Almost Optimal Strategies in One Clock Priced Timed Games

Patricia Bouyer[1], Kim G. Larsen[2],
Nicolas Markey[1], and Jacob Illum Rasmussen[2]

[1] Lab. Spécification & Vérification, CNRS & ENS Cachan, France
{bouyer,markey}@lsv.ens-cachan.fr
[2] Department of Computer Science, Aalborg University, Denmark
{kgl,illum}@cs.aau.dk

Abstract. We consider timed games extended with cost information, and prove computability of the optimal cost and of ε-optimal memoryless strategies in timed games with one clock. In contrast, this problem has recently been proved undecidable for timed games with three clocks.

1 Introduction

An interesting direction of real-time model checking that has recently received substantial attention is to extend and re-target timed automata technology towards optimal scheduling and planning [1, 15, 9]. In particular, as part of this effort, the notion of priced timed automata [6, 5] has been promoted as a useful extension of the classical model of timed automata [4]. In this extended model each location q is associated with a cost c_q giving the cost of a unit of time spent in q. Thus, each run of a priced timed automaton has an accumulated cost, based on which a variety of optimization problems may be formulated.

Several of the established results concerning priced timed automata are concerned with reachability questions. In [3] cost-bounded reachability was shown decidable. [6] and [5] independently show computability of the cost-optimal reachability for priced (or weighted) timed automata using different adaptations of the so-called region technique. In [13, 15] the notion of priced zone is developed allowing efficient implementation of cost-optimal reachability as witnessed by the competitive tool UPPAAL Cora [16]. Also the problem of computing optimal *infinite* schedules (in terms of minimal limit-ratios) has been shown computable [8]. Finally cost-optimal reachability has been shown decidable in a setting with multiple cost-variables [14].

In this paper we consider the more challenging problem of the computation of *cost-optimal winning strategies* for priced timed *game* automata, *i.e.* a game where the controller tries to win at minimal cost and opponent tries to maximize the cost. Consider the priced timed game with the single clock x depicted in Fig. 1. Here the (circle) locations c_1 and c_2 are controllable whereas (square) locations u_1 and u_2 are uncontrollable with cost-rates being 3, 4, 1 and 1, respectively. All four locations have $x \leq 1$ as invariant. Besides transitions between these four locations, additional transitions are indicated to (triangle) locations for which the optimal costs of winning (for any value of x) are assumed to have already been computed (we call those cost functions *outside cost functions* in the sequel). Obviously, c_1 and c_2 have winning strategies for all values of x by uniformly exiting to their respective outside locations (triangle), c_1^{out} and c_2^{out}. However, this strategy is, clearly, suboptimal for both locations. Alternatively, consider the superior strategy for c_2 depicted in Fig. 2. that guarantees cost no larger than depicted in the corresponding cost function. Then it can be shown that this strategy guarantees the optimal cost.

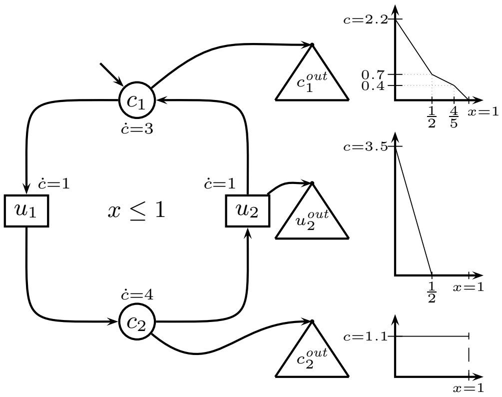
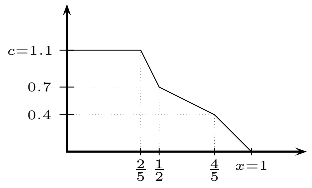


Fig. 1. Sample PTGA with outside cost functions.



$$\sigma(c_2, x) = \begin{cases} c_2^{out} & \text{if } 0 \leq x < 2/5 \\ c_2 & \text{if } 2/5 \leq x < 1/2 \\ u_2 & \text{if } 1/2 \leq x \leq 1 \end{cases}$$

Fig. 2. An optimal strategy in c_2, and the associated cost function.

In [12] the problem of computing cost-optimal winning strategies has been studied and shown computable for *acyclic* priced timed games. Furthermore, in [11] it is proven that computing optimal winning strategies for one-clock PTGA with stopwatch cost (*i.e.* cost are either zero or one) is decidable. [2] and [10] provide partial solutions to the general case of non-acyclic games: under the assumption of certain non-Zenoness behaviour of the underlying priced timed automata it is shown that it suffices only to consider strategies guaranteed to win within some given number k of steps, or alternatively to unfold the given game k times and reduce the problem to solving an acyclic game. To see how restricted these results are, it may be observed that the priced timed game in Fig. 1 does not belong to any of the above classes. In fact, in [11] it has recently been shown that the problem of determining cost-optimal winning strategies for priced timed games is not computable. Most recently, it has been shown that this negative result holds even for priced timed (game) automata with no more than three clocks [7].

In this paper we completely solve the computation of cost-optimal winning strategies for arbitrary priced timed (game) automata *with one clock*: we offer an algorithm for computing optimal costs, explain why optimal strategies need not always exist, whereas memoryless ε-optimal strategies exist and can be computed.

2 Definitions

We write x for the (unique) clock variable, and $\mathcal{X} = \{x\}$. A clock constraint for clock x is an expression of the form $x \in I$ where I is an interval over the reals with integer (or infinite) bounds which can have strict or non-strict bounds. As a shortcut, we may use expressions like $x \geq 5$ instead of $x \in [5, +\infty[$. The set of all clock constraints is denoted $\mathcal{B}(\mathcal{X})$. That a valuation $v\colon \mathcal{X} \to \mathbb{R}_+$ satisfies a clock constraint g is defined in a natural way (v satisfies $x \in I$ whenever $v(x) \in I$), and we then write $v \models g$. We denote by v_0 the valuation that assigns zero to clock x, by $v + t$ ($t \in \mathbb{R}_+$) the valuation that assigns $v(x) + t$ to $x \in \mathcal{X}$.

A *cost function* is a piecewise affine function $f\colon \mathbb{R}_+ \to \mathbb{R}_+ \cup \{+\infty\}$ with negative slopes. We also require that if $\{+\infty\} \in f((n, n+1))$ for some integer n, then $f((n, n+1)) = \{+\infty\}$, and that f is continuous over all intervals $(n, n+1)$. We write CF for the set of all cost functions.

We define an extended notion of priced timed games, with outside cost functions and urgent locations. Those extra features will be needed throughout the proof. A 1-*clock priced timed game with outside cost functions* (PTG_f for short) is a tuple $G = (Q_c, Q_u, Q_f, Q_{\mathrm{urg}}, Q_{\mathrm{init}}, f_{\mathrm{goal}}, T, \eta, P)$ where

- Q_c is a finite set of *controllable locations*, Q_u is a finite set of *uncontrollable locations*. Those sets are disjoint, and we define $Q = Q_c \cup Q_u$;
- Q_f is the set of final locations (it is disjoint from Q).
- $Q_{\mathrm{urg}} \subseteq Q_u$ indicates urgent uncontrollable locations;
- $Q_{\mathrm{init}} \subseteq Q$ is the set of *initial* locations;
- $f_{\mathrm{goal}}\colon Q_f \to \mathrm{CF}$ assigns to each final location a cost function;
- $T \subseteq Q \times \mathcal{B}(\mathcal{X}) \times 2^{\mathcal{X}} \times (Q \cup Q_f)$ is the set of *transitions*;
- $\eta\colon Q \to \mathcal{B}(\mathcal{X})$ defines the *invariants* of each location;
- $P\colon Q \cup T \to \mathbb{N}$ is the *cost (or price) function*.

Standard (1-clock) priced timed games [2, 10] are PTG_f with $Q_{\mathrm{urg}} = \varnothing$ and, for any $q \in Q_f$, $f_{\mathrm{goal}}(q)(\mathbb{R}_+) = \{0\}$ or $\{+\infty\}$.

In the following, G will always refer to a PTG_f, and we will not always rewrite the corresponding tuple. Similarly, G' will denote a PTG_f whose components are "primed".

We assume (w.l.o.g., see [6]) that the clock is bounded, *i.e.*, there exists an integer M such that for every location $q \in Q$, $\eta(q) \Rightarrow x \leq M$.

Let G be a PTG_f. The semantics of G is given as a labeled timed transition system $\mathcal{T} = (S, S_{\mathrm{init}}, \to)$ where $S \subseteq (Q \cup Q_f) \times \mathbb{R}_+$ is the set of states[3], $S_{\mathrm{init}} = Q_{\mathrm{init}} \times \{v_0\}$ is the set of initial states, and the transitions relation $\to \subseteq S \times \mathbb{R}_+ \times S$ is defined as:

1. *(discrete transition)* $(q, v) \xrightarrow{c} (q', v')$ if $q \notin Q_f$ and there exists $(q, g, R, q') \in T$ such that $v(x) \models g$, $v' = [R \leftarrow 0]v$, $v'(x) \models \eta(q')$, and $c = P(q, g, R, q')$;
2. *(delay transition)* $(q, v) \xrightarrow{c} (q, v + t)$ if $q \notin Q_{\mathrm{urg}} \cup Q_f$, and $\forall 0 \leq t' \leq t$, $v + t' \models \eta(q)$, and $c = t \cdot P(q)$.

[3] Formally, $S \subseteq (Q \cup Q_f) \times (\mathbb{R}_+)^{\mathcal{X}}$, but we identify v with $v(x)$ here.

A *run* of G is a (finite) path in the underlying transition system. Given $T, U \subseteq S$, we write[4] $\mathsf{Run}_G(T,U)$ for the set of runs of G issued from $t \in T$ and ending in $u \in U$. Given a run ϱ and a position $v \in \varrho$ along that run, the prefix of ϱ ending in v is denoted by $\varrho_{|v}$. A run is *maximal* if either it is infinite, or no discrete transition is possible (even after a delay transition). A maximal run is *accepting* if it is finite and ends in a final location. Let $\varrho = s_0 \xrightarrow{c_0} s_1 \xrightarrow{c_1} \cdots \xrightarrow{c_{n-1}} s_n$ be a run. Its cost, denoted $\mathsf{cost}(\varrho)$, is either $\sum_{i=0}^{n-1} c_i$ if ϱ is not accepting, or $\sum_{i=0}^{n-1} c_i + f_{\text{goal}}(q_n)(v_n(x))$, where $(q_n, v_n) = s_n$ if ϱ is accepting. An accepting run is *winning* if it has finite cost.

Example. Reconsider the example depicted in Fig. 1. Here, a sample winning run is $\varrho = (c_1, 0) \xrightarrow{0} (u_1, 0) \xrightarrow{0.4} (u_1, 0.4) \xrightarrow{0} (c_2, 0.4) \xrightarrow{0.4} (c_2, 0.5) \xrightarrow{0} (c_2^{out}, 0.5)$ which has cost $\mathsf{cost}(\varrho) = 0.4 \times 1 + 0.1 \times 4 + f_{\text{goal}}(c_2^{out})(0.5) = 1.9$. □

A *strategy* is then a function $\sigma\colon \mathsf{Run}_G(Q \times \mathbb{R}_+, Q_c \times \mathbb{R}_+) \to \{\lambda\} \cup Q \cup Q_f$. Informally, a strategy tells in all controllable locations, what has to be done, and the special symbol λ indicates to delay. A strategy σ is *memoryless* if $\sigma(\varrho) = \sigma(\varrho')$ as soon as ϱ and ϱ' end in the same state.

Let σ be a strategy in G, and ϱ_0 a run in G ending in (q_0, x_0). A run $\varrho = (q_0, x_0) \xrightarrow{c_0} (q_1, x_1) \xrightarrow{c_1} \cdots \xrightarrow{c_{n-1}} (q_n, x_n)$ is a (σ, ϱ_0)-run if for all delay- (or discrete-) transitions $(q_i, x_i) \xrightarrow{c_i} (q_{i+1}, x_{i+1})$ where $q_i \in Q_c$, we have

- $\forall x \in [x_i, x_{i+1}[,\ \sigma(\varrho_0 \cdot \varrho_{|x}) = \lambda$,
- $\sigma(\varrho_0 \cdot \varrho_{|x_i}) = q_{i+1}$.

where $\varrho_0 \cdot \varrho$ denotes the (usual) concatenation. In that case, we say that ϱ is *compatible* with σ after ϱ_0 (or that it is an *outcome* of σ after ϱ_0). We write $\mathsf{Run}_{G,\sigma}(\varrho_0, U)$ for the set of such runs ending in U.

A strategy σ is said *accepting* after (run) ϱ_0 whenever all maximal runs in $\mathsf{Run}_{G,\sigma}(\varrho_0)$ are accepting. If a strategy is not accepting from ϱ_0, we set its cost in G after ϱ_0, $\mathsf{Cost}_G(\sigma, \varrho_0)$, to $+\infty$. Otherwise its cost in G after ϱ_0 is given as: $\mathsf{Cost}_G(\sigma, \varrho_0) = \sup\{\mathsf{cost}(\varrho) \mid \varrho \in \mathsf{Run}_{G,\sigma}(\varrho_0, Q_f \times \mathbb{R}_+)\}$. Obviously, for any two runs ϱ_0 and ϱ_1 ending in (q,x), the sets $\{\mathsf{Cost}_G(\sigma, \varrho_0) \mid \sigma \text{ strategy in } G\}$ and $\{\mathsf{Cost}_G(\sigma, \varrho_1) \mid \sigma \text{ strategy in } G\}$ are equal. An accepting strategy σ after ϱ_0 is *winning* if $\mathsf{Cost}_G(\sigma, \varrho_0)$ is finite. We define for every state s of G, the *optimal cost* of winning from s as $\inf\{\mathsf{Cost}_G(\sigma, \varrho_0) \mid \sigma \text{ strategy in } G\}$ for some run ϱ_0 ending in s. We denote it $\mathsf{OptCost}_G(s)$. If $\mathsf{OptCost}_G(s) < +\infty$, the state s is said *winning* in G. In that case, for every $\varepsilon > 0$, for every run ϱ_0 ending in s, there exists a winning strategy σ s.t. $\mathsf{OptCost}_G(s) \leq \mathsf{Cost}_G(\sigma, \varrho_0) < \mathsf{OptCost}_G(s) + \varepsilon$, and we say that σ is *ε-optimal* from ϱ_0. A strategy σ is *optimal* from ϱ_0 if $\mathsf{Cost}_G(\sigma, \varrho_0) = \mathsf{OptCost}_G(s)$ where ϱ_0 ends in state s.

A strategy σ in G is *(ε, N)-acceptable* (with $\varepsilon > 0$, and $N \in \mathbb{N}$) whenever: (1) it is memoryless, (2) it is ε-optimal, (3) there exist N (consecutive) intervals $(I_i)_{1 \leq i \leq N}$ partitioning $[0,1]$ such that for every location q, for every $1 \leq i \leq N$, for every integer $\alpha < M$, the function $x \mapsto \mathsf{Cost}_G(\sigma, (q,x))$ is affine on every interval $\alpha + I_i$, and the function $x \mapsto \sigma(q,x)$ is constant on $\alpha + I_i$.

[4] In the sequel, we might omit the subscripts G when they are clear from the context.

3 Main result

The main result of this paper is that optimal cost is computable and that almost-optimal memoryless strategies always exist and can be effectively computed. This is summarized by the following theorem:

Theorem 1. *Let G be a PTG$_f$. Then for every location q in G, the function $x \mapsto \mathsf{OptCost}_G((q, x))$ is computable and piecewise-affine. Moreover, for every $\varepsilon > 0$, there exists (and we can effectively compute) a strategy σ in G such that σ is memoryless and ε-optimal in every state.*

We will even prove a stronger result, which is that there exists $N \in \mathbb{N}$ such that for every $\varepsilon > 0$, we can effectively compute an (ε, N)-acceptable strategy σ. The rest of this paper is devoted to a proof of this result.

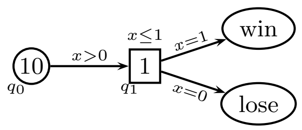


Fig. 3. A game with no optimal strategy

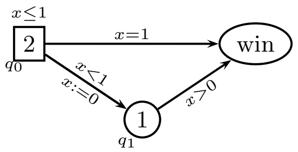


Fig. 4. A game where optimal strategies require memory

There are PTG$_f$ for which no optimal strategies exist, as exemplified by Fig. 3: from q_0, the optimal cost is 1, but a winning strategy consists in delaying in q_0 for some duration $\delta > 0$, yielding a cost of $1 + 9\delta$. This is why we compute, in the general case, ε-optimal strategies. In the same way, as witnessed by Fig 4, it might be the case that optimal strategies exist but require some amount of memory: in the example of Fig 4, state $(q_0, x = 0)$ is winning with optimal cost 2, but no memoryless strategy can achieve that cost for sure.

4 Simplifying Transformations

In this section, we first explain how to restrict to simpler games while preserving the same optimal costs, and we then show how we can inductively compute optimal cost on those simpler games. We also explain how to compute almost-optimal strategies for those simpler games, and how to "lift" those strategies to the original game.

Our transformations consist in two steps: (i) we restrict to PTG$_f$ where the clock is bounded by 1 (denoted $[0, 1]$-PTG$_f$) (Section 4); (ii) we restrict to a $[0, 1]$-PTG$_f$ without resetting transition (Section 4). For each transformation, we prove that:

- the optimal cost in each state of the original game is identical to the optimal cost in some corresponding state in the transformed game,

– we can derive an ε-optimal strategy in the original game from some ε'-optimal strategy in the transformed game.

Section 5 is then devoted to computing the optimal cost and an almost-optimal strategy in the simpler game. For the sake of simplicity, we assume here that there are no discrete costs on transitions. A slight adaptation of the transformation for removing resets can be given for handling discrete costs as well.

Restricting to a PTG$_f$ bounded by 1. The idea of this construction is to reset the clock each time it reaches 1, and to record in the discrete structure what should be the real integer part of the value of the clock (the clock will only store the fractional part of its real value).

Let G be a PTG$_f$. We build another PTG$_f$ G' such that for every $q' \in Q'$, $\eta'(q')$ implies $0 \leq x \leq 1$, and G' is correct for computing optimal cost, in a sense which will be made clear later.

As we have assumed that PTG$_f$ are bounded, we set M the constant bounding G, and we define:

$$\begin{cases} Q'_x &= \{q_{[\alpha,\alpha+1]} \mid q \in Q_x \text{ and } 0 \leq \alpha < M\} \text{ for every } x \in \{c, u, f, \text{urg}\} \\ Q'_{\text{init}} &= \{q_{[0,1]} \mid q \in Q_{\text{init}}\} \end{cases}$$

The set of transitions T' is composed of the following transitions (if g is a guard, $g - \alpha$ denotes the same guard translated by $-\alpha$):

$$\begin{cases} q_{[\alpha,\alpha+1]} \xrightarrow{(g-\alpha)\cap(0\leq x<1)} q'_{[\alpha,\alpha+1]} & \text{if } (q \xrightarrow{g} q') \in T \text{ and } \alpha+1 < M \\ q_{[M-1,M]} \xrightarrow{(g-\alpha)\cap(0\leq x\leq 1)} q'_{[M-1,M]} & \text{if } (q \xrightarrow{g} q') \in T \\ q_{[\alpha,\alpha+1]} \xrightarrow[x:=0]{(g-\alpha)\cap(0\leq x<1)} q'_{[0,1]} & \text{if } (q \xrightarrow[x:=0]{g} q') \in T \text{ and } \alpha+1 < M \\ q_{[M-1,M]} \xrightarrow[x:=0]{(g-\alpha)\cap(0\leq x\leq 1)} q'_{[0,1]} & \text{if } (q \xrightarrow[x:=0]{g} q') \in T \\ q_{[\alpha-1,\alpha]} \xrightarrow[x:=0]{x=1} q_{[\alpha,\alpha+1]} & \text{if } 0 < \alpha < M \end{cases}$$

The invariant η' is defined by $\eta'(q_{[\alpha,\alpha+1]}) = (0 \leq x \leq 1) \wedge (\eta(q) - \alpha)$ if $q \in Q$. The cost function P' is defined by $P'(q_{[\alpha,\alpha+1]}) = P(q)$. The function f'_{goal} is defined by $f'_{\text{goal}}(q_{[\alpha,\alpha+1]})(x) = f_{\text{goal}}(q)(x + \alpha)$ for every $0 \leq x \leq 1$.

Note that all guards and invariants of G' are included in $[0, 1]$, we say that G' is a $[0, 1]$-*PTG*$_f$.

We define f the function which maps every state (q, x) of G onto the state $(q_{[\alpha,\alpha+1]}, x - \alpha)$ of G' such that $0 \leq x - \alpha \leq 1$ and $x < M$ integer implies $x = \alpha$. We now state the following correctness result.

Proposition 2. *For every state (q, x) in G, $\mathsf{OptCost}_G(q, x) = \mathsf{OptCost}_{G'}(f(q, x))$. Moreover, for every $\varepsilon > 0$ and $N \in \mathbb{N}$, given an (ε, N)-acceptable strategy in G', we can compute an (ε, N)-acceptable strategy in G, and vice-versa.*

Removing resetting transitions from SCCs. We have restricted to games with a single clock. A strong property of this model is that each time a resetting transition is taken, then the *very same state* is visited (because the valuation is each time v_0). The construction for removing resetting transitions takes advantage of this property.

Let G be a PTG_f with n resetting transitions. From the previous reduction, we may assume that all the invariants and guards in G imply that $0 \leq x \leq 1$. We build a PTG_f G', made of $n+1$ copies of G, such that no strongly connected component (SCC for short) of G' contains a resetting transition.

We thus define $Q'_c = Q_c \times \{0, ..., n\}$, $Q'_u = Q_u \times \{0, ..., n\}$, and $Q'_f = (Q_f \times \{0, ..., n\}) \cup \{r\}$. A location $(q, i) \in Q'_u$ is urgent iff $q \in Q_{\text{urg}}$. We let $Q'_{\text{init}} = Q_{\text{init}} \times \{0\}$. The outside cost functions are given by $f'_{\text{goal}}((q, i)) = f_{\text{goal}}(q)$, and $f_{\text{goal}}(r) = +\infty$. The invariant is given by $\eta'((q, i)) = \eta(q)$ for $q \in Q$. Transitions are defined as follows:

$$\begin{cases} ((q,i) \xrightarrow{g} (q',i)) \in T' & \text{if } (q \xrightarrow{g} q') \in T \text{ and } i \leq n \\ ((q,i) \xrightarrow[x:=0]{g} (q',i+1)) \in T' & \text{if } (q \xrightarrow[x:=0]{g} q') \in T \text{ and } i < n \\ ((q,n) \xrightarrow{g} r) \in T' & \text{if } (q \xrightarrow[x:=0]{g} q') \in T \text{ and } i = n \end{cases}$$

Last, we set $P'((q, i)) = P'(q)$ for every $q \in Q$, and the price of each transition of T' defined above is the price of the corresponding transition in T.

Proposition 3. *For every state (q, x) in the game G, $\mathsf{OptCost}_G((q, x))$ equals $\mathsf{OptCost}_{G'}(((q, 0), x))$. Moreover, for every $\varepsilon' > 0$ and $N' \in \mathbb{N}$, given an (ε', N')-acceptable strategy in G', we can compute a $(2\varepsilon', N')$-acceptable strategy in G.*

We have thus reduced our problem to computing optimal cost and almost-optimal winning strategies in G'. In G', this can be done by first computing it in the nth copy of G, and then in the $(n-1)$th copy of G, etc.

5 Computing almost-optimal strategies

We have restricted our problem to $[0, 1]$-PTG_f without resets. We can also easily restrict to such PTG_f containing only one SCC: if we can compute the optimal costs and an (ε, N)-acceptable strategy on an SCC, we will be able to handle the general case by working first on the deepest SCC, and then replace it by the corresponding outside function (and an (ε, N)-acceptable strategy).

Thus, we now assume that we only work on a $[0, 1]$-PTG_f without resets and based on an SCC. We prove the following result, which will imply Theorem 1.

Theorem 4. *Let G be a $[0, 1]$-PTG_f without reset such that $(Q_c \cup Q_u, T)$ is an SCC (or contains only one location). Then:*

- *H1.* $\mathsf{OptCost}_G(q, x)$ *is computable for every $q \in Q$ and every $x \in [0, 1]$;*
- *H2.* *for every location $q \in Q$, $x \in [0, 1] \mapsto \mathsf{OptCost}_G(q, x)$ is a cost function whose finitely many segments either have slope $-c$ where $c \in P(Q)$, or are fragments of the outside cost functions of G;*

H3. there exists an integer N such that, for any $\varepsilon > 0$, we can compute an (ε, N)-acceptable strategy in G for every $q \in Q$ and every $x \in [0,1]$.

The rest of this section is devoted to the proof of this theorem, which is by induction on the number of non-urgent locations in G. First we prove the base case of the induction, that is when the game is only composed of urgent locations, or of a single controllable location.

- Proving properties H1 and H2 in the case where G contains only one location is handled straightforwardly, by combining the outside cost functions of G with the cost rate of the location. Property H3 requires more care. Let q be a (controllable) location with a bunch of outside cost functions $\{f_{\text{goal}}(q') \mid q' \in Q_f\}$. Define the function $s\colon x \to \min\{f_{\text{goal}}(q', x) \mid q' \in Q_f\}$. Then $\mathsf{OptCost}_G(q,x) = \inf_{x \le x' \le 1} P(q) \cdot (x' - x) + s(x')$. Let $\varepsilon > 0$. We then define the strategy σ as follows:

$$\sigma(q,x) = \begin{cases} q' \text{ if } \mathsf{OptCost}_G(q,x) = f_{\text{goal}}(q')(x) \\ \lambda \text{ if } \mathsf{OptCost}_G(q,x) < s(x) \text{ and either } s(1) < +\infty \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{or } x \le 1 - \varepsilon/(2P(q)) \\ q' \text{ if } \mathsf{OptCost}_G(q,x) < s(x),\ s(1) = +\infty,\ 1 - \varepsilon/(2P(q)) < x < 1, \\ \quad \text{and } \lim_{x \to 1^-} f_{\text{goal}}(q')(x) = \lim_{x \to 1^-} s(x) \end{cases}$$

  It is not difficult to check that σ is (ε, N)-acceptable for some N which is independent of ε.
- The case where G contains an SCC with only urgent (thus uncontrollable) locations is also straightforward, since the opponent can force the game to never reach a final location, and the optimal cost is then $+\infty$. If the game is composed of a single urgent location, then this is also easy.

We now assume that G is an SCC composed of at least two locations, n of which are non-urgent. We select one of the non-urgent locations having least cost, and denote it with $q_{\min}$, and, depending on the nature (controllable or not) of $q_{\min}$, we explain how we prove that Theorem 4 holds for G if it holds for SCCs having at most $(n-1)$ non-urgent locations.

Case: $q_{\min}$ is controllable. For handling this case, we will prove that the rough intuition that there is no need to delay twice in $q_{\min}$, but we better delay longer in $q_{\min}$ is indeed correct.

From the game G, we construct a game G', made of two copies of G, such that each SCC of the new game contains one location less (see Fig. 5). We define $Q'_c = (Q_c \setminus \{q_{\min}\}) \times \{0,1\} \cup \{q_{\min}\}$, $Q'_u = Q_u \times \{0,1\}$, $Q'_f = Q_f \times \{0,1\} \cup \{r\}$, $Q'_{\text{urg}} = Q_{\text{urg}} \times \{0,1\}$, $Q'_{\text{init}} = Q_{\text{init}} \times \{0\}$, $f'_{\text{goal}}((q,i)) = f_{\text{goal}}(q)$ if $q \in Q_f$, and $f'_{\text{goal}}(r) = +\infty$, $\eta'((q,i)) = \eta(q)$, $\eta'(q_{\min}) = \eta(q_{\min})$, $P'((q,i)) = P(q)$ for every $(q,i) \in Q'_c \cup Q'_u$. The set of transitions is

$$\begin{aligned} T' = &\{(q,i) \xrightarrow{g,R} (q',i) \mid q \xrightarrow{g,R} q', \text{and } q,q' \neq q_{\min}\} \\ &\cup \{(q,0) \xrightarrow{g,R} q_{\min},\ (q,1) \xrightarrow{g,R} r \mid (q \xrightarrow{g,R} q_{\min}) \in T\} \\ &\cup \{q_{\min} \xrightarrow{g,R} (q',1) \mid (q_{\min} \xrightarrow{g,R} q') \in T\}. \end{aligned}$$

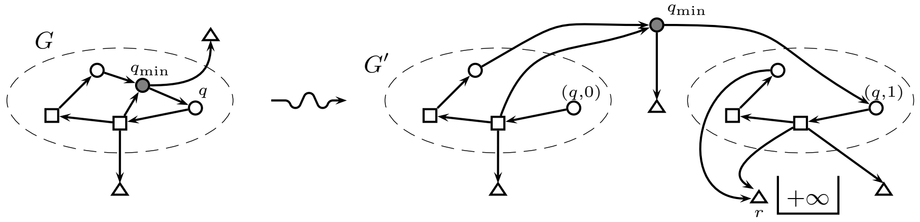


Fig. 5. Case $q_{\min}$ (in grey) controllable

We prove the following lemma, which establishes properties H1 and H2.

Lemma 5. *For every $(q,x) \in (Q \setminus \{q_{\min}\}) \times [0,1]$, we have $\mathit{OptCost}_G(q,x) = \mathit{OptCost}_{G'}((q,0),x)$. For every $x \in [0,1]$, $\mathit{OptCost}_G(q_{\min},x) = \mathit{OptCost}_{G'}(q_{\min},x)$.*

It remains to prove property H3. We fix the integer N' for G'. We fix some $\varepsilon > 0$, and take $\varepsilon' = \frac{\varepsilon}{3}$. We take σ' an (ε', N')-acceptable strategy in G'. We then define σ as follows:

$$\sigma(q,x) = \begin{cases} \sigma'((q,1),x) & \text{if } \mathsf{Cost}_{G'}(\sigma',((q,1),x)) \leq \mathsf{OptCost}_{G'}(q_{\min},x) \\ \sigma'((q,0),x) & \text{otherwise} \end{cases} \tag{1}$$

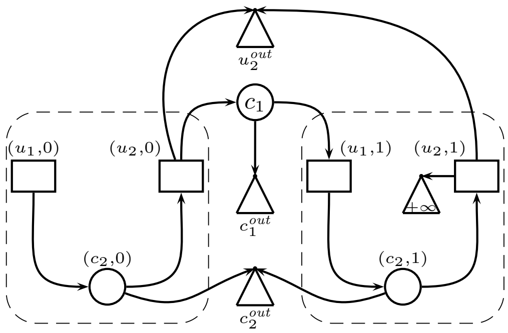


Fig. 6. Running example after unwinding.

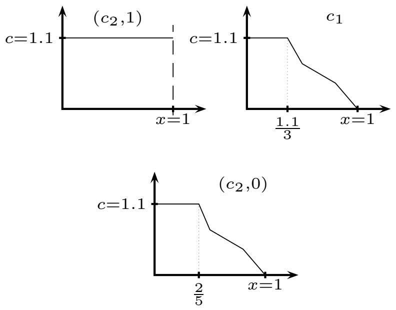


Fig. 7. Optimal costs.

Example. Returning to the running example of Fig. 1 with u_1 and u_2 urgent, performing the above transformation with respect to c_1 gives the PTG_f depicted in Fig. 6. The optimal cost functions are depicted in Fig. 7 and the resulting winning strategy for c_2 is, according to (1), the strategy of $(c_2,1)$ when $x \leq \frac{1.1}{3}$ and $(c_2,0)$ otherwise. □

Obviously, the strategy σ is memoryless. We need to establish that the function $x \mapsto \mathsf{Cost}_G(\sigma,(q,x))$ consists of at most N pieces, and that σ is ε-optimal.

Proposition 6. *Strategy σ is winning and there exists a fixed (independant of ε) integer N such that σ is (ε,N)-acceptable.*

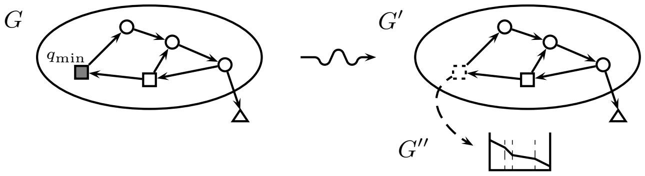


Fig. 8. When it is uncontrollable, $q_{\min}$ is made urgent (in dash line here).

Case: $q_{\min}$ is uncontrollable. The intuition is that the opponent will prefer delays in other locations than $q_{\min}$ whenever possible. We attempt to enforce this by a transformation of the game where location $q_{\min}$ is urgent, as depicted in Fig. 8. Formally, given a $[0,1]$-PTG$_f$ without resets G, we define G' with $Q'_{\mathrm{urg}} = Q_{\mathrm{urg}} \cup \{q_{\min}\}$ and $Q'_u = Q_u \backslash \{q_{\min}\}$.

Obviously enough, since we restrict the possible moves for the opponent in G', we have for every state (q,x), $\mathsf{OptCost}_{G'}(q,x) \leq \mathsf{OptCost}_G(q,x)$.

However, the converse inequality is not correct over $[0,1]$, and we will need a more complex construction to handle this case. We now explain how to iteratively compute the optimal costs in G. Fig. 9 gives an overview of the computation described below.

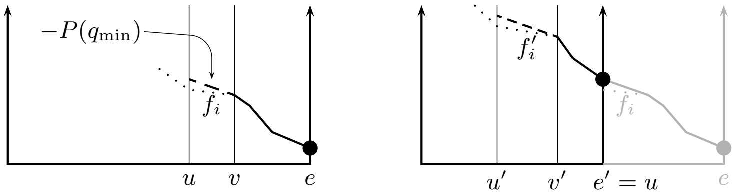


Fig. 9. Successive computations when $q_{\min}$ is uncontrollable

Clearly, we can compute $\mathsf{OptCost}_G(q_{\min},1)$ (indeed, $\mathsf{OptCost}_G(q_{\min},1) = \mathsf{OptCost}_{G'}(q_{\min},1)$, since when $x=1$, time cannot elapse any more and the same moves are available in G' and in G). This initializes our iterative computation.

Now, assume we can compute $\mathsf{OptCost}_G(q_{\min},e)$ for some $e \in [0,1]$. We can apply the induction hypotheses H1—H3 to G'. In particular, $f\colon x \in [0,e] \mapsto \mathsf{OptCost}_{G'}(q_{\min},x)$ is a cost function satisfying the requirements of item H2. Writing f_1, ..., f_n for the successive affine functions constituting f, we pick the smallest index i such that for every $j > i$, function f_j has slope less than or equal to $-P(q_{\min})$. If $i > 0$, we note $[u,v]$ the domain of f_i (see Fig. 9).

Lemma 7. *If $i = 0$, for all $(q,x) \in Q \times [0,e]$, $\mathit{OptCost}_G(q,x) = \mathit{OptCost}_{G'}(q,x)$. If $i > 0$, for all $(q,x) \in Q \times [v,e]$, $\mathit{OptCost}_G(q,x) = \mathit{OptCost}_{G'}(q,x)$.*

We now explain how to compute $\mathsf{OptCost}_G(q_{\min},x)$ for $x \in [u,v]$; we prove the following lemma:

Lemma 8. *If $i > 0$, then for all $(q, x) \in Q \times [u, v]$, we have $\mathsf{OptCost}_G(q_{\min}, x) = (v - x)P(q_{\min}) + f(v)$.*

The optimal cost in states (q, x) with $x \in [u, v]$ can then be computed by considering the PTG_f G'', restricted to $x \in [u, v]$, and obtained from G' by making $q_{\min}$ a goal location with cost function equal to $x \mapsto \mathsf{OptCost}_G(q_{\min}, x)$, which is then viewed as an outside cost function, see Fig. 9.

We can then repeat the procedure above on the interval $[0, u]$ (*i.e.* by setting $e = u$): compute $f' \colon x \mapsto \mathsf{OptCost}_{G'}(q_{\min}, x)$ with $x \in [0, u]$, select an interval $[u', v']$ where f'_i has slope larger than or equal to $-P(q_{\min})$, and so on, replace that part with an affine function with slope $-P(q_{\min})$, and continue with the interval $[0, u']$. We now explain why this process terminates: since they have slopes strictly greater than $-P(q_{\min})$, f_i and f'_i are fragments of outside cost functions, according to hypothesis H2. If they have different slopes, then they are obviously parts of two different fragments of outside cost functions. If they have the same slopes, then they are fragments of two different parts of outside cost functions, since they are joined by affine functions with slopes less than (or equal to $-P(q_{\min})$). Since there are only finitely many affine functions constituting the outside cost functions, our procedure terminates.

At each step of the procedure above, we can also compute (ε, N)-acceptable strategies, and merge them.

6 Conclusion

In this paper we have proven that optimal cost for arbitrary priced timed games with one clock is a computable problem, and that ε-optimal memoryless strategies may effectively be obtained. The complexity of our procedure is quite high, running in 3-EXPTIME, while the best known lower bound for this problem is PTIME. Our future works of course include tightening these bounds.

As a consequence of our result it may be shown that the iterative semi-algorithm proposed in [10] always terminates for priced timed games with one clock. Cost functions cost^i_G are inductively defined, which for any location $q \in Q$ and any clock value v, give the optimal cost of winning from the state (q, v) within at most i steps (we count the number of steps in a run ρ by the number of delay-and-action fractions). Now Theorem 4 ensures that we can find a fixed N such that for *any* $\varepsilon > 0$ we can compute an (ε, N)-acceptable strategy. In particular this guarantees that we can find ε-optimal strategies which are guaranteed to win within $N \cdot |Q|$ steps for any $\varepsilon > 0$. Consequently, $\langle \mathsf{cost}^i_G \rangle_{i=1}^{\infty}$ (the semi-algorithm of [10]) converges after at most $N \cdot |Q|$ iterations to the optimal cost of winning. A prototype implementation of this iterative algorithm is available at `http://www.cs.aau.dk/~illum/tools/1ptga/`.

As future work we would like to determine what happens with priced timed games using two clocks, but this seems really difficult as our approach heavily relies on the fact that there is only one clock.

References

1. Y. Abdeddaïm, E. Asarin, and O. Maler. Scheduling with timed automata. 2006.
2. R. Alur, M. Bernadsky, and P. Madhusudan. Optimal reachability in weighted timed games. In *Proc. 31st Intl Coll. Automata, Languages and Programming (ICALP'04)*, LNCS 3142, p. 122–133. Springer, 2004.
3. R. Alur, C. Courcoubetis, and Th. A. Henzinger. Computing accumulated delays in real-time systems. In *Proc. 5th Intl Conf. Computer Aided Verification (CAV'93)*, LNCS 697, p. 181–193. Springer, 1993.
4. R. Alur and D. Dill. A theory of timed automata. *Theor. Comp. Science*, 126(2):183–235, 1994.
5. R. Alur, S. La Torre, and G. J. Pappas. Optimal paths in weighted timed automata. In *Proc. 4th Intl Workshop Hybrid Systems: Computation and Control (HSCC'01)*, LNCS 2034, p. 49–62. Springer, 2001.
6. G. Behrmann, A. Fehnker, Th. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. Vaandrager. Minimum-cost reachability for priced timed automata. In *Proc. 4th Intl Workshop Hybrid Systems: Computation and Control (HSCC'01)*, LNCS 2034, p. 147–161. Springer, 2001.
7. P. Bouyer, Th. Brihaye, and N. Markey. Improved undecidability results on weighted timed automata. *Inf. Proc. Letters*, (5):188–194, June 2006.
8. P. Bouyer, E. Brinksma, and K. G. Larsen. Staying alive as cheaply as possible. In *Proc. 7th Intl Workshop Hybrid Systems: Computation and Control (HSCC'04)*, LNCS 2993, p. 203–218. Springer, 2004.
9. P. Bouyer, E. Brinksma, and K. G. Larsen. Optimal infinite scheduling for multi-priced timed automata. *Form. Meth. in Syst. Design*, 2006. To appear.
10. P. Bouyer, F. Cassez, E. Fleury, and K. G. Larsen. Optimal strategies in priced timed game automata. In *Proc. 24th Conf. Foundations of Software Technology & Theoretical Computer Science (FST&TCS'04)*, LNCS 3328, p. 148–160. Springer, 2004.
11. Th. Brihaye, V. Bruyère, and J.-F. Raskin. On optimal timed strategies. In *Proc. 3rd Intl Conf. Formal Modeling and Analysis of Timed Systems (FORMATS'05)*, LNCS 3821, p. 49–64. Springer, 2005.
12. S. La Torre, S. Mukhopadhyay, and A. Murano. Optimal-reachability and control for acyclic weighted timed automata. In *Proc. 2nd IFIP Intl Conf. Theoretical Computer Science (IFIPTCS'02)*, IFIP Conf. Proc. 223, p. 485–497. Kluwer, 2002.
13. K. G. Larsen, G. Behrmann, E. Brinksma, A. Fehnker, Th. Hune, P. Pettersson, and J. Romijn. As cheap as possible: Efficient cost-optimal reachability for priced timed automata. In *Proc. 13th Intl Conf. Computer Aided Verification (CAV'01)*, LNCS 2102, p. 493–505. Springer, 2001.
14. K. G. Larsen and J. I. Rassmussen. Optimal conditional reachability for multi-priced timed automata. In *Proc. 8th Intl Conf. Foundations of Software Science and Computation Structures (FoSSaCS'05)*, LNCS 3441, p. 234–249. Springer, 2005.
15. J. I. Rasmussen, K. G. Larsen, and K. Subramani. Resource-optimal scheduling using priced timed automata. In *Proc. 10th Intl Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'04)*, LNCS 2988, p. 220–235. Springer, 2004.
16. UPPAAL CORA. `http://www.cs.aau.dk/~behrmann/cora/`, Jan. 2006.