $a_i = a'_i$ for all i, we have $\sigma(\pi) = \sigma(\pi')$.

Key properties for a POMDP (or MDP) are the probability of reaching a target, and the expected reward cumulated until this occurs. Let O denote the target (e.g., a set of observations of a POMDP). Under a specific strategy σ, we denote these two properties by $Pr^{\sigma}_{\mathsf{M}}(\mathtt{F}\,O)$ and $\mathbb{E}^{\sigma}_{\mathsf{M}}(\mathtt{F}\,O)$, respectively.

Usually, we are interested in the *optimal* (minimum or maximum) values $Pr^{opt}_{\mathsf{M}}(\mathtt{F}\,O)$ and $\mathbb{E}^{opt}_{\mathsf{M}}(\mathtt{F}\,O)$, where $opt \in \{\min, \max\}$. For a MDP or POMDP M:

$$\begin{array}{ll} Pr^{\min}_{\mathsf{M}}(\mathtt{F}\,O) \overset{\text{def}}{=} \inf_{\sigma\in\Sigma_{\mathsf{M}}} Pr^{\sigma}_{\mathsf{M}}(\mathtt{F}\,O) & \quad \mathbb{E}^{\min}_{\mathsf{M}}(\mathtt{F}\,O) \overset{\text{def}}{=} \inf_{\sigma\in\Sigma_{\mathsf{M}}} \mathbb{E}^{\sigma}_{\mathsf{M}}(\mathtt{F}\,O) \\ Pr^{\max}_{\mathsf{M}}(\mathtt{F}\,O) \overset{\text{def}}{=} \sup_{\sigma\in\Sigma_{\mathsf{M}}} Pr^{\sigma}_{\mathsf{M}}(\mathtt{F}\,O) & \quad \mathbb{E}^{\max}_{\mathsf{M}}(\mathtt{F}\,O) \overset{\text{def}}{=} \sup_{\sigma\in\Sigma_{\mathsf{M}}} \mathbb{E}^{\sigma}_{\mathsf{M}}(\mathtt{F}\,O) \end{array}$$

Beliefs. For POMDPs, determining the optimal probabilities and expected rewards defined above is undecidable [24], making exact solution intractable. A useful construction, e.g., as a basis of approximate solutions, is the translation from a POMDP M to a *belief MDP* $\mathcal{B}(\mathsf{M})$, an equivalent (fully observable) MDP, whose (continuous) state space comprises *beliefs*, which are probability distributions over the state space of M. Intuitively, although we may not know which of several observationally-equivalent states we are currently in, we can determine the likelihood of being in each one, based on the probabilistic behaviour of M. A formal definition is given below.

Definition 4 (Belief MDP). *Let* $\mathsf{M}{=}(S, \bar{s}, A, P, R, \mathcal{O}, obs)$ *be a POMDP. The* belief MDP *of* M *is given by* $\mathcal{B}(\mathsf{M}){=}(Dist(S), \delta_{\bar{s}}, A, P^{\mathcal{B}}, R^{\mathcal{B}})$ *where, for any beliefs*

$b, b' \in Dist(S)$ and action $a \in A$:

$$\begin{aligned} P^{\mathcal{B}}(b,a)(b') &= \textstyle\sum_{s \in S} b(s) \cdot \Big(\sum_{o \in \mathcal{O} \wedge b^{a,o}=b'} \sum_{s' \in S \wedge obs(s')=o} P(s,a)(s') \Big) \\ R^{\mathcal{B}}(b,a) &= \textstyle\sum_{s \in S} R(s,a) \cdot b(s) \end{aligned}$$

and $b^{a,o}$ is the belief reached from b by performing a and observing o, i.e.:

$$b^{a,o}(s') \;=\; \begin{cases} \frac{\sum_{s \in S} P(s,a)(s') \cdot b(s)}{\sum_{s \in S} b(s) \cdot \left(\sum_{s'' \in S \wedge obs(s'')=o} P(s,a)(s'') \right)} & \textit{if } obs(s')=o \\ 0 & \textit{otherwise.} \end{cases}$$

The optimal values for the belief MDP equal those for the POMDP, e.g. we have:

$$Pr^{\max}_{\mathsf{M}}(\mathtt{F}\, O) = Pr^{\max}_{\mathcal{B}(\mathsf{M})}(\mathtt{F}\, T_O) \;\text{ and }\; \mathbb{E}^{\max}_{\mathsf{M}}(\mathtt{F}\, O) = \mathbb{E}^{\max}_{\mathcal{B}(\mathsf{M})}(\mathtt{F}\, T_O)$$

where $T_O = \{ b \in Dist(S) \,|\, \forall s \in S. \, (b(s)>0 \rightarrow obs(s) \in O) \}$.

3 Partially Observable Probabilistic Timed Automata

In this section, we define *partially observable probabilistic timed automata* (POPTAs), which generalise the existing model of probabilistic timed automata (PTAs) with the notion of partial observability from POMDPs explained in Sec. 2. We define the syntax of a POPTA, explain its semantics (as an infinite-state POMDP) and define and discuss the *digital clocks* semantics of a POPTA.

Time & clocks. As in classical timed automata [2], we model real-time behaviour using non-negative, real-valued variables called *clocks*, whose values increase at the same rate as real time. Assuming a finite set of clocks $\mathcal{X}$, a *clock valuation* v is a function $v : \mathcal{X} \rightarrow \mathbb{R}$ and we write $\mathbb{R}^{\mathcal{X}}$ for the set of all clock valuations. Clock valuations obtained from v by incrementing all clocks by a delay $t \in \mathbb{R}$ and by resetting a set $X \subseteq \mathcal{X}$ of clocks to zero are denoted $v+t$ and $v[X{:=}0]$, respectively, and we write $\mathbf{0}$ if all clocks are 0. A (closed, diagonal-free) *clock constraint* ζ is either a conjunction of inequalities of the form $x \leqslant c$ or $x \geqslant c$, where $x \in \mathcal{X}$ and $c \in \mathbb{N}$, or `true`. We write $v \models \zeta$ if clock valuation v satisfies clock constraint ζ and use $CC(\mathcal{X})$ for the set of all clock constraints over $\mathcal{X}$.

Syntax of POPTAs. To explain the syntax of POPTAs, we first consider the simpler model of PTAs and then show how it extends to POPTAs.

Definition 5 (PTA syntax). *A PTA is a tuple* $\mathsf{P}{=}(L, \bar{l}, \mathcal{X}, A, inv, enab, prob, r)$ *where:*

- L *is a finite set of* locations *and* $\bar{l} \in L$ *is an* initial location;
- $\mathcal{X}$ *is a finite set of* clocks *and* A *is a finite set of* actions;
- $inv : L \rightarrow CC(\mathcal{X})$ *is an* invariant condition;
- $enab : L \times A \rightarrow CC(\mathcal{X})$ *is an* enabling condition;
- $prob : L \times A \rightarrow Dist(2^{\mathcal{X}} \times L)$ *is a* probabilistic transition function;
- $r{=}(r_L, r_A)$ *is a* reward structure *where* $r_L : L \rightarrow \mathbb{R}$ *is a* location reward function *and* $r_A : L \times A \rightarrow \mathbb{R}$ *is an* action reward function.

A state of a PTA is a pair (l, v) of location $l \in L$ and clock valuation $v \in \mathbb{R}^{\mathcal{X}}$. Time $t \in \mathbb{R}$ can elapse in the state only if the invariant $inv(l)$ remains continuously satisfied while time passes and the new state is then $(l, v{+}t)$. An action a is enabled in the state if v satisfies $enab(l, a)$ and, if it is taken, then the PTA moves to location l' and resets the clocks $X \subseteq \mathcal{X}$ with probability $prob(l, a)(X, l')$. PTAs have two kinds of rewards: location rewards, which are accumulated at rate $r_L(l)$ while in location l and action rewards $r_A(l, a)$, which are accumulated when taking action a in location l. PTAs equipped with reward structures are a probabilistic extension of linearly-priced timed automata [5].

Definition 6 (POPTA syntax). *A partially observable PTA (POPTA) is a tuple* $\mathsf{P} = (L, \bar{l}, \mathcal{X}, A, inv, enab, prob, r, \mathcal{O}_L, obs_L)$ *where:*

- $(L, \bar{l}, \mathcal{X}, A, inv, enab, prob, r)$ *is a* PTA;
- $\mathcal{O}_L$ *is a finite set of* observations;
- $obs_L : L \to \mathcal{O}_L$ *is a* location observation function.

For any locations $l, l' \in L$ *with* $obs_L(l){=}obs_L(l')$*, we require that* $inv(l){=}inv(l')$ *and* $enab(l, a){=}enab(l', a)$ *for all* $a \in A$.

The final condition ensures the semantics of a POPTA yields a valid POMDP: recall states with the same observation are required to have identical available actions. Like for POMDPs, for simplicity, we also assume that the initial location is observable, i.e., there exists $\bar{o} \in \mathcal{O}_L$ such that $obs_L(l){=}\bar{o}$ if and only if $l{=}\bar{l}$.

The notion of observability for POPTAs is similar to the one for POMDPs, but applied to locations. Clocks, on the other hand, are always observable. The requirement that the same choices must be available in any observationally-equivalent states, implies the same delays must be available in observationally-equivalent states, and so unobservable clocks could not feature in invariant or enabling conditions. The inclusion of unobservable clocks would therefore necessitate modelling the system as a game with the elapse of time being under the control of a second (environment) player. The underlying semantic model would then be a partially observable stochastic game (POSG), rather than a POMDP. However, unlike POMDPs, limited progress has been made on efficient computational techniques for this model (belief space based techniques, for example, do not apply in general [12]). Even in the simpler case of non-probabilistic timed games, allowing unobservable clocks requires algorithmic analysis to restrict the class of strategies considered [9,14].

Encouragingly, however, we will later show in Sec. 5 that POPTAs with observable clocks were always sufficient for our modelling and analysis.

Restrictions on POPTAs. At this point, we need to highlight a few syntactic restrictions on the POPTAs treated in this paper. Firstly, we emphasise that clock constraints appearing in a POPTA, i.e., in its invariants and enabling conditions, are required to be *closed* (no strict inequalities) and *diagonal-free* (no comparisons of clocks). This is a standard restriction when using the digital clocks discretisation [20] which we work with in this paper.

Secondly, a specific (but minor) restriction for POPTAs is that resets can only be applied to clocks that are non-zero. The reasoning behind this is outlined later

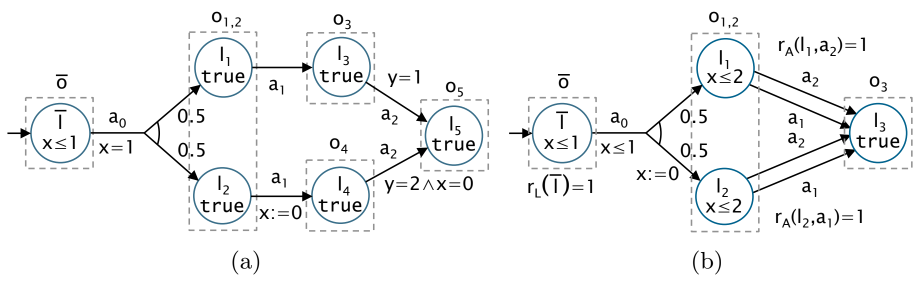


Fig. 1. Examples of partially observable PTAs (see Examples 1 and 2).

in Example 2. Checking this restriction can easily be done when exploring the discrete (digital clocks) semantics of the model – see below and Sec. 4.

Semantics of POPTAs. We now formally define the semantics of a POPTA P, which is given in terms of an infinite-state POMDP. This extends the standard semantics of a PTA [27] (as an infinite MDP) with the same notion of observability we gave in Sec. 2 for POMDPs. The semantics, $[\![\mathsf{P}]\!]_{\mathbb{T}}$, is parameterised by a *time domain* $\mathbb{T}$, giving the possible values taken by clocks. For the standard (dense-time) semantics of a POPTA, we take $\mathbb{T} = \mathbb{R}$. Later, when we discretise the model, we will re-use this definition, taking $\mathbb{T} = \mathbb{N}$. When referring to the "standard" semantics of P we will often drop the subscript $\mathbb{R}$ and write $[\![\mathsf{P}]\!]$.

Definition 7 (POPTA semantics). *Let* $\mathsf{P}=(L, \bar{l}, \mathcal{X}, A, inv, enab, prob, r, \mathcal{O}_L, obs_L)$ *be a POPTA. The* semantics *of* P*, with respect to the time domain* $\mathbb{T}$*, is the POMDP* $[\![\mathsf{P}]\!]_{\mathbb{T}}=(S, \bar{s}, A \cup \mathbb{T}, P, R, \mathcal{O}_L{\times}\mathbb{T}^{\mathcal{X}}, obs)$ *such that:*

- $S = \{(l, v) \in L{\times}\mathbb{T}^{\mathcal{X}} \mid v \models inv(l)\}$ *and* $\bar{s} = (\bar{l}, \mathbf{0})$*;*
- *for* $(l, v) \in S$ *and* $a \in A \cup \mathbb{T}$*, we have* $P((l, v), a) = \mu$ *if and only if:*
  - *(time transitions)* $a \in \mathbb{T}$*,* $\mu = \delta_{(l,v+a)}$ *and* $v{+}t \models inv(l)$ *for all* $0{\leqslant}t{\leqslant}a$*;*
  - *(action transition)* $a \in A$*,* $v \models enab(l, a)$ *and for* $(l', v') \in S$*:*

$$\mu(l', v') = \textstyle\sum_{X \subseteq \mathcal{X} \wedge v'=v[X:=0]} prob(l, a)(X, l')$$

- *for any* $(l, v) \in S$ *and* $a \in A \cup \mathbb{T}$*, we have* $R((l, v), a) = \begin{cases} r_L(l){\cdot}a & \text{if } a \in \mathbb{T} \\ r_A(l, a) & \text{if } a \in A \end{cases}$
- *for any* $(l, v) \in S$*, we have* $obs(l, v) = (obs_L(l), v)$*.*

Example 1. Consider the POPTA in Fig. 1(a) with clocks x, y. Locations are grouped according to their observations, and we omit enabling conditions equal to `true`. We aim to maximise the probability of observing o_5. If locations were fully observable, we would leave $\bar{l}$ when $x{=}y{=}1$ and then, depending on whether the random choice resulted in a transition to l_1 or l_2, wait 0 or 1 time units, respectively, before leaving the location. This would allow us to move immediately from l_3 or l_4 to l_5, meaning observation o_5 is seen with probability 1. However, in the POPTA, we need to make the same choice in l_1 and l_2 since they yield the same observation. As a result, at most one of the transitions leaving locations l_3 and l_4 is enabled, and the probability of observing o_5 is thus at most 0.5.

Digital clocks. Since the semantics of a POPTA (like for a PTA) is an infinite-state model, for algorithmic analysis, we first need to construct a *finite* representation. In this paper, we propose to use the *digital clocks* approach, generalising a technique already used for PTAs [20], which in turn adapts one for timed automata [16]. In short, this approach discretises a POPTA model by transforming its real-valued clocks to clocks taking values from a bounded set of integers.

For clock $x \in \mathcal{X}$, let $\mathbf{k}_x$ denote the greatest constant to which x is compared in the clock constraints of POPTA P. If the value of x exceeds $\mathbf{k}_x$, its exact value will not affect the satisfaction of any invariants or enabling conditions, and thus not affect the behaviour of P. The digital clocks semantics, written $[\![\mathsf{P}]\!]_{\mathbb{N}}$, can be obtained from Defn. 7, taking $\mathbb{T}$ to be $\mathbb{N}$ instead of $\mathbb{R}$. We also need to redefine the operation $v{+}t$, which now adds a delay $t \in \mathbb{N}$ to a clock valuation $v \in \mathbb{N}^{\mathcal{X}}$: we say that $v{+}t$ assigns the value $\min\{v(x){+}t, \mathbf{k}_x{+}1\}$ to each clock $x \in \mathcal{X}$.

Under the restrictions on POPTAs described above, the digital semantics of a POPTA preserves the key properties required in this paper, namely optimal probabilities and expected cumulative rewards for reaching a specified observation. This is captured by the following theorem (the proof is available in [26]).

Theorem 1. *If* P *is a closed, diagonal-free POPTA which resets only non-zero clocks, then, for any set of observations* O *of* P *and* $opt \in \{\min, \max\}$*, we have:*

$$Pr^{opt}_{[\![\mathsf{P}]\!]_{\mathbb{R}}}(\mathtt{F}\,O) = Pr^{opt}_{[\![\mathsf{P}]\!]_{\mathbb{N}}}(\mathtt{F}\,O) \quad and \quad \mathbb{E}^{opt}_{[\![\mathsf{P}]\!]_{\mathbb{R}}}(\mathtt{F}\,O) = \mathbb{E}^{opt}_{[\![\mathsf{P}]\!]_{\mathbb{N}}}(\mathtt{F}\,O).$$

The proof relies on showing probabilistic and expected reward values agree on the belief MDPs underlying the POMDPs representing the dense time and digital clocks semantics. This requires introducing the concept of a belief PTA for a POPTA (analogous to a belief MDP for a POMDP) and results for PTAs [20].

Example 2. The POPTA P in Fig. 1(b) demonstrates why our digital clocks approach (Thm. 1) is restricted to POPTAs which reset only non-zero clocks. We aim to minimise the expected reward accumulated before observing o_3 (rewards are shown in Fig. 1(b) and are zero if omitted). If locations were fully observable, the minimum reward would be 0, achieved by leaving $\bar{l}$ immediately and then choosing a_1 in l_1 and a_2 in l_2. However, if we leave $\bar{l}$ immediately, l_1 and l_2 are indistinguishable (we observe $(o_{1,2},(0))$ when arriving in both), so we must choose the same action in these locations, and hence the expected reward is 0.5.

Consider the strategy that waits $\varepsilon \in (0,1)$ before leaving $\bar{l}$, accumulating a reward of ε. This is possible only in the dense-time semantics. We then observe either $(o_{1,2},(\varepsilon))$ in l_1, or $(o_{1,2},(0))$ in l_2. Thus, we see if x was reset, determine if we are in l_1 or l_2, and take action a_1 or a_2 accordingly such that no further reward is accumulated before seeing o_3, for a total reward of ε. Since ε can be arbitrarily small, the minimum (infimum) expected reward for $[\![\mathsf{P}]\!]_{\mathbb{R}}$ is 0. However, for the digital clocks semantics, we can only choose a delay of 0 or 1 in $\bar{l}$. For the former, the expected reward is 0.5, as described above; for the latter, we can again pick a_1 or a_2 based on whether x was reset, for a total expected reward of 1. Hence the minimum expected reward for $[\![\mathsf{P}]\!]_{\mathbb{N}}$ is 0.5, as opposed to 0 for $[\![\mathsf{P}]\!]_{\mathbb{R}}$.

4 Verification and Strategy Synthesis for POPTAs

We now present our approach for verification and strategy synthesis for POPTAs using the digital clock semantics given in the previous section.

Property specification. First, we define a temporal logic for the formal specification of quantitative properties of POPTAs. This is based on a subset (we omit temporal operator nesting) of the logic presented in [27] for PTAs.

Definition 8 (Properties). *The syntax of our logic is given by the grammar:*

$$\begin{array}{ll} \phi ::= \mathtt{P}_{\bowtie p}[\psi] \mid \mathtt{R}_{\bowtie q}[\rho] & \psi ::= \alpha\, \mathtt{U}^{\leqslant t}\, \alpha \mid \alpha\, \mathtt{U}\, \alpha \\ \alpha ::= \mathtt{true} \mid o \mid \neg o \mid \zeta \mid \alpha \wedge \alpha \mid \alpha \vee \alpha & \rho ::= \mathtt{I}^{=t} \mid \mathtt{C}^{\leqslant t} \mid \mathtt{F}\, \alpha \end{array}$$

where o is an observation, ζ is a clock constraint, $\bowtie \in \{\leqslant, <, \geqslant, >\}$, $p \in \mathbb{Q} \cap [0,1]$, $q \in \mathbb{Q}_{\geqslant 0}$ and $t \in \mathbb{N}$.

A property ϕ is an instance of either the probabilistic operator $\mathtt{P}$ or the expected reward operator $\mathtt{R}$. As for similar logics, $\mathtt{P}_{\bowtie p}[\psi]$ means the probability of path formula ψ being satisfied is $\bowtie p$, and $\mathtt{R}_{\bowtie q}[\rho]$ the expected value of reward operator ρ is $\bowtie q$. For the probabilistic operator, we allow time-bounded ($\alpha\, \mathtt{U}^{\leqslant t}\, \alpha$) and unbounded ($\alpha\, \mathtt{U}\, \alpha$) until formulas, and adopt the usual equivalences such as $\mathtt{F}\, \alpha \equiv \mathtt{true}\, \mathtt{U}\, \alpha$ ("eventually α"). For the reward operator, we allow $\mathtt{I}^{=t}$ (location reward at time instant t), $\mathtt{C}^{\leqslant t}$ (reward accumulated until time t) and $\mathtt{F}\, \alpha$ (the reward accumulated until α becomes true). Our propositional formulas (α) are Boolean combinations of observations and clock constraints.

We omit nesting of $\mathtt{P}$ and $\mathtt{R}$ operators for two reasons: firstly, the digital clocks approach that we used to discretise time is not applicable to nested properties (see [20] for details); and secondly, it allows us to use a consistent property specification for either verification or strategy synthesis problems (the latter is considerably more difficult in the context of nested formulas [4]).

Definition 9 (Property semantics). *Let P be a POPTA with location observation function obs_L and semantics $[\![\mathsf{P}]\!]$. We define satisfaction of a property ϕ from Defn. 8 with respect to a strategy $\sigma \in \Sigma_{[\![\mathsf{P}]\!]}$ as follows:*

$$\begin{array}{lcl} [\![\mathsf{P}]\!], \sigma \models \mathtt{P}_{\bowtie p}[\,\psi\,] & \iff & Pr^{\sigma}_{[\![\mathsf{P}]\!]}(\{\omega \in IPaths_{[\![\mathsf{P}]\!]} \mid \omega \models \psi\}) \bowtie p \\ [\![\mathsf{P}]\!], \sigma \models \mathtt{R}_{\bowtie q}[\,\rho\,] & \iff & \mathbb{E}^{\sigma}_{[\![\mathsf{P}]\!]}(rew(\rho)) \bowtie q \end{array}$$

Satisfaction of a path formula ψ by path ω, denoted $\omega \models \psi$ and the random variable $rew(\rho)$ for a reward operator ρ are defined identically as for PTAs. Due to lack of space, we omit their formal definition here and refer the reader to [27]. For a propositional formula α and state $s = (l, v)$ of $[\![\mathsf{P}]\!]$, we have $s \models o$ if and only if $obs_L(l)=o$ and $s \models \zeta$ if and only if $v \models \zeta$. Boolean operators are standard.

Verification and strategy synthesis. Given a POPTA P and property ϕ, we are interested in solving the dual problems of *verification* and *strategy synthesis.*

Definition 10 (Verification). *The* verification *problem is: given a POPTA* P *and property* ϕ*, decide if* $[\![\mathsf{P}]\!],\sigma \models \phi$ *holds for all strategies* $\sigma \in \Sigma_{[\![\mathsf{P}]\!]}$.

Definition 11 (Strategy synthesis). *The* strategy synthesis *problem is: given POPTA* P *and property* ϕ*, find, if it exists, a strategy* $\sigma \in \Sigma_{[\![\mathsf{P}]\!]}$ *such that* $[\![\mathsf{P}]\!],\sigma \models \phi$.

The verification and strategy synthesis problems for ϕ can be solved similarly, by computing *optimal values* for either probability or expected reward objectives:

$$\begin{array}{ll} Pr^{\min}_{[\![\mathsf{P}]\!]}(\psi) = \inf_{\sigma \in \Sigma_{[\![\mathsf{P}]\!]}} Pr^{\sigma}_{[\![\mathsf{P}]\!]}(\psi) & \mathbb{E}^{\min}_{[\![\mathsf{P}]\!]}(\rho) = \inf_{\sigma \in \Sigma_{[\![\mathsf{P}]\!]}} \mathbb{E}^{\sigma}_{[\![\mathsf{P}]\!]}(\rho) \\ Pr^{\max}_{[\![\mathsf{P}]\!]}(\psi) = \sup_{\sigma \in \Sigma_{[\![\mathsf{P}]\!]}} Pr^{\sigma}_{[\![\mathsf{P}]\!]}(\psi) & \mathbb{E}^{\max}_{[\![\mathsf{P}]\!]}(\rho) = \sup_{\sigma \in \Sigma_{[\![\mathsf{P}]\!]}} \mathbb{E}^{\sigma}_{[\![\mathsf{P}]\!]}(\rho) \end{array}$$

and, where required, also synthesising an *optimal strategy*. For example, verifying $\phi{=}\mathtt{P}_{\geqslant p}[\,\psi\,]$ requires computation of $Pr^{\min}_{[\![\mathsf{P}]\!]}(\psi)$ since ϕ is satisfied by all strategies if and only if $Pr^{\min}_{[\![\mathsf{P}]\!]}(\psi){\geqslant}p$. Dually, consider synthesising a strategy for which $\phi'{=}\mathtt{P}_{\leqslant p}[\,\psi\,]$ holds. Such a strategy exists if and only if $Pr^{\min}_{[\![\mathsf{P}]\!]}(\psi){\leqslant}p$ and, if it does, we can use the optimal strategy that achieves the minimum value. A common practice in probabilistic verification to simply query the optimal values directly, using *numerical* properties such as $\mathtt{P}_{\mathtt{min}=?}[\,\psi\,]$ and $\mathtt{R}_{\mathtt{max}=?}[\,\rho\,]$.

As mentioned earlier, when solving POPTAs (or POMDPs), we may only be able to under- and over-approximate optimal values, which requires adapting the processes sketched above. For example, if we have determined lower and upper bounds $p^\flat \leqslant Pr^{\min}_{[\![\mathsf{P}]\!]}(\psi) \leqslant p^\sharp$. We can verify that $\phi{=}\mathtt{P}_{\geqslant p}[\,\psi\,]$ holds if $p^\flat \geqslant p$ or ascertain that ϕ does not hold if $p \geqslant p^\sharp$. But, if $p^\flat < p < p^\sharp$, we need to refine our approximation to produce tighter bounds. An analogous process can be followed for the case of strategy synthesis. The remainder of this section therefore focuses on how to (approximately) compute optimal values and strategies for POPTAs.

Numerical computation algorithms. Approximate numerical computation of either optimal probabilities or expected reward values on a POPTA P is performed with the sequence of steps given below, each of which is described in more detail subsequently. We compute both an under- and an over-approximation. For the former, we also generate a strategy which achieves this value.

(A) We modify POPTA P, reducing the problem to computing optimal values for a *probabilistic reachability* or *expected cumulative reward* property [27];
(B) We apply the *digital clocks* discretisation of Sec. 3 to reduce the infinite-state semantics $[\![\mathsf{P}]\!]_{\mathbb{R}}$ of P to a *finite-state POMDP* $[\![\mathsf{P}]\!]_{\mathbb{N}}$;
(C) We build and solve a *finite abstraction* of the (infinite-state) belief MDP $\mathcal{B}([\![\mathsf{P}]\!]_{\mathbb{N}})$ of the POMDP from (B), yielding an *over-approximation*;
(D) We synthesise and analyse a strategy for $[\![\mathsf{P}]\!]_{\mathbb{N}}$, giving an *under-approximation*;
(E) If required, we *refine* the abstraction's precision and repeat (C) and (D).

(A) Property reduction. As discussed in [27] (for PTAs), checking $\mathtt{P}$ or $\mathtt{R}$ properties of the logic of Defn. 8 can always be reduced to checking either a probabilistic reachability ($\mathtt{P}_{\bowtie p}[\mathtt{F}\,\alpha]$) or expected cumulative reward ($\mathtt{R}_{\bowtie q}[\mathtt{F}\,\alpha]$) property on a modified model. For example, time-bounded probabilistic reachability

($\mathtt{P}_{\bowtie p}[\mathtt{F}^{\leqslant t}\,\alpha]$) can be transformed into probabilistic reachability ($\mathtt{P}_{\bowtie p}[\mathtt{F}\,(\alpha \wedge y \leqslant t)]$) where y is a new clock added to the model. We refer to [27] for full details.

(B) Digital clocks. We showed in Sec. 3 that, assuming certain simple restrictions on the POPTA P, we can construct a finite POMDP $[\![\mathsf{P}]\!]_{\mathbb{N}}$ representing P by treating clocks as bounded integer variables. The translation itself is relatively straightforward, involving a syntactic translation of the PTA (to convert clocks), followed by a systematic exploration of its finite state space. At this point, we also check satisfaction of the restrictions on POPTAs described in Sec. 3.

(C) Over-approximation. We now solve the finite POMDP $[\![\mathsf{P}]\!]_{\mathbb{N}}$. For simplicity, here and below, we describe the case of maximum reachability probabilities (the other cases are very similar) and thus need to compute $Pr^{\max}_{[\![\mathsf{P}]\!]_{\mathbb{N}}}(\mathtt{F}\,O)$. We first compute an *over-approximation*, i.e. an *upper* bound on the maximum probability. This is computed from an approximate solution to the belief MDP $\mathcal{B}([\![\mathsf{P}]\!]_{\mathbb{N}})$, whose construction we outlined in Sec. 2. This MDP has a continuous state space: the set of beliefs $Dist(S)$, where S is the state space of $[\![\mathsf{P}]\!]_{\mathbb{N}}$.

To approximate its solution, we adopt the approach of [29] which computes values for a finite set of representative beliefs G whose convex hull is $Dist(S)$. Value iteration is applied to the belief MDP, using the computed values for beliefs in G and interpolating to get values for those not in G. The resulting values give the required upper bound. We use [29] as it works with *unbounded* (infinite horizon) and *undiscounted* properties. There are many other similar approaches [28], but these are formulated for discounted or finite-horizon properties.

The representative beliefs can be chosen in a variety of ways. We follow [23], where $G = \{\frac{1}{M}v \mid v \in \mathbb{N}^{|S|} \wedge \sum_{i=1}^{|S|} v(i){=}M\}$, i.e. a uniform *grid* with *resolution* M. A benefit is that interpolation is very efficient, using a process called triangulation [13]. A downside is that the grid size is exponential M.

(D) Under-approximation. Since it is preferable to have two-sided bounds, we also compute an *under-approximation*: here, a lower bound on $Pr^{\max}_{[\![\mathsf{P}]\!]_{\mathbb{N}}}(\mathtt{F}\,O)$. To do so, we first synthesise a finite-memory strategy σ^* for $[\![\mathsf{P}]\!]_{\mathbb{N}}$ (which is often a required output anyway). The choices of this strategy are built by stepping through the belief MDP and, for the current belief, choosing an action that achieves the values returned by value iteration in (C) above – see for example [28]. We then compute, by building and solving the finite Markov chain induced by $[\![\mathsf{P}]\!]_{\mathbb{N}}$ and σ^*, the value $Pr^{\sigma^*}_{[\![\mathsf{P}]\!]_{\mathbb{N}}}(\mathtt{F}\,O)$ which is a lower bound for $Pr^{\max}_{[\![\mathsf{P}]\!]_{\mathbb{N}}}(\mathtt{F}\,O)$.

(E) Refinement. Finally, although no a priori bound can be given on the error between the generated under- and over-approximations (recall that the basic problem is undecidable), asymptotic convergence of the grid based approach *is* guaranteed [29]. In practice, if the computed approximations do not suffice to verify the required property (or, for strategy synthesis, σ^* does not satisfy the property), then we increase the grid resolution M and repeat steps (C) and (D).

5 Implementation and Case Studies

We have built a prototype tool for verification and strategy synthesis of POPTAs and POMDPs as an extension of PRISM [19]. We extended the existing modelling language for PTAs, to allow model variables to be specified as observable or hidden. The tool performs the steps outlined in Sec. 4, computing a pair of bounds for a given property and synthesising a corresponding strategy. We focus on POPTAs, but the tool can also analyse POMDPs directly. The software, details of all case studies, parameters and properties are available online at:

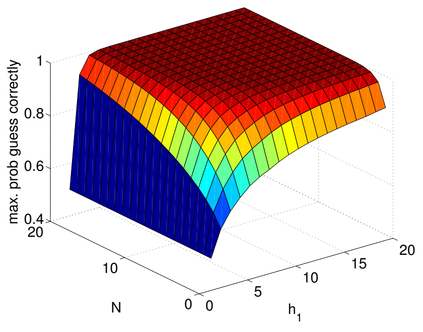
`http://www.prismmodelchecker.org/files/formats15poptas/`

We have developed three case studies to evaluate the tool and techniques, discussed in more detail below. In each case, nondeterminism, probability, real-time behaviour *and* partial observability are all essential aspects required for analysis.

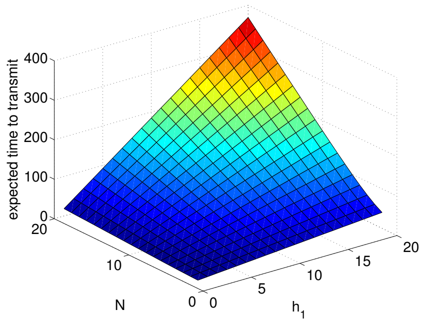
The NRL pump. The NRL (Naval Research Laboratory) pump [17] is designed to provide reliable and secure communication over networks of nodes with 'high' and 'low' security levels. It prevents a covert channel leaking information from 'high' to 'low' through the *timing* of messages and acknowledgements. Communication is buffered and *probabilistic* delays are added to acknowledgements from 'high' in such a way that the potential for information leakage is minimised, while maintaining network performance. A PTA model is considered in [21].

We model the pump as a POPTA using a hidden variable for a secret value $z \in \{0, 1\}$ (initially set uniformly at random) which 'high' tries to covertly communicate to 'low'. This communication is attempted by adding a delay of h_0 or h_1, depending on the value of z, whenever sending an acknowledgement to 'low'. In the model, 'low' sends N messages to 'high' and tries to guess z based on the time taken for its messages to be acknowledged. We consider the maximum probability 'low' can (either eventually or within some time frame) correctly guess z. We also study the expected time to send all messages and acknowledgements. These properties measure the security and performance aspects of the pump. Results are presented in Fig. 2 varying h_1 and N (we fix h_0=2). They show that increasing either the difference between h_0 and h_1 (i.e., increasing h_1) or the number N of messages sent improve the chance of 'low' correctly guessing the secret z, at the cost of a decrease in network performance. On the other hand, when h_0=h_1, however many messages are sent, 'low', as expected, learns nothing of the value being sent and at best can guess correctly with probability 0.5.

Task-graph scheduler. Secondly, we consider a task-graph scheduling problem adapted from [7], where the goal is to minimise the *time* or *energy consumption* required to evaluate an arithmetic expression on multiple processors with different speeds and energy consumption. We extend both the basic model of [7] and the extension from [27] which uses PTAs to model *probabilistic* task execution times. A new 'low power' state to one processor, allowing it to save energy when not in use, but which incurs a delay when waking up to execute a new task. This state is entered with probability *sleep* after each task is completed. We assume that the scheduler cannot observe whether the processor enters this lower



(a)



(b)

Fig. 2. Analysing security/performance of the NRL pump: (a) Maximum probability of covert channel success; (b) Maximum expected transmission time.

power state, and hence the model is a POPTA. We generate optimal schedulers (minimising expected execution time or energy usage) using strategy synthesis.

Non-repudiation protocol. Our third case study is a non-repudiation protocol for information transfer due to Markowitch & Roggeman [25]. It is designed to allow an originator O to send information to a recipient R while guaranteeing *non-repudiation*, that is, neither party can deny having participated in the information transfer. The initialisation step of the protocol requires O to *randomly* select an integer N in the range $1, \ldots, K$ that is never revealed to R during execution. In previous analyses [22,27], modelling this step was not possible since no notion of (non-)observability was used. We resolve this by building a POPTA model of the protocol including this step, thus matching Markowitch & Roggeman's original specification. In particular, we include a hidden variable to store the random value N. We build two models: a basic one, where R's only malicious behaviour corresponds to stopping early; and a second, more complex model, where R has access to a decoder. We compute the maximum probability that R gains an unfair advantage (gains the information from O while being able to deny participating). Our results (see Table 1) show that, for the basic model, this probability equals $1/K$ and R is more powerful in the complex model.

Experimental results. Table 1 summarises a representative set of experimental results from the analysis of our three case studies. All were run on a 2.8 GHz PC with 8GB RAM. The table shows the parameters used for each model (see the web page cited above for details), the property analysed and various statistics from the analysis: the size of the POMDP obtained through the digital clocks semantics; number of observations; number of hidden values (i.e., the maximum number of states with the same observation); the grid size (resolution M and total number of points); the time taken; and the results obtained. For comparison, in the rightmost column, we show what result is obtained if the POPTA is treated as a PTA (by making everything observable).

On the whole, we find that the performance of our prototype is good, especially considering the complexity of the POMDP solution methods and the

Case study (parameters)		Property	Verification/strategy synthesis of POPTA							PTA result
			States ($[\![\mathsf{P}]\!]_{\mathbb{N}}$)	Num. obs.	Num. hidd.	Res. (M)	Grid points	Time (s)	Result (bounds)	
pump (h_1 N)	16 2	$\mathtt{P}_{\mathtt{max}=?}[\mathtt{F}\ \mathit{guess}]$	243	145	3	2	342	0.7	$[0.940, 0.992]$	1.0
	16 2		243	145	3	40	4,845	4.0	$[0.940, 0.941]$	1.0
	16 16		1,559	803	3	2	2,316	16.8	$[0.999, 0.999]$	1.0
pump (h_1 N D)	8 4 50	$\mathtt{P}_{\mathtt{max}=?}[\mathtt{F}^{\leqslant D}\mathit{guess}]$	12,167	7,079	3	2	17,256	11.0	$[0.753, 0.808]$	1.0
	8 4 50		12,167	7,079	3	12	68,201	36.2	$[0.763, 0.764]$	1.0
	16 8 50		26,019	13,909	3	2	38,130	52.8	$[0.501, 0.501]$	1.0
	16 8 100		59,287	31,743	3	2	86,832	284.8	$[0.531, 0.532]$	1.0
scheduler basic (*sleep*)	0.25	$\mathtt{R}_{\mathtt{min}=?}[\mathtt{F}\ \mathit{done}]$ (exec. time)	5,002	3,557	2	2	6,447	3.2	$[14.69, 14.69]$	14.44
	0.5		5,002	3,557	2	2	6,447	3.1	$[17.0, 17.0]$	16.5
	0.75		5,002	3,557	2	4	9,337	3.1	$[19.25, 19.25]$	18.5
scheduler basic (*sleep*)	0.25	$\mathtt{R}_{\mathtt{min}=?}[\mathtt{F}\ \mathit{done}]$ (energy cons.)	5,002	3,557	2	4	9,337	3.1	$[1.335, 1.335]$	1.237
	0.5		5,002	3,557	2	2	6,447	3.1	$[1.270, 1.270]$	1.186
	0.75		5,002	3,557	2	2	6,447	3.2	$[1.204, 1.204]$	1.155
scheduler prob (*sleep*)	0.25	$\mathtt{R}_{\mathtt{min}=?}[\mathtt{F}\ \mathit{done}]$ (exec. time)	6,987	5,381	2	2	8,593	5.8	$[15.00, 15.00]$	14.75
	0.5		6,987	5,381	2	2	8,593	5.8	$[17.27, 17.27]$	16.77
	0.75		6,987	5,381	2	4	11,805	5.0	$[19.52, 19.52]$	18.77
scheduler prob (*sleep*)	0.25	$\mathtt{R}_{\mathtt{min}=?}[\mathtt{F}\ \mathit{done}]$ (energy cons.)	6,987	5,381	2	4	11,805	5.3	$[1.335, 1.335]$	1.3
	0.5		6,987	5,381	2	2	8,593	5.0	$[1.269, 1.269]$	1.185
	0.75		6,987	5,381	2	2	8,593	5.8	$[1.204, 1.204]$	1.155
nrp basic (K)	4	$\mathtt{P}_{\mathtt{max}=?}[\mathtt{F}\ \mathit{unfair}]$	365	194	5	8	5,734	0.8	$[0.25, 0.281]$	1.0
	4		365	194	5	24	79,278	5.9	$[0.25, 0.25]$	1.0
	8		1,273	398	9	4	23,435	4.8	$[0.125, 0.375]$	1.0
	8		1,273	398	9	8	318,312	304.6	$[0.125, 0.237]$	1.0
nrp complex (K)	4	$\mathtt{P}_{\mathtt{max}=?}[\mathtt{F}\ \mathit{unfair}]$	1,501	718	5	4	7,480	2.1	$[0.438, 0.519]$	1.0
	4		1,501	718	5	12	72,748	14.8	$[0.438, 0.438]$	1.0
	8		5,113	1,438	9	2	16,117	6.1	$[0.344, 0.625]$	1.0
	8		5,113	1,438	9	4	103,939	47.1	$[0.344, 0.520]$	1.0

Table 1. Experimental results from verification/strategy synthesis of POPTAs.

fact that we use a relatively simple grid mechanism. We are able to analyse POPTAs whose integer semantics yields POMDPs of up to 60,000 states, with experiments usually taking just a few seconds and, at worst, 5-6 minutes. These are, of course, smaller than the standard PTA (or MDP) models that can be verified, but we were still able to obtain useful results for several case studies.

The values in the rightmost column of Table 1 illustrate that the results obtained with POPTAs would not have been possible using a PTA model, i.e., where all states of the model are observable. For the *pump* example, the PTA gives probability 1 of guessing correctly ('low' can simply read the value of the secret). For the *scheduler* example, the PTA model gives a scheduler with better time/energy consumption but that cannot be implemented in practice since the power state is not visible. For the *nrp* models, the PTA gives probability 1 of unfairness as the recipient can read the random value the originator selects.

Another positive aspect is that, in many cases, the bounds generated are very close (or even equal, in which case the results are exact). For the *pump* and *scheduler* case studies, we included results for the smallest grid resolution M required to ensure the difference between the bounds is at most 0.001. In many cases, this is achieved with relatively small values (for the *scheduler* example, in particular, M is at most 4). For *nrp* models, we were unable to do this when K=8 and instead include the results for the largest grid resolution for which POMDP solution was possible: higher values could not be handled within the memory constraints of our test machine. We anticipate being able to improve this in the future by adapting more advanced approximation methods for POMDPs [28].

6 Conclusions

We have proposed novel methods for verification and control of partially observable probabilistic timed automata, using a temporal logic for probabilistic, real-time properties and reward measures. We developed techniques based on a digital clocks discretisation and a belief space approximation, then implemented them in a tool and demonstrated their effectiveness on several case studies.

Future directions include more efficient approximation schemes, zone-based implementations and development of the theory for unobservable clocks. Allowing unobservable clocks, as mentioned previously, will require moving to partially observable stochastic games and restricting the class of strategies.

Acknowledgments. This work was partly supported by the EPSRC grant "Automated Game-Theoretic Verification of Security Systems" (EP/K038575/1). We also grateful acknowledge support from Google Summer of Code 2014.

References

1. de Alfaro, L.: The verification of probabilistic systems under memoryless partial-information policies is hard. In: Proc. PROBMIV'99. pp. 19–32 (1999)
2. Alur, R., Dill, D.: A theory of timed automata. Theoretical Computer Science 126, 183–235 (1994)
3. Baier, C., Bertrand, N., Größer, M.: On decision problems for probabilistic Büchi automata. In: Proc. FOSSACS'08. LNCS, vol. 4962, pp. 287–301. Springer (2008)
4. Baier, C., Größer, M., Leucker, M., Bollig, B., Ciesinski, F.: Controller synthesis for probabilistic systems. In: Proc. TCS'06. pp. 493–506. Kluwer (2004)
5. Behrmann, G. et al.: Minimum-cost reachability for linearly priced timed automata. In: Proc. HSCC'01. LNCS, vol. 2034, pp. 147–162. Springer (2001)
6. Bouyer, P., D'Souza, D., Madhusudan, P., Petit, A.: Timed control with partial observability. In: Proc. CAV'03. LNCS, vol. 2725, pp. 180–192 (2003)
7. Bouyer, P., Fahrenberg, U., Larsen, K., Markey, N.: Quantitative analysis of real-time systems using priced timed automata. Comm. of the ACM 54(9), 78–87 (2011)
8. Cassandra, A.: A survey of POMDP applications (1998), `http://pomdp.org/pomdp/papers/applications.pdf`, presented at the AAAI Fall Symposium, 1998
9. Cassez, F., David, A., Larsen, K., Lime, D., Raskin, J.F.: Timed control with observation based and stuttering invariant strategies. In: Proc. ATVA'07. LNCS, vol. 4762, pp. 192–206 (2007)
10. Cerný, P., Chatterjee, K., Henzinger, T., Radhakrishna, A., Singh, R.: Quantitative synthesis for concurrent programs. In: Proc. CAV'11. LNCS, vol. 6806, pp. 243–259. Springer (2011)
11. Chatterjee, K., Chmelik, M., Tracol, M.: What is decidable about partially observable Markov decision processes with omega-regular objectives. In: CSL'13. LIPIcs, vol. 23, pp. 165–180. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2013)

12. Chatterjee, K., Doyen, L.: Partial-observation stochastic games: How to win when belief fails. ACM Transactions on Computational Logic 15(2) (2014)
13. Eaves, B.: A course in triangulations for solving equations with deformations. Springer (1984)
14. Finkbeiner, B., Peter, H.: Template-based controller synthesis for timed systems. In: Proc. TACAS'12. LNCS, vol. 7214, pp. 392–406 (2012)
15. Giro, S., Rabe, M.: Verification of partial-information probabilistic systems using counterexample-guided refinements. In: Proc. ATVA'12. LNCS, vol. 7561, pp. 333–348. Springer (2012)
16. Henzinger, T., Manna, Z., Pnueli, A.: What good are digital clocks? In: Proc. ICALP'92. LNCS, vol. 623, pp. 545–558. Springer (1992)
17. Kang, M., Moore, A., Moskowitz, I.: Design and assurance strategy for the NRL pump. Computer 31(4), 56–64 (1998)
18. Kemeny, J., Snell, J., Knapp, A.: Denumerable Markov Chains (1976)
19. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Proc. CAV'11. LNCS, vol. 6806, pp. 585–591. Springer (2011)
20. Kwiatkowska, M., Norman, G., Parker, D., Sproston, J.: Performance analysis of probabilistic timed automata using digital clocks. FMSD 29, 33–78 (2006)
21. Lanotte, R., Maggiolo-Schettini, A., Tini, S., Troina, A., Tronci, E.: Automatic analysis of the NRL pump. ENTCS 99, 245–266 (2004)
22. Lanotte, R., Maggiolo-Schettini, A., Troina, A.: Automatic analysis of a non-repudiation protocol. In: Proc. QAPL'04. ENTCS, vol. 112, pp. 113–129 (2005)
23. Lovejoy, W.: Computationally feasible bounds for partially observed Markov decision processes. Operations Research 39(1), 162–175 (1991)
24. Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. Artif. Intell. 147(1–2), 5–34 (2003)
25. Markowitch, O., Roggeman, Y.: Probabilistic non-repudiation without trusted third party. In: Proc. Workshop on Security in Communication Networks (1999)
26. Norman, G., Parker, D., Zou, X.: Verification and control of partially observable probabilistic real-time systems (2015), `http://arxiv.org/abs/1506.06419`
27. Norman, G., Parker, D., Sproston, J.: Model checking for probabilistic timed automata. FMSD 43(2), 164–190 (2013)
28. Shani, G., Pineau, J., Kaplow, R.: A survey of point-based POMDP solvers. Autonomous Agents and Multi-Agent Systems 27(1), 1–51 (2013)
29. Yu, H.: Approximate Solution Methods for Partially Observable Markov and Semi-Markov Decision Processes. Ph.D. thesis, MIT (2006)