# Revisiting Robustness in Priced Timed Games

**Shibashis Guha[1], Shankara Narayanan Krishna[2], Lakshmi Manasa[2], and Ashutosh Trivedi[2]**

1 **Department of Computer Science & Engineering, IIT Delhi, India**
shibashis@cse.iitd.ernet.in
2 **Department of Computer Science & Engineering, IIT Bombay, India**
{krishnas,manasa,trivedi}@cse.iitb.ac.in

## Abstract

Priced timed games are optimal-cost reachability games played between two players – the controller and the environment – by moving a token along the edges of infinite graphs of configurations of priced timed automata. The goal of the controller is to reach a given set of target locations as cheaply as possible, while the goal of the environment is the opposite. Priced timed games are known to be undecidable for timed automata with 3 or more clocks, while they are known to be decidable for automata with 1 clock. In an attempt to recover decidability for priced timed games Bouyer, Markey, and Sankur studied robust priced timed games where the environment has the power to slightly perturb delays proposed by the controller. Unfortunately, however, they showed that the natural problem of deciding the existence of optimal limit-strategy – optimal strategy of the controller where the perturbations tend to vanish in the limit – is undecidable with 10 or more clocks. In this paper we revisit this problem and improve our understanding of the decidability of these games. We show that the limit-strategy problem is already undecidable for a subclass of robust priced timed games with 5 or more clocks. On a positive side, we show the decidability of the existence of almost optimal strategies for the same subclass of one-clock robust priced timed games by adapting a classical construction by Bouyer at al. for one-clock priced timed games.

**1998 ACM Subject Classification** F.1.1 Models of Computation

**Keywords and phrases** Priced Timed Games, Decidability, Optimal strategies, Robustness

**Digital Object Identifier** 10.4230/LIPIcs.FSTTCS.2015.261

## 1 Introduction

Two-player zero-sum games on priced timed automata provide a mathematically elegant modeling framework for the control-program synthesis problem in real-time systems. In these games, two players – the *controller* and the *environment* – move a token along the edges of the infinite graph of configurations of a timed automaton to construct an infinite execution of the automaton in order to optimize a given performance criterion. The optimal strategy of the controller in such game then corresponds to control-program with the optimal performance. By priced timed games (PTGs) we refer to such games on priced timed automata with optimal reachability-cost objective. The problem of deciding the existence of the optimal controller strategy in PTGs is undecidable [8] with 3 or more clocks, while it is known to be decidable [5] for automata with 1 clock. Also, the $\varepsilon$-optimal strategies can be computed for priced timed games under the non-Zeno assumption [1, 4]. Unfortunately, however, the optimal controller strategies obtained as a result of solving games on timed automata may not be physically realizable due to unrealistic assumptions made in the modeling using timed automata, regarding the capability of the controller in enforcing precise delays. This severely

35th IARCS Annual Conf. Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2015).
Editors: Prahladh Harsha and G. Ramalingam; pp. 261–277

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

limits the application of priced timed games in control-program synthesis for real-time systems.

In order to overcome this limitation, Bouyer, Markey, and Sankur [7] argued the need for considering the existence of robust optimal strategies and introduced two different robustness semantics – *excess* and *conservative* – in priced timed games. The key assumption in their modeling is that the controller may not be able to apply an action at the exact time delays suggested by the optimal strategy. This phenomenon is modeled as a *perturbation* game where the time delay suggested by the controller can be perturbed by a bounded quantity. Notice that such a perturbation may result in the guard of the corresponding action being disabled. In the conservative semantics, it is the controller's responsibility to make sure that the guards are satisfied after the perturbation. On the other hand, in the excess semantics, the controller is supposed to make sure that the guard is satisfied before the perturbation: an action can be executed even when its guard is disabled ("excess") post perturbation and the valuations post perturbation will be reflected in the next state. The game based characterization for robustness in timed automata under "excess" semantics was first proposed by Bouyer, Markey, and Sankur [6] where they study the parameterized robust (qualitative) reachability problem and show it to be EXPTIME-complete. The "conservative" semantics were studied for reachability and Büchi objectives in [14] and shown to be PSPACE-complete. For a detailed survey on robustness in timed setting we refer to an excellent survey by Markey [12].

Bouyer, Markey, and Sankur [7] showed that the problem for deciding the existence of the optimal strategy is undecidable for priced timed games with 10 or more clocks under the excess semantics. In this paper we further improve the understanding of the decidability of these games. However, to keep the presentation simple, we restrict our attention to turn-based games under excess semantics. To further generalize the setting, we permit both positive and negative price rates with the restriction that the accumulated cost in any cycle is non-negative (akin to the standard no-negative-cycle restriction in shortest path game problems on finite graphs). We improve the undecidability result of [7] by proving that optimal reachability remains undecidable for robust priced timed automata with 5 clocks. Our second key result is that, for a fixed $\delta$, the cost optimal reachability problem for one clock priced timed games with no-negative-cycle restriction is decidable for robust priced timed games with given bound on perturbations. To the best of our knowledge, this is the first decidability result known for robust timed games under the excess semantics. A closely related result is [9], where decidability is shown for robust timed games under the conservative semantics for a fixed $\delta$.

## 2 Preliminaries

We write $\mathbb{R}$ for the set of reals and $\mathbb{Z}$ for the set of integers. Let $\mathcal{C}$ be a finite set of real-valued variables called *clocks*. A *valuation* on $\mathcal{C}$ is a function $\nu : \mathcal{C} \to \mathbb{R}$. We assume an arbitrary but fixed ordering on the clocks and write $x_i$ for the clock with order $i$. This allows us to treat a valuation $\nu$ as a point $(\nu(x_1), \nu(x_2), \ldots, \nu(x_n)) \in \mathbb{R}^{|\mathcal{C}|}$. Abusing notations slightly, we use a valuation on $\mathcal{C}$ and a point in $\mathbb{R}^{|\mathcal{C}|}$ interchangeably. For a subset of clocks $X \subseteq \mathcal{C}$ and valuation $\nu \in \mathbb{R}^{|\mathcal{C}|}$, we write $\nu[X{:=}0]$ for the valuation where $\nu[X{:=}0](x) = 0$ if $x \in X$, and $\nu[X{:=}0](x) = \nu(x)$ otherwise. The valuation $\mathbf{0} \in \mathbb{R}^{|\mathcal{C}|}$ is a special valuation such that $\mathbf{0}(x) = 0$ for all $x \in \mathcal{C}$. A clock constraint over $\mathcal{C}$ is a subset of $\mathbb{R}^{|\mathcal{C}|}$. We say that a constraint is *rectangular* if it is a conjunction of a finite set of constraints of the form $x \bowtie k$, where $k \in \mathbb{Z}$, $x \in \mathcal{C}$, and $\bowtie \in \{<, \leq, =, >, \geq\}$. For a constraint $g \in \varphi(\mathcal{C})$, we write $[\![g]\!]$ for the set of valuations in $\mathbb{R}^{|\mathcal{C}|}$ satisfying $g$. We write $\varphi(\mathcal{C})$ for the set of rectangular constraints over $\mathcal{C}$.

We use the terms constraints and guards interchangeably.

Following [5] we introduce priced timed games with external cost function on target locations (see [10]). For this purpose, we define a *cost function*[5] as a piecewise affine continuous function $f : \mathbb{R}^n_{\geq 0} \to \mathbb{R} \cup \{+\infty, -\infty\}$. We write $\mathcal{F}$ for the set of all cost functions.

▶ **Definition 2.1** (Priced Timed Games)**.** A turn-based two player *priced timed game* is a tuple $\mathcal{G} = (L_1, L_2, L_{init}, \mathcal{C}, X, \eta, T, f_{goal})$ where $L_i$ is a finite set of *locations* of Player $i$, $L_{init} \subseteq L_1 \cup L_2$(let $L_1 \cup L_2 = L$) is a set of initial locations, $\mathcal{C}$ is an (ordered) set of *clocks*, $X \subseteq L \times \varphi(\mathcal{C}) \times 2^{\mathcal{C}} \times (L \cup T)$ is the *transition relation*, $\eta\colon L \to \mathbb{Z}$ is the price function, $T$ is the set of target locations, $T \cap L = \emptyset$; and $f_{goal} : T \to \mathcal{F}$ assigns external cost functions to target locations.

We refer to Player 1 as the controller and Player 2 as the environment. A priced timed game begins with a token placed on some initial location $\ell$ with valuation $\mathbf{0}$ and cost accumulated being so far being 0. At each round, the player who controls the current location $\ell$ chooses a delay $t$ (to be elapsed in $l$) and an outgoing transition $e = (\ell, g, r, \ell') \in X$ to be taken after $t$ delay at $\ell$. The clock valuation is then updated according to the delay $t$, the reset $r$, the cost is incremented by $\eta(\ell) \cdot t$ and the token is moved to the location $\ell'$. The two players continue moving the token in this fashion, and give rise to a sequence of locations and transitions called a *play* of the game. A configuration or state of a PTG is a tuple $(\ell, \nu, c)$ where $\ell \in L$ is a location, $\nu \in \mathbb{R}^{|\mathcal{C}|}$ is a valuation, and $c$ is the cost accumulated from the start of the play. We assume, w.l.o.g [2], that the clock valuations are bounded.

▶ **Definition 2.2** (PTG semantics)**.** The semantics of a PTG $\mathcal{G}$ is a labelled state-transition game arena $[\![\mathcal{G}]\!] = (\mathcal{S} = S_1 \uplus S_2, S_{init}, A, E, \pi, \kappa)$ where

- $S_j = L_j \times \mathbb{R}^{|\mathcal{C}|}$ are the Player $j$ states with $\mathcal{S} = S_1 \uplus S_2$,
- $S_{init} \subseteq \mathcal{S}$ are initial states s.t. $(\ell, \nu) \in S_{init}$ if $\ell \in L_{init}$, $\nu = \mathbf{0}$,
- $A = \mathbb{R}_{\geq 0} \times X$ is the set of *timed moves*,
- $E : (\mathcal{S} \times A) \to \mathcal{S}$ is the transition function s.t. for $s = (\ell, \nu), s' = (\ell', \nu') \in \mathcal{S}$ and $\tau = (t, e) \in A$ the function $E(s, \tau)$ is defined if $e = (\ell, g, r, \ell')$ is a transition of the PTG and $\nu \in [\![g]\!]$; moreover $E(s, \tau) = s'$ if $\nu' = (\nu + t)[r{:=}0]$ (we write $s \xrightarrow{\tau} s'$ when $E(s, \tau) = s'$);
- $\pi : \mathcal{S} \times A \to \mathbb{R}$ is the price function such that $\pi((\ell, \nu), (t, e)) = \eta(\ell) \cdot t$; and
- $\kappa : \mathcal{S} \to \mathbb{R}$ is an external cost function such that $\kappa(\ell, \nu)$ is defined when $\ell \in T$ such that $\kappa(\ell, \nu) = f_{goal}(\ell)(\nu)$.

A *play* $\rho = \langle s_0, \tau_1, s_1, \tau_2, \ldots, s_n \rangle$ is a finite sequence of states and actions s.t. $s_0 \in S_{init}$ and $s_i \xrightarrow{\tau_{i+1}} s_{i+1}$ for all $0 \leq i < n$. The infinite plays are defined in an analogous manner. For a finite play $\rho$ we write its last state as $\text{last}(\rho) = s_n$. For a (infinite or finite) play $\rho$ we write $\text{stop}(\rho)$ for the index of first target state and if it doesn't visit a target state then $\text{stop}(\rho) = \infty$. We denote the set of plays as $\mathsf{Plays}_{\mathcal{G}}$. For a play $\rho = \langle s_0, (t_1, a_1), s_1, (t_2, a_2), \ldots \rangle$ if $\text{stop}(\rho) = n < \infty$ then $\mathsf{Cost}_{\mathcal{G}}(\rho) = \kappa(s_n) + \sum_{j=1}^{n} \pi(s_{i-1}, (t_i, a_i))$ else $\mathsf{Cost}_{\mathcal{G}}(\rho) = +\infty$.

A *strategy* of player $j$ in $\mathcal{G}$ is a function $\sigma : \mathsf{Plays}_{\mathcal{G}} \to A$ such that for a play $\rho$ the function $\sigma(\rho)$ is defined if $\text{last}(\rho) \in S_j$. We say that a strategy $\sigma$ is memoryless if $\sigma(\rho) = \sigma(\rho')$ when $\text{last}(\rho) = \text{last}(\rho')$, otherwise we call it memoryful. We write $\mathsf{Strat}_1$ and $\mathsf{Strat}_2$ for the set of strategies of player 1 and 2, respectively.

A play $\rho$ is said to be *compatible to a strategy* $\sigma$ of player $j \in \{1, 2\}$ if for every state $s_i$ in $\rho$ that belongs to Player $j$, $s_{i+1} = \sigma(s_i)$. Given a pair of strategies $(\sigma_1, \sigma_2) \in \mathsf{Strat}_1 \times \mathsf{Strat}_2$, and a state $s$, the outcome of $(\sigma_1, \sigma_2)$ from $s$ denoted $\mathsf{Outcome}(s, \sigma_1, \sigma_2)$ is the unique play that starts at $s$ and is compatible with both strategies. Given a player 1 strategy $\sigma_1 \in \mathsf{Strat}_1$

we define its cost $\mathsf{Cost}_{\mathcal{G}}(s, \sigma_1)$ as $\sup_{\sigma_2 \in \mathsf{Strat}_2}(\mathsf{Cost}(\mathsf{Outcome}(s, \sigma_1, \sigma_2)))$. We now define the *optimal reachability-cost* for Player 1 from a state $s$ as

$$\mathsf{OptCost}_{\mathcal{G}}(s) = \inf_{\sigma_1 \in \mathsf{Strat}_1} \sup_{\sigma_2 \in \mathsf{Strat}_2} (\mathsf{Cost}(\mathsf{Outcome}(s, \sigma_1, \sigma_2))).$$

A strategy $\sigma_1 \in \mathsf{Strat}_1$ is said to be optimal from $s$ if $\mathsf{Cost}_{\mathcal{G}}(s, \sigma_1) = \mathsf{OptCost}_{\mathcal{G}}(s)$. Since the optimal strategies may not always exist [5] we define $\epsilon$ optimal strategies. For $\epsilon > 0$ a strategy $\sigma_\epsilon \in \mathsf{Strat}_1$ is called $\epsilon$-optimal if $\mathsf{OptCost}_{\mathcal{G}}(s) \leq \mathsf{Cost}_{\mathcal{G}}(s, \sigma_\epsilon) < \mathsf{OptCost}_{\mathcal{G}}(s) + \epsilon$. Given a PTG $\mathcal{G}$ and a bound $K \in \mathbb{Z}$, the *cost-optimal* reachability problem for PTGs is to decide whether there exists a strategy for player 1 such that $\mathsf{OptCost}_{\mathcal{G}}(s) \leq K$ from some starting state $s$.

▶ **Theorem 2.3** ([3]). *Cost-optimal reachability problem is undecidable for PTGs with* 3 *clocks.*

▶ **Theorem 2.4** ([5, 11, 13]). *The $\epsilon$-optimal strategy is computable for* 1 *clock PTGs.*

## 3 Robust Semantics

Under the robust semantics of priced timed games the environment player – also called as the perturbator – is more privileged as it has the power to perturb any delay chosen by the controller by an amount in $[-\delta, \delta]$, where $\delta > 0$ is a pre-defined bounded quantity. However, in order to ensure time-divergence there is a restriction that the time delay at all locations of the RPTG must be $\geq \delta$. There are the following two perturbation semantics as defined in [7].

- *Excess semantics.* At any controller location, the time delay $t$ chosen by the controller is altered to some $t' \in [t - \delta, t + \delta]$ by the perturbator. However, the constraints on the outgoing transitions of the controller locations are evaluated with respect to the time elapse $t$ chosen by the controller. If the constraint is satisfied with respect to $t$, then the values of all variables which are not reset on the transition are updated with respect to $t'$; the variables which are reset obtain value 0.
- *Conservative semantics.* In this, the constraints on the outgoing transitions are evaluated with respect to $t'$.

In both semantics, the delays chosen by perturbator at his locations are not altered, and the constraints on outgoing transitions are evaluated in the usual way, as in PTG.

A Robust-Priced Timed Automata (RPTA) is an RPTG which has only controller locations. At all these locations, for any time delay $t$ chosen by controller, perturbator can implicitely perturb $t$ by a quantity in $[-\delta, \delta]$. The excess as well as the conservative perturbation semantics for RPTA are defined in the same way as in the RPTG. Note that our RPTA coincides with that of [7] when the cost functions at all target locations are of the form $cf : \mathbb{R}^n_{\geq 0} \to \{0\}$. Our RPTG are turn-based, and have cost funtions at the targets, while RPTGs studied in [7] are concurrent.

▶ **Definition 3.1** (Excess Perturbation Semantics). Let $\mathcal{R} = (L_1, L_2, L_{init}, \mathcal{C}, X, \eta, T, f_{goal})$ be a RPTG. Given a $\delta > 0$, the excess perturbation semantics of RPTG $\mathcal{R}$ is a LTS $[\![\mathcal{R}]\!] = (\mathcal{S}, A, E)$ where $\mathcal{S} = S_1 \cup S_2 \cup (T \times \mathbb{R}_{\geq 0})$, $A = A_1 \cup A_2$ and $E = E_1 \cup E_2$. We define the set of states, actions and transitions for each player below.

- $S_1 = L_1 \times \mathbb{R}^{|\mathcal{C}|}$ are the controller states,

- $S_2 = (L_2 \times \mathbb{R}^{|\mathcal{C}|}) \cup (S_1 \times \mathbb{R}_{\geq 0} \times X)$ are the perturbator states. The first kind of states are encountered at perturbator locations. The second kind of states are encountered when controller chooses a delay $t \in \mathbb{R}_{\geq 0}$ and a transition $e \in X$ at a controller location.
- $A_1 = \mathbb{R}_{\geq 0} \times X$ are controller actions
- $A_2 = (\mathbb{R}_{\geq 0} \times X) \cup [-\delta, \delta]$ are perturbator actions. The first kind of actions $(\mathbb{R}_{\geq 0} \times X)$ are chosen at states of the form $L_2 \times \mathbb{R}^{|\mathcal{C}|} \in S_2$, while the second kind of actions are chosen at states of the form $S_1 \times \mathbb{R}_{\geq 0} \times X \in S_2$,
- $E_1 = (S_1 \times A_1 \times S_2)$ is the set of controller transitions such that for a controller state $(l, \nu)$ and a controller action $(t, e)$, $E_1((l, \nu), (t, e))$ is defined iff there is a transition $e = (l, g, a, r, l')$ in $\mathcal{R}$ such that $\nu + t \in [\![g]\!]$.
- $E_2 = S_2 \times A_2 \times (S_1 \cup S_2 \cup (T \times \mathbb{R}_{\geq 0}))$ is the set of perturbator transitions such that
  - For a perturbator state of the type $(l, \nu)$ and a perturbator action $(t, e)$, we have $(l', \nu') = E_2((l, \nu), (t, e))$ iff there is a transition $e = (l, g, a, r, l')$ in $\mathcal{R}$ such that $\nu + t \in [\![g]\!]$, $\nu' = (\nu + t)[r := 0]$,
  - For a perturbator state of type $((l, \nu), t, e)$ and a perturbator action $\varepsilon \in [-\delta, \delta]$, we have $(l', \nu') = E_2(((l, \nu), t, e), \varepsilon)$ iff $e = (l, g, a, r, l')$, and $\nu' = (\nu + t + \varepsilon)[r := 0]$.

We now define the cost of the transitions, denoted as $\mathsf{Cost}(t, e)$ as follows :

- For controller transitions : $(l, \nu) \xrightarrow{(t,e)} ((l, \nu), t, e)$ : the cost accumulated is $\mathsf{Cost}(t, e) = 0$.
- For perturbator transitions :
  - From perturbator states of type $(l, \nu)$ : $(l, \nu) \xrightarrow{t,e} (l', \nu')$, the cost accumulated is $\mathsf{Cost}(t, e) = t * \eta(l)$.
  - From perturbator states of type $((l, \nu), t, e)$ : $((l, \nu), t, e) \xrightarrow{\varepsilon} (l', \nu')$, the cost accumulated is $(t + \varepsilon) * \eta(l)$. Note that although this transition has no edge choice involved and the perturbation delay chosen is $\varepsilon \in [-\delta, \delta]$, the controller action $(t, e)$ chosen in the state $(l, \nu)$ comes into effect in this transition. Hence for the sake of uniformity, we denote the cost accumulated in this transition to be $\mathsf{Cost}(t + \varepsilon, e) = (t + \varepsilon) * \eta(l)$.

Note that we check satisfiability of the constraint $g$ before the perturbation; however, the reset occurs after the perturbation. The notions of a path and a winning play are the same as in PTG. We shall now adapt the definitions of cost of a play, and a strategy for the excess perturbation semantics. Let $\rho = \langle s_1, (t_1, e_1), s_2, (t_2, e_2), \cdots (t_{n-1}, e_{n-1}), s_n \rangle$ be a path in the LTS $[\![\mathcal{R}]\!]$. Given a $\delta > 0$, for a finite play $\rho$ ending in target location, we define $\mathsf{Cost}^{\delta}_{\mathcal{R}}(\rho) = \sum_{i=1}^{n} \mathsf{Cost}(t_i, e_i) + f_{goal}(l_n)(\nu_n)$ as the sum of the costs of all transitions as defined above along with the value from the cost function of the target location $l_n$. Also, we re-define the cost of a strategy $\sigma_1$ from a state $s$ for a given $\delta > 0$ as $\mathsf{Cost}^{\delta}_{\mathcal{R}}(s, \sigma_1) = \sup_{\sigma_2 \in \mathsf{Strat}_2(\mathcal{R})} \mathsf{Cost}^{\delta}_{\mathcal{R}}(\mathsf{Outcome}(s, \sigma_1, \sigma_2))$. Similarly, $\mathsf{OptCost}^{\delta}_{\mathcal{R}}$ is the optimal cost under excess perturbation semantics for a given $\delta > 0$ defined as

$$\mathsf{OptCost}^{\delta}_{\mathcal{R}}(s) = \inf_{\sigma_1 \in \mathsf{Strat}_1(\mathcal{R})} \sup_{\sigma_2 \in \mathsf{Strat}_2(\mathcal{R})} (\mathsf{Cost}^{\delta}_{\mathcal{R}}(\mathsf{Outcome}(s, \sigma_1, \sigma_2))).$$

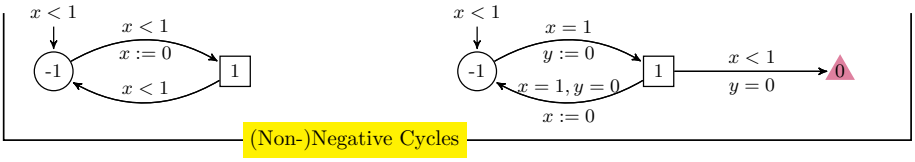
Since optimal strategies may not always exist, we define $\epsilon-$optimal strategies such that for every $\epsilon > 0$, $\mathsf{OptCost}^{\delta}_{\mathcal{R}}(s) \leq \mathsf{Cost}^{\delta}_{\mathcal{R}}(s, \sigma_1) < \mathsf{OptCost}^{\delta}_{\mathcal{R}}(s) + \epsilon$. Given a $\delta$ and a RPTG $\mathcal{R}$ with a single clock $x$, a strategy $\sigma_1$ is called $(\epsilon, N)-$*acceptable* [5] for $\epsilon > 0, N \in \mathbb{N}$ when (1)it is memoryless, (2)it is $\epsilon-$optimal and (3)there exist $N$ consecutive intervals $(I_i)_{1 \leq i \leq N}$ partitioning $[0, 1]$ such that for every location $l$, for every $1 \leq i \leq N$ and every integer $\alpha < M$ (where $M$ is the maximum bound on the clock value), the function that maps the clock values $\nu(x)$ to the cost of the strategy $\sigma_1$ at every state $(l, \nu(x))$, $(\nu(x) \mapsto \mathsf{Cost}^{\delta}_{\mathcal{R}}((l, \nu(x)), \sigma_1))$ is

affine for every interval $\alpha + I_i$. Also, the strategy $\sigma_1$ is constant over the values $\alpha + I_i$ at all locations, that is, when $\nu(x) \in \alpha + I_i$, the strategy $\sigma_1(l, \nu(x))$ is constant. The number $N$ is an important attribute of the strategy as it establishes that the strategy does not fluctuate infinitely often and is implementable.

Now, we shall define limit variations of costs, strategies and values as $\delta \to 0$. The *limit-cost* of a controller strategy $\sigma_1$ from state $s$ is defined over all plays $\rho$ starting from $s$ that are compatible with $\sigma_1$ as:

$$\mathsf{LimCost}_{\mathcal{R}}(s, \sigma_1) = \lim_{\delta \to 0} \sup_{\sigma_2 \in \mathsf{Strat}_2(\mathcal{R})} \mathsf{Cost}^{\delta}_{\mathcal{R}}(\mathsf{Outcome}(s, \sigma_1, \sigma_2)).$$

The *limit strategy upper-bound problem* [7] for excess perturbation semantics asks, given a RPTG $\mathcal{R}$, state $s = (l, \mathbf{0})$ with cost 0 and a rational number $K$, whether there exists a strategy $\sigma_1$ such that $\mathsf{LimCost}_{\mathcal{R}}(s, \sigma_1) \leq K$. The following are the main results of [7].




▶ **Theorem 3.2** (Known results [7]).

1. *The limit-strategy upper-bound problem is undecidable for RPTA and RPTG under excess perturbation semantics, for* $\geq 10$ *clocks.*
2. *For a fixed* $\delta \in [0, \frac{1}{3}]$*, and a given RPTA* $\mathcal{A}$*, a target location* $l$ *and a rational* $K$*, it is undecidable whether* $\inf_{\sigma_1} \sup_{\sigma_2} cost_{\sigma_1,\sigma_2}(\rho) < K$ *such that* $\rho$ *ends in* $l$*.* $cost_{\sigma_1,\sigma_2}(\rho)$ *is the cost of the unique run* $\rho$ *obtained from the pair of strategies* $(\sigma_1, \sigma_2)$*.*

We consider a semantic subclass of RPTGs in which the accumulated cost of any cycle is non-negative: that is, any iteration of a cycle will always have a non-negative cost. Consider the two cycles depicted. The one on top has a non-negative cost, while the one below always has a negative cost. In the cycle below, the perturbator will not perturb, since that will lead to a target state. In the rest of the paper, we consider this semantic class of RPTGs (RPTAs), and prove decidability and undecidability results; however, we will refer to them as RPTGs(RPTAs). Our key contributions are the following theorems.

▶ **Theorem 3.3.** *The limit-strategy upper-bound problem is undecidable for RPTA with 5 clocks, location prices in* $\{0, 1\}$*, and cost functions* $cf : \mathbb{R}^n_{\geq 0} \to \{0\}$ *at all target locations.*

▶ **Theorem 3.4.** *Given a 1-clock RPTG* $\mathcal{R}$ *and a* $\delta > 0$*, we can compute* $\mathsf{OptCost}^{\delta}_{\mathcal{R}}(s)$ *for every state* $s = (l, \nu)$*. For every* $\epsilon > 0$*, there exists an* $N \in \mathbb{N}$ *such that the controller has an* $(\epsilon, N)$*-acceptable strategy.*

The rest of the paper is devoted to the proof sketches of these two theorems, while we give detailed proofs in [10].

## 4 Undecidability with 5 clocks

In this section, we improve the result of [7] by showing that the limit strategy upper bound problem is undecidable for robust priced timed automata with 5 or more clocks. The undecidability result is obtained using a reduction to the halting problem of two-counter machines.

A two-counter machine has counters $C_1$ and $C_2$, and a list of instructions $I_1, I_2, \ldots, I_n$, where $I_n$ is the *halt instruction*. For each $1 \leq i \leq n-1$, $I_i$ is one of the following instructions: `increment` $c_b$`:` $c_b := c_b + 1$; *goto* $I_j$, for $b = 1$ or 2, `decrement` $c_b$ `with zero test:` *if* $(c_b = 0)$ *goto* $I_j$ *else* $c_b := c_b - 1$; *goto* $I_j$, where $c_1, c_2$ represent the counter values. The initial values of both counters are 0. Given the initial configuration $(I_1, 0, 0)$ the halting problem for two counter machines is to find if the configuration $(I_n, c_1, c_2)$ is reachable, with $c_1, c_2 \geq 0$. This problem is known to be undecidable.

We simulate the two counter machine using a RPTA with 5 clocks $x_1, z, x_2, y_1$ and $y_2$ under the excess perturbation semantics. The counters are encoded in clocks $x_1$ and $z$ as $x_1 = \frac{1}{2^i} + \varepsilon_1$ and $z = \frac{1}{2^j} + \varepsilon_2$ where $i, j$ are respectively the values of counters $C_1, C_2$, and $\varepsilon_1$ and $\varepsilon_2$ denote accumulated values due to possible perturbations. Clocks $x_2$, $y_1$ and $y_2$ help with the rough work. The simulation is achieved as follows: for each instruction, we have a module simulating it. Upon entering the module, the clocks are in their normal form i.e. $x_1 = \frac{1}{2^i} + \varepsilon_1, z = \frac{1}{2^j} + \varepsilon_2$ and $x_2 = 0$ and $y_1 = y_2 = 0$.

## 4.1 Increment module

The module in Figure 1 simulates the increment of counter $C_1$. The value of counter $C_2$ remains unchanged since the value of clock $z$ remains unchanged at the exit from the module. Upon entering $A$ the clock values are $x_1 = \frac{1}{2^i} + \varepsilon_1, z = \frac{1}{2^j} + \varepsilon_2, x_2 = y_1 = y_2 = 0$. Here $\varepsilon_1$ and $\varepsilon_2$ respectively denote the perturbations accumulated so far. We denote by $\alpha$, the value of clock $x_1$, i.e. $\frac{1}{2^i} + \varepsilon_1$. Thus at $A$, the delay is $1 - \alpha$. Note that the dashed edges are unperturbed (this is a short hand notation. A small gadget that implements this is described in [10], so $x_1 = 1$ on entering $B$. No time elapse happens at $B$, and at $C$, controller chooses a delay $t$. This $t$ must be $\frac{\alpha}{2}$ to simulate the increment correctly. $t$ can be perturbed by an amount $\delta$ by the perurbator, where $\delta$ can be both positive or negative, obtaining $x_2 = t + \delta, x_1 = 0, y_1 = 1 - \alpha + t + \delta$ on entering $D$. At $D$, the delay is $\alpha - t - \delta$. Thus the total delay from the entry point $A$ in this module to the mChoice module is 1 time unit. At the entry of the *mChoice* (*mChoice* and Restore modules are in [10]) module, the clock values are $x_1 = \alpha - t - \delta, z = 1 + \frac{1}{2^j} + \varepsilon_2, x_2 = \alpha, y_1 = 1, y_2 = 0$. To correctly simulate the increment of $C_1$, $t$ should be exactly $\frac{\alpha}{2}$.

At the mChoice module, perturbator can either continue the simulation by going through the Restore module or verify the correctness of controller's delay (check $t = \frac{\alpha}{2}$). The mChoice module adds 3 units to the values of $x_1, x_2$ and $z$, and resets $y_1, y_2$. Due to the mChoice module, the clock values are $x_1 = 3 + \alpha - t - \delta, z = 4 + \frac{1}{2^j} + \varepsilon_2, x_2 = 3 + \alpha, y_1 = 1, y_2 = 0$. If perturbator chooses to continue the simulation, then Restore module brings all the clocks back to normal form. Hence upon entering $F$, the clock values are $x_1 = \alpha - t - \delta, z = \frac{1}{2^j} + \varepsilon_2, x_2 = y_1 = 1, y_2 = 0$. This value of $x_1$ is $\frac{\alpha}{2} + \varepsilon_1$, since $t = \frac{\alpha}{2}$ and $\varepsilon_1 = -\delta$, the perturbation effect.

Let us now see how perturbator verifies $t = \frac{\alpha}{2}$ by entering the Choice module. The Choice module also adds 3 units to the values of $x_1, x_2$ and $z$, and resets $y_1, y_2$. The module $Test\ Inc_{>}^{C_1}$ is invoked to check if $t > \frac{\alpha}{2}$, and the module $Test\ Inc_{<}^{C_1}$ is invoked to check if $t < \frac{\alpha}{2}$. Note that using the mChoice module and the Choice module one after the other, the clock values upon entering $Test\ Inc_{>}^{C_1}$ or $Test\ Inc_{<}^{C_1}$ are $x_1 = 6 + \alpha - t - \delta, z = 7 + \frac{1}{2^j} + \varepsilon_2, x_2 = 6 + \alpha, y_1 = 0, y_2 = 0$.

**Test $\mathbf{Inc_{>}^{C_1}}$**: The delay at $A'$ is $1 - \alpha + t + \delta$, obtaining $x_2 = 7 + t + \delta$, and the cost accumulated is $1 - \alpha + t + \delta$. At $B'$, $1 - t - \delta$ time is spent, obtaining $x_1 = 1 - t - \delta$. Finally, at $C'$, a time $t + \delta$ is spent, and at $D'$, one time unit, making the total cost accumulated $2 - \alpha + 2t + 2\delta$ at the target location. The cost function at the target assigns the cost 0 for

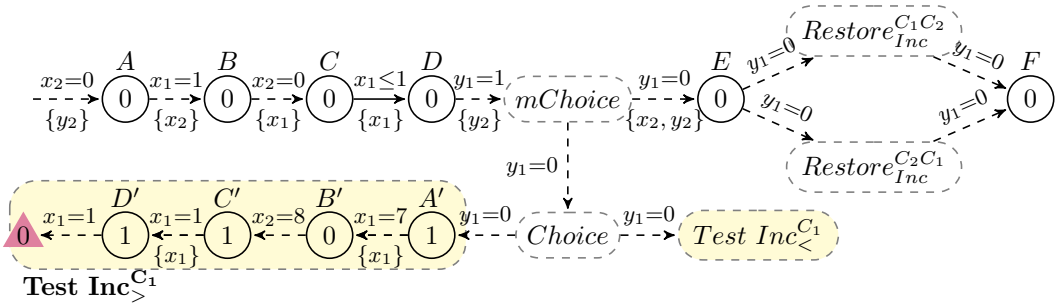



**Figure 1 Increment $C_1$ module** : The module keeps the fractional part of the clock $z$ unchanged. The dashed edges represent unperturbed edges (detailed in [10]).

all valuations, hence the total cost to reach the target is $2 + 2t - \alpha + 2\delta$ which is greater than $2 + 2\delta$ iff $2t - \alpha > 0$, i.e. iff $t > \frac{\alpha}{2}$.

▶ **Lemma 4.1.** *Assume that an increment $C_b$ ($b \in \{0, 1\}$) module is entered with the clock valuations in their normal forms. Then controller has a strategy to reach either location $l_j$ corresponding to instruction $I_j$ of the two-counter machine or a target location is reached with cost at most $2 + |2\delta|$, where $\delta$ is the perturbation added by perturbator.*

## 4.2 Complete Reduction

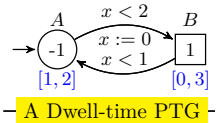
The entire reduction consists of constructing a module corresponding to each instruction $I_i$, $1 \leq i \leq n$, of the two-counter machine. The first location of the module corresponding to instruction $I_1$ is the initial location. We simulate the halting instruction $I_n$ by a target location with cost function $cf : \mathbb{R}^5_{\geq 0} \to \{0\}$. We denote the robust timed automaton simulating the two counter machine by $\mathcal{A}$, $s$ is the initial state $(l, \mathbf{0}, \mathbf{0})$.

▶ **Lemma 4.2.** *The two counter machine halts if and only if there is a strategy $\sigma$ of controller such that $limcost_{\mathcal{A}}(\sigma, s) \leq 2$.*

The details of the decrement and zero test modules are in [10]. They are similar to the increment module; if player 2 desires to verify the correctness of player 1's simulation, a cost $> 2 + |2\delta|$ is accumulated on reaching a target location iff player 1 cheats. In the limit, as $\delta \to 0$, the limcost will be $> 2$ iff controller cheats. The other possibility to obtain a limcost $> 2$ is when the two counter machine does not halt.

# 5 Decidability of One-clock RPTG

In order to show the decidability of the optimal reachability game for 1 clock RPTG $\mathcal{R}$ and a fixed $\delta > 0$, we perform a series of reachability and optimal cost preserving transformations. The idea is to reduce the RPTG into a simpler priced timed game, while preserving the optimal costs. The advantages of this conversion is that the semantics of PTGs are easier to understand, and one could adapt known algorithms to solve PTGs. On the other hand, the PTGs that we obtain are 1-clock PTGs with dwell-time requirement (having restrictions on minimum as well as maximum amount of time spent at certain locations), see for example, a

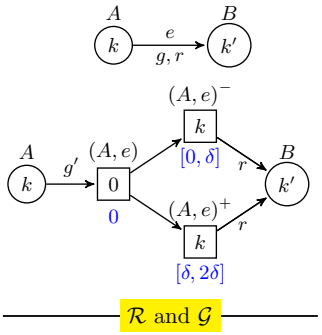


dwell-time PTG with two locations $A, B$. A minimum of 1 and a maximum of two units of time should be spent at $A$, while a maximum of 3 time units can be spent at $B$. If we wish to model this using standard PTGs, we need one extra clock and we can not use the decidability results of 1 clock PTG to show the decidability of our model. We show in Section 5.4 how to solve 1-clock PTGs with dwell-time requirements.

Our transformations are as follows: (i) for a given $\delta$, our first transformation reduces the RPTG $\mathcal{R}$ into a dwell-time PTG $\mathcal{G}$ (Section 5.1); (ii) our second transformation restricts to dwell-time PTGs where the clock is bounded by $1+\delta$. To achieve this, we use a notion of *fractional resets*, and denote these PTGs as $\mathcal{G}_{\mathcal{F}}$ (Section 5.2); (iii) our third and last transformation restricts $\mathcal{G}_{\mathcal{F}}$ without resets (Section 5.3). The reset-free dwell-time PTG is denoted ${\mathcal{G}_{\mathcal{F}}}'$. For each transformation, we prove that the optimal cost in each state of the original game is the same as the optimal cost at some corresponding state of the new game. We also show that an $(\epsilon, N)$-strategy of the original game can be computed from some $(\epsilon', N')$-strategy in the new game. The details of each transformation and correctness is established in subequent sections. We then solve ${\mathcal{G}_{\mathcal{F}}}'$ employing a technique inspired by [5] while ensuring that the robust semantics are satisfied.

## 5.1 Transformation 1: RPTG $\mathcal{R}$ to dwell-time PTG $\mathcal{G}$

Given a one clock RPTG $\mathcal{R} = (L_1, L_2, \{x\}, X, \eta, T, f_{goal})$ and a $\delta > 0$, we construct a dwell-time PTG $\mathcal{G} = (L_1, L_2 \cup L', \{x\}, X', \eta', T, f_{goal})$. All the controller, perturbator locations of $\mathcal{R}$ ($L_1$ and $L_2$) are carried over respectively as player 1, player 2 locations in $\mathcal{G}$. In addition, we have some new player 2 locations $L'$ in $\mathcal{G}$. The dwell-time PTG $\mathcal{G}$ constructed has dwell-time restrictions for the new player 2 locations $L'$. The locations of $L'$ are either urgent, or have a a dwell-time of $[\delta, 2\delta]$ or $[0, \delta]$. All the perturbator transitions of $\mathcal{R}$ are retained as it is in $\mathcal{G}$. Every transition in $\mathcal{R}$ from a controller location $A$ to some location $B$ is replaced in $\mathcal{G}$ by a game graph as shown. Let $e = (A, g, r, B)$ be the transition from a controller location $A$ to a location $B$ with guard $g$, and reset $r$. Depending on the guard $g$, in the transformed game graph, we have the new guard $g'$. If $g$ is $x = H$, then $g'$ is $x = H - \delta$, while if $g$ is $H < x < H + 1$, then $g'$ is $H - \delta < x < H + 1 - \delta$, for $H > 0$. When $g$ is $0 < x < K$, then $g'$ is $0 \leq x < K - \delta$ and $x = 0$ stays unchanged. It can be seen that doing this transformation to all the controller edges of a RPTG $\mathcal{R}$ gives rise to a dwell-time PTG $\mathcal{G}$.



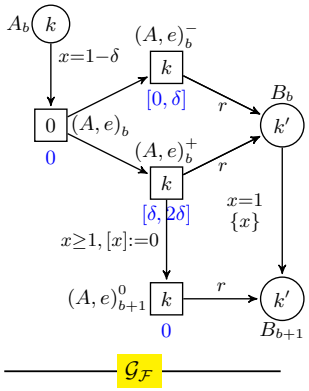

Lets consider the transition from $A$ to $B$ in $\mathcal{R}$. Assume that the transition from $A$ to $B$ (called edge $e$) had a constraint $x = 1$, and assume that $x = \nu$ on entering $A$. Then, in $\mathcal{R}$, controller elapses a time $1 - \nu$, and reaches $B$; however on reaching $B$, the value of $x$ is in the range $[1 - \delta, 1 + \delta]$ depending on the perturbation. Also, the cost accumulated at $A$ is $k * (1 - \nu + \gamma)$, where $\gamma \in [-\delta, \delta]$. To take into consideration these semantic restrictions of $\mathcal{R}$, we transform the RPTG $\mathcal{R}$ into a dwell-time PTG $\mathcal{G}$. First of all, we change the constraint $x = 1$ into $x = 1 - \delta$ from $A$ (a player 1 location) and enter a new player 2 location $(A, e)$. This player 2 location is an urgent location. The correct strategy for player 1 is to spend a time $1 - \nu - \delta$ at $A$ (corresponding to the time $1 - \nu$ he spent at $A$ in $\mathcal{R}$). At $(A, e)$, player 2 can either proceed to one of the player 2 locations $(A, e)^-$ or $(A, e)^+$. The player 2 location $(A, e)$ models perturbator's choices of positive or negative perturbation in $\mathcal{R}$. If player 2 goes to $(A, e)^-$, then on reaching $B$, the value of $x$ is in the interval $[1 - \delta, 1]$ (this

corresponds to perturbator's choice of $[-\delta, 0]$ in $\mathcal{R}$) and if he goes to $(A,e)^+$, then the value of $x$ at $B$ is in the interval $[1, 1+\delta]$ (this corresponds to perturbator's choice of $[0,\delta]$ in $\mathcal{R}$). The reset happening in the transition from $A$ to $B$ in $\mathcal{R}$ is now done on the transition from $(A,e)^-$ to $B$ and from $(A,e)^+$ to $B$. Thus, note that the possible ranges of $x$ as well as the accumulated cost in $\mathcal{R}$ while reaching $B$ are preserved in the transformed dwell-time PTG.

▶ **Lemma 5.1.** *Let $\mathcal{R}$ be a RPTG and $\mathcal{G}$ be the corresponding dwell-time PTG obtained using the transformation above. Then for every state $s$ in $\mathcal{R}$, $\mathsf{OptCost}_{\mathcal{R}}(s) = \mathsf{OptCost}_{\mathcal{G}}(s)$. An $(\epsilon, N)-$strategy in $\mathcal{R}$ can be computed from a $(\epsilon, N)-$strategy in $\mathcal{G}$ and vice versa.*

Proof in [10].

## 5.2 Transformation 2: Dwell-time PTG $\mathcal{G}$ to Dwell-time FRPTG $\mathcal{G}_{\mathcal{F}}$




Recall that the locations of the dwell-time PTG $\mathcal{G}$ is $L_1 \cup L_2 \cup L'$ where $L_1 \cup L_2$ are the set of locations of $\mathcal{R}$, and $L'$ are new player 2 locations introduced in $\mathcal{G}$. In this section, we transform the dwell-time PTG $\mathcal{G}$ into a dwell-time PTG $\mathcal{G}_{\mathcal{F}}$ having the restriction that the value of $x$ is in [0,1] at all locations corresponding to $L_1 \cup L_2$, and is in $[0, 1+\delta]$ at all locations corresponding to $L'$. While this transformation is the same as that used in [5], the main difference is that we introduce special resets called *fractional resets* which reset only the integral part of clock $x$ while its fractional part is retained. For instance, if the value of $x$ was 1.3, then the operation $[x] := 0$ makes the value of $x$ to be 0.3. Given a one clock, dwell-time PTG $\mathcal{G} = (L_1, L_2 \cup L', \{x\}, X, \eta, T, f_{goals})$ with $M$ being the maximum value that can be assumed by clock $x$, we define a dwell-time PTG with fractional resets (FRPTG) $\mathcal{G}_{\mathcal{F}}$. In $\mathcal{G}_{\mathcal{F}}$, we have $M+1$ copies of the locations in $L_1 \cup L_2$ as well as the locations in $L'$ with dwell time $[0,\delta]$, $[0,0]$. These $M+1$ copies of $L'$ have the same dwell-time restrictions in $\mathcal{G}_{\mathcal{F}}$. The copies are indexed by $i, 0 \leq i \leq M$, capturing the integral part of clock $x$ in $\mathcal{G}$. Finally, we have in $\mathcal{G}$, the locations of $L'$ with dwell-time restriction $[\delta, 2\delta]$. For each such location $(A,e)^+$, we have in $\mathcal{G}_{\mathcal{F}}$, the locations $(A,e)_i^+$ and $(A,e)_{i+1}^0$ for $0 \leq i \leq M$. The dwell-time restriction for $(A,e)_i^+$ is same as $(A,e)^+$, while locations $(A,e)_{i+1}^0$ are urgent. The prices of locations are carried over as they are in the various copies.

The transitions in $\mathcal{G}_{\mathcal{F}}$ consists of the following: (1) $l_i \xrightarrow{(g-i)\cap 0\leq x<1} m_i$[1] if $l \xrightarrow{g} m \in X$; (2) $l_i \xrightarrow{(g-i)\cap 0\leq x<1;\{x\}} m_0$ if $l \xrightarrow{g;\{x\}} m \in X$; (3) $l_i \xrightarrow{x=1,\{x\}} l_{i+1}$, for $l \in L_1 \cup L_2$, and $(A,e)_i^+ \xrightarrow{x\geq 1,[x]:=0} (A,e)_{i+1}^0$ for $i < M$. Consider for example, the constraint $g'$ between $A$ and $(A,e)$ as $x = (b+1) - \delta$ in $\mathcal{G}$. Then the value of $x$ is $b + (1-\delta)$ for $b < M$ when $(A,e)^+$ is entered in $\mathcal{G}$. The location $(A,e)^+$ with $\nu(x) = b + (1-\delta)$ is represented in $\mathcal{G}_{\mathcal{F}}$ as $(A,e)_b^+$ with $\nu(x) = 1 - \delta$. If player 2 spends $[\delta, 2\delta]$ time at $(A,e)^+$ in $\mathcal{G}$, then $\nu(x) \in [b+1, b+1+\delta]$. If there are no resets to goto $B$, then $\nu(x) \in [b+1, (b+1)+\delta]$ at $B$. Correspondingly in $\mathcal{G}_{\mathcal{F}}$, $\nu(x) \in [1, 1+\delta]$ at $(A,e)_b^+$. By construction, $B_b$ is not reachable, since we check $0 \leq x < 1$ on the transition to $B_b$. The fractional reset is employed to obtain $x = \delta$ while moving to $(A,e)_{b+1}^0$. This ensures that $x = \delta$ on reaching $B_{b+1}$, thereby

[1] $g - i$ represents the constraint obtained by shifting the constraint by $-i$

preserving the perturbation, and keeping $x < 1$. A normal reset would have destroyed the value obtained by perturbation. The mapping $f$ between states of $\mathcal{G}$ and $\mathcal{G}_{\mathcal{F}}$ is as follows: $f(l, x) = (l_b, x - b)$, $b < M$, and $x \in [b, b + 1]$, $l \in L_1 \cup L_2$, $f((A, e), x) = ((A, e)_b, x - b)$, $b < M$, and $x \in [b, b + 1]$, $f((A, e)^-, x) = ((A, e)_b^-, x - b)$, $b < M$, and $x \in [b, b + 1]$. Finally, $f((A, e)^+, x) = ((A, e)_b^+, x - b)$, $b < M$, and $x \in [b, b + 1] \cup [b + 1, b + 2]$. Note that in the last case, the value of $x - b$ can exceed 1 but is less than or equal to $1 + \delta$.

▶ **Lemma 5.2.** *For every state $(l, \nu)$ in $\mathcal{G}$, $\mathsf{OptCost}_{\mathcal{G}}(l, \nu)$ in $\mathcal{G}$ is the same as $\mathsf{OptCost}_{\mathcal{G}_{\mathcal{F}}}(f(l, \nu))$ in $\mathcal{G}_{\mathcal{F}}$. For every $\epsilon > 0$, $N \in \mathbb{N}$, an $(\epsilon, N)$-acceptable strategy in $\mathcal{G}$ can be computed from an $(\epsilon, N)$-acceptable strategy in $\mathcal{G}_{\mathcal{F}}$ and vice versa.*

## 5.3 Transformation 3: Dwell-time FRPTG $\mathcal{G}_{\mathcal{F}}$ to resetfree FRPTG $\mathcal{G}_{\mathcal{F}}{}'$

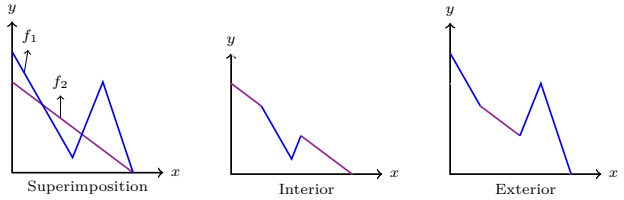
We now apply the final transformation to the FRPTG $\mathcal{G}_{\mathcal{F}}$ and construct a reset-free version of the FRPTG denoted $\mathcal{G}_{\mathcal{F}}{}'$. Assume that there are a total of $n$ resets (including fractional resets) in the FRPTG. $\mathcal{G}_{\mathcal{F}}{}'$ consists of $n + 1$ copies of the FRPTG : $\mathcal{G}_{\mathcal{F}0}, \mathcal{G}_{\mathcal{F}1}, \ldots, \mathcal{G}_{\mathcal{F}n}$. Given the locations $L$ of the FRPTG, the locations of $\mathcal{G}_{\mathcal{F}i}$ are $L^i$, $0 \leq i \leq n$. $\mathcal{G}_{\mathcal{F}0}$ starts with $l^0$, where $l$ is the initial location of the FRPTG and continues until a resetting transition happens. At the first resetting transition, $\mathcal{G}_{\mathcal{F}0}$ makes a transition to $\mathcal{G}_{\mathcal{F}1}$. The $n$th copy is directed to a sink target location $S$ with cost function $cf : \mathbb{R}_{\geq 0} \to \{+\infty\}$ on the $(n + 1)$th reset. Note that each $\mathcal{G}_{\mathcal{F}i}$ is reset-free. One crucial property of each $\mathcal{G}_{\mathcal{F}i}$ is that on entering with some value of $x$ in $[0, \delta]$, the value of $x$ only increases as the transitions go along in $\mathcal{G}_{\mathcal{F}i}$; moreover, $x \leq 1 + \delta$ in each $\mathcal{G}_{\mathcal{F}i}$ by construction. The formal details and proof of Lemma 5.3 can be found in [10]. Using the cost function of $S$ and those of the targets, we compute the optimal cost functions for all the locations of the deepest component $\mathcal{G}_{\mathcal{F}n}$. The cost functions of the locations of $\mathcal{G}_{\mathcal{F}i}$ are used to compute that of $\mathcal{G}_{\mathcal{F}i-1}$, and so on until the cost function of $l^0$, the starting location of $\mathcal{G}_{\mathcal{F}0}$ is computed. An example can be seen in [10].

▶ **Lemma 5.3.** *For every state $(l, \nu)$ in $\mathcal{G}_{\mathcal{F}}$, $\mathsf{OptCost}_{\mathcal{G}_{\mathcal{F}}}(l, \nu) = \mathsf{OptCost}_{\mathcal{G}_{\mathcal{F}}{}'}(l^0, \nu)$, where $\mathcal{G}_{\mathcal{F}}{}'$ is the resetfree* FRPTG*. For every $\epsilon > 0, N \in \mathbb{N}$, given an $(\epsilon, N)$-acceptable strategy $\sigma'$ in $\mathcal{G}_{\mathcal{F}}{}'$, we can compute a $(2\epsilon, N)$-acceptable strategy $\sigma$ in $\mathcal{G}_{\mathcal{F}}$ and vice versa.*

## 5.4 Solving the Resetfree FRPTG

Before we sketch the details, let us introduce some key notations. Observe that after our simplifying transformations, the cost functions $cf$ are piecewise-affine continuous functions that assign a value to every valuation $x \in [0, 1 + \delta]$ (construction of FRPTG ensures $x \leq 1+\delta$ always). The *interior* of two cost functions $f_1$ and $f_2$ is a cost function $f_3 : [0, 1 + \delta] \to \mathbb{R}$ defined by $f_3(x) = \min(f_1(x), f_2(x))$. Similarly, the *exterior* of $f_1$ and $f_2$ is a cost function $f_4 : [0, 1 + \delta] \to \mathbb{R}$ defined as $f_4(x) = \max(f_1(x), f_2(x))$. Clearly, $f_3$ and $f_4$ are also piecewise-affine continuous. The interior and exterior can be easily computed by *superimposing* $f_1$ and $f_2$ as shown graphically in the example by computing lower envelope and upper envelope respectively.

We now work on the reset-free components $\mathcal{G}_{\mathcal{F}i}$, and give an algorithm to compute $OptCost_{\mathcal{G}_{\mathcal{F}i}}(l, \nu)$ for every state $(l, \nu)$ of $\mathcal{G}_{\mathcal{F}i}$, $\nu(x) \in [0, 1 + \delta]$. We also show the existence of an $N$ such that for any $\epsilon > 0$, and every $l \in L^i, \nu(x) \in [0, 1 + \delta]$, an $(\epsilon, N)$-acceptable strategy can be computed. Consider the location of $\mathcal{G}_{\mathcal{F}i}$ that has the smallest price and call it $l_{min}$. If this is a player 1 location, then intuitively, player 1 would want to spend as much time as possible here, and if this is a player 2 location, then player 2 would want to spend as less time as possible here. By our assumption, all the cycles in $\mathcal{G}_{\mathcal{F}i}$ are non-negative, and

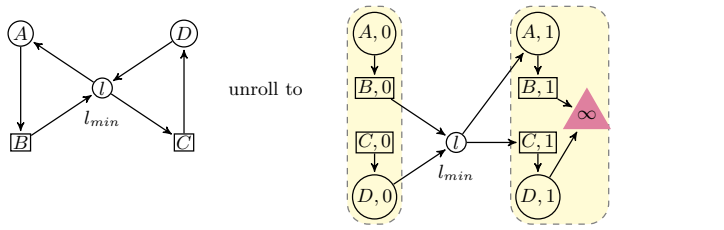



Example Illustrating SuperImposition, Interior and Exterior

hence if $l_{min}$ is part of a cycle, revisiting it will only increase the total cost if at all. Player 1 thus would like to spend all the time he wants to during the first visit itself. We now prove that this is indeed the case. We consider two cases separately.

### 5.4.1 $l_{min}$ is a Player 1 location

We split $\mathcal{G}_{\mathcal{F}i}$ such that $l_{min}$ is visited only once. We transform $\mathcal{G}_{\mathcal{F}i}$ into $\mathcal{G}_{\mathcal{F}}{}''$ which has two copies of all locations except $l_{min}$ such that corresponding to every location $l \neq l_{min}$, we have the copies $(l,0)$ and $(l,1)$. A special target location $S$ is added with cost function assigning $+\infty$ to all clock valuations.




Duplicate $L - l_{min}$

Given the transitions $X$ of $\mathcal{G}_{\mathcal{F}i}$, the FRPTG $\mathcal{G}_{\mathcal{F}}{}''$ has the following transitions.

- if $l \xrightarrow{g} l' \in X$ and $l, l' \neq l_{min}$ then $(l,0) \xrightarrow{g} (l',0)$ and $(l,1) \xrightarrow{g} (l',1)$
- if $l \xrightarrow{g} l' \in X$ and $l' = l_{min}$ then $(l,0) \xrightarrow{g} l_{min}$ and $(l,1) \xrightarrow{g} S$,
- if $l_{min} \xrightarrow{g} l$, then $l_{min} \xrightarrow{g} (l,1)$

▶ **Lemma 5.4.** *For every state $(l,\nu)$ if $\nu \in [0, 1+\delta]$ and $l \neq l_{min}$, we have that $\mathit{OptCost}_{\mathcal{G}_{\mathcal{F}i}}(l,\nu) = \mathit{OptCost}_{\mathcal{G}_{\mathcal{F}}{}''}((l,0),\nu)$ and $\mathit{OptCost}_{\mathcal{G}_{\mathcal{F}i}}(l_{min},\nu) = \mathit{OptCost}_{\mathcal{G}_{\mathcal{F}}{}''}(l_{min},\nu)$.*

We give an intuition for Lemma 5.4. Locations $(l,0)$ have all the transitions available to location $l$ in $\mathcal{G}_{\mathcal{F}i}$. Also, any play in $\mathcal{G}_{\mathcal{F}}{}''$ which is compatible with a winning strategy of player 1 in $\mathcal{G}_{\mathcal{F}i}$ contains only one of the locations $(l,0),(l,1)$ by construction of $\mathcal{G}_{\mathcal{F}}{}''$. The outcomes from $(l,0)$ are more favourable than $(l,1)$ for $l$ as a player 1 location. Based on these intuitions, we conclude that $\mathsf{OptCost}_{\mathcal{G}_{\mathcal{F}i}}(l,\nu)$ is same as that for $((l,0),\nu)$. This observation also leads to the $\epsilon-$optimal strategy being the same as that for $(l,0)$. Given a strategy $\sigma'$ in $\mathcal{G}_{\mathcal{F}}{}''$, we construct $\sigma$ in $\mathcal{G}_{\mathcal{F}i}$ as $\sigma(l,\nu) = \sigma'((l,0),\nu)$. Further, any strategy that revisits $l_{min}$ in $\mathcal{G}_{\mathcal{F}i}$ cannot be winning for player 1, since all cycles are non-negative; we end up at $S$ with cost $\infty$ in $\mathcal{G}_{\mathcal{F}}{}''$. However, all strategies that do not revisit $l_{min}$ in $\mathcal{G}_{\mathcal{F}i}$ are preserved in $\mathcal{G}_{\mathcal{F}}{}''$, and hence $\mathsf{OptCost}_{\mathcal{G}_{\mathcal{F}i}}(l_{min},\nu) = \mathsf{OptCost}_{\mathcal{G}_{\mathcal{F}}{}''}(l_{min},\nu)$.

We iteratively solve the part of $\mathcal{G}_{\mathcal{F}}{}''$ with locations indexed 1 (i.e; $(l,1)$) in the same fashion (picking minimal price locations) each time obtaining a smaller PTG. Computing the cost function of the minimal price location of the last such PTG, and propagating this backward, we compute the cost function of $l_{min}$. We then use the cost function of $l_{min}$ to solve the part of $\mathcal{G}_{\mathcal{F}}{}''$ with locations indexed 0 (i.e; $(l,0)$).

**Algorithm 1:** Optimal Cost Algorithm when $l_{min}$ is a Player 1 location.

Let $l_1, \ldots, l_n$ be the successors of $l_{min}$ with optcost functions $f_1, f_2 \cdots f_n$.;
**STEP 1 : Superimpose :** Superimpose all the optcost functions $f_1, f_2 \cdots f_n$.;
**STEP 2 : Interior :** Take the interior of the superimposition; call it $f$.;
Let $f$ be composed of line segments $g_1, g_2 \cdots g_m$ such that $g_i \in \{f_1, \ldots, f_n\}$, for all $i$.
$\forall\ k$, let the domain of $g_k$ be $[u_k, v_k]$. Set $i = m$.;
**STEP 3 : Selective Replacement :** **while** $i \geq 1$ **do**
  **if** *slope of* $g_i \leq -\eta(l_{min})$ **then**
    replace $g_i$ with line $h_i$ with slope $-\eta(l_{min})$ and passing through $(v_i, g_i(v_i))$;
    Let $h_i$ intersect $g_j$ (largest $j < i$) at some point $x = v_j''$, $v_j'' \in [u_j, v_j]$;
    Update domain of $g_j$ from $[u_j, v_j]$ to $[u_j, v_j'']$;
    **if** $j < i - 1$ **then**
      Remove functions $g_{j+1}$ to $g_{i-1}$ from $f$
    Set $i = j$;
  **else**
    $i = i - 1$;
**STEP 4 : Refresh Interior :** Take the interior after STEP 3 and call it $f'$.;
**if** $l'' \to l_{min}$ **then**
  update the optcost function of $l''$

**Computing the Optcost function of $l_{min}$:** Algorithm 1 computes the optcost function for a player 1 location $l_{min}$, assuming all the constraints on outgoing transitions from $l_{min}$ are the same, namely $x \in [0, 1]$. We discuss adapting the algorithm to work for transitions with different constraints in [10]. A few words on the notation used: if a location $l$ has price $\eta(l)$, then slope associated with $l$ is $-\eta(l)$ (see STEP 3 in Algorithm 1).

Let $l_1, \ldots, l_n$ be the successors of $l_{min}$, with cost functions $f_1, \ldots, f_n$. Each of these cost functions are piecewise affine continuous over the domain $[0, 1]$. The first thing to do is to superimpose $f_1, \ldots, f_n$, and obtain the cost function $f$ corresponding to the interior of $f_1, \ldots, f_n$ ($l_{min}$ is a player 1 location and would like to obtain the minimal cost, hence the interior). The line segments comprising $f$ come from the various $f_i$. Let $dom(f) = [0, 1]$ be composed of $0 = u_{i_1} \leq v_{i_1} = u_{i_2} \leq \ldots u_{i_m} \leq v_{i_m} = 1$ : that is, $f(x) = f_{i_j}(x)$, $dom(f_{i_j}) = [u_{i_j}, v_{i_j}]$, for $i_j \in \{1, 2, \ldots, n\}$ and $1 \leq j \leq m$. Let us denote $f_{i_j}$ by $g_j$, for $1 \leq j \leq m$. Then, $f$ is composed of $g_1, g_2, \ldots, g_m$, and $dom(f)$ is composed of $dom(g_1), \ldots, dom(g_m)$ from left to right. Let $dom(g_i) = [u_i, v_i]$. Step 2 of the algorithm achieves this.

For a given valuation $\nu(x)$, if $l_{min}$ is an urgent location, then player 1 would go to a location $l_k$ if the interior $f$ is such that $f(\nu(x)) = g_k(\nu(x))$(the least cost is given by $g_k$, obtained from the outside cost function of $l_k$). If $l_{min}$ is not an urgent location, then player 1 would prefer delaying $t$ units at $l_{min}$ so that $\nu(x) + t \in [u_i, v_i]$ rather than goto some location $l_i$ if $g_i(\nu(x)) > \eta(l_{min})(v_i - \nu(x))$. Again, $g_i$ is a part of the ouside cost function of $l_i$, and player 1 prefers delaying time at $l_{min}$ rather than goto $l_i$ since that minimizes the cost. In this case, the cost function $f$ is refined by replacing the line segment $g_i$ over $[u_i, v_i]$ by another line segment $h_i$ passing through $(v_i, g_i(v_i))$, and having a slope $-\eta(l_{min})$. Step 3 of the algorithm does this.

Recall that by our transformation 2, the value of clock $x$ in any player 1 location is $\leq 1 - \delta$. The value of $x$ is in $[1 - \delta, 1 + \delta]$ only at a player 2 location $((A, e)_b)^+$ in the FRPTG, section 5.2). Hence, the domain of cost functions for player 1 locations is actually $[0, 1 - \delta]$,

and not $[0, 1+\delta]$. Let the domain of $g_m$ be $[u_m, 1]$. Then we can split $g_m$ into two functions $g_m^1, g_m^2$ with domains $[u_m, 1-\delta]$ and $[1-\delta, 1]$. Now, we ensure that no time is spent in the player 1 location $l_{min}$ over $dom(g_m^2)$, by not applying step 3 of the algorithm for $g_m^2$. This way, selective replacement of the cost functions $g_i$ occur only in the domain $[0, 1-\delta]$, and we remain faithful to transformation 2, and the semantics of RPTGs.

**Computing Almost Optimal Strategies:** The strategy corresponding to this computed optcost is derived as follows. $f'$ is the optcost of location $l_{min}$ computed in Step 4 of the algorithm. $f'$ is composed of two kinds of functions (a) the functions $g_i$ computed in step 2 as a result of the interior of superimposition and (b) functions $h_i$ which replaced some functions $g_j$ from $f$, corresponding to delay at $l_{min}$. For functions $h_j$ of $f'$ with domain $[u_j, v_j]$, we prescribe the strategy to delay at $l_{min}$ till $x = v_j$ when entered with clock $x \in [u_j, v_j]$. For functions $g_i$, that come from $f$ at Step 2, where $g_i$ is part of some optcost function $f_k$, ($f_k$ is the optcost function of one of the successors $l_k$ of $l_{min}$), the strategy dictates moving immediately to $l_k$ when entered with clock $x \in [u_i, v_i]$.

**Termination:** Finally, we prove the existence of a number $N$, the number of affine segments that appear in the cost functions of all locations. Start with the resetfree FRPTG with $m$ locations having $p$ segments in the outside cost functions. Let $\alpha(m, p)$ denote the total number of affine segments appearing in cost functions across all locations. The transformation of resetfree components $\mathcal{G}_{\mathcal{F}}$ into $\mathcal{G}_{\mathcal{F}}{}''$ gives rise to two smaller resetfree FRPTGs with $m-1$ locations each, after separating out $l_{min}$. The resetfree FRPTG $(\mathcal{G}_{\mathcal{F}}, 1)$ with $m-1$ locations indexed with 1 of the form $(l, 1)$ are solved first, these cost functions are added as outside cost functions to solve $l_{min}$, and finally, the cost function of $l_{min}$ is added as an outside cost function to solve the resetfree FRPTG $(\mathcal{G}_{\mathcal{F}}, 0)$ with $m-1$ locations indexed with 0 of the form $(l, 0)$. Taking into account the new sink target location added, we have $\leq p+1$ segments in outside cost functions in $(\mathcal{G}_{\mathcal{F}}, 1)$. This gives atmost $\beta = \alpha(m-1, p+1)$ segments in solving $(\mathcal{G}_{\mathcal{F}}, 1)$, and $\alpha(1, p+\beta) = \gamma$ segments to solve $l_{min}$, and finally $\alpha(m-1, p+\gamma)$ segments to solve $(\mathcal{G}_{\mathcal{F}}, 0)$. Solving this, one can easily check that $\alpha(m, p)$ is atmost triply exponential in the number of locations $m$ of the resetfree component $\mathcal{G}_{\mathcal{F}}$. Obtaining a bound of the number of affine segments, it is easy to see that Algorithm 1 terminates; the time taken to compute almost optimal strategies and optcost functions is triply exponential.

We illustrate the computation of Optcost of a Player 1 location in Figure 2. The proof of Lemma 5.5 is given in [10], while Lemma 5.6 follows from Lemma 5.5 and Step 4 of Algorithm 1.

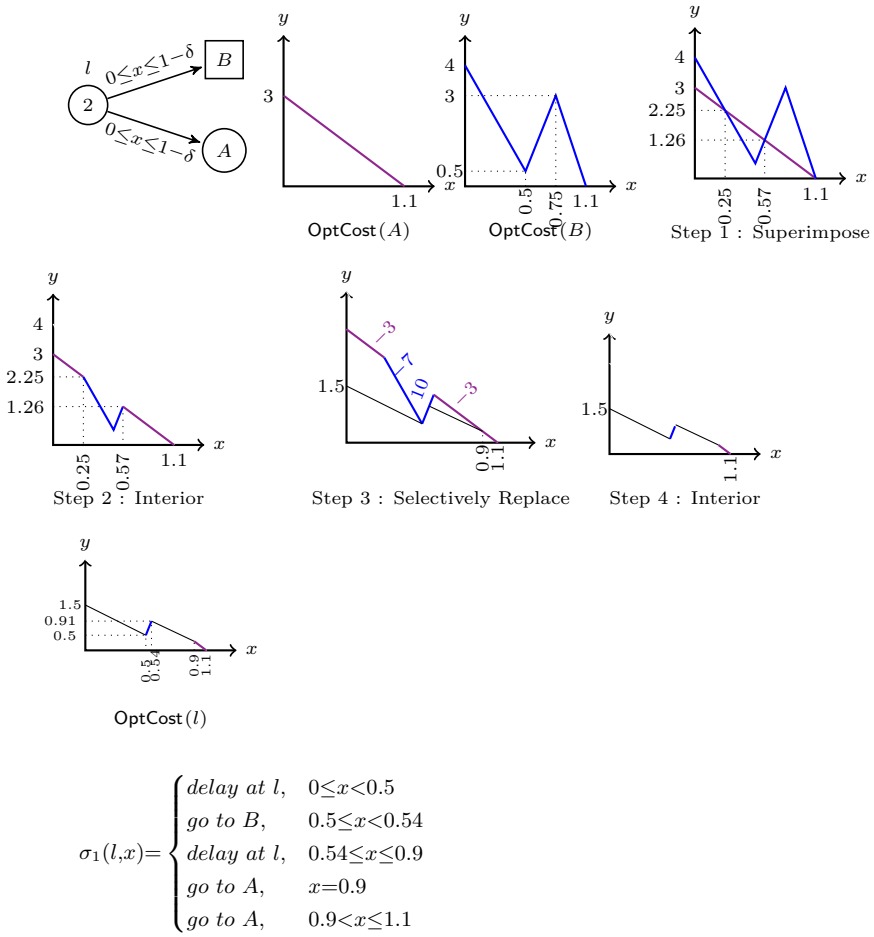
▶ **Lemma 5.5.** *In Algorithm 1, if a function $g_i$ (in $f$ of Step 2) has domain $[u_i, v_i]$ and slope $\leq -\eta(l)$ then $\mathsf{OptCost}(l, \nu) = (v_i - \nu) * \eta(l) + g(v_i)$.*

▶ **Lemma 5.6.** *The function $f'$ in Algorithm 1 computes the optcost at any location $l$. That is, $\forall x \in [0, 1]$, $\mathsf{OptCost}_{\mathcal{G}}(l, x) = f'(x)$.*

Note that the strategy under construction is a player 1 strategy, and player 1 has no control over the interval $[1, 1+\delta]$. $x \in [1, 1+\delta]$ after a positive perturbation, and is under player 2's control. Thus, at a player 1 location, proving for $x \in [0, 1]$ suffices.

### 5.4.2 $l_{min}$ is a Player 2 location

If $l_{min}$ is a player 2 location in the reset-free component $\mathcal{G}_{\mathcal{F}_i}$, then intuitively, player 2 would want to spend as little time as possible there. Keeping this in mind, we first run




$$\sigma_1(l,x)=\begin{cases} \textit{delay at } l, & 0\leq x<0.5 \\ \textit{go to } B, & 0.5\leq x<0.54 \\ \textit{delay at } l, & 0.54\leq x\leq 0.9 \\ \textit{go to } A, & x=0.9 \\ \textit{go to } A, & 0.9<x\leq 1.1 \end{cases}$$

**Figure 2** Optcost Computation for a Player 1 location ($\delta = 0.1$): we can keep the guards as $0 \leq x \leq 1$ and not apply Step 3 for $x \in [1-\delta, 1]$.

steps 1, 2 of Algorithm 1 by taking the exterior of $f_1, \ldots, f_n$ instead of the interior(player 2 would maximise the cost). There is no time elapse at $l_{min}$ on running steps 1,2 of the algorithm. Let $f$ be the computed exterior using steps 1,2. If $f$ comprises of functions $g_i$ having a greater slope than $-\eta(l)$, then Finally, while doing Step 4, we take the exterior of the replaced functions $h_i$ and old functions $g_i$. Recall that our transformations resulted in 3 kinds of player 2 locations : urgent, those with dwell-time restriction $[0, \delta]$ and finally those with $[\delta, 2\delta]$. The 3 cases are discussed in detail in [10].

## 6 Conclusion and Future Work

In this paper we studied excess robust semantics and provided the first decidability result for excess semantics and improved the known undecidability result with 10 clocks to 5 clocks. To the best of our knowledge, the other known decidability result for robust timed games is under the conservative semantics for a fixed $\delta$, [9]. As a consequence of our decidability result, the reachability problem for 1 clock PTG with arbitrary prices is shown to be decidable too under the assumption that the PTG does not have any negative cost cycle. The decidability we show

is for a fixed perturbation bound $\delta > 0$. We use $\delta$ in the constraints of the dwell-time PTG after the first transformation for ease of understanding the robust semantics. Implementing this in step 3 of Algorithm 1 and ensuring no time elapse in the interval $[1 - \delta, 1]$ takes no extra effort while $l_{min}$ is a player 1 location. In that sense, we could have avoided explicit use of $\delta$ in the constraints in our simplifying transformations, and taken the appropriate steps in the algorithm itself. The existence of limit-strategy with $\delta \to 0$ seems rather hard. Our construction would not directly extend to limit-strategy problem as it is heavily dependant on the fixed $\delta$.

## References


1 R. Alur, M. Bernadsky, and P. Madhusudan. Optimal reachability for weighted timed games. In J. Díaz, J. Karhumäki, A. Lepistö, and D Sannella, editors, *Proc. ICALP'04*, volume 3142 of *LNCS*, pages 122–133. Springer, 2004.

2 G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. W. Vaandrager. Minimum-cost reachability for priced timed automata. In M. D. Di Benedetto and A. L. Sangiovanni-Vincentelli, editors, *Proc. HSCC'01*, volume 2034 of *LNCS*, pages 147–161, Heidelberg, 2001. Springer.

3 P. Bouyer, T. Brihaye, and N. Markey. Improved undecidability results on weighted timed automata. *Information Processing Letters*, 98:188–194, 2006.

4 P. Bouyer, F. Cassez, E. Fleury, and K. G. Larsen. Optimal strategies in priced timed game automata. In K. Lodaya and M. Mahajan, editors, *FSTTCS'04*, volume 3328 of *LNCS*, pages 148–160. Springer, 2004.

5 Patricia Bouyer, Kim Guldstrand Larsen, Nicolas Markey, and Jacob Illum Rasmussen. Almost optimal strategies in one clock priced timed games. In *FSTTCS 2006: Foundations of Software Technology and Theoretical Computer Science, 26th International Conference, Kolkata, India, December 13-15, 2006, Proceedings*, pages 345–356, 2006.

6 Patricia Bouyer, Nicolas Markey, and Ocan Sankur. Robust reachability in timed automata: A game-based approach. In *Automata, Languages, and Programming*, volume 7392 of *Lecture Notes in Computer Science*, pages 128–140. Springer, 2012.

7 Patricia Bouyer, Nicolas Markey, and Ocan Sankur. Robust weighted timed automata and games. In Víctor Braberman and Laurent Fribourg, editors, *Proceedings of the 11th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'13)*, volume 8053 of *Lecture Notes in Computer Science*, pages 31–46, Buenos Aires, Argentina, August 2013. Springer.

8 T. Brihaye, V. Bruyère, and J. Raskin. On optimal timed strategies. In P. Pettersson and W. Yi, editors, *Proc. FORMATS'05*, volume 3829 of *LNCS*, pages 49–64. Springer, 2005.

9 Krishnendu Chatterjee, Thomas A. Henzinger, and Vinayak S. Prabhu. Timed parity games: Complexity and robustness. *Logical Methods in Computer Science*, 7(4), 2011.

10 Shibashis Guha, Shankara Narayanan Krishna, Lakshmi Manasa, and Ashutosh Trivedi. Revisiting robustness in priced timed games. *CoRR*, abs/1507.05787, 2015.

11 T. D. Hansen, R. Ibsen-Jensen, and P. B. Miltersen. A faster algorithm for solving one-clock priced timed games. In PedroR. D'Argenio and Hernán Melgratti, editors, *CONCUR 2013 – Concurrency Theory*, volume 8052 of *Lecture Notes in Computer Science*, pages 531–545. Springer, 2013.

12 N. Markey. Robustness in real-time systems. In *Industrial Embedded Systems (SIES), 2011 6th IEEE International Symposium on*, pages 28–34, June 2011.

13 Michal Rutkowski. Two-player reachability-price games on single-clock timed automata. In *QAPL*, pages 31–46, 2011.

14 Ocan Sankur, Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust controller synthesis in timed automata. In *CONCUR 2013 – Concurrency Theory*, volume 8052 of *Lecture Notes in Computer Science*, pages 546–560. Springer, 2013.