Optimal Strategies in Priced Timed Game Automata

Patricia Bouyer[1⋆], Franck Cassez[2⋆], Emmanuel Fleury[3], Kim G. Larsen[3]

[1] LSV, UMR 8643, CNRS & ENS de Cachan, France
Email: bouyer@lsv.ens-cachan.fr

[2] IRCCyN, UMR 6597, CNRS, France
Email: cassez@irccyn.ec-nantes.fr

[3] Computer Science Department, BRICS[⋆⋆], Aalborg University, Denmark
Email: {fleury,kgl}@cs.auc.dk

Abstract. Priced timed (game) automata extend timed (game) automata with costs on both locations and transitions. In this paper we focus on reachability priced timed game automata and prove that the optimal cost for winning such a game is computable under conditions concerning the non-zenoness of cost. Under stronger conditions (strictness of constraints) we prove that in case an optimal strategy exists, we can compute a state-based winning optimal strategy.

1 Introduction

Optimal Scheduling in Timed Systems. In recent years the application of model-checking techniques to scheduling problems has become an established line of research. Static scheduling problems with timing constraints may often be formulated as reachability problems on timed automata, viz. as the possibility of reaching a given goal state. Real-time model checking tools such as KRONOS and UPPAAL have been applied on a number of industrial and benchmark scheduling problems [13, 15].

Often the scheduling strategy needs to take into account uncertainty with respect to the behavior of an environmental context. In such situations the scheduling problem becomes a dynamic (timed) game between the controller and the environment, where the objective for the controller is to find a *dynamic* strategy that will guarantee the game to end in a goal state [5, 11, 17].

Optimality of schedules may be obtained within the framework of timed automata by associating with each run a performance measure. Thus it is possible to compare runs and search for the optimal run from an initial configuration to a final (goal) target. The most obvious performance measure for timed automata is clearly that of time itself. Time-optimality for timed automata was first considered in [10] and proved computable in [18]. The related problem of synthesizing time-optimal winning strategies for timed game automata was shown computable in [4].

⋆ Work partially supported by ACI Cortos, a program of the French government. Visits to Aalborg supported by CISS, Aalborg University, Denmark
⋆⋆ Basic Research in Computer Science (www.brics.dk).

More recently, the ability to consider more general performance measures has been given. Priced extensions of timed automata have been introduced where a cost c is associated with each location ℓ giving the cost of a unit of time spent in ℓ. In [2] cost-bound reachability has been shown decidable. [6] and [3] independently solve the cost-optimal reachability problem for priced timed automata. Efficient incorporation in UPPAAL is provided by use of so-called priced zones as a main data structure [16]. More recently in [9], the problem of computing optimal *infinite* schedules (in terms of minimal limit-ratios) is solved for the model of priced timed automata.

The Optimal Control Problem for Timed Games. In this paper we combine the notions of game and price and solve the problem of cost-optimal winning strategies for priced timed game automata. The problem we consider is: "Given a priced timed game automaton A, a goal location Goal, what is the optimal cost we can achieve to reach Goal in A?". We refer to this problem as the Optimal Control Problem (OCP). Consider the example of a priced timed game automaton given in Fig. 1. Here the cost-rates (cost per time unit) in locations ℓ_0, ℓ_2 and ℓ_3 are 5, 10 and 1 respectively. In ℓ_1 the environment may choose to move to either ℓ_2 or ℓ_3 (dashed arrows are uncontrollable). However, due to the invariant $y = 0$ this choice must be made instantaneous. Obviously, once ℓ_2 or ℓ_3 has been reached the optimal strategy for the controller is to move to Goal immediately (however there is a discrete cost (resp. 1 and 7) on each discrete transition). The crucial (and only remaining) question is how long the controller should wait in ℓ_0 before taking the transition to ℓ_1. Obviously, in order for the controller to win this duration must be no more than two time units. However, what is the optimal choice for the duration in the sense that the overall cost of reaching Goal is minimal? Denote by t the chosen delay in ℓ_0. Then $5t + 10(2 - t) + 1$ is the minimal cost through ℓ_2 and $5t + (2 - t) + 7$ is the minimal cost through ℓ_3. As the environment chooses between these two transitions the best choice for the controller is to delay $t \leq 2$ such that $\max(21 - 5t, 9 + 4t)$ is minimum, which is $t = \frac{4}{3}$ giving a minimal cost of $14\frac{1}{3}$.

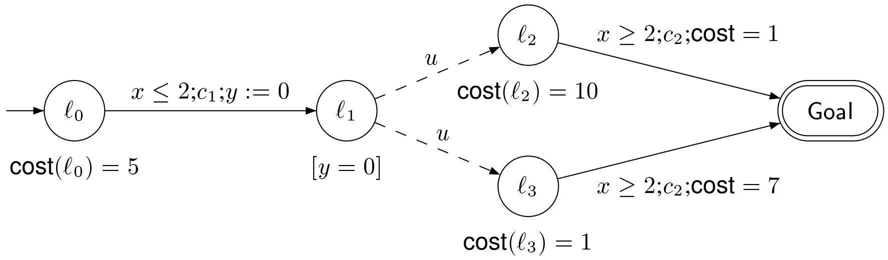

Fig. 1. A Reachability Priced Time Game Automaton $\mathcal{A}$

Related Work. Acyclic priced (or weighted) timed games have been studied in [14] and the more general case of non-acyclic games have been recently considered in [1]. In [1], the problem they consider is "compute the optimal cost within k steps": we refer to this bounded problem as the k-OCP. This is a weaker version than the one we consider (OCP) and roughly corresponds to unfolding the game k times and to reducing the problem to solving an acyclic game. In [1], the authors focus on the complexity of the k-OCP rather than on the decidability of the OCP and give a clever (exponential) bound on the number of regions that appear after unfolding the game k times. In the conclusion the authors also indicate that under some non-Zenoness assumption (similar to the one we use in theorem 6) the number of iterations required to compute the optimal cost (OCP) is finite and thus that, under this assumption, any game can be reduced to an "optimal game in finite number of steps". However both our work and [1] fail in solving the general OCP without any (non-Zenoness) assumption.

In this work (following our research report [7]) that was done simultaneously and independently from [1], we don't provide any complexity bound for (k-)OCP, but rather focus on the synthesis of winning strategies and their structural properties. The method we use is radically different from the one proposed in [14, 1] and our main contributions (which extend previous works) are then the following:

- in both above-mentioned papers, the definition of the optimal cost is based on a recursive definition of a function (like the O function given in definition 11, page 8) that can be very complex (e.g. in [1]); we propose a new run-based definition (definition 9) of the optimal cost that is more natural and enables us to obtain new results. For instance the definition of the optimal cost in [14, 1] is based on an infimum-supremum computation: if the optimal cost is c the algorithm does not give any hint whether c is actually realized (there is a strategy of cost c) or if c is the limit of the optimal cost (there is a family of strategies of cost $c+\varepsilon$ for all $\varepsilon > 0$). In our settings, we can compute the optimal cost and answer the question whether an optimal strategy exists or not (corollaries 1 and 2). Moreover we provide a proof that non-Zenoness implies termination of our algorithm (theorem 6).
- in addition to the previous new results on optimal cost computation that extend the ones in [14, 1] we also tackle the problem of strategy synthesis. In particular we study the properties of the strategies (memoryless, cost-dependence) needed to achieve the optimal cost which is a natural question that arises in game theory. For example, in [1] setting, it could be the case that in two instances of the unfolding of the game, the values of a strategy for a given state are different. In this paper we prove that if an optimal strategy exists then one can effectively construct an optimal strategy which only depends on the current state and on the accumulated cost since the beginning of the play. We also prove that under some assumptions, if an optimal strategy exists then a state-based cost-independent strategy exists and can be effectively computed (theorem 7).
- finally the algorithms we obtain can be implemented [8] in HyTech.

Proofs are omitted but can be found in [7].

2 Reachability Timed Games (RTG)

In this paper we focus on *reachability games*, where the control objective is to enforce that the system eventually evolves into a particular state. It is classical in the literature to define *reachability timed games (RTG)* [5, 11, 17] to model control problems. In this section we recall some known general results about RTG.

Timed Transition Systems and Games.

Definition 1 (Timed Transition Systems (TTS)). *A* timed transition system *is a tuple $S = (Q, Q_0, \mathsf{Act}, \longrightarrow)$ where Q is a set of states, $Q_0 \subseteq Q$ is the set of initial states,* Act *is a finite set of actions, disjoint from $\mathbb{R}_{\geq 0}$, $\longrightarrow \subseteq Q \times \Sigma \times Q$ is a set of edges. We let $\Sigma = \mathsf{Act} \cup \mathbb{R}_{\geq 0}$. If $(q, e, q') \in \longrightarrow$, we also write $q \xrightarrow{e} q'$.*

We make the following common assumptions about TTSs:

- 0-DELAY: $q \xrightarrow{0} q'$ if and only if $q = q'$,
- ADDITIVITY: if $q \xrightarrow{d} q'$ and $q' \xrightarrow{d'} q''$ with $d,\ d' \in \mathbb{R}_{\geq 0}$, then $q \xrightarrow{d+d'} q''$,
- CONTINUITY: if $q \xrightarrow{d} q'$, then for every d' and d'' in $\mathbb{R}_{\geq 0}$ such that $d = d' + d''$, there exists q'' such that $q \xrightarrow{d'} q'' \xrightarrow{d''} q'$,
- DETERMINISM: if $q \xrightarrow{e} q'$ and $q \xrightarrow{e} q''$ with $e \in \Sigma$, then $q' = q''$.

A *run* $\rho = q_0 \xrightarrow{t_0} q_0' \xrightarrow{e_0} q_1 \xrightarrow{t_1} q_1' \xrightarrow{e_1} \cdots q_n \xrightarrow{t_n} q_n' \xrightarrow{e_n} q_{n+1} \ldots$ in S is a finite or infinite sequence of alternating time ($t_i \in \mathbb{R}_{\geq 0}$) and discrete ($e_i \in \mathsf{Act}$) steps. $\mathsf{States}(\rho) = \{q_0, q_0', , q_1, q_1', \ldots, q_n, q_n', \ldots\}$ is the set of states encountered on ρ. We denote by $\mathit{first}(\rho) = q_0$ and if ρ is finite and has n alternating time and discrete steps $\mathit{last}(\rho) = q_n$. $\mathsf{Runs}(q, S)$ is the set of (finite and infinite) runs in S starting from q. The set of runs of S is $\mathsf{Runs}(S) = \bigcup_{q \in Q} \mathsf{Runs}(q, S)$. We use $q \xrightarrow{e}$ as a shorthand for "$\exists q'$ s.t. $q \xrightarrow{e} q'$" and extends this notation to finite runs $\rho \xrightarrow{e}$ whenever $\mathit{last}(\rho) \xrightarrow{e}$.

Definition 2 (Timed Games (TG)). *A* timed game $G = (Q, Q_0, \mathsf{Act}, \longrightarrow)$ *is a TTS such that* Act *is partitioned into* controllable *actions* Act_c *and* uncontrollable actions Act_u.

Strategies, Reachability Games. A strategy [17] is a function that during the cause of the game constantly gives information as to what the controller should do in order to win the game. In a given situation the strategy could suggest the controller to either i) "do a particular controllable action" or ii) "do nothing at this point in time, just wait" which will be denoted by the special symbol λ. For instance if one wants to delay until some clock value x reaches $\frac{4}{3}$ (as would be a good strategy in the location ℓ_0 of Fig. 1) then the strategy would be: for $x < \frac{4}{3}$ do λ and for $x = \frac{4}{3}$ do the control action from ℓ_0 to ℓ_1.

Definition 3 (Strategy). *Let $G = (Q, Q_0, \mathsf{Act}, \longrightarrow)$ be a TG. A* strategy *f over G is a partial function from* $\mathsf{Runs}(G)$ *to* $\mathsf{Act}_c \cup \{\lambda\}$.

We denote $\mathsf{Strat}(G)$ the set of strategies over G. A strategy f is *state-based* whenever $\forall \rho, \rho' \in \mathsf{Runs}(G), \mathit{last}(\rho) = \mathit{last}(\rho')$ implies that $f(\rho) = f(\rho')$. State-based strategies are also called *memoryless* strategies in game theory [11, 19]. The possible runs that may be realized when the controller follows a particular strategy is defined by the following notion of outcome (see e.g. [11]):

Definition 4 (Outcome). *Let $G = (Q, Q_0, \mathsf{Act}, \longrightarrow)$ be a TG and f a strategy over G. The* outcome $\mathsf{Outcome}(q, f)$ *of f from q in G is the subset of* $\mathsf{Runs}(q, G)$ *defined inductively by:*

– $q \in \mathsf{Outcome}(q, f)$,
– *if* $\rho \in \mathsf{Outcome}(q, f)$ *then* $\rho' = \rho \xrightarrow{e} q' \in \mathsf{Outcome}(q, f)$ *if* $\rho' \in \mathsf{Runs}(q, G)$ *and one of the following three conditions hold:*
  1. $e \in \mathsf{Act}_u$,
  2. $e \in \mathsf{Act}_c$ *and* $e = f(\rho)$,
  3. $e \in \mathbb{R}_{\geq 0}$ *and* $\forall 0 \leq e' < e, \exists q'' \in Q$ *s.t.* $last(\rho) \xrightarrow{e'} q'' \wedge f(\rho \xrightarrow{e'} q'') = \lambda$.
– *for an infinite run* ρ, $\rho \in \mathsf{Outcome}(q, f)$ *if all the finite prefixes of* ρ *are in* $\mathsf{Outcome}(q, f)$.

Note that some strategies may block the evolution at some point for instance if condition 3 above is not satisfied. One has to be careful when synthesizing strategies to ensure condition 3 and this is not trivial (see [7], theorem 2 for details).

Definition 5 (Reachability Timed Games (RTG)). *A* reachability timed game $G = (Q, Q_0, \mathsf{Goal}, \mathsf{Act}, \longrightarrow)$ *is a timed game* $(Q, Q_0, \mathsf{Act}, \longrightarrow)$ *with a distinguished set of* goal states $\mathsf{Goal} \subseteq Q$ *such that for all* $q \in \mathsf{Goal}$, $q \xrightarrow{e} q'$ *implies* $q' \in \mathsf{Goal}$.

If G is a RTG, a run ρ is a *winning run* if $\mathsf{States}(\rho) \cap \mathsf{Goal} \neq \emptyset$. The set of winning runs in G from q is denoted $\mathsf{WinRuns}(q, G)$.

For reachability games one has to choose a semantics for uncontrollable actions: either i) they can only spoil the game and it is up to the controller to do some controllable action to win ([5, 17, 14]) or ii) if at some state s only an uncontrollable action is enabled but forced to happen and leads to a winning state then s is winning. The choice we make is to follow the framework used by La Torre *et al* in [14, 1] where uncontrollable actions cannot help to win. This choice is made for the sake of simplicity (mainly for the proof of theorem 3). However, we can handle any reasonable semantics like ii) above but the proofs are more involved (see [7]).

We now formalize the previous notions. A *maximal run* ρ is either an infinite run (supposing strict alternation of delays and actions) or a finite run ρ that satisfies either (i) $last(\rho) \in \mathsf{Goal}$ or ii) $\forall t \ \geq 0$, if $\rho \xrightarrow{t} q' \xrightarrow{a}$ then $a \in \mathsf{Act}_u$ (i.e. the only possible next discrete actions from $last(\rho)$, if any, are uncontrollable actions). A strategy f is *winning* from q if all maximal runs in $\mathsf{Outcome}(q, f)$ are in $\mathsf{WinRuns}(q, G)$. A state q in a RTG G is *winning* if there exists a winning strategy f from q in G. We denote by $\mathcal{W}(G)$ the set of winning states in G and $\mathsf{WinStrat}(q, G)$ the set of winning strategies from q over G.

Control of Linear Hybrid Games. In the remainder of this section we summarize previous results [11, 17, 20] obtained for particular classes of RTG: Linear Hybrid Games (LHG).

Let X be a finite set of real-valued variables. We denote *Lin*(X) the set of linear constraints over the variables in X. *Lin*$_c(X)$ is the subset of convex linear constraints over X. A *valuation* of the variables in X is a mapping from X to $\mathbb{R}$ (thus an element of $\mathbb{R}^X$). For a valuation v and a *linear assignment*[4] α we denote $v[\alpha]$ the valuation defined by $v[\alpha](x) = \alpha(x)(v)$. *Assign*$(X)$ is the set of linear assignments over X. For $r : X \longrightarrow \mathbb{Q}$ and $\delta \in \mathbb{R}_{\geq 0}$ we denote $v + r \cdot \delta$ the valuation s.t. for all $x \in X$, $(v + r \cdot \delta)(x) = v(x) + r(x) \cdot \delta$.

[4] A linear assignment assigns to each variable a linear expression.

Definition 6 (LHG [12]). *A* Linear Hybrid Game $H = (L, \ell_0, \mathsf{Act}, X, E, \mathsf{inv}, \mathit{Rate})$ *is a tuple where L is a finite set of* locations*, $\ell_0 \in L$ is the* initial *location,* $\mathsf{Act} = \mathsf{Act}_c \cup \mathsf{Act}_u$ *is the set of* actions *(controllable and uncontrollable actions), X is a finite set of* real-valued variables*, $E \subseteq L \times \mathit{Lin}(X) \times \mathsf{Act} \times \mathit{Assign}(X) \times L$ is a finite set of* transitions*, $\mathsf{inv} : L \longrightarrow \mathit{Lin}_c(X)$ associates to each location its* invariant*, $\mathit{Rate} : L \longrightarrow (X \longrightarrow \mathbb{Q})$ associates to each location and variable an* evolution rate*. A* reachability *LHG is a LHG with a distinguished set of locations $\mathsf{Goal} \subseteq L$ (with no outgoing edges). It defines the set of goal states $\mathsf{Goal} \times \mathbb{R}^X$.*

The semantics of a LHG $H = (L, \ell_0, \mathsf{Act}, X, E, \mathsf{inv}, \mathit{Rate})$ is a TTS $S_H = ((L \times \mathbb{R}^X, (\ell_0, \mathbf{0}), \mathsf{Act}, \longrightarrow))$ where $\longrightarrow$ consists of: i) *discrete steps:* $(\ell, v) \xrightarrow{e} (\ell', v')$ if there exists $(\ell, g, e, \alpha, \ell') \in E$ s.t. $v \models g$ and $v' = v[\alpha]$; ii) *time steps:* $(\ell, v) \xrightarrow{\delta} (\ell, v')$ if $\delta \in \mathbb{R}_{\geq 0}$, $v' = v + \mathit{Rate}(\ell) \cdot \delta$ and $v, v' \in \mathsf{inv}(\ell)$.

For reachability LHG, the computation of the winning states is based on the definition of a *controllable predecessors* operator [11, 17]. Let $Q = L \times \mathbb{R}^X$. For a subset $X \subseteq Q$ and $a \in \mathsf{Act}$ we define $\mathsf{Pred}^a(X) = \{q \in Q \mid q \xrightarrow{a} q', q' \in X\}$. The controllable and uncontrollable discrete predecessors of X are defined by $\mathsf{cPred}(X) = \bigcup_{c \in \mathsf{Act}_c} \mathsf{Pred}^c(X)$ and $\mathsf{uPred}(X) = \bigcup_{u \in \mathsf{Act}_u} \mathsf{Pred}^u(X)$. A notion of *safe* timed predecessors of a set X w.r.t. a set Y is also needed. Intuitively a state q is in $\mathsf{Pred}_t(X, Y)$ if from q we can reach $q' \in X$ by time elapsing and along the path from q to q' we avoid Y. Formally this is defined by:

$$\mathsf{Pred}_t(X, Y) = \{q \in Q \mid \exists \delta \in \mathbb{R}_{\geq 0} \text{ s.t. } q \xrightarrow{\delta} q',\ q' \in X \text{ and } \mathit{Post}_{[0,\delta]}(q) \subseteq \overline{Y}\}$$

where $\mathit{Post}_{[0,\delta]}(q) = \{q' \in Q \mid \exists t \in [0, \delta] \text{ s.t. } q \xrightarrow{t} q'\}$. Now we are able to define the *controllable predecessors* operator π as follows:

$$\pi(X) = \mathsf{Pred}_t\left(X \cup \mathsf{cPred}(X), \mathsf{uPred}(\overline{X})\right) \tag{1}$$

Note that this definition of π captures the choice that uncontrollable actions cannot be used to win. A symbolic version of the π operator can be defined on LHG [11, 17]. Hence there is a semi-algorithm $\mathsf{CompWin}$ which computes the least fixed point of $\lambda X.\{\mathsf{Goal}\} \cup \pi(X)$ as the limit of an increasing sequence of sets of states (starting with the initial state Goal). If H is a reachability LHG, the result of the computation $\mu X.\{\mathsf{Goal}\} \cup \pi(X)$ is denoted $\mathsf{CompWin}(H)$.

Theorem 1 (Symbolic Algorithm for LHG [11]). *$\mathcal{W}(S_H) = \mathsf{CompWin}(H)$ for a reachability LHG H and hence $\mathsf{CompWin}$ is a symbolic semi-algorithm for computing the winning states of a reachability LHG.*

As for controller synthesis the previous algorithm allows us to compute the winning states of a game but the extraction of strategies is not made particularly explicit. The proof of the following theorem (given in [7]) provides a symbolic algorithm (assuming time determinism) that synthesizes winning

Theorem 2 (Synthesis of Winning Strategies [7]). *Let H be a LHG. If the semi-algorithm $\mathsf{CompWin}$ terminates for H, then we can compute a polyhedral[5] strategy which is winning in each state of $\mathsf{CompWin}(H)$ and state-based.*

[5] A strategy f is polyhedral if for all $a \in \mathsf{Act}_c \cup \{\lambda\}$, $f^{-1}(a)$ is a finite union of convex polyhedra for each location of the LHG.

3 Priced Timed Games (PTG)

In this section we define *Priced Timed Games (PTG)*. We focus on *reachability PTG (RPTG)* where the aim is to reach a particular state of the game at the *lowest* possible cost. We give a new run-based definition of the *optimal cost*. We then relate our definition with the one given in [14] (note that the definition of [1] seems close to the one in [14] but it is not clear enough for us how close they are) and prove both definitions are indeed equivalent.

Priced Timed Games.

Definition 7 (Priced Timed Transition Systems (PTTS)). *A* priced timed transition system *is a pair* (S, Cost) *where* $S = (Q, Q_0, \mathsf{Act}, \longrightarrow)$ *is a TTS and* Cost *is a* cost function *i.e. a mapping from* $\longrightarrow$ *to* $\mathbb{R}_{\geq 0}$ *that satisfies:*

- PRICE ADDITIVITY: *if* $q \xrightarrow{d} q'$ *and* $q' \xrightarrow{d'} q''$ *with* $d,\ d' \in \mathbb{R}_{\geq 0}$, *then the following holds:* $\mathsf{Cost}(q \xrightarrow{d+d'} q'') = \mathsf{Cost}(q \xrightarrow{d} q') + \mathsf{Cost}(q' \xrightarrow{d'} q'')$.
- BOUNDED COST RATE: *there exists* $K \in \mathbb{N}$ *such that for every* $q \xrightarrow{d} q'$ *where* $d \in \mathbb{R}_{\geq 0}$, $\mathsf{Cost}(q \xrightarrow{d} q') \leq d.K$

For a transition $q \xrightarrow{e} q'$, $\mathsf{Cost}(q \xrightarrow{e} q')$ *is the* cost *of the transition and we note* $q \xrightarrow{e,p} q'$ *if* $p = \mathsf{Cost}(q \xrightarrow{e} q')$.

All notions concerning runs on TTS extend straightforwardly to PTTS. Let S be a PTTS and $\rho = q_0 \xrightarrow{e_1} q_1 \xrightarrow{e_2} \dots \xrightarrow{e_n} q_n$ a finite run[6] of S. The *cost* of ρ is defined by $\mathsf{Cost}(\rho) = \sum_{i=0}^{n-1} \mathsf{Cost}(q_i \xrightarrow{e_{i+1}} q_{i+1})$.

Definition 8 (Priced Timed Games). *A* priced timed game *(PTG) (resp. Reachability PTG) is a pair* $G = (S, \mathsf{Cost})$ *such that* S *is a TG (resp. RTG) and* Cost *is a cost function.*

All the notions like strategies, outcomes, winning states are already defined for (R)TG and carry over in a natural way to (R)PTG. The *cost* $\mathsf{Cost}(q, f)$ of a winning strategy $f \in \mathsf{WinStrat}(q, G)$ is defined by: $\mathsf{Cost}(q, f) = \sup\{\mathsf{Cost}(\rho) \mid \rho \in \mathsf{Outcome}(q, f)\}$.

Definition 9 (Optimal Cost for a RPTG). *Let* G *be a RPTG and* q *be a state in* G. *The* reachable costs set $\mathsf{Cost}(q)$ *from* q *in* G *is defined by:*

$$\mathsf{Cost}(q) = \{\mathsf{Cost}(q, f) \mid f \in \mathsf{WinStrat}(q, G)\}$$

The optimal cost from q in G *is* $\mathsf{OptCost}(q) = \inf \mathsf{Cost}(q)$. *The* optimal cost *in* G *is* $\sup_{q \in Q_0} \mathsf{OptCost}(q)$ *where* Q_0 *denotes the set of initial states.*

Definition 10 (Optimal Strategies for a RPTG). *Let* G *be a RPTG and* q *a state in* G. *A winning strategy* $f \in \mathsf{WinStrat}(q, G)$ *is said to be* optimal *whenever* $\mathsf{Cost}(q, f) = \mathsf{OptCost}(q)$.

[6] We are not interested in defining the cost of an infinite run as we will only use costs of winning runs which must be finite in the games we play.

Optimal winning strategies do not always exist, even for RPTGs deriving from timed automata (see [7]). A family of winning strategies (f_ε) which get arbitrarily close to the optimal cost may be rather determined. Our aim is many-fold. We want to 1) compute the optimal cost of winning, 2) decide whether there is an optimal strategy, and 3) in case there is an optimal strategy compute one such strategy. Before giving a solution to the previous problems we relate our definition of cost optimality to the one given in [14, 1].

Recursive Definition of the Optimal Cost. In [14, 1] a method for computing the optimal cost in priced timed games is introduced: it is defined as the optimal cost one can expect from a state by a function satisfying a set of recursive equations, and not using a run-based definition as we did in the last subsection. We give hereafter the definition of the function used in [14] and prove that it does correspond to our run-based definition of optimal cost. In [1], a similar but more involved definition is proposed, we do not detail this last definition here.

Definition 11 (The O function (Adapted from [14])). *Let G be a RPTG. Let O be the function from Q to $\mathbb{R}_{\geq 0} \cup \{+\infty\}$ that is the least fixed point*[7] *of the following functional:*

$$O(q) = \inf_{\substack{q \xrightarrow{t,p} q' \\ t \in \mathbb{R}_{\geq 0}}} \max \begin{cases} \min\left(\left(\min_{\substack{q' \xrightarrow{c,p'} q'' \\ c \in \mathsf{Act}_c}} p + p' + O(q'') \right), p + O(q') \right) & (1) \\ \sup_{\substack{q \xrightarrow{t',p'} q'' \\ t' \leq t}} \ \max_{\substack{q'' \xrightarrow{u,p''} q''' \\ u \in \mathsf{Act}_u}} p' + p'' + O(q''') & (2) \end{cases} \qquad (\lozenge)$$

The following theorem relates the two definitions:

Theorem 3. *Let $G = (S, \mathsf{Cost})$ be a RPTG induced by a LHG and Q its set of states. Then $O(q) = \mathsf{OptCost}(q)$ for all $q \in Q$.*[8]

4 Reducing Priced Timed Games to Timed Games

In this section we show that computing the optimal cost to win a priced timed game amounts to solving a control problem (without cost).

Priced Timed Game Automata. Let X be a finite set of real-valued variables called clocks. We denote $\mathcal{B}(X)$ the set of constraints φ generated by the grammar: $\varphi ::= x \sim k \mid \varphi \wedge \varphi$ where $k \in \mathbb{Z}$, $x, y \in X$ and $\sim \in \{<, \leq, =, >, \geq\}$. A *valuation* of the variables in X is a mapping from X to $\mathbb{R}_{\geq 0}$ (thus an element of $\mathbb{R}_{\geq 0}^X$). For a valuation v and a set $R \subseteq X$ we denote $v[R]$ the valuation that agrees with v on $X \setminus R$ and is zero on R. We denote $v + \delta$ for $\delta \in \mathbb{R}_{\geq 0}$ the valuation s.t. for all $x \in X$, $(v + \delta)(x) = v(x) + \delta$.

[7] The righthand-sides of the equations for $O(q)$ defines a functional $\mathcal{F}$ on $(Q \longrightarrow \mathbb{R}_{\geq 0} \cup \{+\infty\})$. $(Q \longrightarrow \mathbb{R}_{\geq 0} \cup \{+\infty\})$ equipped with the natural lifting of $\leq$ on $\mathbb{R}_{\geq 0} \cup \{+\infty\}$ constitutes a complete lattice. Also $\mathcal{F}$ can be quite easily seen to be a monotonic functional on this lattice. It follows from Tarski's fixed point theory that the least fix point of $\mathcal{F}$ exists.

[8] Note that if a state $q \in Q$ is not winning, both $O(q)$ and $\mathsf{OptCost}(q)$ are $+\infty$.

Definition 12 (PTGA). *A* Priced Timed Game Automaton A *is a tuple* $(L, \ell_0, \mathsf{Act}, X, E, \mathsf{inv}, f)$ *where* L *is a finite set of* locations, $\ell_0 \in L$ *is the* initial *location,* $\mathsf{Act} = \mathsf{Act}_c \cup \mathsf{Act}_u$ *is the set of* actions *(partitioned into controllable and uncontrollable actions),* X *is a finite set of* real-valued clocks, $E \subseteq L \times \mathcal{B}(X) \times \mathsf{Act} \times 2^X \times L$ *is a finite set of* transitions, $\mathsf{inv} : L \longrightarrow \mathcal{B}(X)$ *associates to each location its* invariant, $f : L \cup E \longrightarrow \mathbb{N}$ *associates to each location a* cost rate *and to each discrete transition a* cost. *A* reachability *PTGA (RPTGA) is a PTGA with a distinguished set of locations* $\mathsf{Goal} \subseteq L$ *(with no outgoing edges). It defines the set of goal states* $\mathsf{Goal} \times \mathbb{R}^X_{\geq 0}$.

The semantics of the PTGA is a PTTS $S_A = ((L \times \mathbb{R}^X_{\geq 0}, (\ell_0, \mathbf{0}), \mathsf{Act}, \longrightarrow), \mathsf{Cost})$ where $\longrightarrow$ consists of: i) *discrete steps:* $(\ell, v) \xrightarrow{e} (\ell', v')$ if there exists $(\ell, g, e, R, \ell') \in E$ s.t. $v \models g$ and $v' = v[R]$; $\mathsf{Cost}((\ell, v) \xrightarrow{e} (\ell', v')) = f(\ell, g, e, R, \ell')$; ii) *time steps:* $(\ell, v) \xrightarrow{\delta} (\ell, v')$ if $\delta \in \mathbb{R}_{\geq 0}$, $v' = v + \delta$ and $v, v' \in \mathsf{inv}(\ell)$; and $\mathsf{Cost}((\ell, v) \xrightarrow{\delta} (\ell, v')) = \delta \cdot f(\ell)$. Note that this definition of Cost gives a cost function as defined in Def. 7.

From Optimal Reachability Game to Reachability Game. Assume we want to compute the optimal cost to win a reachability priced timed game automaton A. We define a (usual and unpriced) LHG H as follows: we use a variable $cost$ in the LHG to stand for the cost value. We build H with the same discrete structure as A and specify a rate for $cost$ in each location: if the cost increases with a rate of $+k$ per unit of time in A, then we set the derivative of $cost$ to be $-k$ in H; if the cost of a discrete transition is $+k$ in A, then we update $cost$ by $cost := cost - k$ in H. To each state q in (the semantics of) A there are many corresponding states (q, c) in H, where c is the value of the $cost$ variable. For such a state (q, c) we denote $\exists cost.(q, c)$ the state q. If X is a set of states in (the semantics of) H then $\exists cost.X = \{q \mid \exists c \geq 0 \mid (q, c) \in X\}$. From the PTGA of Fig. 1 we obtain the LHG of Fig. 2.

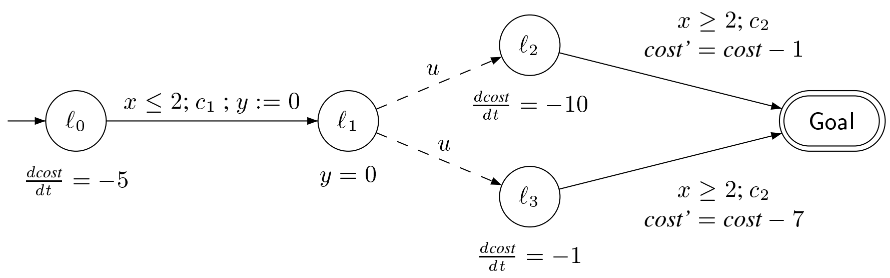

Fig. 2. The Linear Hybrid Game H.

Now we solve the following control problem on the LHG: "can we win in H with the goal states $\mathsf{Goal} \wedge cost \geq 0$?" Intuitively speaking we are asking the question: "what is the minimal amount of resource ($cost$) needed to win the control game H?" For a PTGA A we can compute the winning states of H with the semi-algorithm $\mathsf{CompWin}$ (defined at the end of section 2) and if it terminates the wining set of states $W_H = \mathsf{CompWin}(H)$ is a union of zones of the form $(\ell, R \wedge cost \succ h)$ where ℓ is a location,

$R \subseteq \mathbb{R}^X_{\geq 0}$, h is a piece-wise affine function on R and $\succ \in \{>, \geq\}$ (because π preserves this kind of sets). Hence we have the answer to the optimal reachability game: we intersect the set of initial states with the set of winning states W_H, and in case it is not empty, the projection on the $cost$ axis yields a constraint on the cost like $cost \succ k$ with $k \in \mathbb{Q}_{\geq 0}$ and $\succ \in \{>, \geq\}$. By definition of winning set of states in reachability games, *i.e.* this is the largest set from which we can win, no cost lower than or equal to k is winning and we can deduce that k is the optimal cost. Also we can decide whether there is an optimal strategy or not: if $\succ$ is equal to $>$ there is no optimal strategy and if $\succ$ is $\geq$ there is one.

Note that with our reduction of optimal control of PTGA to control of LHG, the cost information becomes part of the state and that the runs in A and H are closely related. The correctness of the reduction is then given by the next theorem.

Theorem 4. *Let A be a RPTGA and H its corresponding LHG (as defined above). If the semi-algorithm* $\mathsf{CompWin}$ *terminates for H and if $W_H = \mathsf{CompWin}(H)$, then: 1)* $\mathsf{CompWin}$ *terminates for A and $W_A \stackrel{def}{=} \mathsf{CompWin}(A) = \exists cost.W_H$; and 2) $(q, c) \in W_H \iff$ there exists $f \in \mathsf{WinStrat}(q, W_A)$ with $\mathsf{Cost}(q, f) \leq c$.*

Computation of the Optimal Cost and Strategy. Let $X \subseteq \mathbb{R}^n_{\geq 0}$. The *upward closure* of X, denoted $\uparrow X$ is the set $\uparrow X = \{x' \mid \exists x \in X \text{ s.t. } x' \geq x\}$.

Theorem 5. *Let A be a RPTGA and H its corresponding LHG. If the semi-algorithm* $\mathsf{CompWin}$ *terminates for H then for $q \in W_A$, $\uparrow\mathsf{Cost}(q) = \{c \mid (q, c) \in W_H\}$.*

Corollary 1 (Optimal Cost). *Let A be a RPTGA and H its corresponding LHG. If the semi-algorithm* $\mathsf{CompWin}$ *terminates for H then $\uparrow\mathsf{Cost}(\ell_0, \mathbf{0})$ is computable and is of the form $cost \geq k$ (left-closed) or $cost > k$ (left-open) with $k \in \mathbb{Q}_{\geq 0}$. In addition we get that $\mathsf{OptCost}(l_0, \mathbf{0}) = k$.*

Corollary 2 (Existence of an Optimal Strategy). *Let A be a RPTGA. If $\uparrow\mathsf{Cost}(\ell_0, \mathbf{0})$ is left-open then there is no optimal strategy. Otherwise we can compute a winning and optimal strategy.*

Termination Criterion & Optimal Strategies.

Theorem 6. *Let A be a RPTGA satisfying the following hypotheses: 1) A is bounded, i.e. all clocks in A are bounded ; 2) the cost function of A is* strictly non-zeno*, i.e. there exists some $\kappa > 0$ such that the accumulated cost of every cycle in the region automaton associated with A is at least κ. Then the semi-algorithm* $\mathsf{CompWin}$ *terminates for H, where H is the LHG associated with A.*

Note that the strategy built in corollary 2 is state-based for H but is *a priori* no more state-based for A: indeed the strategy for H depends on the current value of the cost (which is part of the state in H). The strategy for A is thus dependent on the run and not memoryless. More precisely it depends on the last state (ℓ, v) of the run and on the accumulated cost along the run.

Nevertheless, we now give a sufficient condition for the existence of optimal cost-independent strategies and exhibit a restricted class of automata for which this conditions holds.

Theorem 7. *Let A be a RPTGA and H the associated LHG. If* $\mathsf{CompWin}$ *terminates for H and W_H is a union of sets of the form $(\ell, R, cost \geq h)$ then there exists a state-based strategy f defined over $W_A = \exists cost.W_H$ s.t. for each $q \in W_A$, $f \in \mathsf{WinStrat}(q, W_A)$ and* $\mathsf{Cost}(q, f) = \mathsf{OptCost}(q)$.

Note that under the previous conditions we build a strategy f which is *uniformly optimal i.e.* optimal for all states of W_A. A syntactical criterion to enforce the condition of theorem 7 is that the constraints (guards) on controllable actions are non-strict and constraints on uncontrollable actions are strict.

Remarks on the hypotheses in Theorems 6 and 7. The hypothesis on A being bounded is not restrictive because all priced timed automata can be transformed into bounded priced timed automata having the same behaviours (see for example [16]). The strict non-zenoness of the cost function can be checked on priced timed game automata: indeed it is sufficient to check whether there is a cycle whose price is 0 in the so-called "corner-point abstraction" (see [6, 9]) ; then, if there is no cycle with cost 0, it means that the cost is strictly non-zeno, otherwise, it is not strictly non-zeno.

5 Conclusion

In this paper we have given a new run-based definition of cost optimality for priced timed games. This definition enables us to prove the following results: the optimal cost can be computed for the class of priced timed game automata with a strictly non-zeno cost. Moreover we can decide whether there exists an optimal strategy which could not be done in previous works [14, 1]. In case an optimal strategy exists we can compute a witness. Finally we give some additional results concerning the type of information needed by the optimal strategy and exhibit a class of priced timed game automata for which optimal state-based (no need to keep track of the cost information) can be synthetized. Our strategy extraction algorithm has been implemented using the tool HYTECH [8].

Our future work will be on extending the class of systems for which termination is ensured. Our claim is that there is no need for the strict non-zenoness hypothesis for termination. Another direction will consist in extending our work to optimal safety games where we want to minimize for example the cost per time unit along infinite schedules whatever the environment does, which would naturally extends both this current work and [9].

References

1. R. Alur, M. Bernadsky, and P. Madhusudan. Optimal reachability in weighted timed games. In *Proc. 31st Int. Coll. Automata, Languages and Programming (ICALP'04)*, *LNCS 3142*, pp. 122–133. Springer, 2004.
2. R. Alur, C. Courcoubetis, and T. Henzinger. Computing accumulated delays in real-time systems. In *Proc. 5th Int. Conf. Computer Aided Verification (CAV'93)*, *LNCS 697*, pp. 181–193. Springer, 1993.

3. R. Alur, S. La Torre, and G. Pappas. Optimal paths in weighted timed automata. In *Proc. 4th Int. Work. Hybrid Systems: Computation and Control (HSCC'01)*, *LNCS 2034*, pp. 49–62. Springer, 2001.
4. E. Asarin and O. Maler. As soon as possible: Time optimal control for timed automata. In *Proc. 2nd Int. Work. Hybrid Systems: Computation and Control (HSCC'99)*, *LNCS 1569*, pp. 19–30. Springer, 1999.
5. E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *Proc. IFAC Symp. System Structure and Control*, pp. 469–474. Elsevier Science, 1998.
6. G. Behrmann, A. Fehnker, T. Hune, K. Larsen, P. Pettersson, J. Romijn, and F. Vaandrager. Minimum-cost reachability for priced timed automata. In *Proc. 4th Int. Work. Hybrid Systems: Computation and Control (HSCC'01)*, *LNCS 2034*, pp. 147–161. Springer, 2001.
7. P. Bouyer, F. Cassez, E. Fleury and K. Larsen. Optimal Strategies on Priced Timed Game Automata. BRICS Report Series, February 2004.
8. P. Bouyer, F. Cassez, E. Fleury and K. Larsen. Synthesis of Optimal Strategies Using HyTech. In Proc. Games in Design and Verification (GDV'04), ENTCS. Elsevier, 2004. To appear.
9. P. Bouyer, E. Brinksma, and K. Larsen. Staying alive as cheaply as possible. In *Proc. 7th Int. Work. Hybrid Systems: Computation and Control (HSCC'04)*, *LNCS 2993*, pp. 203–218. Springer, 2004.
10. C. Courcoubetis and M. Yannakakis. Minimum and maximum delay problems in real-time systems. *Formal Methods in System Design*, 1(4):385–415, 1992.
11. L. De Alfaro, T. Henzinger, and R. Majumdar. Symbolic algorithms for infinite-state games. In *Proc. 12th Int. Conf. Concurrency Theory (CONCUR'01)*, *LNCS 2154*, pp. 536–550. Springer, 2001.
12. T. Henzinger. The theory of hybrid automata. In *Proc. 11th IEEE Annual Symp. Logic in Computer Science (LICS'96)*, pp. 278–292. IEEE Computer Society Press, 1996.
13. T. Hune, K. Larsen, and P. Pettersson. Guided synthesis of control programs using UPPAAL. In *Proc. IEEE ICDS Int. Work. Distributed Systems Verification and Validation*, pp. E15–E22. IEEE Computer Society Press, 2000.
14. S. La Torre, S. Mukhopadhyay, and A. Murano. Optimal-reachability and control for acyclic weighted timed automata. In *Proc. 2nd IFIP Int. Conf. Theoretical Computer Science (TCS 2002)*, *IFIP Proceedings 223*, pp. 485–497. Kluwer, 2002.
15. K. Larsen. Resource-efficient scheduling for real time systems. In *Proc. 3rd Int. Conf. Embedded Software (EMSOFT'03)*, *LNCS 2855*, pp. 16–19. Springer, 2003. Invited talk.
16. K. Larsen, G. Behrmann, E. Brinksma, A. Fehnker, T. Hune, P. Pettersson, and J. Romijn. As cheap as possible: Efficient cost-optimal reachability for priced timed automata. In *Proc. 13th Int. Conf. Computer Aided Verification (CAV'01)*, *LNCS 2102*, pp. 493–505. Springer, 2001.
17. O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Proc. 12th Annual Symp. Theoretical Aspects of Computer Science (STACS'95)*, *LNCS 900*, pp. 229–242. Springer, 1995.
18. P. Niebert, S. Tripakis, and S. Yovine. Minimum-time reachability for timed automata. In *Proc. 8th IEEE Mediterranean Conf. Control and Automation*, 2000.
19. W. Thomas. On the synthesis of strategies in infinite games. In *Proc. 12th Annual Symp. Theoretical Aspects of Computer Science (STACS'95)*, *LNCS 900*, pp. 1–13. Springer, 1995. Invited talk.
20. H. Wong-Toi. The synthesis of controllers for linear hybrid automata. In *Proc. 36th IEEE Conf. Decision and Control*, pp. 4607–4612. IEEE Computer Society Press, 1997.