On Optimal Timed Strategies

Thomas Brihaye[1], Véronique Bruyère[1], and Jean-François Raskin[2]

[1] Faculté des Sciences, Université de Mons-Hainaut,
Avenue du Champ de Mars 6, B-7000 Mons, Belgium

[2] Département d'Informatique, Université Libre de Bruxelles,
Boulevard du Triomphe CP 212, B-1050-Bruxelles, Belgium

Abstract. In this paper, we study timed games played on weighted timed automata. In this context, the reachability problem asks if, given a set T of locations and a cost C, Player 1 has a strategy to force the game into T with a cost less than C no matter how Player 2 behaves. Recently, this problem has been studied independently by Alur et al and by Bouyer et al. In those two works, a semi-algorithm is proposed to solve the reachability problem, which is proved to terminate under a condition imposing the non-zenoness of cost. In this paper, we show that in the general case the existence of a strategy for Player 1 to win the game with a bounded cost is undecidable. Our undecidability result holds for weighted timed game automata with five clocks. On the positive side, we show that if we restrict the number of clocks to one and we limit the form of the cost on locations, then the semi-algorithm proposed by Bouyer et al always terminates.

1 Introduction

Weighted timed automata are an extension of timed automata with costs : each discrete transition has an associated non-negative integer cost to be paid when the transition is taken, and each location has an associated cost rate that has to be paid with respect to the time spent in the location. If the most important problem for timed automata is *reachability*, the natural extension for weighted timed automata is *optimal cost reachability*, that is, given an initial state, what is the minimum cost to be paid to reach a given location. This problem has been solved independently in [6] and [8]. The complexity of this problem is similar to the complexity of classical reachability in timed automata [3]. The more general problem of model-checking on weighted timed automata is investigated in [10].

Timed automata and weighted timed automata are models for closed systems, where every transition is controlled. If we want to distinguish between actions of a *controller* and actions of an *environment* we have to consider *timed games* on those formalisms. In one round of the timed game played on a timed automaton, Player 1 (the controller) chooses an action a and a time $t \geq 0$, Player 2 (the environment) updates the state of the automaton either by playing an uncontrollable action at time $t' \leq t$ or by playing the action a at time t as proposed by Player 1. We say that Player 1 has a winning *strategy* to reach a set

T of target locations if it can force Player 2 to update the automaton in a way that the control of the automaton eventually reaches a location of T. When the timed game is played on a weighted timed automaton, we can ask if Player 1 can force Player 2 to update the control of the automaton in a way to reach T with a cost bounded by a given value. We can also ask to compute the optimal cost for Player 1 winning such a game.

While games on timed automata are already well studied, see for example [11], [1] and [2], and are known to be decidable, only preliminary results about games on weighted timed automata are known. First results on reachability with an optimal cost appear in [7], where the cost is equal the time spent to reach a target location in a timed automaton. Optimal reachabitility is aslo studied in [13] with any costs and weighted automata that are acyclic. In [4], Alur et al study the k-bounded optimal game reachability problem, i.e. given an initial state s of a weighted timed automaton $\mathcal{A}$, a cost bound C and a set T of locations, determine if Player 1 has a strategy to enforce the game started in state s into a location of T within k rounds, while ensuring that the cost is bounded by C. Their algorithmic solution has an exponential-time worst case complexity. In [9], the authors study winning strategies to reach a set of target locations with an optimal cost in a weighted timed automaton $\mathcal{A}$. To compute the optimal cost and to synthetize an optimal winning strategy, they provide a semi-algorithm for which they can guarantee the termination under a condition called *strict non-zenoness of cost*. This condition imposes that every cycle in the region automaton of $\mathcal{A}$ has a cost bounded away from zero. The general case where this condition is not imposed, is left open in both papers [4] and [9].

In this paper, we consider timed games played on a weighted timed automaton as they are introduced in [4], and following the lines of [9] we study the two problems of the existence of a winning strategy with a bounded cost, and of the existence of a winning strategy with an optimal cost (Section 2). We prove the unexpected negative result that for weighted timed automata, the existence of a winning strategy with a cost bounded by a given value is undecidable (Section 3, Theorem 1). The proof is based on a reduction of the halting problem for two-counter machines. The weighted timed automaton simulating the two-counter machine has five clocks and a cost rate equal to 0 or 1 on the locations. On the positive side, we show that if we restrict the number of clocks to one and we limit the cost rate to 0 or d where d is a fixed integer, then the two problems mentioned above are decidable (Section 4, Corollary 3). The proof follows the approach of [9] but we can prove the termination of their semi-algorithm without the non-zenoness of cost hypothesis.

2 Timed games

In this section, we recall the notion of timed game on a weighted timed automaton as it is defined in [4]. In this context we introduce the concept of winning strategy and the related cost problems as mentioned in [9]. We begin with the definition of weighted timed automaton.

2.1 Weighted timed automata

Let X be a finite set of clocks. Let $\mathbb{R}_+$ be the set of all non-negative reals and let $\mathbb{N}$ be the set of all non-negative integers. A clock valuation is a map $\nu : X \to \mathbb{R}_+$. The set of constraints over X, denoted $G(X)$, is the set of boolean combinations of constraints of the form $x \sim \alpha$ or $x - y \sim \alpha$ where $x, y \in X$, $\alpha \in \mathbb{N}$, and $\sim \in \{<, \leq, =, \geq, >\}$. The way a clock valuation ν over X satisfies a constraint g over X is defined naturally; it is denoted by $\nu \models g$.

Definition 1. *A* weighted timed automata*, WTA for short, is a tuple* $\mathcal{A} = (L, L_F, X, \Sigma, \delta, \mathit{Inv}, W_L, W_\delta)$ *where* L *is a finite set of locations,* $L_F \subseteq L$ *is a set of target locations,* Σ *is a finite set of actions that contains the special symbol* u*,* $\delta \subseteq L \times \Sigma \times G(X) \times 2^X \times L$ *is a transition relation,* $\mathit{Inv} : L \to G(X)$ *is an invariant function,* $W_L : L \to \mathbb{N}$ *gives the cost for each location, and* $W_\delta : \delta \to \mathbb{N}$ *gives the cost for each transition.*

For a transition $e = (l, a, g, Y, l') \in \delta$, the label of e is a, and it is denoted by $\mathsf{Action}(e)$. Transitions labeled with u model *uncontrolled* transitions. The other ones are the *controlled* transitions.

A *state* of $\mathcal{A}$ is a pair $q = (l, \nu)$ where $l \in L$ is a location and ν is a valuation over X. Let Q denote the set of all states. For a clock valuation ν and a value $t \in \mathbb{R}_+$, $\nu + t$ denotes the clock valuation ν' where $\nu'(x) = \nu(x) + t$, for each $x \in X$. For any clock valuation ν, and any subset of clocks $Y \subseteq X$, $\nu[Y := 0]$ denotes the clock valuation ν' such that $\nu'(x) = \nu(x)$ for any $x \in X \setminus Y$ and $\nu'(x) = 0$ for any $x \in Y$.

A *timed transition* in $\mathcal{A}$ is of the form $(l, \nu) \to^t (l, \nu+t)$, where $(l, \nu), (l, \nu+t) \in Q$, $t \in \mathbb{R}_+$, and $\nu + t' \models \mathit{Inv}(l)$ for every t', $0 \leq t' \leq t$. A *discrete transition* in $\mathcal{A}$ is of the form $(l, \nu) \to^e (l', \nu')$ where e is a transition $(l, a, g, Y, l') \in \delta$ such that $\nu \models \mathit{Inv}(l)$, $\nu \models g$, $\nu' = \nu[Y := 0]$ and $\nu' \models \mathit{Inv}(l')$.

In this paper, without loss of generality, we make the assumption that a WTA $\mathcal{A}$ is *c-deterministic*, i.e. if $q \to^e q'$ and $q \to^{e'} q''$ with e, e' two controlled transitions such that $\mathsf{Action}(e) = \mathsf{Action}(e')$, then $q' = q''$.

Hypothesis 1. A WTA $\mathcal{A}$ is supposed to be c-deterministic.

A *run* ρ of a WTA $\mathcal{A}$ is a finite or infinite sequence of alternating timed and discrete transitions

$$\rho = q_1 \to^{t_1} q_1' \to^{e_1} q_2 \to^{t_2} q_2' \to^{e_2} \cdots \to^{t_k} q_k' \to^{e_k} q_{k+1} \cdots .$$

The run ρ is also denoted as $q_1 \to^{t_1 \cdot e_1} q_2 \to^{t_2 \cdot e_2} \cdots \to^{t_k \cdot e_k} q_{k+1} \cdots$. When ρ is the finite run $q_1 \to^{t_1 \cdot e_1} \cdots \to^{t_k \cdot e_k} q_{k+1}$, with $q_i = (l_i, \nu_i)$ for each i, we define the *cost* $W(\rho)$ of ρ as

$$W(\rho) = \sum_{i=1}^{k} W_L(l_i) \cdot t_i + \sum_{i=1}^{k} W_\delta(e_i).$$

2.2 Timed games and related cost problems

We now present the notion of timed game on a WTA and some related problems.

The *timed game* on a WTA $\mathcal{A} = (L, L_F, X, \Sigma, \delta, Inv, W_L, W_\delta)$ is played by two players, *Player 1* (the *controller*) and *Player 2* (the *environment*). Let $\Sigma_u = \Sigma \setminus \{u\}$. At any state q, Player 1 picks a time t and an action $a \in \Sigma_u$ such that there is a transition $q \rightarrow^{t \cdot e} q'$ with $\mathsf{Action}(e) = a$. Player 2 has two choices:

- either it can wait for time t', $0 \leq t' \leq t$, and execute a transition $q \rightarrow^{t' \cdot e'} q''$ with $\mathsf{Action}(e') = u$,
- or it can decide to wait for time t and execute the[1] transition $q \rightarrow^{t \cdot e} q'$ proposed by Player 1.

The game then evolves to a new state (according to the choice of Player 2) and the two players proceed to play as before.

Comments 1. In the definition of a timed game, it is implicitly supposed that Player 1 can always formulate a choice (t, a) in any reachable state q of the game.

We present the concept of strategy. A *(Player 1) strategy* is a function $\lambda : Q \mapsto \mathbb{R}_+ \times \Sigma_u$. A finite or infinite run $\rho = q_1 \rightarrow^{t_1 \cdot e_1} q_2 \rightarrow^{t_2 \cdot e_2} \cdots \rightarrow^{t_k \cdot e_k} q_{k+1} \cdots$ is said to be played[2] *according to* λ if for every i, if $\lambda(q_i) = (t'_i, a_i)$, then either $t_i \leq t'_i$ and $\mathsf{Action}(e_i) = u$, or $t_i = t'_i$ and $\mathsf{Action}(e_i) = a_i$. The run ρ is *winning* if for some i, $q_i = (l_i, \nu_i)$ with $l_i \in L_F$ being a target location. Suppose that q_i is the first state of ρ such that $l_i \in L_F$, and let ρ' be the prefix run of ρ equal to $q_1 \rightarrow^{t_1 \cdot e_1} \cdots \rightarrow^{t_{i-1} \cdot e_{i-1}} q_i$. Then we say that $W(\rho')$ is *the cost of* ρ *to reach* L_F and we abusively denote it by $W(\rho)$. Given a state q and a strategy λ, we define $\mathsf{Outcome}(q, \lambda)$ as the set of runs starting from q and played according to λ. The strategy λ is *winning* from state q if all runs of $\mathsf{Outcome}(q, \lambda)$ are winning.

Finally, we define two notions of cost as proposed in [9], and we state the problems studied in this paper. The *cost* $\mathsf{Cost}(q, \lambda)$ associated with a winning strategy λ and a state q is defined by

$$\mathsf{Cost}(q, \lambda) = \sup\{W(\rho) \mid \rho \in \mathsf{Outcome}(q, \lambda)\}.$$

Intuitively, the presence of the supremum is explained by the fact that Player 2 tries to make choices that lead to cost $W(\rho)$ as large as possible. The *optimal cost* $\mathsf{OptCost}(q)$ is then equal to

$$\mathsf{OptCost}(q) = \inf\{\mathsf{Cost}(q, \lambda) \mid \lambda \text{ is a winning strategy}\}.$$

A winning strategy λ from q is called *optimal* whenever $\mathsf{Cost}(q, \lambda) = \mathsf{OptCost}(q)$.

Problem 1. Given a WTA $\mathcal{A}$, a state q of $\mathcal{A}$ and a constant $c \in \mathbb{N}$, decide if there exists a winning strategy λ from q such that $\mathsf{Cost}(q, \lambda) \leq c$.

[1] Recall that $\mathcal{A}$ is assumed to be c-deterministic.

[2] This definition is from [4]. A third condition appears in the definition given in [9],[11].

Problem 2. Given a WTA $\mathcal{A}$ and a state q of $\mathcal{A}$, determine the optimal cost $\mathsf{OptCost}(q)$ and decide whether there exists an optimal winning strategy.

Comments 2. Concerning Problem 2, there is an optimal winning strategy from state q iff the infimum can be replaced by a minimum in the definition of $\mathsf{OptCost}(q)$. Notice that Problem 1 is decidable if Problem 2 can be solved. Indeed, there exists a winning strategy λ from q such that $\mathsf{Cost}(q, \lambda) \leq c$ iff either $\mathsf{OptCost}(q) < c$, or $\mathsf{OptCost}(q) = c$ and there is an optimal strategy from q.

3 Undecidability results

This section is devoted to the main result of this article, that is, Problems 1 is undecidable. By Comments 2, it follows Problem 2 cannot be solved.

Theorem 1. *Problem 1 is undecidable.*

Proof. The idea of the proof is the following one. Given a two-counter machine M, we will construct a WTA $\mathcal{A}$ and propose a timed game on $\mathcal{A}$. In this game, Player 1 will simulate the execution of M, and Player 2 will observe the possible simulation errors done by Player 1. We will prove that for a well-chosen state q, there exists a winning strategy λ from q with $\mathsf{Cost}(q, \lambda) \leq 1$ iff the machine M halts. It will follow that Problem 1 is undecidable.

We here consider the classical model of two-counter machine [12]. The two counters are denoted by c_1 and c_2, and the different types of labeled instructions are given in Table 1.[3] A *configuration* of the machine M is given by a triple

zero test	k : if $c_i = 0$ then goto k' else goto k''
increment	$k : c_i := c_i + 1$
decrement	$k : c_i := c_i - 1$
stop	k : `STOP`

Table 1. The possible instructions of a two-counter machine.

(k, c_1, c_2) which represents the (label of the) current instruction of M and two counter values. The first instruction of M is supposed to be labeled by k_0 and the stop instruction for which M halts, is supposed to be labeled by k_s. The initial configuration of M is thus $(k_0, 0, 0)$.

We first define how the counter values are encoded in the states of $\mathcal{A}$. We encode the value of counter c_1 using three clocks x_1, y_1, z_1 and the value of counter c_2 using three clocks x_2, y_2, z_2[4]. The clock values are always between 0 and 1. To keep the notation simple, we use the same notation to denote the clock or its value. When clear from the context, we often drop the subscript, that is, counter c is described by clocks x, y and z. Counter c_i, $i = 1, 2$, has value $n \in \mathbb{N}$,

$$c_i = n \tag{1}$$

[3] We assume that there is a zero test before each decrementation instruction such that the counter value is not modified each time it is equal to zero.

[4] An encoding using five clocks is possible, but the exposition would be more technical.

iff one of the following three conditions is satisfied :

- $0 \le x_i \le y_i \le z_i \le 1$, $\quad y_i - x_i = \frac{1}{2^{n+1}}$, $\quad$ and $x_i + (1 - z_i) = \frac{1}{2^{n+1}}$,
- $0 \le z_i \le x_i \le y_i \le 1$, $\quad y_i - x_i = \frac{1}{2^{n+1}}$, $\quad$ and $x_i - z_i = \frac{1}{2^{n+1}}$,
- $0 \le y_i \le z_i \le x_i \le 1$, $\quad (1 - x_i) + y_i = \frac{1}{2^{n+1}}$, $\quad$ and $x_i - z_i = \frac{1}{2^{n+1}}$.

The first condition is given in Figure 1.[5] We say that the encoding is in *normal form* if $x_i = 0$ (see Figure 2).

Fig. 1. One among the three encodings of $c_1 = n$, with $\alpha + \beta = \frac{1}{2^{n+1}}$.

Fig. 2. The encoding of $c_1 = n$ in normal form.

The automaton $\mathcal{A} = (L, L_F, X, \Sigma, \delta, Inv, W_L, W_\delta)$ has thus a set X of six clocks (x_i, y_i and z_i, $i = 1, 2$). The costs given by function W_L to the locations are either 0 or 1. The function W_δ assigns a null cost to each transition.[6] The set L contains a location for each label k of the machine M, which is labeled by σ_k in a way to remember the label k. For each such k, the related location l is as depicted in Figure 3 where i is equal to 1 or 2. We notice that the control spends no time in location l, and that one of the two counters, c_i, is encoded in normal form. This is the way configurations (k, c_1, c_2) of the machine M are encoded by states (l, ν) of the automaton $\mathcal{A}$ with locations l like in Figure 3. In particular, the stop instruction of M which is labeled by k_s is encoded by a location l like in Figure 3, such that σ_{k_s} replaces σ_k and $l \in L_F$ is a target location.

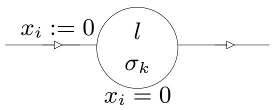

Fig. 3. Location labeled by σ_k

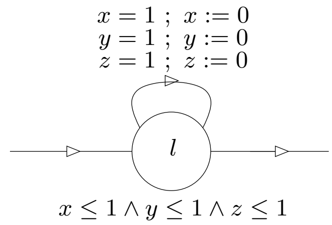

Fig. 4. Widget to let the value of a counter unchanged.

In the sequel, we present *widgets* used by Player 1 to simulate the instructions of the machine M. These widgets are fragments of the automaton $\mathcal{A}$; they are depicted in Figures 4–11. In these figures, target locations $l \in L_F$ are surrounded by a double circle, uncontrolled transitions are labeled by the action u, and controlled transitions are those that are not labeled. It is supposed that controlled transitions leaving a given location are labeled by distinct actions of Σ_u, in a way to have a c-deterministic WTA $\mathcal{A}$ (see Hypothesis 1). Notice that the constructed automaton $\mathcal{A}$ will satisfy the assumptions of Comments 1.

[5] The two other conditions are cyclic– or mod 1, representations of the first condition.
[6] In the following figures, the cost if not indicated is supposed to be equal to zero.

With the construction of these widgets and a particular state q of $\mathcal{A}$, we will see that the machine M halts iff Player 1 has a winning strategy λ from q with $\mathsf{Cost}(q, \lambda) \leq 1$. Let us describe this idea, the complete proof will be given later:

- If M halts, then the strategy of Player 1 is to faithfully simulate the instructions of M. If Player 2 lets Player 1 playing, then the cost of simulating M equals 0, otherwise the cost equals 1. In both cases the game always reaches a target location. This shows that λ is a winning strategy with $\mathsf{Cost}(q, \lambda) \leq 1$.
- Suppose that M does not halt. Either the timed game simulates the instructions of M and thus never finishes. Or it does not simulate the instructions of M and Player 2 is able to force the game to reach a target location with a cost strictly greater than 1. Therefore in both cases, Player 1 has no winning strategy λ with $\mathsf{Cost}(q, \lambda) \leq 1$.

Widget W_1 *to let a counter value unchanged* - The first widget allows, when time elapses in a location l, to keep the value of counter c unchanged. Such a widget is useful when, for instance, the value of one counter is incremented while the value of the other counter is not modified. See Figure 4. If the control enters location l at time t with clock values x, y, z encoding the value n of counter c, and leave location l at time $t'' \geq t$, then for all t', $t \leq t' \leq t''$, the current clock values x', y', z' still encode the value n. Indeed the clock values cyclically rotate among the three possible conditions for encoding n (see (1)).

The widget W_1 is often useful in combination with other widgets. To keep the figures of those widgets readable, we often omit widget W_1 inside them.

Widget W_2 *for normal form* - Figure 5 presents a widget to put a counter encoding in normal form. When the control enters location l with clocks values x, y, z encoding the value n of counter c, the control reaches location l' with x, y, z encoding n and $x = 0$. The control instantaneoulsy leaves location l' due to the invariant $x = 0$.

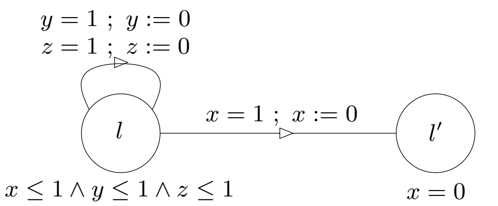

Fig. 5. Widget to put a counter encoding in normal form.

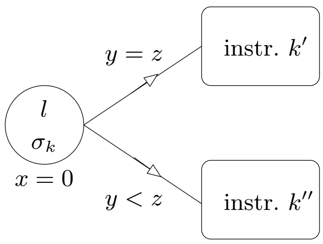

Fig. 6. Widget for zero test.

Widget W_3 *for zero test* - We here indicate how to simulate a zero test instruction, i.e. an instruction k : if $c = 0$ then goto k' else goto k''. The widget for zero test is given in Figure 6. We assume that the control reaches location l with the value n of counter c encoded by x, y, z in normal form[7], that is, $x = 0$, $y = \frac{1}{2^{n+1}}$ and

[7] This is always possible by using widget W_2.

$z = 1 - \frac{1}{2^{n+1}}$. We notice that location l is like locations described in Figure 3. No time can elapse in l. Clearly to test that $n = 0$ is equivalent to test that $y = z$ as done in this widget.

Widget W_4 *for increment* - In this paragraph, we indicate how to simulate an increment instruction $k : c := c + 1$. While the previous widgets have controlled transitions only, and null costs on every location, the widget for incrementing counter c uses two uncontrolled transitions, and have cost equal to 1 for certain locations. This widget is composed of several parts.

(1) First part of widget W_4.
Consider Figure 7. We can suppose that the control reaches location l_0 with the

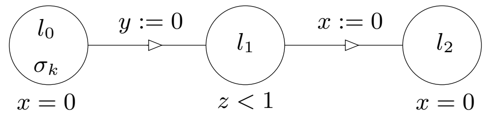

Fig. 7. First part of the widget for increment.

value n of counter c encoded by x, y, z in normal form, such that $x = 0$, $y = \frac{1}{2^{n+1}}$ and $z = 1 - \frac{1}{2^{n+1}}$. The transition from l_0 to l_1 has to be taken immediately. As the transition from l_1 to l_2 is controlled, Player 1 has to choose the amount of time t that it waits in l_1 before taking the transition to l_2. Because of the invariant labeling l_1, we know that $t < \frac{1}{2^{n+1}}$. When entering location l_2, the clock values are as follows: $x = 0$, $y = t$ and $z = 1 - \frac{1}{2^{n+1}} + t$. Note that to faithfully simulate the increment of counter c, Player 1 should choose $t = \frac{1}{2^{n+2}}$. It is easy to verify that in location l_2,

$$t = \frac{1}{2^{n+2}} \Leftrightarrow y + z = 1. \tag{2}$$

So, we are in the following situation: to verify that Player 1 has faithfully chosen t to simulate the increment of counter c, we simply have to check that in l_2, $y + z = 1$. Hereafter, we show how Player 2 observes in location l_2 the possible simulation errors of Player 1. Notice that in l_2, the clock values x, y, z satisfy $0 = x < y < z \leq 1$.

(2) Part of widget W_4 to check if $y + z \neq 1$.
For clarity, we distinguish the case where (i) $y + z > 1$ from the case where (ii) $y + z < 1$. We begin with Case (i). The widget $\mathsf{W}^{>}$ is given in Figure 8. Notice that the first location of this widget is equal to the last one of the widget of Figure 7, and that the first transition is uncontrolled. Location l_7 is a target location, i.e. $l_7 \in L_F$. The idea is as follows: we use the cost $W(\rho)$ of the run ρ from l_2 to l_7 to compute the value $y + z$. The cost of each location is null except for locations l_4 and l_6 where $W_L(l_4) = 1$ and $W_L(l_6) = 1$. Let ρ be a run from l_2 to l_7 such that y and z are clock values in l_2. Recall that in location l_2, the clock values x, y, z satisfy $0 = x < y < z \leq 1$. We can verify that the cost of ρ is

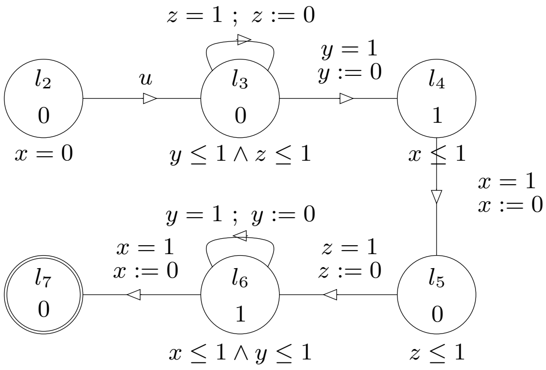

Fig. 8. Widget $\mathsf{W}^{>}$.

equal to $y + z$ (a cost y in location l_4 and a cost z in location l_6). Hence we have

$$y + z > 1 \Leftrightarrow W(\rho) > 1. \tag{3}$$

We now consider Case (ii). The widget $\mathsf{W}^{<}$ is given in Figure 9. As for widget $\mathsf{W}^{>}$ the first location of this widget is equal to location l_2 of Figure 7, and the first transition is uncontrolled. Location l'_6 is a target location. The idea is similar to Case (i) : along the run ρ' from l_2 to l'_6, the value n of counter c is left unchanged, and the cost of ρ' is equal to $(1 - y) + (1 - z)$ (a cost $1 - y$ in l'_3 and a cost $1 - z$ in l'_5). As $y + z < 1$ is equivalent to $(1 - y) + (1 - z) > 1$, then

$$y + z < 1 \Leftrightarrow W(\rho') > 1. \tag{4}$$

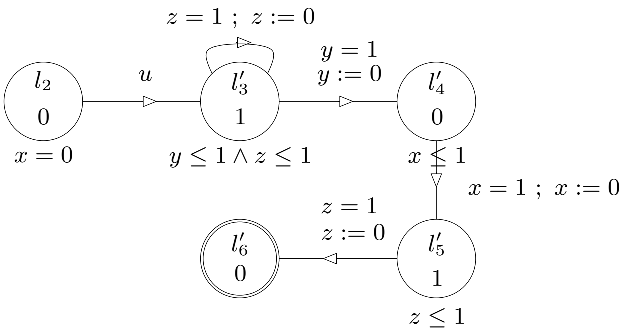

Fig. 9. Widget $\mathsf{W}^{<}$.

(3) Complete widget for increment.
The complete widget for increment is composed of the widgets given in Figures 7, 8 and Figure 9, as it is schematically given in Figure 10. The counter that we want to increment has value n. First the control enters the first part of the widget for incrementation with $x = 0$, $y = \frac{1}{2^{n+1}}$, $z = 1 - \frac{1}{2^{n+1}}$. As we have seen before, Player 1 has to choose the amount of time t that it waits in l_1 before taking the transition to l_2. The only way to reach l_2 with $y + z = 1$ is to simulate faithfully the increment of the counter (see (2)). Then in location l_2, Player 1 proposes to

Player 2 to move the control to the widget that encodes the next instruction of the machine M. Player has three choices: either accept the move of Player 1, or move the control to the widget $\mathsf{W}^{>}$, or move the control to the widget $\mathsf{W}^{<}$.

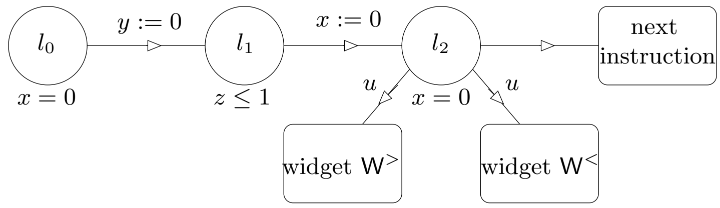

Fig. 10. Widget W_4 for increment.

So, looking at Figure 10, the situation is as follows. Suppose that Player 1 faithfully simulates the increment instruction, i.e. $y + z = 1$ (see (2)). Either Player 2 lets the game evolving to the next instruction of M, and the cost remains null. Or it decides to use one of the two widgets $\mathsf{W}^{>}$, $\mathsf{W}^{<}$, and the game reaches a target location with a cost equal to 1 (see (3) and (4)). So whatever the Player 2's decision, the cost is bounded by 1. Suppose now that Player 1 does not simulate the increment instruction, i.e. $y + z \neq 1$, then Player 2 can take a decision such that the game reaches a target location with a cost strictly greater than 1. Indeed, if $y + z > 1$, it decides to use the widget $\mathsf{W}^{>}$ (see (3)), otherwise it uses the widget $\mathsf{W}^{<}$ (see (4)).

Widget W_5 *for decrement* - As for the increment, the widget for decrement is in several parts. We only present the first part in details, where Player 1 has to faithfully simulate the decrement. The other parts where Player 2 observes the possible errors of Player 1 are identical to Cases (i), (ii) of the increment widget.

Let us assume that we enter location l_0 of the widget of Figure 11 with $x = 0$, $y = \frac{1}{2^{n+1}}$ and $z = 1 - \frac{1}{2^{n+1}}$. We also assume that $n > 1$ (see footnote 3).

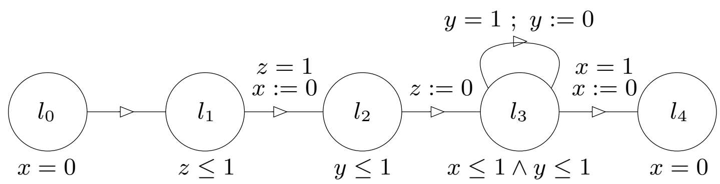

Fig. 11. First part of the widget for decrement.

When the control leaves location l_1, the clock values are respectively equal to $x = 0$, $z = 1$, and $y = \frac{1}{2^{n+1}} + \frac{1}{2^{n+1}}$. Then Player 1 has to choose the amount of time t that it waits in location l_2 before taking the transition to l_3. To faithfully simulate the decrement, Player 1 should choose $t = \frac{1}{2^n}$. In location l_4, we are now in the same situation as in location l_2 of the increment widget (see Figure 10): $t = \frac{1}{2^n} \Leftrightarrow y + z = 1$. So, we just have to plug in l_4 the two widgets $\mathsf{W}^{>}$, $\mathsf{W}^{<}$ and a transition to the next instruction of the machine M. The situation is the same as for the increment. Indeed if Player 1 faithfully simulates the decrement instruction, then the cost is bounded by 1 whatever the Player 2's decision. If Player 1 does not simulate it, then Player 2 can take a decision such that the game reaches a target location with a cost strictly greater than 1.

It should now be clear why we can reduce the halting of a two-counter machine to the existence of a winning strategy for Player 1 to reach a target location with a cost bounded by 1. Let M be a two-counter machine and $\mathcal{A}$ the WTA constructed from the widgets as above. The target locations of $\mathcal{A}$ are either the location associated with the stop instruction of M, or the target locations of the widgets of Figures 8 and 9. Let $q = (l, \nu)$ be the state of $\mathcal{A}$ encoding the initial configuration $(k_0, 0, 0)$ of M, that is, l is the location labeled by σ_{k_0}, and ν is the clock valuation such that $x_1 = x_2 = 0$ and $y_1 = z_1 = y_2 = z_2 = \frac{1}{2}$. Let us prove that M halts iff there exists a winning strategy λ from q with $\mathsf{Cost}(q, \lambda) \leq 1$.

Suppose that M halts, then the strategy λ of Player 1 is to faithfully simulate the instructions of M. Let ρ be a run of $\mathsf{Outcome}(q, \lambda)$. If along ρ, Player 2 lets Player 1 simulating M, then ρ reaches the target location of $\mathcal{A}$ associated with the stop instruction of M with a cost $W(\rho) = 0$. If Player 2 decides to use one of the two widgets $\mathsf{W}^{>}$, $\mathsf{W}^{<}$, then ρ reaches the target location of this widget with $W(\rho) = 1$. Therefore, λ has a winning strategy from q satisfying $\mathsf{Cost}(q, \lambda) \leq 1$.

Suppose that there is a winning strategy λ from q with $\mathsf{Cost}(q, \lambda) \leq 1$. Assume that M does not halt, the contradiction is obtained as follows. If λ consists in simulating the instructions of M, then Player 2 decides to let Player 1 simulating M. The corresponding run $\rho \in \mathsf{Outcome}(q, \lambda)$ will never reach a target location since M does not halt. This is impossible since λ is winning. Thus suppose that λ does not simulate the instructions of M, and let $\rho \in \mathsf{Outcome}(q, \lambda)$. As soon as Player 2 observes a simulation error along ρ, it decides to use one of the widgets $\mathsf{W}^{>}$, $\mathsf{W}^{<}$ such that ρ reaches the target location of this widget with $W(\rho) > 1$. This is impossible since λ is winning with a cost $\mathsf{Cost}(q, \lambda) \leq 1$. □

4 Symbolic analysis of timed games

4.1 The Pre operator

In order to symbolically analyse timed games, we present a controllable predecessor operator. The main result is Proposition 1 relating the iteration of this operator with the existence of a winning strategy with a bounded cost. The content of this section is close to [9], but with a different and simpler presentation.[8]

Let $\mathcal{A} = (L, L_F, X, \Sigma, \delta, Inv, W_L, W_\delta)$ be a WTA. An *extended state* of $\mathcal{A}$ is a tuple (l, ν, w) where $l \in L$ is a location, ν is a clock valuation over X, and $w \in \mathbb{R}_+$ is called the *credit*. Intuitively, the credit models a sufficient amount of resource that allows Player 1, when in state (l, ν), to reach a target location of L_F whatever Player 2 decides to do, with a cost less than or equal to w. The set of extended states is denoted by Q_E.

We now define the following Pre operator.

[8] In [9], timed games on WTA's are reduced to games on linear hybrid automata where the cost is one of the variables.

Definition 2. *Let $\mathcal{A}$ be a WTA and $R \subseteq Q_E$. Then $(l, \nu, w) \in \mathsf{Pre}(R)$ iff there exist $t \in \mathbb{R}_+$ and a controlled transition $e \in \delta$ such that*

- *there exists an extended state $(l', \nu', w') \in R$, with $(l, \nu) \to^{t \cdot e} (l', \nu')$, and $w \geq w' + W_L(l) \cdot t + W_\delta(e)$,*
- *and for every t', $0 \leq t' \leq t$, every uncontrolled transition $e' \in \delta$, and every state (l', ν') such that $(l, \nu) \to^{t' \cdot e'} (l', \nu')$, there exists an extended state $(l', \nu', w') \in R$ with $w \geq w' + W_L(l) \cdot t' + W_\delta(e')$.*

The Pre operator satisfies the following nice properties. Given a WTA $\mathcal{A}$, we define the set $\mathsf{Goal} = \{(l, \nu, w) \mid l \in L_F \text{ and } w \geq 0\}$, and the set

$$\mathsf{Pre}^*(\mathsf{Goal}) = \bigcup_{k \geq 0} \mathsf{Pre}^k(\mathsf{Goal}).^9$$

A set $R \subseteq Q_E$ of extended states is said *upward closed* if whenever $(l, \nu, w) \in R$, then $(l, \nu, w') \in R$ for all $w' \geq w$.

Lemma 1. *1. For all $R \subseteq Q_E$, the set $\mathsf{Pre}(R)$ is upward closed.*
2. The set Goal and $\mathsf{Pre}^*(\mathsf{Goal})$ are upward closed.

Proposition 1. *Let $\mathcal{A}$ be a WTA. Then $(l, \nu, w) \in \mathsf{Pre}^*(\mathsf{Goal})$ iff there exists a winning strategy λ from state $q = (l, \nu)$ such that $\mathsf{Cost}(q, \lambda) \leq w$.*

Proposition 1 leads to several comments in the case a symbolic representation[10] for $\mathsf{Pre}^*(\mathsf{Goal})$ can be computed. In such a case, we say that $\mathsf{Pre}^*(\mathsf{Goal})$ has an *effective representation.*

Comments 3. By Proposition 1, Problem 1 is decidable if (i) $\mathsf{Pre}^*(\mathsf{Goal})$ has an effective representation, and (ii) the belonging of an extended state (l, ν, w) to $\mathsf{Pre}^*(\mathsf{Goal})$ can be effectively checked. We now know from Theorem 1 that one of the conditions (i), (ii) cannot be fulfilled in general.

Comments 4. Let $\mathcal{A}$ be a WTA and $q = (l, \nu)$ be a state of $\mathcal{A}$. Problem 2 asks to determine the optimal cost $\mathsf{OptCost}(q)$. This is possible under the following hypotheses: (i) $\mathsf{Pre}^*(\mathsf{Goal})$ has an effective representation, (ii) the value $\inf\{w \mid (l, \nu, w) \in \mathsf{Pre}^*(\mathsf{Goal})\}$ can be effectively computed. This value is exactly $\mathsf{OptCost}(q)$.

Moreover the existence of an optimal winning strategy from q is decidable if one can determine the value $c = \mathsf{OptCost}(q)$, and the belonging of (l, ν, c) to $\mathsf{Pre}^*(\mathsf{Goal})$ can be effectively checked. Indeed, an optimal strategy exists iff c is the minimum value of the set $\{w \mid (l, \nu, w) \in \mathsf{Pre}^*(\mathsf{Goal})\}$ (see Comments 2).

In [9], Problem 2 has been solved for the class of WTA's $\mathcal{A}$ such that the cost function of is strictly non-zeno, i.e. every cycle in the region automaton associated with $\mathcal{A}$ has a cost which is bounded away from zero. The authors of

[9] For $k = 0$, $\mathsf{Pre}^k(\mathsf{Goal}) = \mathsf{Goal}$, and for $k > 0$, $\mathsf{Pre}^k(\mathsf{Goal}) = \mathsf{Pre}\left(\mathsf{Pre}^{k-1}(\mathsf{Goal})\right)$.

[10] For instance this representation could be given in a decidable logical formalism like the first-order theory of the reals with order and addition.

this paper translate Problem 2 into some linear hybrid automata where the cost is one of the variables. For this class of hybrid automata, the conditions mentioned above in these comments are fulfilled. Of course the automaton we have contructed in the proof of Theorem 1 does not fall into this class of automata.

4.2 One clock

In Section 3, Problem 1 was shown undecidable by a reduction of the halting problem of a two-counter machine. The WTA in the proof uses five clocks, has no cost on the transitions and cost 0 or 1 on the locations. We here study WTA's with one clock and such that for any location l, $W_L(l) \in \{0, d\}$ with $d \in \mathbb{N}$ a given constant. For this particular class of automata, we solve Problem 2 by following the lines of Comments 4. By Comments 2, Problem 1 is thus also solved. The proof is only detailed for $d = 1$.

To facilitate the computation of the Pre operator, we first introduce another operator denoted by π, that is largely inspired from the one of [9]. We need to generalize some notation to extended states: a timed transition $(l, \nu) \rightarrow^t (l', \nu')$ is extended to $(l, \nu, w) \rightarrow^t (l, \nu', w - W_L(l) \cdot t)$, similarly with $(l, \nu) \rightarrow^e (l', \nu')$ extended to $(l, \nu, w) \rightarrow^e (l', \nu', w - W_\delta(e))$. Given $R \subseteq Q_E$ and $a \in \Sigma$ we define

$$\mathsf{Pre}^a(R) = \{r \in Q_E \mid \exists r' \in R \text{ such that } r \rightarrow^e r' \text{ with } \mathsf{Action}(e) = a\},$$

as well as $\mathsf{cPre}(R) = \cup_{a \in \Sigma_u} \mathsf{Pre}^a(R)$, and $\mathsf{uPre}(R) = \mathsf{Pre}^u(R)$. We also define the following set tPre(R,S), with $R, S \subseteq Q_E$. Intuitively, an extended state r is in $\mathsf{tPre}(R, S)$ if from r we can reach r' by time elapsing and along the timed transition from r to r' we avoid S. This set is defined by

$$\mathsf{tPre}(R, S) = \big\{r \in Q_E \mid \exists t \in \mathbb{R}_+ \text{ with } r \rightarrow^t r', r' \in R, \text{ and } \mathsf{Post}_{[0,t]}(s) \subseteq \overline{S}\big\}$$

where $\mathsf{Post}_{[0,t]}(s) = \{r' \in Q_E \mid \exists t', 0 \leq t' \leq t, \text{ such that } r \rightarrow^{t'} r'\}$. The new operator π is then defined by :

$$\pi(R) = \mathsf{tPre}\big(\mathsf{cPre}(R), \mathsf{uPre}(\overline{R})\big). \tag{5}$$

The next lemmas indicate useful properties of the various operators.

Lemma 2. *1.* $\mathsf{cPre}(R_1 \cup R_2) = \mathsf{cPre}(R_1) \cup \mathsf{cPre}(R_2)$,
2. $\mathsf{uPre}(R_1 \cup R_2) = \mathsf{uPre}(R_1) \cup \mathsf{uPre}(R_2)$,
3. $\mathsf{tPre}(R_1 \cup R_2, S) = \mathsf{tPre}(R_1, S) \cup \mathsf{Pre}_t(R_2, S)$,
4. $\mathsf{tPre}(R, S_1 \cup S_2) = \mathsf{tPre}(R, S_1) \cap \mathsf{Pre}_t(R, S_2)$.

Lemma 3. *1. If $R \subseteq Q_E$ is upward closed, then $\pi(R) = \mathsf{Pre}(R)$.*
2. $\mathsf{Pre}^*(\mathsf{Goal}) = \pi^*(\mathsf{Goal})$.

We now study WTA's $\mathcal{A}$ with one clock x, such that $W_L(l) \in \{0, 1\}$ for every location l. Let C be the largest constant used in the guards of $\mathcal{A}$. As done in [5] for timed automata, we define an equivalence relation on Q_E in order to obtain a partition of this set.

Definition 3. *Let* (ν, w), $(\nu', w') \in \mathbb{R}_+^2$*. Then* $(\nu, w) \sim (\nu', w')$ *if the following conditions hold.*

1. *Either* $\lfloor \nu \rfloor = \lfloor \nu' \rfloor$, *or* $\nu, \nu' > C$; $\quad \lfloor w \rfloor = \lfloor w' \rfloor$;
2. *For* $\nu, \nu' \leq C$, *fract*$(\nu) = 0$ *iff fract*$(\nu') = 0$; $\quad$ *fract*$(w) = 0$ *iff fract*$(w') = 0$;
3. *For* $\nu, \nu' \leq C$, *fract*$(\nu) +$ *fract*$(w) \sim 1$ *iff fract*$(\nu') +$ *fract*$(w') \sim 1$, *with* $\sim \in \{<, =, >\}$.

An example of equivalence relation $\sim$ is given in Figure 12. We extend the relation $\sim$ to Q_E by defining $(l, \nu, w) \sim (l', \nu', w')$ iff $l = l'$ and $(\nu, w) \sim (\nu', w')$. Let $\mathcal{P}$ be the partition of Q_E obtained with this relation.

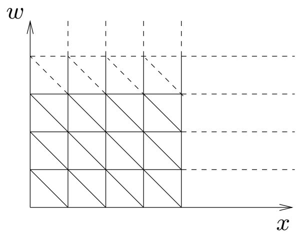

Fig. 12. The relation $\sim$ with $C = 4$.

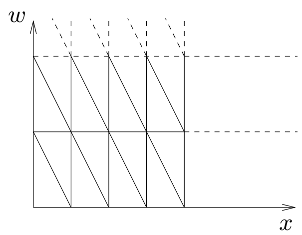

Fig. 13. The partition $\mathcal{P}_2$

The partition $\mathcal{P}$ is *stable* under π, that is, given $R \in \mathcal{P}$, $\pi(R)$ is a union of equivalence classes of $\mathcal{P}$. The reader could convince himself as follows. Let $R \in \mathcal{P}$. Clearly, the sets $\mathsf{cPre}(R)$ and $\mathsf{uPre}(R)$ are union of equivalences classes of $\mathcal{P}$. Now due to Lemma 2, it remains to check that given $R, S \in \mathcal{P}$, the set $\mathsf{tPre}(R, S)$ is a union of equivalence classes taking into account that $W_L(l) \in \{0, 1\}$. We summarize this result in the next lemma.

Lemma 4. $\mathcal{P}$ *is stable under* π.

By this lemma, the next corollary is straightforward since Goal is a union of equivalence classes of $\mathcal{P}$ and by Lemmas 1 and 3.

Corollary 1. *The set* $\mathsf{Pre}^*(\mathsf{Goal})$ *is a union of equivalence classes of* $\mathcal{P}$*. Given a state* q *of* $\mathcal{A}$*, the optimum cost* $\mathsf{OptCost(q)}$ *is a non-negative integer*[11].

Even if the proposed partition $\mathcal{P}$ is infinite, we are able to prove that the computation of $\mathsf{Pre}^*(\mathsf{Goal})$ terminates. We first define the *set* $\mathsf{Up}(\mathcal{P})$ *of upward closed sets w.r.t.* $\mathcal{P}$: $\mathsf{Up}(\mathcal{P}) = \{R \mid R = \cup R_i, R_i \in \mathcal{P} \text{ and } R \text{ is upward closed}\}$.

Lemma 5. *The partially ordered set* $\langle \mathsf{Up}(\mathcal{P}), \supseteq \rangle$ *is Artinian*[12].

Corollary 2. $\mathsf{Pre}^*(\mathsf{Goal})$ *can be effectively computed.*

Looking at Comments 4, we get the next corollary.

[11] It is possible to find an example of WTA with two clocks and an optimum cost which is rational.

[12] Every decreasing chain is finite.

Corollary 3. *Let $\mathcal{A}$ be a WTA with one clock and such that $W_L(l) \in \{0, 1\}$ for all locations l. Then Problems 1 and 2 can be solved.*

Comments 5. The arguments given in this section are easily extended to a cost function $W_L(l) \in \{0, d\}$ for any location l, where $d \geq 1$ is a fixed integer. The same approach holds but with a partition $\mathcal{P}_d$ different from $\mathcal{P}$. This partition is similar to $\mathcal{P}$, except that we only need horizontal lines of the form $w = d \cdot n$ (with $n \in \mathbb{N}$) and each anti-diagonal of the form $x + w = c$ is removed and replaced by the lines of equations $d \cdot x + w = d \cdot n$ (with $n \in \mathbb{N}$). See Figure 13.

References

1. L. de Alfaro and T.A. Henzinger and R. Majumdar. Symbolic algorithms for infinite-state games. In *Proceedings of CONCUR'01*, *Lect. Notes Comput. Sci.* **2154**, 536–550, Springer, 2001.
2. L. de Alfaro and M. Faella and T.A. Henzinger and R. Majumdar and M. Stoelinga". The element of surprise in timed games". In *Proceedings of CONCUR'03*, *Lect. Notes Comput. Sci.* **2761**, 144–158, Springer, 2003.
3. R. Alur and P. Madhusudan. Decision problems for timed automata: A survey. In *Proceedings of SFM'04*, *Lect. Notes Comput. Sci.* **3185**, 1–24, Springer, 2004.
4. R. Alur, M. Bernadsky, and P. Madhusudan. Optimal reachability for weighted timed games. In *Proceedings of ICALP'04*, *Lect. Notes Comput. Sci.* **3142**, 122–133, Springer, 2004.
5. R. Alur and D.L. Dill. A theory of timed automata. *Theoret. Comput. Sci.* **126**, 183–235, 1994.
6. R. Alur, S. La Torre, and G.J. Pappas. Optimal paths in weighted timed automata. In *Proceedings of HSCC'01*, *Lect. Notes Comput. Sci.* **2034**, 49–62, Springer, 2001.
7. E. Asarin and O. Maler. As soon as possible: Time optimal control for timed automata. In *Proceedings of HSCC'99*, *Lect. Notes Comput. Sci.* **1569**, 19–30, Springer, 1999.
8. G. Behrmann, A. Fehnker, T. Hune, K. Larsen, P. Pettersson, J. Romijn, and F. Vaandrager. Minimum-cost reachability for priced timed automata. In *Proceedings of HSCC'01*, *Lect. Notes Comput. Sci.* **2034**, 147–161. Springer-Verlag, 2001.
9. P. Bouyer, F. Cassez, E. Fleury, and K.G. Larsen. Optimal strategies in priced timed game automata. In *Proceedings of FSTTCS'04*, **3328** *Lect. Notes Comput. Sci.* **3328**, 148–160, Springer, 2004.
10. T. Brihaye, V. Bruyère, and J.-F. Raskin. Model-Checking for Weighted Timed Automata. In *Proceedings of FORMATS-FTRTFT'04*, *Lect. Notes Comput. Sci.* **3253** , 277–292, Springer, 2004.
11. O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Proceedings of STACS'95*, *Lect. Notes Comput. Sci.* **900**, 229–242, Springer, 1995.
12. M. Minsky. *Computation: Finite and Infinite Machines.* Prentice Hall, 1967.
13. S. La Torre, S. Mukhopadhyay, and A. Murano. Optimal-reachability and control for acyclic weighted timed automata. In *Proceedings of IFIP TCS'02*, *IFIP Conference Proceedings* **223**, 485–497, Kluwer, 2002.