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 games for 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 it is decidable whether there is an optimal strategy in which case an optimal strategy can be computed. Our results extend previous decidability result which require the underlying game automata to be acyclic. Finally, our results are encoded in a first prototype in HyTech which is applied on a small case-study.

1 Introduction

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 [1,10,15,20,22,26].

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 [6,13,24].

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

⋆ Work partially supported by ACI Cortos, a program of the French government.
† Basic Research in Computer Science (www.brics.dk).

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 [12] and proved computable in [25]. The related problem of synthesizing time-optimal winning strategies for timed game automata was shown computable in [5].

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. [7] and [4] 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 [23]. In [27] the implementation of cost-optimal reachability is improved considerably by exploiting the duality with linear programming problems over zones (min-cost flow problems). More recently [9], the problem of computing optimal *infinite* schedules (in terms of minimal limit-ratios) is solved for the model of priced timed automata.

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 under conditions concerning the strictness of constraints and non-zenoness of cost. In [21] the authors solve the problem of computing the optimal cost for **acyclic** priced timed game. The existing results mentioned above related to timed game automata and priced timed automata respectively, are all based on various extensions of the so-called classical region- and zone-techniques. In this paper the solution is obtained in a radically different way and we extend the result of [21] into many directions: first we give a new (run-based) definition of cost-optimality; second we solve the problem of computing the optimal cost for the class of priced timed game automata having a strictly non-zeno cost; third we can decide whether there exists an optimal strategy (*i.e.* achieving the optimal cost); fourth we can synthetize an optimal strategy in case one exists.

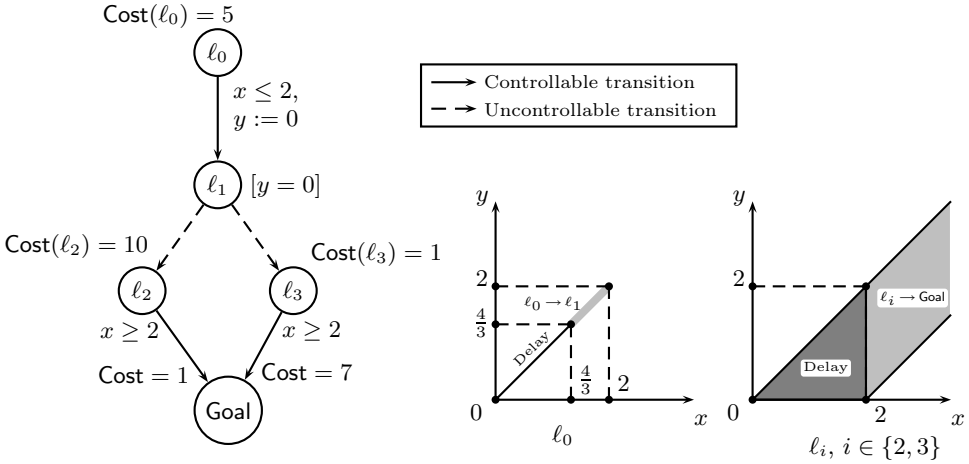


Fig. 1. *A small Priced Time Game Automata. Optimal strategy has cost* $14\frac{1}{3}$.

Consider the priced timed game automaton in Fig. 1. Here the cost-rates 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. 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 $14\frac{1}{3}$. In Fig. 1 we illustrate the optimal strategy for all states reachable from the initial state provided by our HyTech-implementation.

The outline of the paper is as follows: in section 2 we recall some basics about *reachability timed games*. Section 3 introduces *priced timed games (PTG)* where we give a new run-based definition of optimality. We also relate our run-based definition of optimality to the recursive one previously given in [21]. Section 4 is the core of the paper where we show how to compute the optimal cost of a PTG and optimal strategies. Finally section 5 reports on preliminary experiments with our implementation in HyTech. For proofs of all theorems we refer the interested reader to [8].

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)* [6,13,24] to model control problems. In this section we recall some known general results about RTG and we finally give an additional result about the controller (strategy) synthesis for RTG. Indeed controller synthesis is well defined for *safety* games but some additional care is needed for RTG as shown later in the section.

Timed Transition Systems and Games.

Definition 1 (Timed Transition Systems). *A* timed transition system *(TTS) is a tuple* $S = (Q, Q_0, \mathsf{Act}, \longrightarrow)$ *with:*

- 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* $\mathbb{R}_{\geq 0}$*. We denote* $\Sigma = \mathsf{Act} \cup \mathbb{R}_{\geq 0}$
- $\longrightarrow \subseteq Q \times \Sigma \times Q$ *is a set of edges. 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[1]: if $q \xrightarrow{e} q'$ and $q \xrightarrow{e} q''$ with $e \in \Sigma$, then $q' = q''$.

A *run* in S is a finite or infinite sequence $\rho = q_0 \xrightarrow{e_1} q_1 \xrightarrow{e_2} \dots \xrightarrow{e_n} q_n \dots$. $\mathsf{States}(\rho) = \{q_0, q_1, \dots, q_n, \dots\}$ is the set of states encountered on ρ. We denote by $\mathit{first}(\rho) = q_0$ and $\mathit{last}(\rho) = q_n$ if ρ is finite and ends in 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 – Adapted from [13]). *A* timed game *(TG)* $G = (Q, Q_0, \mathsf{Act}, \longrightarrow)$ *is a TTS such that* Act *is partitioned into* controllable *actions* Act_c *and* uncontrollable actions Act_u.

Strategies, Reachability Games. A strategy [24] 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" 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 [13,28]. The possible runs that may be realized when the controller follows a particular strategy is defined by the following notion of $\mathsf{Outcome}$ ([13]):

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* $e = f(\rho)$,

[1] Determinism is not essential in our work but it simplifies proofs in the sequel.

3. $e \in \mathbb{R}_{\geq 0}$ *and* $\forall 0 \leq e' < e, \exists q'' \in Q$ *s.t.* $last(\rho) \xrightarrow{e'} q'' \wedge f(\rho \xrightarrow{e'} q'') = \lambda$.

– *an infinite run* ρ *is in* $\in \mathsf{Outcome}(q, f)$ *if all the finite prefixes of* ρ *are in* $\mathsf{Outcome}(q, f)$.

A strategy is *realizable*, whenever the following holds: for all $\rho \in \mathsf{Outcome}(q, f)$ s.t. f is defined on ρ and $f(\rho) = \lambda$, there exists some $\delta > 0$ such that for all $0 \leq t < \delta$, there exist q' with $\rho \xrightarrow{t} q' \in \mathsf{Outcome}(q, f)$ and $f(\rho \xrightarrow{t} q') = \lambda$. Strategies which are not realizable are not interesting because they generate empty set of outcomes. Note that realizability is a weaker notion than the one of implementability considered in [11,14]. It is easy to provide examples of non-realizable strategies (see [8]). We want to avoid this and offer as a secondary result means of synthesizing *realizable* strategies.

In the following, we will restrict our attention to realizable strategies and simply refer to them as strategies.

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 $\mathsf{States}(\rho) \cap \mathsf{Goal} \neq \emptyset$. We denote $\mathsf{WinRuns}(q, G)$ the set of winning runs in G from q.

In the literature one can find (at least) two definitions of the meaning of uncontrollable actions: i) in [6,24] uncontrollable actions can be used to win the game whereas ii) in [21] they cannot help to win the game.

We follow the framework used by La Torre *et al* in [21] 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 the semantics of [24] (case i) as well but the proofs are more involved[2].

We now formalize the previous notions. A *maximal run* ρ is either an infinite run or a finite run that satisfies: $\forall t \geq 0$, $\rho \xrightarrow{t} q' \xrightarrow{a}$ implies $a \in \mathsf{Act}_u$, thus the next discrete actions from $last(\rho)$, if any, are uncontrollable actions. A strategy f is *winning* from q if all runs in $\mathsf{Outcome}(q, f)$ are finite and all maximal runs in $\mathsf{Outcome}(q, f)$ are in $\mathsf{WinRuns}(q, G)$. Note that f must be realizable. 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. We note $\mathsf{WinStrat}(q, G)$ the set of winning (and realizable) strategies from q over G.

In the remainder of this section we summarize previous results obtained for particular classes of RTG: Linear Hybrid Games (LHG). Due to lack of space we will not define this model here but refer to [16] for details.

The computation of the winning states is based on the definition of a *controllable predecessors* operator [13,24]. Let $G = (Q, Q_0, \mathsf{Goal}, \mathsf{Act}, \longrightarrow)$ be a RTG. For

[2] The definition of π later on must be patched as well as the definition of the O function in Def. 10. Theorem 2 still holds for this case as it only depends on the winning set of states.

a subset $X \subseteq Q$ and $a \in \mathsf{Act}$ we define $\mathsf{Pred}^a(X) = \{q \in Q \mid q \xrightarrow{a} q', q' \in X\}$. Now the controllable and uncontrollable discrete predecessors of X are defined by $\mathsf{cPred}(X) = \bigcup_{c \in \mathsf{Act}_c} \mathsf{Pred}^c(X)$, respectively $\mathsf{uPred}(X) = \bigcup_{u \in \mathsf{Act}_u} \mathsf{Pred}^u(X)$. We also need a notion of *safe* timed predecessors of a set X w.r.t. a set Y. 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:

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

where $Post_{[0,\delta]}(q) = \{q' \in Q \mid \exists t \in [0, \delta] \text{ s.t. } q \xrightarrow{t} q'\}$. 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. We denote by $\mathsf{CompWin}$ the semi-algorithm 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 G is a LHG, the result of the computation $\mu X.\{\mathsf{Goal}\} \cup \pi(X)$ is denoted $\mathsf{CompWin}(G)$.

Theorem 1 (Symbolic Algorithm for LHG [13,17]). $\mathcal{W}(G) = \mathsf{CompWin}(G)$ *for a LHG G and hence* $\mathsf{CompWin}$ *is a symbolic semi-algorithm for computing the winning states of a LHG. Moreover* $\mathsf{CompWin}$ *terminates for the subclass of Initialized Rectangular Games [17].*

As for controller synthesis the previous algorithms allow us to compute the winning states of a game but the extraction of strategies is not made particularly explicit. As a secondary result we provide a symbolic algorithm (assuming time determinism) that synthesizes realizable strategies as claimed in the following theorem:

Theorem 2 (Synthesis of Realizable Strategies). *Let G be a time deterministic LHG. If the semi-algorithm* $\mathsf{CompWin}$ *terminates for G, then we can compute a polyhedral*[3] *strategy which is winning (and realizable) in each state of $\mathcal{W}(G)$ and state-based.*

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*. Then we review some previous work [21] on **acyclic** *weighted timed automata* by Salvatore La Torre *et al* where a definition of optimal cost is given as a *state-based* optimal cost function. We conclude this section by relating the two definitions and proving their equivalence.

[3] 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.

Priced Timed Games.

Definition 6 (Priced Timed Transition Systems). *A* priced timed transition system (PTTS) *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* $\mathsf{Cost}(q \xrightarrow{d+d'} q'') = \mathsf{Cost}(q \xrightarrow{d} q') + \mathsf{Cost}(q' \xrightarrow{d'} q'')$.
- 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 *is the cost of* $q \xrightarrow{e} q'$.

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[4] 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 7 (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:

$$\mathsf{Cost}(q, f) = \sup \{\mathsf{Cost}(\rho) \mid \rho \in \mathsf{Outcome}(q, f)\} \tag{2}$$

Definition 8 (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*[5].

Definition 9 (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* $\mathsf{Cost}(q, f) = \mathsf{OptCost}(q)$.

In the presence of RPTGs described by priced timed automata with strict guards an optimal winning strategy may not always exist, rather a family of strategies f_ε which get arbitrarily close to the optimal cost of winning may be 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.

[4] 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.

[5] An alternative definition would be to take the inf if we consider that the choice of the initial state is "controllable".

Recursive Definition of the Optimal Cost. In [21] Salvatore La Torre *et al* introduced a method for computing the optimal cost in **acyclic** priced timed games. In this paper the authors define 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 [21] and prove that it does correspond to our run-based definition of optimal cost.

Definition 10 (The O function (Adapted from [21])). *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[6] of the following functional:*

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

This definition can be justified by the following arguments: item (2) of Def. 10 gives the maximum cost that an uncontrollable action can lead to if it is taken before t; note that by definition $\sup \emptyset = -\infty$ and that (2) is always defined and the outermost max is thus always defined; item (1) gives the best you can expect if a controllable action can be fired; if from q' no controllable action can be taken, then either (i) there is a time step leading to some q' with $O(q')$ finite or (ii) no such state q' is reachable from q: as our semantics specifies that no uncontrollable action can be used to win, we can not win from q (except if $q \in \mathsf{Goal}$) and the optimal cost will be $+\infty$. We have the following theorem that 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 $q \in Q$.*[7]

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). The idea is the following:

[6] 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.

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

From Optimal Reachability Game to Reachability Game. Assume we want to compute the optimal cost to win a priced timed game $\mathcal{A}$. We define a (usual and unpriced) timed game $\mathcal{A}_C$ as follows: we use a variable *cost* to stand for the cost value. We build $\mathcal{A}_C$ with the same discrete structure as $\mathcal{A}$ and specify a rate for *cost* in each location: if the cost increases with a rate of $+k$ per unit of time in $\mathcal{A}$, then we set the derivative of *cost* to be $-k$ in $\mathcal{A}_C$; if the cost of a discrete transition is $+k$ in $\mathcal{A}$, then we update *cost* by $cost := cost - k$ in $\mathcal{A}_C$. Now we solve the following control problem: can we win in $\mathcal{A}_C$ with the goal states being $\mathsf{Goal} \wedge cost \geq 0$? Note that if $\mathcal{A}$ is a priced timed automaton [4,7] (game) then $\mathcal{A}_C$ is a (simple) linear hybrid automaton [16]. Intuitively speaking we want to solve a control game with a distinguished variable *cost* that decreases when time elapses and when a discrete transition is fired according to what it costs in the priced timed game. So we are asking the question: "what is the minimal amount of resource (*cost*) needed to win the control game $\mathcal{A}_C$?" In the case of $\mathcal{A}$ we can compute the winning states of $\mathcal{A}_C$ (with an algorithm for solving hybrid games [13,29]) and if it terminates we have the answer to the optimal reachability game: we intersect the set of initial states with the set of winning states, and in case it is not empty, the projection on the *cost* axis yields a constraint on the cost like $cost > 1$. 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 1 is winning and we can deduce that 1 is the optimal cost. Also we can deduce there is no optimal strategy because of the strict inequality.

The rest of this section is devoted to formalizing this reduction and to proving that this reduction is correct. Then we focus on the computation of optimal strategies and we investigate conditions under which we can compute the optimal cost (*i.e.* a termination criterion).

Definition 11 (RTG associated to a RPTG). *Let* $G = ((Q, Q_0, \mathsf{Goal}, \mathsf{Act}, \longrightarrow_G), \mathsf{Cost})$ *be a RPTG. We associate to* G *the RTG* $\mathsf{Cont}(G) = (Q \times \mathbb{R}_{\geq 0}, Q_0 \times \mathbb{R}_{\geq 0}, \mathsf{Goal} \times \mathbb{R}_{\geq 0}, \mathsf{Act}, \longrightarrow_{\mathsf{Cont}(G)})$ *where* $(q, c) \xrightarrow{e}_{\mathsf{Cont}(G)} (q', c') \iff q \xrightarrow{e, c-c'}_G q'$. *In the sequel we abstract away the subscript of* $\longrightarrow$ *as it will be clear from the context which transition relation is referred to.*

Note that with our reduction, the cost information becomes part of the state and that the runs in G and $\mathsf{Cont}(G)$ are closely related. Now we focus on subclasses of reachability timed games, namely those obtained by enriching timed automata [3] with costs (Priced or Weighted Timed Automata [4,7]). This enables us to rely on symbolic algorithms and to have computability results.

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 $x \in X$, $(v + \delta)(x) = v(x) + \delta$.

Definition 12 (PTGA). *A* Priced Timed Game Automaton $A = (L, \ell_0, \mathsf{Act}, X, E, \mathsf{inv}, f)$ *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 *(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* $\mathsf{Goal} \subseteq L$. *It defines the set of goal states* $\mathsf{Goal} \times \mathbb{R}^X_{\geq 0}$.

The semantics of a PTGA $A = (L, \ell_0, \mathsf{Act}, X, E, \mathsf{inv}, f)$ is a PTG $G_A = ((L \times \mathbb{R}^X_{\geq 0}, (\ell_0, \mathbf{0}), \mathsf{Act}, \longrightarrow), \mathsf{Cost})$ 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. 6.

Lemma 1 (PTGA to LHG). *Let A be a PTGA. There exists a LHG H with a distinguished extra variable cost such that* $\mathsf{Cont}(G_A) = G_H$[8].

The correctness of the reduction is given by the following theorem:

Theorem 4. *Let A be a RPTGA and H its corresponding LHG (as given by lemma 1). If the semi-algorithm* CompWin *terminates for* G_H *and if* $W_H = \mathsf{CompWin}(G_H)$*, then: 1)* CompWin *terminates for* G_A *and* $W_A \stackrel{\text{def}}{=} \mathsf{CompWin}(G_A) = \exists cost.W_H$*; and 2)* $(q, c) \in W_H \iff \exists f \in \mathsf{WinStrat}(q, W_A)$ *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 $\uparrow X = \{x' \mid \exists x \in X \mid x' \geq x\}$.

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

Proof. This is a direct consequence of theorem 4. □

Corollary 1 (Optimal Cost). *Let A be a RPTGA and H its corresponding LHG. If the semi-algorithm* CompWin *terminates for* G_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* $\mathsf{OptCost}(A) = k$.

[8] Note that G_H is the TG that gives the semantics of H.

Proof. The fact that it is of the form $cost \geq k$ or $cost > k$ is direct from theorem 5. Now k is a rational number because we are considering LHG and symbolic algorithms. The iterative computation of the π operator generates only polyhedra defined by rational inequations. As it terminates the result follows. □

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 realizable, winning, optimal strategy.*

Proof. Direct consequence of theorem 4. □

Note that in the proof of corollary 2 we build a state-based strategy for H which is 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. However it only depends on the last state (ℓ, v) of the run and on the accumulated cost along the run.

Termination Criterion & Optimal Strategies.

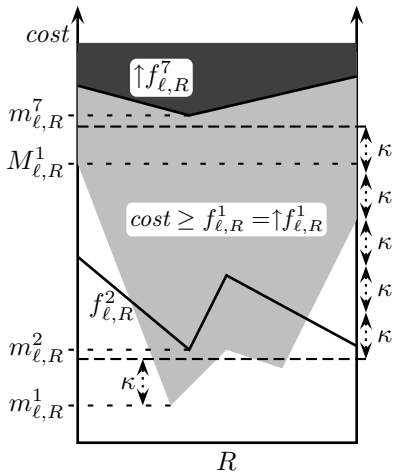
Theorem 6. *Let A be a RPTGA satisfying the following hypotheses:*

- *A is bounded, i.e. all clocks in A are bounded ;*
- *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 CompWin *terminates for G_H, where H is the LHG associated with A.*

Sketch of proof. After a finite number of iterations of CompWin we obtain a set $\mathcal{R}$ of regions of the form $(\ell, R, cost \geq f)$ where f is a piecewise affine function on R[9]. Assume $f^i_{\ell,R}$ is the cost function obtained after reaching the i-th occurrence of (ℓ, R) by computing backward with the semi-algorithm CompWin.

We know that the semi-algorithm CompWin terminates for G_H without the cost variable (this is a timed automaton game as in [6,24]) and this entails that abstracting away the cost function in $\mathcal{R}$ gives a finite number of *cost-free* regions (ℓ, R). To prove termination on G_H it suffices to prove that for each such region (ℓ, R) there is some $i \in \mathbb{N}$ such that $\uparrow f^i_{\ell,R}(R) \subseteq \uparrow f^1_{\ell,R}(R)$. As all clocks are bounded there exists a maximum $M^i_{\ell,R}$ and a minimum $m^i_{\ell,R}$ cost value for the function $f^i_{\ell,R}$ on region (ℓ, R) (see the figure on the righthand side for the case the PTGA has only one clock).



[9] Note that cost constraints could be of the form $cost > f$ as well, but this does not affect our termination argument.

Now if we encounter the $(i+1)$-th occurrence of (ℓ, R) when computing backward with CompWin, we have $m^{i+1}_{\ell,R} \geq m^1_{\ell,R} + i \times \kappa$ (see the previous figure) as each cycle increases the cost of at least κ for any point in R (strictly non-zenoness of the cost). Define $n(\ell, R) = \left\lceil \frac{M^1_{\ell,R} - m^1_{\ell,R}}{\kappa} \right\rceil$. As soon as we have encountered the $(n(\ell, R))$-th occurrence of (ℓ, R) (7 on the previous figure) we have $m^{n(\ell,R)}_{\ell,R} \geq M^1_{\ell,R}$ and thus $\uparrow f^{n(\ell,R)}_{\ell,R}(R) \subseteq \uparrow f^1_{\ell,R}(R)$.

On each branch obtained in the tree corresponding to the (backward) computation of CompWin, once (ℓ, R) has appeared $n(\ell, R)$ times no better cost will be added for region (ℓ, R). Hence CompWin terminates. □

It is not straightforward to build an optimal state-based (without the accumulated cost) strategy as shown by the PTGA A given in Fig. 2. The most nat-

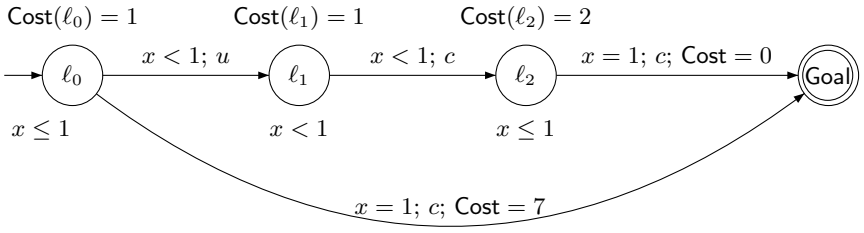


Fig. 2. Priced Timed Game A

ural way to define a state-based (without cost) strategy would be to take in each state (ℓ, v) the action given by the strategy in H in the state (ℓ, v, c) with some minimal c. Doing this would result in a strategy f such that $f(\ell_1, x < 1) = \lambda$. Such a strategy is however not winning. In this particular case, we can build an optimal strategy f^* the cost of which is 8: $f^*(\ell_0, x < 1) = \lambda$, $f^*(\ell_0, x = 1) = c$, $f^*(\ell_1, x < 1) = c$, $f^*(\ell_2, x < 1) = \lambda$ and $f^*(\ell_2, x = 1) = c$. This strategy is optimal in $(\ell_0, \mathbf{0})$ but is not (and needs not to be) optimal in state ℓ_1 for example. From this observation we see that it is difficult to exhibit an algorithm for building a state-based (with no cost) winning strategy.

Nevertheless, we now exhibit a restricted class of automata for which we can synthesize optimal state-based (without the cost information) strategies automatically. One of the challenges of future work is to enlarge this class of automata.

Theorem 7. *Let A be a RPTGA satisfying the following hypotheses:*

1. *A is bounded ;*
2. *the cost function of A is strictly non-zeno ;*
3. *constraints on controllable actions are non-strict ;*
4. *constraints on uncontrollable actions are strict*

Let $W_A = \mathsf{CompWin}(G_A)$ *be the set of winning states. There exists a state-based strategy* f *defined over* W_A *s.t. for each* $(\ell, v) \in W_A$, $f \in \mathsf{WinStrat}((\ell, v), W_A)$ *and* $\mathsf{Cost}((\ell, v), f) = \mathsf{OptCost}(\ell, v)$.

Note that under the previous conditions we build a strategy f which is *globally optimal i.e.* optimal for all states of W_A.

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 [23]). 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 [7,9]) ; 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. As illustrated by Fig. 2, hypotheses on the syntax of the guards seem quite natural to get Theorem 7.

5 Preliminary Experiments

An example has been implemented in HYTECH [19]. We can compute optimal strategies for reachability priced timed game automata: first we get the winning set of states by applying CompWin (see Section 2) ; then, we extract a winning optimal strategy (if any). The code of this example can be found in [8] and on the web page `http://www.lsv.ens-cachan.fr/aci-cortos/ptga/`.

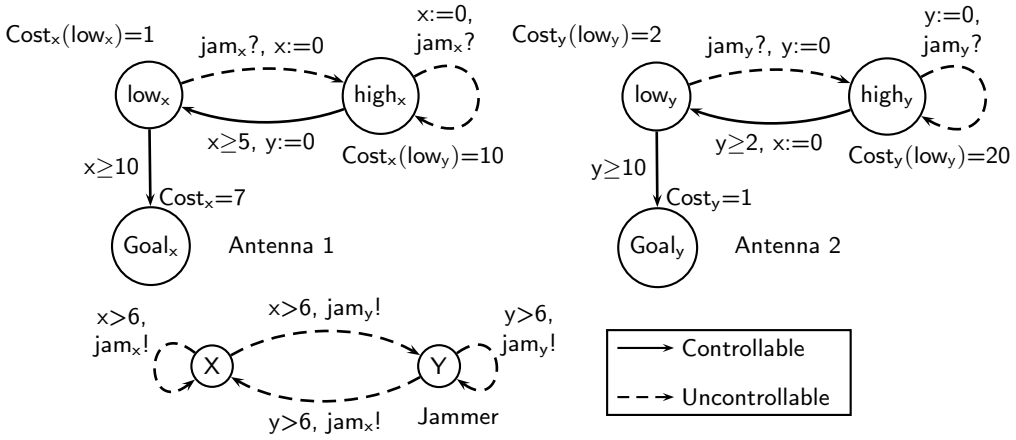


Fig. 3. *Mobile Phone Example.*

We consider a mobile phone with two antennas emitting on different channels. Making the initial connection with the base station takes 10 time units whatever

antenna is in use. Statistically, a jam of the transmission (*e.g.* collision with another phone) may appear every 6 time units in the worst case. When a collision is observed, the antenna tries to transmit with a higher level of energy for a while (at least 5 time units for Antenna 1 and at least 2 time units for Antenna 2) and then can switch back to the lower consumption mode. But switching back to the low consumption mode requires more resources and forces to interrupt the other transmission (Antenna 1 resets variable y of Antenna 2 and vice-versa). The overall cost rate (consumption per time unit) for the mobile phone in a product state $s = (\mathsf{low_x}, \mathsf{high_y}, \mathsf{Y})$ is the sum of the rates of Antenna 1 and Antenna 2 (both are working) *i.e.* $1 + 20 = 21$ and $\mathsf{Cost}(s) = 21$ in our model. Once the connection with the base station is established (either $x \geq 10$ or $y \geq 10$) the message is delivered with an energy consumption depending on the antenna ($\mathsf{Cost} = 7$ for $\mathsf{Antenna\ 1}$ and $\mathsf{Cost} = 1$ for $\mathsf{Antenna\ 2}$). The aim is to connect the mobile phone with an energy consumption (cost) as low as possible whatever happens in the network (jam). This system can be represented by PTGAs (see Fig. 3) and the problem reduces to finding an optimal strategy for reaching one of the goal states $\mathsf{Goal_x}$ or $\mathsf{Goal_y}$. Note that the modelization we propose satisfies the assumptions of Theorem 7, needed to ensure termination and existence of a state-based strategy.

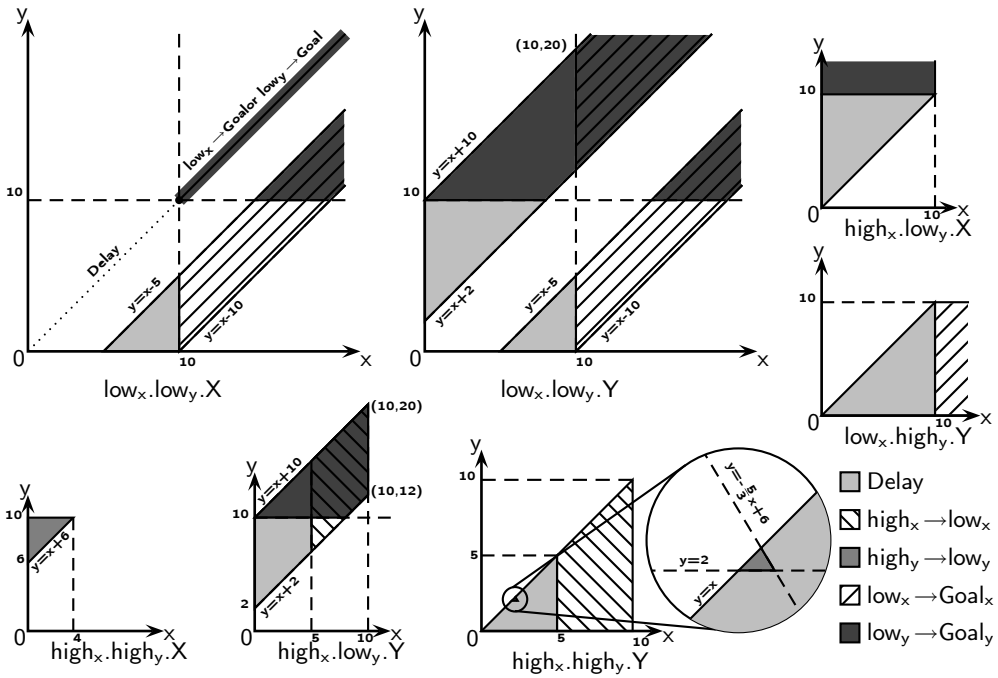


Fig. 4. *Strategy of the* mobile phone *example.*

With our HYTECH implementation we can compute the optimal cost (lowest energy consumption) that we can ensure: it is 109. We can also compute an optimal strategy the graphical representation of which is given for each product location of the system in Fig. 4. The strategy is non trivial and the actions to take depend on complex zones *e.g.* in $(\mathsf{high}_x, \mathsf{low}_y, \mathsf{Y})$ delay when $y \geq x+1 \wedge x \leq 5 \wedge y \leq 10$; Antenna 1 should go to lower consumption mode when $y \geq x + 2 \wedge x \geq 5$. In state $(\mathsf{high}_x, \mathsf{high}_y, \mathsf{Y})$ it is even more complex as described by the zoomed picture: one should delay except if $y \geq 2 \wedge y \leq \frac{5}{3} + 6 \wedge y \leq x$ where Antenna 2 should switch to lower consumption mode. This example demonstrates the power of our algorithm as it is really difficult to come up by hand with such a strategy.

6 Conclusion

In this paper we have given a new run-based definition of cost optimality for priced timed games. This definition enables us to extend previous results on **acyclic** priced timed games [21] and generalize them in the following ways: 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 even for acyclic priced timed games [21]. 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. We have implemented the algorithm we propose in HYTECH and demonstrated the usefulness of our work on a small example of a mobile phone.

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. Y. Abdeddaim. *Modélisation et résolution de problèmes d'ordonnancement à l'aide d'automates temporisés*. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble, France, 2002.
2. R. Alur, C. Courcoubetis, and T. Henzinger. Computing accumulated delays in real-time systems. In *Proc. 5th International Conference on Computer Aided Verification (CAV'93)*, vol. 697 of *LNCS*, pp. 181–193. Springer, 1993.
3. R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science (TCS)*, 126(2):183–235, 1994.
4. 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)*, vol. 2034 of *LNCS*, pp. 49–62. Springer, 2001.

5. 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)*, vol. 1569 of *LNCS*, pp. 19–30. Springer, 1999.
6. E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *Proc. IFAC Symposium on System Structure and Control*, pp. 469–474. Elsevier Science, 1998.
7. 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 International Workshop on Hybrid Systems: Computation and Control (HSCC'01)*, vol. 2034 of *LNCS*, pp. 147–161. Springer, 2001.
8. P. Bouyer, F. Cassez, E. Fleury and K. Larsen. Optimal Strategies on Priced Timed Game Automata. BRICS Report Series, February 2004.
9. P. Bouyer, E. Brinksma, and K. Larsen. Staying alive as cheaply as possible. In *Proc. 7th International Workshop on Hybrid Systems: Computation and Control (HSCC'04)*, LNCS. Springer, 2004. To appear.
10. E. Brinksma, A. Mader, and A. Fehnker. Verification and optimization of a PLC control schedule. *Journal of Software Tools for Technology Transfer (STTT)*, 4(1):21–33, 2002.
11. F. Cassez, T. Henzinger, and J.-F. Raskin. A comparison of control problems for timed and hybrid systems. In *Proc. 5th Int. Workshop on Hybrid Systems: Computation and Control (HSCC'02)*, vol. 2289 of *LNCS*, pp. 134–148. Springer, 2002.
12. C. Courcoubetis and M. Yannakakis. Minimum and maximum delay problems in real-time systems. *Formal Methods in System Design*, 1(4):385–415, 1992.
13. L. De Alfaro, T. Henzinger, and R. Majumdar. Symbolic algorithms for infinite-state games. In *Proc. 12th International Conference on Concurrency Theory (CONCUR'01)*, vol. 2154 of *LNCS*, pp. 536–550. Springer, 2001.
14. M. De Wulf, L. Doyen, and J.-F Raskin. Almost ASAP semantics: From timed models to timed implementations. In *Proc. 7th International Workshop on Hybrid Systems: Computation and Control (HSCC'04)*, LNCS. Springer, 2004. To appear.
15. A. Fehnker. Scheduling a steel plant with timed automata. In *Proc. 6th Int. Conf. Real-Time Computing Systems and Applications (RTCSA'99)*, pp. 280–286. IEEE Computer Society Press, 1999.
16. T. Henzinger. The theory of hybrid automata. In *Proc. 11th IEEE Annual Symposim on Logic in Computer Science (LICS'96)*, pp. 278–292. IEEE Computer Society Press, 1996.
17. T. Henzinger, B. Horowitz, and R. Majumdar. Rectangular hybrid games. In *Proc. 10th International Conference on Concurrency Theory (CONCUR'99)*, vol. 1664 of *LNCS*, pp. 320–335. Springer, 1999.
18. T. Henzinger, P.-H. Ho, and H. Wong-Toi. A user guide to HYTECH. In *Proc. 1st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'95)*, vol. 1019 of *LNCS*, pp. 41–71. Springer, 1995.
19. T. Henzinger, P.-H. Ho, and H. Wong-Toi. HyTech: A model-checker for hybrid systems. *Journal on Software Tools for Technology Transfer (STTT)*, 1(1–2):110–122, 1997.
20. 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.
21. S. La Torre, S. Mukhopadhyay, and A. Murano. Optimal-reachability and control for acyclic weighted timed automata. In *Proc. 2nd IFIP International Confer-*

ence on Theoretical Computer Science (TCS 2002), vol. 223 of *IFIP Conference Proceedings*, pp. 485–497. Kluwer, 2002.
22. K. Larsen. Resource-efficient scheduling for real time systems. In *Proc. 3rd International Conference on Embedded Software (EMSOFT'03)*, vol. 2855 of *LNCS*, pp. 16–19. Springer, 2003. Invited presentation.
23. 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)*, vol. 2102 of *LNCS*, pp. 493–505. Springer, 2001.
24. O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Proc. 12th Annual Symposium on Theoretical Aspects of Computer Science (STACS'95)*, vol. 900, pp. 229–242. Springer, 1995.
25. P. Niebert, S. Tripakis, and S. Yovine. Minimum-time reachability for timed automata. In *Proc. 8th IEEE Mediterranean Conference on Control and Automation*, 2000.
26. P. Niebert and S. Yovine. Computing efficient operations schemes for chemical plants in multi-batch mode. *European Journal of Control*, 7(4):440–453, 2001.
27. J. Rasmussen, K. Larsen, and K. Subramani. Resource-optimal scheduling using priced timed automata. In *Proc. 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'04)*, LNCS. Springer, 2004. To appear.
28. W. Thomas. On the synthesis of strategies in infinite games. In *Proc. 12th Annual Symposium on Theoretical Aspects of Computer Science (STACS'95)*, vol. 900, pp. 1–13. Springer, 1995. Invited talk.
29. H. Wong-Toi. The synthesis of controllers for linear hybrid automata. In *Proc. 36th IEEE Conference on Decision and Control*, pp. 4607–4612. IEEE Computer Society Press, 1997.