On Nonlinear Prices in Timed Automata

Devendra Bhave
IIT Bombay, India
devendra@cse.iitb.ac.in

Shankara Narayanan Krishna
IIT Bombay, India
krishnas@cse.iitb.ac.in

Ashutosh Trivedi *
CU Boulder, USA
ashutosh.trivedi@colorado.edu

Priced timed automata provide a natural model for quantitative analysis of real-time systems and have been successfully applied in various scheduling and planning problems. The optimal reachability problem for linearly-priced timed automata is known to be PSPACE-complete. In this paper we investigate priced timed automata with more general prices and show that in the most general setting the optimal reachability problem is undecidable. We adapt and implement the construction of Audemard, Cimatti, Kornilowicz, and Sebastiani for non-linear priced timed automata using state-of-the-art theorem prover Z3 and present some preliminary results.

1 Introduction

Timed automata, introduced by Alur and Dill [3], extend finite state automata with continuous variables—referred as clocks—that evolve with uniform rates. Time automata syntax permits comparing clocks with integers as guard on transitions and as well as invariants on locations (states), and also allows clock resets as a way to remember the time a transition was last fired. These features of time automata are general enough to permit modeling rich timing properties of real-time systems while providing a decidable verification framework. Timed automata have been quite successful in practice due to their appealing theoretical properties as well as the presence of mature verification tools such as UPPAAL.

Priced timed automata [9, 8] are extensions of timed automata which permit us to model cost associated with staying at locations as well as taking discrete transitions. Priced timed automata are useful in modeling various decision-theoretic problem in the presence of strict timing constraints. The most natural problem studied on these models is the optimal reachability problem (shortest path problem) where the goal is to find the minimum (or maximum) cost to reach a given set of locations.

Linearly-priced timed automata [6] (LPTA), also known as weighted timed automata, are subclasses of priced timed automata where prices change linearly with respect to delay incurred at particular location. For LPTA the optimal reachability problem is known to be decidable and is shown to be PSPACE-complete exploiting a clever extension of region graphs to so-called corner-point abstraction by Bouyer et al. [8]. Alur et al. [4] earlier gave an EXPTIME algorithm to solve the problem with an arbitrary initial state by giving a non-trivial extension of the region graph. Larsen et al. [6, 9] gave a symbolic algorithm to solve the problem, although with some restrictions on the initial state (a corner state with all clocks set to zero). A recent result by Fearnley and Jurdzinksi [13] showed that the PSPACE-hardness results hold for timed automata with two clocks [13]. On the other hand, for timed automata with one clock, reachability-time and reachability-price problems are known to be NL-complete [19].

In practice, however, the requirement for nonlinear pricing models is quite common. As an example consider the optimal scheduling problem of battery usage in embedded systems studied by Jongerden et

*This material is based upon work supported by DARPA under agreement number FA8750-15-2-0096 and by the US National Science Foundation (NSF) under grant numbers CPS-1446900. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. All opinions expressed are those of the authors, and not necessarily of the DARPA or NSF.

Kargahi and Trivedi: Verification and Validation of
Cyber-Physical Systems 2016 (V2CPS-16)
EPTCS 232, 2016, pp. 65–78, doi:10.4204/EPTCS.232.9

al. [16]. In this work authors modeled batteries using kinetic battery model (KiBaM). KiBaM itself is a nonlinear model, but Jongerden et al. [16] manually discretized it to required approximation to model the whole problem as optimization on LPTA. Similar scenarios can be cited from other application domains of priced timed automata such as scheduling [7], resource modeling and analysis [14], and optimal synthesis [15]. However, we believe that providing non-linear price modeling facilities directly in the language of timed automata will further their applicability in system design. Jurdzinski and Trivedi [18] introduced a non-linear subclass of priced timed automata, so-called concavely-priced timed automata, where prices in each location are certain concave prices of valuation and time delays. Exploiting the concave nature of the prices, they showed that the optimal price reachability problem for this class of automata has the same complexity as that of LPTA. Priced timed automata with exponential price functions were studied in a restricted context by Bouyer et al. [10] and Fahrenberg and Larsen [12].

In this paper we uniformly study various subclasses of (non-linear) priced timed automata, and study the boundary between decidable and undecidable variants of PTA. Towards this goal we first show the undecidability of the optimal reachability problem for unrestricted priced timed automata by showing a reduction from the halting problem for two-counter machines. For reasoning with decidable variants, we first introduce a key notion of price-preserving bisimilarity. We exploit this notion to formalize reduction for the optimal cost reachability problem for piecewise-linear priced timed automata to linearly priced timed automata. We also show the decidability of ε-optimal cost reachability for priced timed automata with Lipschitz-continuous prices. Finally, we adapt the construction of Audemard, Cimatti, Kornilowicz, and Sebastiani [5] for bounded model-checking of timed automata using SAT solvers to work for bounded reachability problem for non-linearly priced timed automata using SMT solver Z3 [11]. In conjunction with a decision procedure for the theory of the class of price functions (for instance polynomial prices [17, 11]), our implementation can be used to compute bounded-step cost-optimal schedules for priced timed automata. We demonstrate the applicability of our approach using airplane landing problem [7].

This paper is organized as follows: we begin by defining syntax and semantics of generalized priced timed automata in the next section. We define various pricing models and their hierarchy. We prove key undecidability result in Section 3 and show decidability results in Section 4. Finally, in section 5 we present the details of our implementation and experimental results.

2 Priced Timed Automata

We denote sets of integers, rational numbers and real numbers as $\mathbb{Z}, \mathbb{Q}$ and $\mathbb{R}$ respectively. Their respective non-negative subsets are denoted as $\mathbb{Z}^+, \mathbb{Q}^+$ and $\mathbb{R}^+$.

Let $X = \{x_1, x_2, \ldots, x_n\}$ be the finite set of clocks. A clock valuation is a map $v : X \mapsto \mathbb{R}^+$. Thus, a given clock valuation v maps clock x_i to a value v_i. This fact is written as $v(x_i) = v_i$. In n-tuple form, a clock valuation v is denoted as $(v_1, v_2, \ldots, v_n)$. Given a clock valuation v and $\tau \in \mathbb{R}^+$, $v + \tau$ is the clock valuation defined by $(v_1 + \tau, v_2 + \tau, \ldots, v_n + \tau)$. A guard is any finite conjunction of clauses of the form $x_i \sim c$, where clock $x_i \in X$, constant $c \in \mathbb{Z}^+$ and $\sim$ is one of the comparison operators in set $\{<, \leq, =, >, \geq\}$. Let G be the set of guards. Given a valuation v and a guard $g = \bigwedge_j (x_i \sim c_j)$, $v \vDash g$ means expression $\bigwedge_j (v(x_i) \sim c_j)$ evaluates to true. For $Y \subseteq X$, $v[Y := 0]$ denotes clock valuation in which clocks in Y are reset to 0 while other clocks remain unchanged.

Timed Automata. A timed transition system $\mathscr{T}$ is a tuple (L, X, E) where (i) L is a finite set of locations, (ii) X is a finite set of clocks variables, and (iii) E is set of transitions. A *configuration* of $\mathscr{T}$

is a pair (ℓ, v), where $\ell \in L$ is a location and v in clock valuation over set X. Let $Q_{\mathscr{T}}$ be the set of configurations for the timed transition system $\mathscr{T}$. There are two types of transitions over $Q_{\mathscr{T}}$:

- Delay, $E^{\tau} \subseteq Q_{\mathscr{T}} \times \mathbb{R}^+ \times Q_{\mathscr{T}}$: $(\ell, v) \xrightarrow{t} (\ell, v+t)$, where $t \in \mathbb{R}^+$
- Switch, $E^{e} \subseteq Q_{\mathscr{T}} \times 2^X \times Q_{\mathscr{T}}$: $(\ell, v) \xrightarrow{Y} (\ell', v[Y := 0])$ where $Y \subseteq X$.

We write $E = E^{\tau} \cup E^{e}$ for the set of transitions of timed transition system $\mathscr{T}$.

Definition A timed automaton $\mathscr{A}$ is a tuple (L, X, E, I) where (i) L is a finite set of locations (ii) X is a finite set of clocks (iii) $E \subseteq L \times G \times 2^X \times L$ is a finite set of edges (iv) $I : L \mapsto G$ assigns an invariant to each location.

The semantics of timed automaton $\mathscr{A}$ is given as a timed transition system $\mathscr{T}_{\mathscr{A}} = (L_{\mathscr{A}}, X_{\mathscr{A}}, E_{\mathscr{A}})$, where (i) $L_{\mathscr{A}} = L$ (ii) $X_{\mathscr{A}} = X$ (iii) $E_{\mathscr{A}} = E^{\tau}_{\mathscr{A}} \cup E^{e}_{\mathscr{A}}$, s.t.

- $E^{\tau}_{\mathscr{A}} = \{(\ell, v) \xrightarrow{t} (\ell, v+t) \mid t \in \mathbb{R}^+ \text{ and } \forall \delta \in \mathbb{R}^+, 0 \leq \delta \leq t \Rightarrow (v+\delta) \vDash I(\ell)\}$
- $E^{e}_{\mathscr{A}} = \{(\ell, v) \xrightarrow{Y} (\ell', v[Y{:=}\,0]) \mid Y \subseteq X,\ (\ell, g, Y, \ell') \subseteq E,\ v \vDash g \text{ and } v \vDash I(\ell)\}$

A *run* $\rho = q_0 \to q_1 \to \cdots \to q_m$ of the timed automaton $\mathscr{A}$ is a finite path in the induced timed transition system $\mathscr{T}_{\mathscr{A}}$ where every q_i is configuration in $\mathscr{T}_{\mathscr{A}}$ and $\to$ is either delay or switch edge in $E_{\mathscr{A}}$. We use notation $\rho = q_0 \rightsquigarrow q_m$ for a run from q_0 to q_m. We write $\mathsf{Runs}(q, q')$ for the set of runs from the location q to q'. A run is said to be *canonical* if delay and switch transitions alternate.

Priced timed automata. A priced timed automaton (PTA) is a timed automaton $\mathscr{A} = (L, X, E, I, \pi, \psi)$ augmented with a price functions $\pi : Q_{\mathscr{A}} \times \mathbb{R}^+ \mapsto \mathbb{R}^+$ and $\psi : E \mapsto \mathbb{Z}^+$ which assign prices (costs) for waiting at locations and taking edges, where $Q_{\mathscr{A}}$ is the set of the configurations of timed automaton $\mathscr{A}$. Let $\mathscr{A}$ be a PTA and $\rho = q'_0 \xrightarrow{\tau_1} q_1 \xrightarrow{e_1} q'_1 \xrightarrow{\tau_2} q_2 \xrightarrow{e_2} q'_2 \cdots \xrightarrow{\tau_m} q_m \xrightarrow{e_m} q'_m$ be a canonical run of $\mathscr{T}_{\mathscr{A}}$. Then the cost $C(\rho)$ of the run ρ is equal to $C_d(\rho) + C_s(\rho)$ where $C_d(\rho) = \sum_{k=1}^{m} \pi(q'_{k-1}, \tau_k)$ and $C_s(\rho) = \sum_{k=1}^{m} \psi(e_k)$ are the *duration* and *switching* costs of ρ respectively.

A priced timed transition system $\mathscr{T}$ is a tuple (L, X, E) where (i) L is a set of locations (ii) X is a finite set of clocks (iii) E is a set of transitions. A *configuration* is a tuple (ℓ, v, u), where $\ell \in L$ is a location, v is a clock valuation over set X and $u \in \mathbb{R}$ is current accumulated price. Let $Q_{\mathscr{T}}$ be the set of configurations for timed transition system $\mathscr{T}$. There are two types of transitions defined over $Q_{\mathscr{T}}$:

- Delay, $E^{\tau} \subseteq Q_{\mathscr{T}} \times \mathbb{R}^+ \times Q_{\mathscr{T}}$: $(\ell, v, u) \xrightarrow{t} (\ell, v+t, u')$, where $t \in \mathbb{R}^+$
- Switch, $E^{e} \subseteq Q_{\mathscr{T}} \times 2^X \times Q_{\mathscr{T}}$: $(\ell, v, u) \xrightarrow{Y} (\ell', v[Y{:=}0], u')$ where $Y \subseteq X$.

We write $E = E^{\tau} \cup E^{e}$ for the set of transitions of timed transition system $\mathscr{T}$. A priced timed transition system is said to be *canonical* if for its every run, delay transitions and switch transitions occur in the strict alternation. Observe that two consecutive delay transitions like $(\ell_1, v_1, u_1) \xrightarrow{t_1} (\ell_2, v_2, u_2) \xrightarrow{t_2} (\ell_3, v_3, u_3)$ cannot be combined together as $(\ell_1, v_1, u_1) \xrightarrow{t_1+t_2} (\ell_3, v_3, u_3)$ because for non-linear price functions such clubbing may not yield valid transitions. Similarly, there cannot be consecutive switch transitions without zero delay transition between them.

Let $\mathscr{A} = (L, X, E, I, \pi, \psi)$ be a priced timed automaton. The semantics of $\mathscr{A}$ are given by a canonical priced timed transition system $\mathscr{T}_{\mathscr{A}} = (L_{\mathscr{A}}, X_{\mathscr{A}}, E_{\mathscr{A}})$ such that, $L_{\mathscr{A}} = L$, $X_{\mathscr{A}} = X$, $E_{\mathscr{A}} = E^{\tau}_{\mathscr{A}} \cup E^{e}_{\mathscr{A}}$, s.t.:

- $E^{\tau}_{\mathscr{A}} = \{(\ell, v, u) \xrightarrow{t} (\ell, v+t, u+\pi(\ell, t)) \mid t \in \mathbb{R}^+ \text{ and } \forall \delta \in \mathbb{R}^+, 0 \leq \delta \leq t \Rightarrow (v+\delta) \vDash I(\ell)\}$
- $E^{e}_{\mathscr{A}} = \{(\ell, v, u) \xrightarrow{Y} (\ell', v[Y{:=}0], u + \psi(\gamma)) \mid Y \subseteq X, \text{ transition } \gamma = (\ell, g, Y, \ell') \in E,\ v \vDash g,\ v \vDash I(\ell) \text{ and } v[Y{:=}0]] \vDash I(\ell')\}$.

A run of the transition system $\mathscr{T}_{\mathscr{A}}$ starts with some configuration (ℓ, v, u_0) where $\ell \in L$, $v \in (\mathbb{R}^+)^X$ and $u_0 \in \mathbb{R}$. We do not explicitly specify initial configuration in our definition of priced timed automaton.

Cost-optimal reachability problem Let $\mathscr{A}$ be a priced timed automaton. Given two locations ℓ, ℓ' of A, the optimal cost $\mathsf{OptCost}(\ell, \ell')$, of reaching ℓ' from ℓ is defined as

$$\mathsf{OptCost}(\ell, \ell') = \inf_{\rho \in \mathsf{Runs}(\ell, \ell')} C(\rho).$$

Given priced timed automaton $\mathscr{A}$, locations ℓ, ℓ', and a budget $B \in \mathbb{R}$ the cost-optimal reachability problem is to decide whether $\mathsf{OptCost}(\ell, \ell') \leq B$.

Summary of Our Results. Our first result (Section 3) is that the optimal cost reachability problem for general priced timed automata is undecidable.

Theorem 2.1 *Cost-optimal reachability problem for nonlinearly priced timed automata is undecidable.*

Given this negative result it is justifiable to look for various restricted subclasses of price functions in order to recover decidable variants. The first subclass that we consider is piece-wise linear price functions. A *piecewise linear price function* $f : \mathbb{R}^+ \to \mathbb{R}$ can be represented as tuple $(P^\ell, Y_P^\ell, Y_I^\ell)$ where

- $P^\ell = \langle p_1 = 0, p_2, \ldots, p_n \rangle$ is an increasing sequence of n-points in time. First point p_1 is always at time value zero. Thus, $p_1 (= 0) < p_2 < \ldots < p_n$ holds.
- $Y_P^\ell = \langle y_{p_1}, y_{p_2}, \ldots, y_{p_n} \rangle$ is a sequence of prices such that $y_{p_i} = f_\ell(p_i)$.
- $Y_I^\ell = \langle (m_1, c_1), (m_2, c_2), \ldots, (m_n, c_n) \rangle$ is a sequence of n tuples. Time intervals formed by points in P^ℓ are $I_1 \stackrel{\text{def}}{=} (p_1, p_2), \cdots, I_n \stackrel{\text{def}}{=} (p_n, +\infty)$. Again let $I = \langle I_1, \ldots, I_n \rangle$ be the sequence of intervals. Value of piecewise linear price function f_ℓ in the interval I_k is given by parameters in tuple (m_k, c_k), such that if $\tau \in I_k$, $f(\tau) = m_k \tau + c_k$.

We call tuple $(P^\ell, Y_P^\ell, Y_I^\ell)$ as *structure* of function f. We call a timed automaton is *piecewise linearly priced* if for all configuration (ℓ, v) we have that $\pi((\ell, v), \tau) = f_\ell(\tau)$, where $f_\ell = (P^\ell, Y_P^\ell, Y_I^\ell)$, is a piecewise linear function defined over interval $[0, +\infty)$ and all the constants appearing in its structure are integers. Observe that the standard definition of linearly priced timed automata can be casted as a special case of piecewise linearly priced timed automata such that $f_\ell = (P^\ell, Y_P^\ell, Y_I^\ell)$, where $P^\ell = \langle 0 \rangle$, $Y_P^\ell = \langle 0 \rangle$, and $Y_I^\ell = \langle (k_\ell, 0) \rangle$ such that k_ℓ is rate of change of price at location ℓ. For LPTA the cost-optimal reachability problem is known to be PSPACE-complete [8]. In Section 4 we show the following key result of piecewise linearly priced timed automata.

Theorem 2.2 *The cost-optimal reachability problem for piecewise linearly priced timed automaton is PSPACE-complete.*

This result can easily be extended to piecewise-concave priced timed automata [18].

We also study more general Lipschitz continuously priced timed automata. We say that a function $f : \mathbb{R} \mapsto \mathbb{R}$ is Lipschitz continuous function, if there exists a constant $K \geq 0$, called Lipschitz constant of f, s.t. $\|f(x) - f(y)\| \leq K\|x - y\|$ for all x, y in the domain of f. A timed automaton is then called *Lipschitz continuous priced* if price functions $\pi((\ell, v), \tau) = f_\ell(\tau)$, are Lipschitz continuous for every location ℓ and there exists a constant T such that all the clock valuations are bounded from above by T. For this class of functions the optimal reachability problem may not be computable due to optimal occurring at non-rational points. For this reason we study the following approximate optimal problem.

ε-Cost-optimal reachability problem Let $\mathscr{A}$ be a priced timed automaton. Given $\varepsilon > 0$ and two locations ℓ, ℓ' of $\mathscr{A}$, a budget $B \in \mathbb{R}^+$, the ε-optimal cost problem is to decide whether $\mathsf{OptCost}(\ell, \ell') \leq B + \varepsilon$.

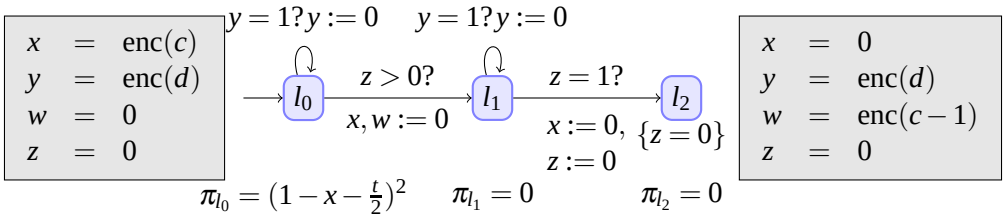

Figure 1: Decrement c module

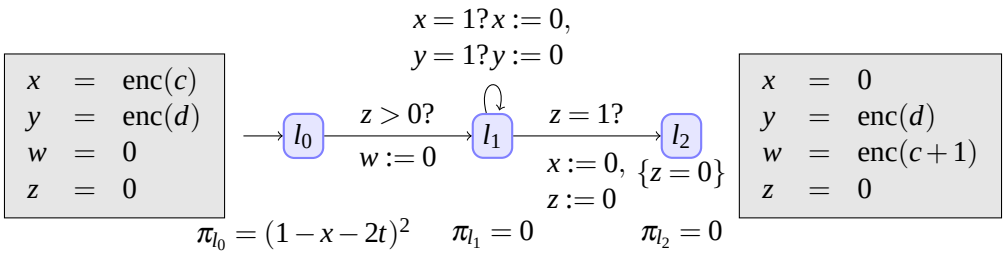

Figure 2: Increment c module

We show in Section 4 the following result for Lipschitz-continuous priced timed automata.

Theorem 2.3 *The ε-Cost-optimal reachability is decidable for Lipschitz-continuous priced timed automata.*

Finally, in Section 5 we give details of our implementation to solve step-bounded cost-optimal reachability problem for general priced timed automata.

3 Undecidability

This section is devoted to the proof of Theorem 2.1. We prove this result by reducing the halting problem for two counter machines to the cost-optimal reachability problem for priced timed automata. A *two-counter machine* M is a tuple (L, C) where $L = \{\ell_0, \ell_1, \ldots, \ell_n\}$ is the set of instructions including a distinguished terminal instruction ℓ_n called HALT, and the set $C = \{c_1, c_2\}$ of two *counters*. The instructions L are of the type: (1) (increment c) $\ell_i : c := c + 1$; goto ℓ_k, (2) (decrement c) $\ell_i : c := c - 1$; goto ℓ_k, (3) (zero-check c) ℓ_i : if $(c > 0)$ then goto ℓ_k else goto ℓ_m, where $c \in C$, $\ell_i, \ell_k, \ell_m \in L$. A configuration of a two-counter machine is a tuple (ℓ, c, d) where $\ell \in L$ is an instruction, and $c, d \in \mathbb{N}$ is the value of counters c_1 and c_2, resp. A run of a two-counter machine is a (finite or infinite) sequence of configurations $\langle k_0, k_1, \ldots \rangle$ where $k_0 = (\ell_0, 0, 0)$ and the relation between subsequent configurations is governed by transitions between respective instructions. The *halting problem* for a two-counter machine asks whether its unique run ends at the terminal instruction ℓ_n. The halting problem for two-counter machines is known [20] to be undecidable.

Proof of Theorem 2.1 We reduce the reachability problem of two counter machines to an instance of the cost-optimal reachability problem $\mathsf{OptCost}(q, q')$ for priced timed automata $\mathscr{A}$ such that desired configuration of two counter machine is reachable from its initial configuration iff there is a run in the automaton $\mathscr{A}$ from q to q' of cost exactly zero.

Let $\mathscr{M}$ be the instance of the two counter machine having counters c and d. We construct a PTA $\mathscr{A}$ from $\mathscr{M}$ using suitable encoding. Valid runs of $\mathscr{M}$ are mapped to valid runs of $\mathscr{A}$ such that their cost is exactly zero. Figure 1 and 2 describes the module simulating counter decrement instruction of $\mathscr{M}$. PTA $\mathscr{A}$ is constructed by composing the various modules. $\mathscr{A}$ uses four clocks – x, y, w and z, out of which x and y encode counters c and d as $x = 1 - \frac{1}{2^c}$ and $y = 1 - \frac{1}{2^d}$. Let enc$(\cdot)$ denote this encoding function. Testing whether c is zero amounts to testing x is zero in the guards of $\mathscr{A}$. Figure 1 describes decrement operation on counter c. It shows clock valuations before entering the module and after exiting the module when simulation is correct. Let t be the amount of time spent in location l_0 during simulation. Let $(x,y,z,w) = (1-\frac{1}{2^c}, 1-\frac{1}{2^d}, 0, 0)$ be the configuration on entering l_0. We want to ensure that the time spent at l_0 is $t = \frac{1}{2^{c-1}}$. The self loop at l_0 ensures that the value of y never crosses 1. If so, the new values of x,y,z,w respectively are $0, 1-(\frac{1}{2^d} - \frac{1}{2^{c-1}})$ or $\frac{1}{2^{c-1}} - \frac{1}{2^d}$, $\frac{1}{2^{c-1}}, 0$. Note that the new value of y after elapse of time t is $1-(\frac{1}{2^d} - \frac{1}{2^{c-1}})$ or $\frac{1}{2^{c-1}} - \frac{1}{2^d}$ depending on whether $d > c$ or not. A time of $1 - \frac{1}{2^{c-1}}$ is spent at location l_1. This gives us the configuration $0, 1-\frac{1}{2^d}, 0, 1-\frac{1}{2^{c-1}}$ on reaching l_2. Note that the self loop on y at location l_1 helps in regaining the value of y to be $1 - \frac{1}{2^d}$ in the case when $d > c$. Note that the cost is 0 iff $t = \frac{1}{2^{c-1}}$. Thus, only correct simulation incurs zero price. Likewise increment module in figure 2 correctly works when $t = \frac{1-x}{2}$.

Observe that after every increment or decrement operation, the value of clock x moves to clock w. Hence, in order to composing $\mathscr{A}$ from individual modules we need to swap the roles of clocks x and w in every alternate modules. Let $\langle c_1, d_1 \rangle$ be initial configuration and $\langle c_2, d_2 \rangle$ be target configuration of $\mathscr{M}$. They map to clock valuation $\nu_1 = (\text{enc}(c_1), \text{enc}(d_1), 0, 0)$ and $\nu_2 = (\text{enc}(c_2), \text{enc}(d_2), 0, 0)$ respectively. To make ν_1 and ν_2 separate locations, we can scale all constants ν_1, ν_2 and $\mathscr{A}$ so as to make clock values in ν_1 and ν_2 integers. The construction is now complete. ∎

4 Decidable Subclasses

Priced Timed Bisimulations. Let $\mathscr{A}$ and $\mathscr{B}$ be timed automata with their timed transition systems $\mathscr{S}_\mathscr{A}$ and $\mathscr{S}_\mathscr{B}$. Let $\mathscr{Q}_\mathscr{A}$ and $\mathscr{Q}_\mathscr{B}$ respective sets of configurations. A binary symmetric relation $\mathscr{R}$ over $\mathscr{Q}_\mathscr{A} \times \mathscr{Q}_\mathscr{B}$ is a *strong timed bisimulation relation* iff for all $a \in (\mathbb{R}^+ \cup 2^X)$

- if $q_1 \xrightarrow{a} q_1'$ and $q_1 \mathscr{R} q_2$ then there exists transition $q_2 \xrightarrow{a} q_2'$ such that $q_1' \mathscr{R} q_2'$
- conversely, if $q_2 \xrightarrow{a} q_2'$ and $q_1 \mathscr{R} q_2$ then there exists transition $q_1 \xrightarrow{a} q_1'$ such that $q_1' \mathscr{R} q_2'$,

where $q_1, q_1' \in \mathscr{Q}_\mathscr{A}$ and $q_2, q_2' \in \mathscr{Q}_\mathscr{B}$. The relation $\mathscr{R}$ is *strong timed bisimilarity* or *strong timed bisimulation equivalence* if it is the largest strong timed bisimulation relation such that $\mathscr{R} \subseteq \mathscr{Q}_\mathscr{A} \times \mathscr{Q}_\mathscr{B}$. Timed automata $\mathscr{A}$ and $\mathscr{B}$ are *strong timed bisimilar* if there exists such $\mathscr{R}$.

Let $\mathscr{A}$ and $\mathscr{B}$ be priced timed automata with their priced timed transition systems $\mathscr{T}_\mathscr{A}$ and $\mathscr{T}_\mathscr{B}$. Let $\mathscr{P}_\mathscr{A}$ and $\mathscr{P}_\mathscr{B}$ be respective sets of priced configurations. A strong timed bisimilarity $\sim$ is said to *price preserving* if for every $a \in (\mathbb{R}^+ \cup 2^X)$

- if $(q_1, u_1) \xrightarrow{a} (q_1', u_1')$ is in $\mathscr{T}_\mathscr{A}$ and $q_1 \sim q_2$ then there exists transition $(q_2, u_2) \xrightarrow{a} (q_2', u_2')$ in $\mathscr{T}_\mathscr{B}$ such that $q_1' \sim q_2'$ and $(u_1' - u_1) = (u_2' - u_2)$
- conversely, if $(q_2, u_2) \xrightarrow{a} (q_2', u_2')$ is in $\mathscr{T}_\mathscr{B}$ and $q_1 \sim q_2$ then there exists transition $(q_1, u_1) \xrightarrow{a} (q_1', u_1')$ in $\mathscr{T}_\mathscr{A}$ such that $q_1' \sim q_2'$ and $(u_1' - u_1) = (u_2' - u_2)$

where $(q_1, u_1), (q_1', u_1') \in \mathscr{P}_\mathscr{A}$ and $(q_2, u_2), (q_2', u_2') \in \mathscr{P}_\mathscr{B}$.

Lemma 4.1 *If $\mathscr{A}$ and $\mathscr{B}$ are two priced timed automata with price preserving timed bisimilarity $\sim$, then for any k length run $\rho^{(k)}_{\mathscr{A}}$ in $\mathscr{A}$, where $\rho^{(k)}_{\mathscr{A}} = (q^0_{\mathscr{A}}, u_0) \xrightarrow{a_1} (q^1_{\mathscr{A}}, u_1) \xrightarrow{a_2} (q^2_{\mathscr{A}}, u_2) \xrightarrow{a_3} \cdots \xrightarrow{a_{k-1}} (q^{k-1}_{\mathscr{A}}, u_{k-1}) \xrightarrow{a_k} (q^k_{\mathscr{A}}, u_k)$, there is a run k length run $\rho^{(k)}_{\mathscr{B}}$ in $\mathscr{B}$, where $\rho^{(k)}_{\mathscr{B}} = (q^0_{\mathscr{B}}, u_0) \xrightarrow{a_1} (q^1_{\mathscr{B}}, u_1) \xrightarrow{a_2} (q^2_{\mathscr{B}}, u_2) \xrightarrow{a_3} \cdots \xrightarrow{a_{k-1}} (q^{k-1}_{\mathscr{B}}, u_{k-1}) \xrightarrow{a_k} (q^k_{\mathscr{B}}, u_k)$, such that, for every $0 \le i \le k$, $q^i_{\mathscr{A}} \sim q^i_{\mathscr{B}}$ holds. u_0 is initial credit.*

As the choice of initial credit is arbitrary and the cost of a run does not depend on the value of initial credit, we claim following lemma.

Lemma 4.2 *Let $\mathscr{A}$ and $\mathscr{B}$ be two priced timed automata with price preserving timed bisimilarity $\sim$. Then following statements are true.*

1. *for every run $\rho_{\mathscr{A}}$ in $\mathscr{A}$, there exists a run $\rho_{\mathscr{B}}$ in $\mathscr{B}$ s. t. cost $C(\rho_{\mathscr{A}}) = C(\rho_{\mathscr{B}})$*
2. *for every run $\rho_{\mathscr{B}}$ in $\mathscr{B}$, there exists a run $\rho_{\mathscr{A}}$ in $\mathscr{A}$ s. t. cost $C(\rho_{\mathscr{A}}) = C(\rho_{\mathscr{B}})$*

4.1 Proof of Theorem 2.2

Lemma 4.3 *For every piecewise linearly priced timed automaton (PwLPTA), there exists linearly priced timed automaton with price preserving strong bisimulation between them.*

Proof We prove this lemma by constructing LPTA explicitly from a given PwLPTA. Rest of this section explains construction and lemma 4.4 proves its correctness.

Construction of LPTA Let $\mathscr{A} = (L_{\mathscr{A}}, X_{\mathscr{A}}, E_{\mathscr{A}}, I_{\mathscr{A}}, \pi_{\mathscr{A}}, \psi_{\mathscr{A}})$ be a PwLPTA. We construct LPTA $\mathscr{B} = (L_{\mathscr{B}}, X_{\mathscr{B}}, E_{\mathscr{B}}, I_{\mathscr{B}}, \pi_{\mathscr{B}}, \psi_{\mathscr{B}})$ from PwLPTA $\mathscr{A}$ as follows:

- Let $\ell \in L_{\mathscr{A}}$ be some location of $\mathscr{A}$. Delay price function for location ℓ, $\pi_{\mathscr{A}}(\ell, \tau) = f_\ell(\tau)$, is piecewise linear with respect to τ. f_ℓ is given by integer restricted structure $(P^\ell, Y^\ell_P, Y^\ell_I)$, where
  - $P^\ell = \langle p_1 = 0, p_2, \ldots, p_n \rangle$
  - $Y^\ell_P = \langle y_{p_1}, y_{p_2}, \ldots, y_{p_n} \rangle$
  - $Y^\ell_I = \langle (m_1, c_1), (m_2, c_2), \ldots, (m_n, c_n) \rangle$ with the following interval sequence

$$I = \langle I_1 \stackrel{\text{def}}{=} (p_1, p_2), I_2 \stackrel{\text{def}}{=} (p_2, p_3), \cdots, I_n \stackrel{\text{def}}{=} (p_n, p_{n+1} = +\infty) \rangle.$$

  We associate each $p_i \in P^\ell$ and each $I_j \in I$ with locations of $L_{\mathscr{B}}$. This association is captured by mapping α^ℓ such that $\alpha^\ell(p_i) = \ell^{p_i}$ and $\alpha^\ell(I_j) = \ell^{(p_j, p_{j+1})}$. Here, ℓ^{p_i} and $\ell^{(p_j, p_{j+1})}$ are the names of locations of $\mathscr{B}$. We define another mapping $\beta^\ell(I_j)$ which returns j^{th} entry in the sequence Y^ℓ_I. This mapping is useful for retrieving parameters of delay cost function in the interval I_j. Let $\theta^\ell = \cup_{i=1}^n \{\ell^{p_i}, \ell^{(p_i, p_{i+1})}\}$. θ^ℓ denotes locations in $L_{\mathscr{B}}$ generated from location $\ell \in L_{\mathscr{A}}$. Then $L_{\mathscr{B}} := \cup_{\ell \in L_{\mathscr{A}}} \theta^\ell$.
- We add one extra clock named x to $\mathscr{B}$. Thus, $X_{\mathscr{B}} := X_{\mathscr{A}} \cup \{x\}$. This clock measures time spent at every location of $\mathscr{A}$. Whenever a run enters any location of $\mathscr{A}$, x is reset to zero.
- An edge $e = (l, \varphi, \lambda, l') \in E_{\mathscr{B}}$ iff there is an edge $e' = (\ell, \chi, \xi, \ell') \in E_{\mathscr{A}}$ such that
  - either $\alpha^\ell(p_i) = l$ or $\alpha^\ell(I_i) = l$
  - either $\alpha^\ell(p_j) = l'$ or $\alpha^\ell(I_j) = l'$
  - $\varphi := \begin{cases} \chi \wedge (x = p_i) & \text{if } \alpha^\ell(p_i) = l \\ \chi \wedge (x \in I_i) & \text{if } \alpha^\ell(I_i) = l \end{cases}$

– $\lambda := \xi \cup \{x\}$

- Location invariant, $I_{\mathscr{B}}(l) = I_{\mathscr{A}}(\ell)$ iff $l \in \theta^{\ell}$
- Location price, $\pi_{\mathscr{B}}(l) := \begin{cases} 0 & \text{if } \alpha^{\ell}(p_i) = l \\ m_i & \text{if } \alpha^{\ell}(I_i) = l \text{ and } \beta^{\ell}(I_i) = (m_i, c_i) \end{cases}$
- Edge price, $\psi_{\mathscr{B}}(e) := \begin{cases} \psi_{\mathscr{A}}(e') + y_{p_i} & \text{if } \alpha^{\ell'}(p_i) = l' \\ \psi_{\mathscr{A}}(e') + c_i & \text{if } \alpha^{\ell'}(I_i) = l' \text{ and } \beta^{\ell'}(I_i) = (m_i, c_i) \end{cases}$

Let ℓ and m be locations of $\mathscr{A}$ and $\mathscr{B}$ respectively. We define following relation between ℓ and m, $\Upsilon = \{(\ell, m) \mid m \in \theta^{\ell}\}$.

Lemma 4.4 *Υ is price preserving timed bisimilarity.*

Proof Let $\mathscr{A}$ = ($L_{\mathscr{A}}$, $X_{\mathscr{A}}$, $E_{\mathscr{A}}$, $I_{\mathscr{A}}$, $\pi_{\mathscr{A}}$, $\psi_{\mathscr{A}}$) be a PwLPTA and $\mathscr{B}$ = ($L_{\mathscr{B}}$, $X_{\mathscr{B}}$, $E_{\mathscr{B}}$, $I_{\mathscr{B}}$, $\pi_{\mathscr{B}}$, $\psi_{\mathscr{B}}$) be LPTA constructed from $\mathscr{A}$ using above construction.

If part: Let t be delay and λ be set of clocks to be reset in $\mathscr{A}$. Now consider following transition in $\mathscr{T}_{\mathscr{A}}$, $(l_1, v_1, u_1) \xrightarrow{(t,\lambda)} (l_1', v_1', u_1')$. Now we try to find simulating transition in $\mathscr{B}$ under relation Υ. We claim its $(l_2, v_1 : 0, u_1) \xrightarrow{(t,\lambda)} (l_2', v_1' : 0, u_1')$. To hold this claim, we choose $l_2 \in L_{\mathscr{B}}$ such that delay t matches with expected interval of l_2. If $t = p_i$ for some i then $l_2 = \alpha^{l_1}(p_i)$. Otherwise t will match with some interval I_j. So $l_2 = \alpha^{l_1}(I_j)$. Thus, $(l_1, l_2) \in \Upsilon$ holds. To place edge in $\mathscr{B}$, construction mandates $(l_1', l_2') \in \Upsilon$. Also the clocks in $X_{\mathscr{A}}$ change identically. Now, let's verify that prices are preserved. For the case where $t = p_i$, $(u_1' - u_1) = y_{p_i} + \psi_{\mathscr{A}}((l_1, l_2))$. Verify that from construction yields same price difference. For the case where $t = I_j$, location price matters. Verify that rates at l_1' and l_2' are the same in the construction. Price change $(u_1' - u_1) = m_j \cdot t + c_j + \psi_{\mathscr{A}}((l_1, l_2))$. Price offset c_j is added to edge cost in the construction. Thus prices are preserved.

Else if part: We consider following transition in $\mathscr{T}_{\mathscr{B}}$, $(l_2, v_2 : 0, u_2) \xrightarrow{(t,\lambda)} (l_2', v_2' : 0, u_2')$. We simulate it on $\mathscr{A}$ to get $(l_1, v_2, u_1) \xrightarrow{(t,\lambda)} (l_1', v_2', u_1')$. If $(l_1, l_2) \in \Upsilon$, then construction offers no choice but to choose l_1' such that $(l_1', l_2') \in \Upsilon$ holds. $v_2' := (v_2 + t)[\lambda := 0]$ follows from construction. Verify that prices are preserved using the same argument as in if part of the proof.

Now we are in position to sketch the proof of Theorem 2.2.

Proof of Theorem 2.2 PSPACE-hardness follows from the fact that LPTA are nothing but PwLPTA with single piece and their cost-optimal reachability is PSPACE-complete. We now explain a PSPACE algorithm for solving cost-optimal reachability for PwLPTA. We construct LPTA $\mathscr{B}$ for given piecewise linearly priced timed automaton $\mathscr{A}$ and solve cost-optimal reachability on $\mathscr{B}$. Construction yields priced timed bisimilarity Υ. Using lemma 4.2, we get $\text{OptCost}(l, l') = \text{opt}\,\{\text{OptCost}(m, m') \mid (l, m) \in \Upsilon \text{ and } (l', m') \in \Upsilon\}$ where l and l' are locations of $\mathscr{A}$, m and m' are locations of $\mathscr{B}$ and opt is either supremum or infimum. ∎

4.2 Proof of Theorem 2.3

Before we sketch a proof of Theorem 2.3, we introduce the concept of iterative approximation for nonlinear price functions.

Let $\mathscr{A}$ = (L, X, E, I, π, ψ) be a priced timed automaton. If for some location ℓ, price function $\pi(\ell, \tau)$ is nonlinear with respect to τ, then $\mathscr{A}$ is nonlinearly priced timed automaton (NLPTA).

Definition We define a PwLPTA $\mathscr{A}_u = (L, X, E, I, \pi_u, \psi)$ be upper bound price approximation of $\mathscr{A}$, if for every location ℓ and time τ, $\pi_u(\ell,\tau) \geq \pi(\ell,\tau)$ and $\pi_u(\ell,\tau)$ is piecewise linear in τ for a fixed ℓ.

Similarly, a PwLPTA $\mathscr{A}_l = (L, X, E, I, \pi_l, \psi)$ is lower bound price approximation of $\mathscr{A}$, if for every location ℓ and time τ, $\pi_l(\ell,\tau) \leq \pi(\ell,\tau)$ and $\pi_l(\ell,\tau)$ is piecewise linear in τ for a fixed ℓ.

Lemma 4.5 *$OptCost_{\mathscr{A}_l}(\ell, \ell') \leq OptCost_{\mathscr{A}}(\ell, \ell') \leq OptCost_{\mathscr{A}_u}(\ell, \ell')$*

Now we are in position to sketch the proof of Theorem 2.3.

Proof of Theorem 2.3 Let $f : \mathbb{R} \mapsto \mathbb{R}$ be Lipschitz continuous function with Lipschitz constant K. Let $x,y \in \mathbb{R}$ be any two arbitrary points in the interval $[x,y]$. The value of f in $[x,y]$ is upper bounded by $\frac{f(x)+f(y)+K(y-x)}{2}$ and lower bounded by $\frac{f(x)+f(y)-K(y-x)}{2}$. Figure 3 shows calculation of these bounds for a Lipschitz continuous function. More precisely, for every $t \in [x,y]$,

$$f(t) \in \Big[\frac{f(x)+f(y)-K(y-x)}{2}, \frac{f(x)+f(y)+K(y-x)}{2}\Big].$$

Assume that f is a rational function. We will first prove decidability of ε-optimal cost reachability problem using this assumption. Later we will drop this assumption.

We now construct two piecewise linear price functions f_l and f_u such that $f_l(t) \leq f(t) \leq f_u(t)$ holds for $0 \leq t \leq T$. Let $T \in \mathbb{R}^+$ is a constant such that all clock valuations are bounded above by T.

Let $\delta \in \mathbb{Q}^+, 0 < \delta \leq T$ be the sampling period. Choice for the value of δ is explained at the end of the proof. We sample f at periodic intervals of δ in the interval $0 \leq t \leq T$. We define a piecewise linear functions

$$\begin{array}{lll} f_l(t) & = f(t) & \text{if } t = N\cdot\delta, \text{where } N \in \mathbb{N} \\ & = \frac{f(N\cdot\delta)+f((N+1)\cdot\delta)-K\delta}{2} & \text{if } t \in \big(N\cdot\delta, (N+1)\cdot\delta\big), \text{where } N \in \mathbb{N} \end{array}$$

$$\begin{array}{lll} f_u(t) & = f(t) & \text{if } t = N\cdot\delta, \text{where } N \in \mathbb{N} \\ & = \frac{f(N\cdot\delta)+f((N+1)\cdot\delta)+K\delta}{2} & \text{if } t \in (N\cdot\delta, (N+1)\cdot\delta), \text{where } N \in \mathbb{N} \end{array}$$

Let $\mathscr{A}$ be priced timed automaton with Lipschitz continuous price functions at all locations. We construct automata $\mathscr{A}_l$ and $\mathscr{A}_u$ by replacing price function at every location while keeping everything else unchanged. Specifically, if price function at location ℓ in $\mathscr{A}$ is $\pi^{(\ell)} = f$, then in $\mathscr{A}_l$, price at location ℓ is $\pi_l^{(\ell)} = f_l$. Likewise we assign price $\pi_u^{(\ell)} = f_u$ to location ℓ of $\mathscr{A}_u$. Observe that $\mathscr{A}_l$ and $\mathscr{A}_u$ are replicas of $\mathscr{A}$ except the difference in location price functions. Since, $\pi_l^{(\ell)}(t) \leq \pi^{(\ell)}(t) \leq \pi_u^{(\ell)}(t)$ holds for all locations ℓ, $\text{OptCost}_{\mathscr{A}_l}(\ell, \ell') \leq \text{OptCost}_{\mathscr{A}}(\ell, \ell') \leq \text{OptCost}_{\mathscr{A}_u}(\ell, \ell')$ follows. Now, for any single delay transition, $\sup\{\|\pi_u^{(\ell)}(t) - \pi_l^{(\ell)}(t)\|\} \leq \|K\delta\|$ over all $0 \leq t \leq T$. Let D be the diameter of region

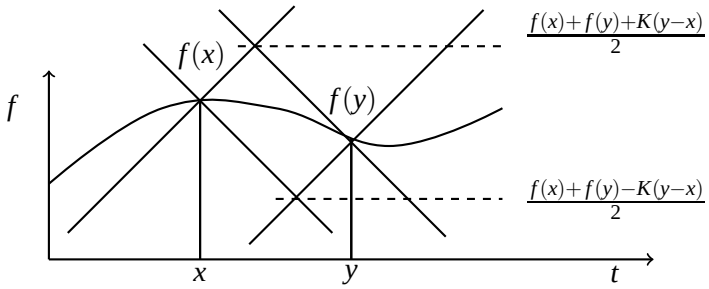

Figure 3: Upper and lower bounds for Lipschitz continuous function in the range $[x,y]$

graph, then $\sup\{\|\mathsf{OptCost}_{\mathscr{A}_l}(\ell,\ell') - \mathsf{OptCost}_{\mathscr{A}_u}(\ell,\ell')\|\} = \varepsilon \leq \|DK\delta\|$. This gives us the bound on ε. We choose $\delta = \frac{\|DK\|}{\varepsilon}$.

In the above construction f is evaluated only at sampling points. We can safely drop the rationality restriction of f by approximating it by rational function f' such that $\|f - f'\| \leq \frac{\|DK\delta\|}{2}$. ∎

5 Step-Bounded Cost-Optimal Reachability Problem

In this section we look into the following step-bounded cost-optimal reachability problem for priced timed automata.

Step-Bounded Cost-optimal reachability problem Let $\mathscr{A}$ be a priced timed automaton. Given two locations ℓ, ℓ' of A, step bound $N \in \mathbb{N}$, the step-bounded optimal cost $\mathsf{OptCost}_N(\ell,\ell')$, is defined as

$$\mathsf{OptCost}_N(\ell,\ell') = \inf_{\rho \in \mathsf{Runs}_N(\ell,\ell')} C(\rho),$$

where $\mathsf{Runs}_N(\ell,\ell')$ are the set of canonical runs between ℓ and ℓ' of length less than or equal to N. Given priced timed automaton $\mathscr{A}$, locations ℓ, ℓ', and a budget $B \in \mathbb{R}^+$ the step-bounded cost-optimal reachability problem is to decide whether $\mathsf{OptCost}_N(\ell,\ell') \leq B$.

In this section we extend the encoding of Audemard, Cimatti, Kornilowicz, and Sebastiani [5] to solve step-bounded optimal-cost reachability problem for priced timed automata. After generating the encoding, we can feed it to SMT solver that support the theory corresponding to the price functions to solve the step-bounded cost-optimal reachability problem.

5.1 Audemard-Cimatti-Kornilowicz-Sebastiani Encoding for PTA

Let $\mathscr{A}_1, \mathscr{A}_2, \ldots \mathscr{A}_n$ be the priced timed automata which are composed into network of automata $\mathscr{A}$. These automata communicate using channels. Let η be the set of channels used in $\mathscr{A}$. If c is a channel, then $c!$ is send operation on the channel c and $c?$ is the blocking receive operation on the channel c.

Original Encoding for Timed Automata. We generate SMT formula for each automata using encoding from Audemard et. al. [5]. As per their scheme, we create one real variable for every clock and create separate one for an extra variable named z, which keeps the track of global time. We add a variable named s of type bitvector at every step which denotes current location. Notation s_ℓ denotes assertion that current location is ℓ. We also create two binary variables for each channel per automaton – one for send and one for receive. For example, if automaton $\mathscr{A}_2$ sends over channel c in current step, we set variable named $\mathscr{A}_2.c!$. This notation helps us to identify automaton which uses that channel in the current step and the type (send or receive) of an operation performed on that channel. We permit to use global clocks. While generating formula for A, it may happen that some of the automata share clock names or location names. For example, automata $\mathscr{A}_1$ and $\mathscr{A}_2$ may both have local clocks named y. But we must distinguish between variables that were created to hold value of y in $\mathscr{A}_1$ and value of y in $\mathscr{A}_2$. We qualify all variables with name of automaton they are the part of. Here, we create real variables named $\mathscr{A}_1.y$ and $\mathscr{A}_2.y$. All of these variables are created for every step of a run in a standard bounded-model-checking fashion. Assertions in Fig. 5.1 describe encoding at current and next step in the formula. We represent next step variables in primed version. For further details refer to [5].

Extension for Priced Timed Automata. Let $\mathscr{A} = (L, X, E, I, \pi, \psi)$ be priced timed automaton. To keep our encoding as general as possible, we describe our SMT formula generation for general priced timed automata. Observe that this class of automata subsume LPTA, concavely-priced PTA, piecewise-linear

$$\bigwedge_{T=(\ell,\varphi,\lambda,\ell')} T \rightarrow \big(s_\ell \wedge \varphi \wedge s'_{\ell'} \wedge \bigwedge_{x\in\lambda}(x'=z') \wedge \bigwedge_{x\notin\lambda}(x'=x) \wedge (z'=z)\big) \tag{1}$$

$$T_\delta \rightarrow \big((s_\ell = s'_\ell) \wedge (z'-z<0) \wedge \bigwedge_{x\in X}(x'=x) \wedge \bigwedge_{a\in\eta}(\neg a)\big) \tag{2}$$

$$T_{\text{null}} \rightarrow \big((s_\ell = s'_\ell) \wedge (z'=z) \wedge \bigwedge_{x\in X}(x'=x) \wedge \bigwedge_{a\in\eta}(\neg a)\big) \tag{3}$$

$$T_{\text{null}} \vee T_\delta \vee \bigvee_{T\in E} T \tag{4}$$

$$\text{price}_0 = 0 \tag{5}$$

$$\bigwedge_{T\in E} T \rightarrow \big(\text{price}' = \text{price} + \psi(T)\big) \tag{6}$$

$$\bigwedge_{\ell\in L} T_\delta \wedge s_\ell \rightarrow \big(\text{price}' = \text{price} + \pi(\ell,\mathbf{x},z-z')\big) \tag{7}$$

$$T_{\text{null}} \rightarrow (\text{price'} = \text{price}) \tag{8}$$

$$\text{price}^{(n)} \bowtie k \tag{9}$$

Figure 4: SMT assertions for priced timed automata

Table 1: Comparison of the performance of our tool with UPPAAL-Cora is shown for ALP problem with 8, 9, and 10 runways, with varying number of airplanes. We report running time (in seconds) for our algorithm (Z3) and DFS and random options for UPPAAL-Cora. TO stands for timeout (>30 mins).

Airplanes	8 runways			9 runways			10 runways		
	Z3	CORA DFS	CORA Random	Z3	CORA DFS	CORA Random	Z3	CORA DFS	CORA Random
1	0.12	< 0.1	< 0.1	0.4	< 0.1	< 0.1	0.30	< 0.1	< 0.1
2	0.09	< 0.1	< 0.1	0.57	< 0.1	< 0.1	0.76	< 0.1	< 0.1
3	0.44	< 0.1	< 0.1	2.52	< 0.1	< 0.1	2.31	< 0.1	< 0.1
4	4.28	2.4	0.04	6.73	4.18	0.08	5.86	7.81	0.06
5	2.73	278.21	0.7	9.61	679.27	0.1	5.09	TO	0.05
6	22.28	TO	0.16	21.34	TO	0.45	20.68	TO	0.32
7	29.23	TO	0.23	201.15	TO	1.15	152.03	TO	1.36
8	89.27	TO	0.79	86.1	TO	1.85	94.88	TO	5.12
9	331	TO	35.09	103.62	TO	151.84	1650.05	TO	277.38
10	889	TO	36.42	667.33	TO	49.04	1309.67	TO	230.69

PTA, and Lipschitz-continuous priced PTA. For each automaton, we represent current accumulated price using real variable named price. We introduce variables price_k at each step. Initially price_0 is set to zero as in 5. When switch transition occurs, we update the price using equation 6. The function $\psi(T)$ denotes edge price for the transition T. Equation 7 is used to specify prices for each delay transition. Quantity $(z-z')$ is the delay incurred at current step and $\mathbf{x}$ is vector of current clock valuations. As price functions are location dependent, we add clause s_ℓ to check whether current location is ℓ and then update price accordingly. For null transitions, prices at current and previous step are identical. To decide whether accumulated price at step n satisfies the condition $\text{price}^{(n)} \bowtie k$, where $\bowtie \in \{<, \leq, >, \geq, =, \neq\}$, we add an assertion as per Eq. 9.

5.2 Experimental Results

We implemented the encoding discussed in the previous subsection as a vtool [1] for analyzing step-bounded optimal-cost for PTA. Out tool invokes state-of-the-art theorem prover Z3 [11] from Microsoft Research. It supports linear and non-linear arithmetic, bit-vectors, arrays, data-types, and quantifiers. For our purpose, Z3 can be used to solve price functions that are given as a polynomial of time-delay and the current valuation. Other non-linear price functions such as log, sin, cos, and exp can be accommodated in this framework using corresponding Taylor series approximations.

In order to show experimental results, we concentrate the standard Airport Landing Scheduling Problem (ALP) from [7]. In order to give comparison with an existing tool we keep the price function linear and compare our tool with state-of-the-art optimal-cost reachability tool Uppaal-Cora [2].

Airport Landing Scheduling Problem. Given number of airplanes each with attributes like type of airplane, landing time window and number of runways, assign a landing time and runways to each airplane such that all airplanes land within their specific landing time window and also comply with safety regulations like mandatory wake turbulence separation delay. There are two possible sources of costs. If airplane travels faster than its designated speed, it lands earlier but consumes more fuel. If airplane landing is delayed, it suffers fuel costs for circling over the airport.

ALP is known to suffer exponential blowup with increasing runways [7]. We used the instances of ALP problem which are distributed with Uppaal CORA demo version. We asked whether there is a schedule such that all airplanes land and total cost is bounded from above by a fixed budget (800). Table 1 shows the results of our experiments. We ran all our experiments on 64-bit Intel® Xeon® CPU E5-2660 v2 running at 2.20GHz with 64 GB RAM. We fixed time limit to 30 minutes for each problem and used single threaded Z3 SMT solver (v 4.3.2).

6 Conclusion and Future Work

We studied priced timed automata with non-linear prices and showed the undecidability of a general class of polynomially-priced timed automata. We then introduced piecewise-linear and Lipschitz-continuous price functions, and recovered decidability in this restricted setting. We also studied step-bounded cost-optimal reachability problem for price timed automata, and implemented an SMT based tool to solve this problem. This problem is of interest since the optimal-cost reachability problem in some cases (under structurally non-Zeno restriction on timed automata along with non-negativity restriction on prices) reduces to step-bounded reachability problem.

Observe that, although our tool does not perform as well as `random-optimal` option of UPPAAL-Cora, it outperforms both `dfs` and `bfs` (not reported here). As a future work, we plan to exploit randomization to scale the performance of our implementation. We believe that these experiments presented here demonstrate the applicability of SMT-based step-bounded verification methodology for medium-sized examples of priced timed automata.

References

[1] Bhave et al. (2015): *PTA-BMC*. `http://www.cse.iitb.ac.in/~devendra/pta-bmc.html`.

[2] Larsen et al. (2006): *UPPAAL-CORA*. `http://people.cs.aau.dk/~adavid/cora/index.html`.

[3] Rajeev Alur & David L. Dill (1994): *A Theory of Timed Automata*. *Theor. Comput. Sci* 126(2), pp. 183–235. DOI: `10.1016/0304-3975(94)90010-8`.

[4] Rajeev Alur, Salvatore La Torre & George J. Pappas (2004): *Optimal Paths in Weighted Timed Automata*. *Theor. Comput. Sci.* 318, pp. 297–322, DOI: `10.1016/j.tcs.2003.10.038`.

[5] G. Audemard, A. Cimatti, A. Kornilowicz & R. Sebastiani (2002): *Bounded Model Checking for Timed Systems*. In: *Formal Techniques for Networked and Distributed Sytems FORTE 2002, LNCS* 2529, Springer, pp. 243–259, DOI: `10.1007/3-540-36135-9_16`.

[6] Gerd Behrmann, Ansgar Fehnker, Thomas Hune, Kim Larsen, Paul Pettersson, Judi Romijn & Frits Vaandrager (2001): *Minimum-Cost Reachability for Priced Time Automata*. In: *HSCC, LNCS* 2034, Springer, pp. 147–161, DOI: `10.1007/3-540-45351-2_15`.

[7] Gerd Behrmann, Kim G Larsen & Jacob I Rasmussen (2005): *Optimal scheduling using priced timed automata*. *Performance Evaluation Review* 32(4), pp. 34–40, DOI: `10.1145/1059816.1059823`.

[8] Patricia Bouyer, Thomas Brihaye, Véronique Bruyère & Jean-François Raskin (2007): *On the Optimal Reachability Problem of Weighted Timed Automata*. *FMSD* 31(2), pp. 135–175, DOI: `10.1007/s10703-007-0035-4`.

[9] Patricia Bouyer, Ed Brinksma & Kim Guldstrand Larsen (2004): *Staying Alive as Cheaply as Possible*. In: *HSCC*, pp. 203–218, DOI: `10.1007/978-3-540-24743-2_14`.

[10] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen & Nicolas Markey (2010): *Timed Automata with Observers Under Energy Constraints*. In: *HSCC*, ACM, pp. 61–70, DOI: `10.1145/1755952.1755963`.

[11] Leonardo De Moura & Nikolaj Bjørner (2008): *Z3: An efficient SMT solver*. In: *Tools and Algorithms for the Construction and Analysis of Systems*, Springer, pp. 337–340, DOI: 10.1007/978-3-540-78800-3_24.

[12] Uli Fahrenberg & Kim G. Larsen (2009): *Discount-Optimal Infinite Runs in Priced Timed Automata*. *ENTCS (INFINITY)* 239, pp. 179 – 191, DOI: 10.1016/j.entcs.2009.05.039.

[13] John Fearnley & Marcin Jurdzinski (2013): *Reachability in Two-Clock Timed Automata Is PSPACE-Complete*. In: *ICALP*, pp. 212–223, DOI: 10.1007/978-3-642-39212-2_21.

[14] Dinko Ivanov, Marin Orlić, Cristina Seceleanu & Aneta Vulgarakis (2010): *REMES tool-chain: a set of integrated tools for behavioral modeling and analysis of embedded systems*. In: *Automated software engineering*, ACM, pp. 361–362, DOI: 10.1145/1858996.1859076.

[15] Zhihao Jiang, Miroslav Pajic, Salar Moarref, Rajeev Alur & Rahul Mangharam (2012): *Modeling and verification of a dual chamber implantable pacemaker*. In: *Tools and Algorithms for the Construction and Analysis of Systems*, Springer, pp. 188–203, DOI: 10.1007/978-3-642-28756-5_14.

[16] M. Jongerden, A. Mereacre, H. Bohnenkamp, B. Haverkort & J. Katoen (2010): *Computing Optimal Schedules of Battery Usage in Embedded Systems*. *Industrial Informatics, IEEE Transactions on* 6(3), pp. 276–286, DOI: 10.1109/TII.2010.2051813.

[17] Dejan Jovanović & Leonardo De Moura (2012): *Solving non-linear arithmetic*. In: *Automated Reasoning*, Springer, pp. 339–354, DOI: 10.1007/978-3-642-31365-3_27.

[18] Marcin Jurdzinski & Ashutosh Trivedi (2008): *Concavely-Priced Timed Automata*. In: *FORMATS*, pp. 48–62, DOI: 10.1007/978-3-540-85778-5_5.

[19] F. Laroussinie, N. Markey & Ph. Schnoebelen (2004): *Model Checking Timed Automata with One or Two Clocks*. In: *CONCUR, LNCS* 3170, Springer, pp. 387–401, DOI: 10.1007/978-3-540-28644-8_25.

[20] Marvin L. Minsky (1967): *Computation: finite and infinite machines*. Prentice-Hall, Inc.