



Inapproximability in Weighted Timed Games

Quentin Guilmant   

Max Planck Institute for Software Systems, Saarland Informatics Campus, Germany

Joël Ouaknine   

Max Planck Institute for Software Systems, Saarland Informatics Campus, Germany

Abstract

We consider two-player, turn-based weighted timed games played on timed automata equipped with (positive and negative) integer weights, in which one player seeks to reach a goal location whilst minimising the cumulative weight of the underlying path. Although the *value problem* for such games (is the value of the game below a given threshold?) is known to be undecidable, the question of whether one can *approximate* this value has remained a longstanding open problem. In this paper, we resolve this question by showing that approximating arbitrarily closely the value of a given weighted timed game is computationally unsolvable.

2012 ACM Subject Classification D.2.4

Keywords and phrases Weighted timed games, undecidability, approximation

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

1 Introduction

Weighted timed games are zero-sum games played by two players on a timed automaton equipped with weights, where one player seeks to reach a goal location whilst minimising the cumulative weight. Such games have been extensively studied for over two decades; we refer the reader to the recent and comprehensive article [10] for a thorough overview of the state of the art.

As noted in [10, Sec. 12], *a longstanding open problem is the approximation of weighted timed games, i.e., whether one can compute an arbitrarily close approximation of the value of a given game.*

Our main result is the following:

► **Theorem 1.** *Given a two-player, turn-based, weighted timed game with integer weights, the problem of approximating its value arbitrarily closely is computationally unsolvable.*

An important open question is whether this result can be extended to timed games in which only non-negative integer weights are allowed.

2 Weighted Timed Games

Let \mathcal{X} be a finite set of **clocks**. **Clock constraints** over \mathcal{X} are expressions of the form $x \sim n$ or $x - y \sim n$, where $x, y \in \mathcal{X}$ are clocks, $\sim \in \{<, \leq, =, \geq, >\}$ is a comparison symbol, and $n \in \mathbb{N}$ is a natural number. We write \mathcal{C} to denote the set of all clock constraints over \mathcal{X} . A **valuation** on \mathcal{X} is a function $\nu : \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$. For $d \in \mathbb{R}_{\geq 0}$ we denote by $\nu + d$ the valuation such that, for all clocks $x \in \mathcal{X}$, $(\nu + d)(x) = \nu(x) + d$. Let $X \subseteq \mathcal{X}$ be a subset of all clocks. We write $\nu[X := 0]$ for the valuation such that, for all clocks $x \in X$, $\nu[X := 0](x) = 0$, and $\nu[X := 0](y) = \nu(y)$ for all other clocks $y \notin X$. For $C \subseteq \mathcal{C}$ a set of clock constraints over \mathcal{X} , we say that the valuation ν **satisfies** C , denoted $\nu \models C$, if and only if all the comparisons in C hold when replacing each clock x by its corresponding value $\nu(x)$.

► **Definition 2.** *A (turn-based) weighted timed game \mathcal{G} is a tuple $\mathcal{G} = (L_{\text{Min}}, L_{\text{Max}}, G, \mathcal{X}, T, w)$, where:*



© Quentin Guilmant and Joël Ouaknine;
licensed under Creative Commons License CC-BY 4.0

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:12



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

23:2 Inapproximability in Weighted Timed Games

- L_{Min} and L_{Max} are the (disjoint) sets of **locations** belonging to Players Min and Max respectively; we let $L = L_{\text{Min}} \cup L_{\text{Max}}$ denote the set of all locations. (In drawings, locations belonging to Min are depicted by blue circles, and those belonging to Max are depicted by red squares.)
- $G \subseteq L_{\text{Min}}$ are the **goal locations**.
- \mathcal{X} is a set of clocks.
- $T \subseteq (L \setminus G) \times 2^C \times 2^X \times L$ is a set of (**discrete**) **transitions**. A transition $\ell \xrightarrow{C,X} \ell'$ enables moving from location ℓ to location ℓ' , provided all clock constraints in C are satisfied, and afterwards resetting all clocks in X to zero.
- $w : (L \setminus G) \cup T \rightarrow \mathbb{Z}$ is a **weight function**.

In the above, we assume that all data (set of locations, set of clocks, set of transitions, set of clock constraints) are finite.

► **Remark 3.** The weight function w associates integer weights to each discrete transition and each non-goal location. It is worth pointing out that in our proof of inapproximability (Thm. 1), only *transitions* may carry negative weights; all *locations* have weights in \mathbb{N} .

Let $\mathcal{G} = (L_{\text{Min}}, L_{\text{Max}}, G, \mathcal{X}, T, w)$ be a weighted timed game. A **configuration** over \mathcal{G} is a pair (ℓ, ν) , where $\ell \in L$ and ν is a valuation on \mathcal{X} . Let $d \in \mathbb{R}_{\geq 0}$ be a **delay** and $t = \ell \xrightarrow{C,X} \ell' \in T$ be a discrete transition. One then has a **delayed transition** (or simply a **transition** if the context is clear) $(\ell, \nu) \xrightarrow{d,t} (\ell', \nu')$ provided that $\nu + d \models C$ and $\nu' = (\nu + d)[X := 0]$. Intuitively, control remains in location ℓ for d time units, after which it transitions to location ℓ' , resetting all the clocks in X to zero in the process. The **weight** of such a delayed transition is $d \cdot w(\ell) + w(t)$, taking account both of the time spent in ℓ as well as the weight of the discrete transition t .

As noted in [10], without loss of generality one can assume that no configuration (other than those associated with goal locations) is deadlocked; in other words, for any location $\ell \in L \setminus G$ and valuation $\nu \in \mathbb{R}_{\geq 0}^{\mathcal{X}}$, there exists $d \in \mathbb{R}_{\geq 0}$ and $t \in T$ such that $(\ell, \nu) \xrightarrow{d,t} (\ell', \nu')$.¹

Let $k \in \mathbb{N}$. A **run** ρ of length k over \mathcal{G} from a given configuration (ℓ_0, ν_0) is a sequence of matching delayed transitions, as follows:

$$\rho = (\ell_0, \nu_0) \xrightarrow{d_0, t_0} (\ell_1, \nu_1) \xrightarrow{d_1, t_1} \dots \xrightarrow{d_{k-1}, t_{k-1}} (\ell_k, \nu_k).$$

The **weight** of ρ is the cumulative weight of the underlying delayed transitions:

$$\text{weight}(\rho) = \sum_{i=0}^{k-1} (d_i \cdot w(\ell_i) + w(t_i)).$$

An infinite run ρ is defined in the obvious way; however, since no goal location is ever reached, its weight is defined to be infinite: $\text{weight}(\rho) = +\infty$.

A run is **maximal** if it is either infinite or cannot be extended further. Thanks to our deadlock-freedom assumption, finite maximal runs must end in a goal location. We refer to maximal runs as **plays**.

¹ The authors of [10] point out that this assumption can be enforced by examining the region graph of the timed game \mathcal{G} , and adding new transitions towards sink locations as required. In the pictorial representations of timed game fragments that appear later in this paper, in the interest of clarity we omit such extraneous transitions and locations; we merely assume instead that neither player allows him- or herself to end up in a deadlocked situation, unless a goal location has been reached.

We now define the notion of **strategy**. Recall that locations of \mathcal{G} are partitioned into sets L_{Min} and L_{Max} , belonging respectively to Players Min and Max. Let Player $P \in \{\text{Min}, \text{Max}\}$, and write $\mathcal{FR}_{\mathcal{G}}^P$ to denote the collection of all non-maximal finite runs of \mathcal{G} ending in a location belonging to Player P . A **strategy** for Player P is a mapping $\sigma_P : \mathcal{FR}_{\mathcal{G}}^P \rightarrow \mathbb{R}_{\geq 0} \times T$ such that for all finite runs $\rho \in \mathcal{FR}_{\mathcal{G}}^P$ ending in configuration (ℓ, ν) with $\ell \in L_P$, the delayed transition $(\ell, \nu) \xrightarrow{d,t} (\ell', \nu')$ is valid, where $\sigma_P(\rho) = (d, t)$ and (ℓ', ν') is some configuration (uniquely determined by $\sigma_P(\rho)$ and ν).

Let us fix a starting configuration (ℓ_0, ν_0) , and let σ_{Min} and σ_{Max} be strategies for Players Min and Max respectively (one speaks of a *strategy profile*). We write $\text{play}_{\mathcal{G}}((\ell_0, \nu_0), \sigma_{\text{Min}}, \sigma_{\text{Max}})$ to denote the unique maximal run starting from configuration (ℓ_0, ν_0) and unfolding according to the strategy profile $(\sigma_{\text{Min}}, \sigma_{\text{Max}})$: in other words, for every strict finite prefix ρ of $\text{play}_{\mathcal{G}}((\ell_0, \nu_0), \sigma_{\text{Min}}, \sigma_{\text{Max}})$ in $\mathcal{FR}_{\mathcal{G}}^P$, the delayed transition immediately following ρ in $\text{play}_{\mathcal{G}}((\ell_0, \nu_0), \sigma_{\text{Min}}, \sigma_{\text{Max}})$ is labelled with $\sigma_P(\rho)$.

Recall that the objective of Player Min is to reach a goal location through a play whose weight is as small possible. Player Max has an opposite objective, trying to avoid goal locations, and, if not possible, to maximise the cumulative weight of any attendant play. This gives rise to the following two symmetrical definitions:

$$\begin{aligned} \overline{\text{Val}}_{\mathcal{G}}(\ell_0, \nu_0) &= \inf_{\sigma_{\text{Min}}} \left\{ \sup_{\sigma_{\text{Max}}} \left\{ \text{weight}(\text{play}_{\mathcal{G}}((\ell_0, \nu_0), \sigma_{\text{Min}}, \sigma_{\text{Max}})) \right\} \right\} \text{ and} \\ \underline{\text{Val}}_{\mathcal{G}}(\ell_0, \nu_0) &= \sup_{\sigma_{\text{Max}}} \left\{ \inf_{\sigma_{\text{Min}}} \left\{ \text{weight}(\text{play}_{\mathcal{G}}((\ell_0, \nu_0), \sigma_{\text{Min}}, \sigma_{\text{Max}})) \right\} \right\}. \end{aligned}$$

$\overline{\text{Val}}_{\mathcal{G}}(\ell_0, \nu_0)$ represents the smallest possible weight that Player Min can possibly achieve, starting from configuration (ℓ_0, ν_0) , against best play from Player Max, and conversely for $\underline{\text{Val}}_{\mathcal{G}}(\ell_0, \nu_0)$: the latter represents the largest possible weight that Player Max can enforce, against best play from Player Min.² As noted in [10], turned-based weighted timed games are *determined*, and therefore $\overline{\text{Val}}_{\mathcal{G}}(\ell_0, \nu_0) = \underline{\text{Val}}_{\mathcal{G}}(\ell_0, \nu_0)$ for any starting configuration (ℓ_0, ν_0) ; we denote this common value by $\text{Val}_{\mathcal{G}}(\ell_0, \nu_0)$.

► **Remark 4.** Note that $\text{Val}_{\mathcal{G}}(\ell_0, \nu_0)$ can take on real numbers, or either of the values $-\infty$ and $+\infty$. Our proof of inapproximability, however, only makes use of games having finite values.

3 Inapproximability

3.1 Probabilistic Finite Automata

We establish value inapproximability for weighted timed games by reducing from an unsolvable approximation problem for probabilistic automata. We start with some definitions.

► **Definition 5 (PFA).** A (two-letter) **probabilistic automaton** is a tuple $\mathcal{A} = (Q, q_1, F, A_a, A_b)$, where:

- $Q = \{q_1, \dots, q_\ell\}$ is a finite set of **states**.
- $q_1 \in Q$ is the **initial state**.
- $F \subseteq Q$ are the **accepting states**.

² Technically speaking, these values may not be literally achievable; however given any $\varepsilon > 0$, both players are guaranteed to have strategies that can take them to within ε of the optimal value.

23:4 Inapproximability in Weighted Timed Games

- $A_a, A_b \in ([0, 1] \cap \mathbb{Q})^{\ell \times \ell}$ are left stochastic **transition matrices** corresponding to letters a and b respectively.³

Given such a probabilistic automaton \mathcal{A} , any word $w \in \{a, b\}^*$ induces a probability distribution on Q , as follows. For the empty word λ , we let the distribution $\mathbb{D}(\lambda) = (1, 0, \dots, 0)^T$, i.e., initially all the probability mass lies in the initial state q_1 . Suppose now that the distribution on Q upon reading word w is $\mathbb{D}(w)$, i.e., the probability $\mathbb{P}_w(q_i)$ of being in state q_i after reading w is precisely the i th component of $\mathbb{D}(w)$. We then let $\mathbb{D}(wa) = A_a \mathbb{D}(w)$ and $\mathbb{D}(wb) = A_b \mathbb{D}(w)$.

Finally, for any word $w \in \{a, b\}^*$, we write $\mathcal{A}(w) = \mathbb{P}_w(F) = \sum_{q \in F} \mathbb{P}_w(q)$ to denote the probability that the automaton \mathcal{A} accepts word w .

The key result we need (the main ingredient of which is due to Condon and Lipton [11]) is the following [15, Thm. 3.3]:

► **Theorem 6.** *There exists an algorithm which takes a Turing machine TM as input and outputs a two-letter probabilistic automaton \mathcal{A} satisfying the following:*

- *if TM does not accept the empty string, then \mathcal{A} accepts no word with probability exceeding $1/10$, and*
- *if TM does accept the empty string, then \mathcal{A} accepts some word with probability at least $1/2$.*

Theorem 6 states, in effect, that the maximum probability with which a given probabilistic automaton accepts some word cannot in general be approximated.⁴ In the remainder of this section, we show how to exploit this fact to establish that the value of a given weighted time game is, in turn, also not approximable in general.

3.2 Reduction Overview

Let probabilistic automaton $\mathcal{A} = (Q, q_1, F, A_a, A_b)$, with $Q = \{q_1, \dots, q_\ell\}$, be fixed for the rest of this paper. Without loss of generality, we may assume that all non-zero probabilistic transitions have weight $1/M$, for some constant $M \in \mathbb{N}$.⁵ In other words, $A_a, A_b \in \{0, 1/M\}^{\ell \times \ell}$.

Players Min and Max will play a weighted timed game \mathcal{G} representing the evolution of \mathcal{A} as it reads a word $w \in \{a, b\}^*$. Min will choose the letters of w , and will simulate running this word through \mathcal{A} , seeking to minimise the cumulative weight of the underlying path in \mathcal{G} . As long as Min faithfully simulates the behaviour of \mathcal{A} , the cumulative weight will remain constant. Any error or ‘cheating’ by Min, however, will be ‘punishable’ by Max in the form of an increase in the cumulative weight, the size of which will be proportional to the magnitude of the error. Naturally, as Max seeks to maximise the cumulative weight of the path, the dominant strategy for him will always be to seek to extract as large a cost as possible.

To this end, the game \mathcal{G} will be equipped with two sets of clocks:

- $Z = \{z_1, \dots, z_\ell\} \cup \{z_F\}$; intuitively, for $i \in \{1, \dots, \ell\}$, z_i is intended to store the current value of the probability of being in state q_i , and z_F is intended to store the probability of being in one of the accepting states in F .
- $X_{\text{Min}} = \{\mu_1, \dots, \mu_\ell\} \cup \{\mu_F\}$; intuitively, for $i \in \{1, \dots, \ell\}$, μ_i is Min’s guess of the value to which to update clock z_i next, and likewise for μ_F and z_F .

³ Left stochasticity means that each column of the matrix sums to 1.

⁴ Technically speaking, we should speak of a *supremum*.

⁵ This can straightforwardly be achieved via an increase in the number of states of \mathcal{A} .

In addition, \mathcal{G} has use of an auxiliary clock t to ensure proper synchronisation, etc.

The game unfolds through a cycle of modules, as follows (see also Fig. 5):

1. Min chooses a letter (a or b) to append to the word that has been played so far.
2. Min compiles her guesses as to how the resulting probabilities of being in each state (q_1 to q_ℓ) should then be updated, and stores the corresponding values in clocks μ_1, \dots, μ_ℓ (and in μ_F in the collection of states in F). In so doing, the game infrastructure ensures that clocks z_1, \dots, z_ℓ and z_F remain untouched (their values at the beginning and at the end of the relevant module are the same).
3. Max extracts a cost (an increase to the cumulative weight) for every gap between the values guessed by Min and the actual freshly computed values of the clocks in Z , according to the transition matrices of \mathcal{A} .
4. The aforementioned gaps are then erased, by updating each of the clocks in Z to assume the value of its counterpart in X_{Min} .
5. Finally, before looping back, Min is given the opportunity to reach the goal location of \mathcal{G} ; this transition is however only available if $z_F \geq 1/2$.

► **Remark 7.** Note in the above that \mathcal{G} contains a transition in which a clock is compared to $1/2$ (rather than an integer). This is clearly easily circumvented by considering an equivalent weighted timed game in which all constants have been multiplied by 2. We opted for the present half-integer formulation as this enables clock values directly to represent probabilities, rather than twice the corresponding probabilities.

Assuming that \mathcal{A} does accept some word w with probability at least $1/2$, Min need not make any error; she only has to guess correctly each letter of w in turn, along with the corresponding exact distribution updates, and eventually z_F will rise to $1/2$ or above, allowing her to reach the goal location at zero cumulative cost.

On the other hand, if no word is accepted by \mathcal{A} with probability exceeding $1/10$, then Min will be forced to make errors in order to enable μ_F , and thus in turn z_F , to reach $1/2$. Such errors will be punished by Max, extracting a cost of at least $0.4M$. Since approximating the value of \mathcal{G} to within $0.1M$ would enable, thanks to Thm. 6, to decide whether the Turing machine TM corresponding to \mathcal{A} halts or not, one concludes that weighted timed game values cannot in general be approximated, since otherwise one could solve the Halting Problem.

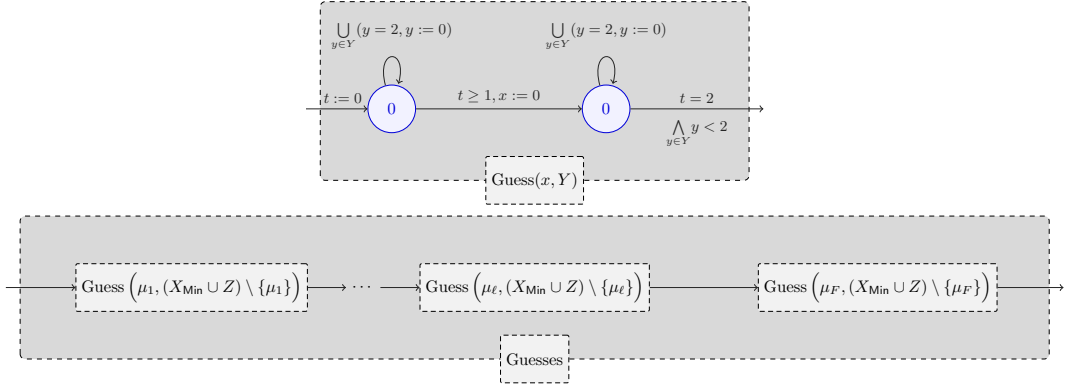
3.3 Modules and Widgets

We now describe a number of modules enabling us to implement the high-level protocol described in the previous section. In what follows, recall our assumption from Sec. 2, made without loss of generality, to the effect that neither player allows him- or herself to be deadlocked; in particular, if a clock constraint on a given transition requires the transition to be taken at a certain specific time, or within a certain time interval, for otherwise the run would deadlock (either immediately or shortly afterwards), then we assume the transition in question is indeed taken at the correct time.

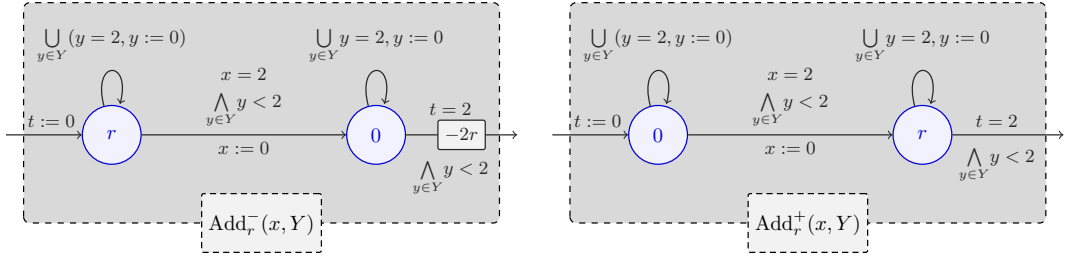
We begin with the modules $\text{Guess}(x, Y)$ and Guesses , depicted in Fig. 1, which enable Min to set the clocks in X_{Min} to arbitrary values of her choosing in $[0, 1]$. Here x stands for an arbitrary clock, and Y for an arbitrary set of clocks not containing x .

The correctness of the following two statements is clear upon inspection; we therefore omit the proofs.

► **Lemma 8.** *Provided $x \notin Y$ and the initial values of all clocks in Y upon entering $\text{Guess}(x, Y)$ lie in $[0, 2)$, then upon exiting $\text{Guess}(x, Y)$ all clocks in Y have their respective initial values, x has value in $[0, 1]$, and the cumulative weight is unchanged.*



■ **Figure 1** The guessing modules, enabling Min to set clocks μ_1, \dots, μ_ℓ and μ_F to arbitrary values of her choosing in $[0, 1]$, whilst leaving the values of clocks in Z unchanged. Recall that blue circles depict locations belonging to Player Min. The value 0 inside these circles, in module $\text{Guess}(x, Y)$, represents the weight of these locations (i.e., the rate at which the cumulative weight changes when control is in one of these locations). The notation $\bigcup_{y \in Y} (y = 2; y := 0)$ represents a collection of transitions, one for each $y \in Y$. Note that any such transition, when enabled (i.e., upon the corresponding clock $y \in Y$ reaching value 2), *must* instantly be taken, otherwise the guard on the transition exiting module $\text{Guess}(x, Y)$ could never be satisfied, and deadlock would ensue.



■ **Figure 2** Modules $\text{Add}_r^-(x, Y)$ and $\text{Add}_r^+(x, Y)$ enable to alter the cumulative weight by $-r\tilde{x}$ and $r\tilde{x}$ respectively, where \tilde{x} denotes the value of clock x upon entering the module and $r \in \mathbb{N}$ is a positive weight. Upon exiting the module, x as well as all clocks in Y have recovered their initial values. Note the negative weight of $-2r$ on the transition exiting module $\text{Add}_r^-(x, Y)$; this is the only place in our weighted timed game \mathcal{G} in which a negative weight is used.

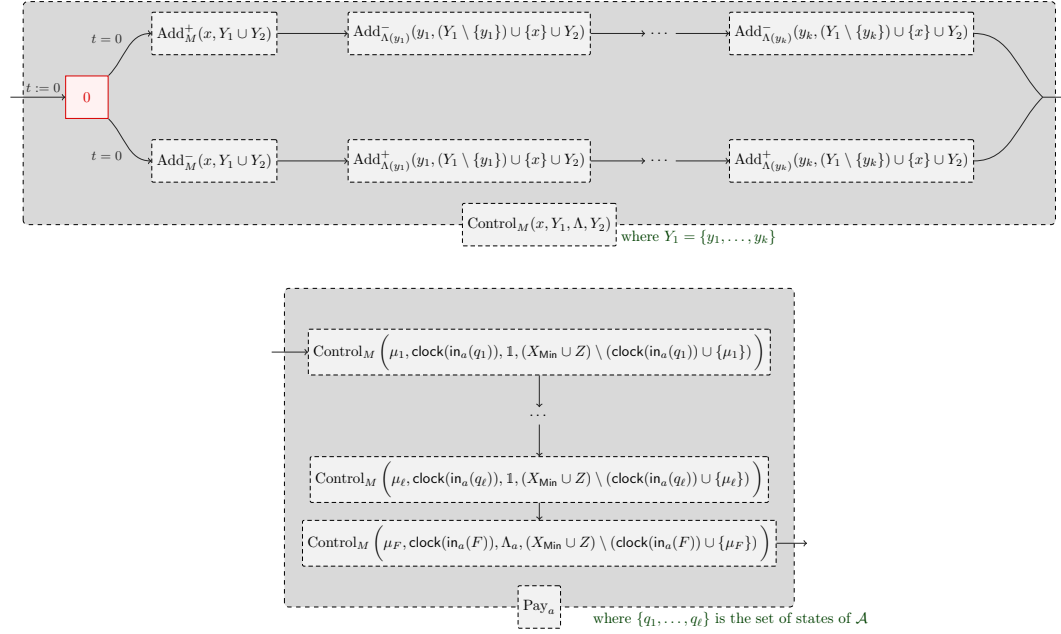
▶ **Corollary 9.** *Provided all clocks in X_{Min} and Z have values in $[0, 2)$ upon entering module Guesses , then upon exit the values of clocks in Z are unchanged, all clocks in X_{Min} have values in $[0, 1]$, and the cumulative weight is unchanged.*

We now introduce modules $\text{Add}_r^-(x, Y)$ and $\text{Add}_r^+(x, Y)$, depicted in Fig. 2. Here $r \in \mathbb{N}$ stands for a positive weight, x is a clock, and Y is a set of clocks not containing x . The role of these two modules is to alter the cumulative weight, as follows:

▶ **Lemma 10.** *Assume $x \notin Y$ and all clocks in $\{x\} \cup Y$ have values in $[0, 2)$ upon entering either $\text{Add}_r^-(x, Y)$ or $\text{Add}_r^+(x, Y)$. Let \tilde{x} denote the initial value of clock x . Then upon exiting $\text{Add}_r^-(x, Y)$, the cumulative weight has changed by $-r\tilde{x}$ (a decrease), whereas upon exiting $\text{Add}_r^+(x, Y)$, the cumulative weight has changed by $r\tilde{x}$ (an increase). Moreover, all clocks in $\{x\} \cup Y$ have recovered their initial values upon exiting either module.*

Once again, the statements are clear upon inspection.

We now turn to the payment modules, depicted in Fig. 3, which enable Player Max



■ **Figure 3** The payment modules, enabling Max to charge Min for guessing errors. x is a clock, and Y_1 and Y_2 are disjoint sets of clocks, neither of which contains x . Only Pay_a is depicted here; Pay_b is defined in entirely symmetrical fashion. The submodule $\text{Control}_M(x, Y_1, \Lambda, Y_2)$ is entered via a location represented as a red square, and hence belonging to Max, who can then choose between the upper and lower paths, whichever increases the cumulative weight (the two paths carry weights of equal magnitude but opposite signs). Note however that Max must act instantly upon entering submodular $\text{Control}_M(x, Y_1, Y_2)$, as the clock guard $t = 0$ would otherwise create a deadlock. The function $\Lambda_a : Z \rightarrow \mathbb{N}$ is defined as follows: $\Lambda_a(z) = \#(\text{out}_a(\text{clock}^{-1}(z)) \cap F)$; in other words, Λ_a retrieves the state of \mathcal{A} corresponding to clock z (call it q), and counts how many non-null a -labelled transitions into F originate from q in \mathcal{A} . This is required in order to properly calculate the total probability of being in F upon reading letter a . The function $\mathbb{1}$ simply returns the value 1 on all inputs.

to extract a cost for guessing errors committed by Min. We first need to introduce some auxiliary definitions.

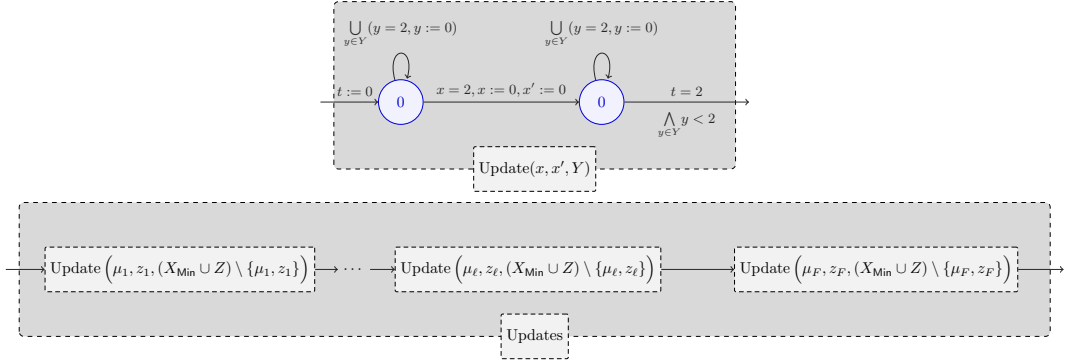
Given a state $q_i \in Q$, let $\text{in}_a(q_i)$ denote the set of all states $q_j \in Q$ such that there is an a -labelled non-null transition from q_j to q_i in \mathcal{A} (and recall, as noted in Sec. 3.2, that all such transitions have weight $1/M$). Formally, $\text{in}_a(q_i) = \{q_j \in Q \mid (A_a)_{i,j} = 1/M\}$. Overloading notation, write $\text{in}_a(F) = \cup_{q \in F} \text{in}_a(q)$. We define $\text{in}_b(q_i)$ and $\text{in}_b(F)$ in similar fashion.

We also define $\text{out}_a(q_i)$ symmetrically, representing the set of states q_j such that there is an a -labelled non-null transition from q_i to q_j , and similarly for $\text{out}_b(q_i)$.

For $q_i \in Q$, let $\text{clock}(q_i) = z_i$, and extend the clock function to sets of states in the obvious way: $\text{clock}(S) = \cup_{q \in S} \text{clock}(q)$.

We also consider the inverse function clock^{-1} , which associates to clock $z_i \in Z$ the state $q_i \in Q$.

► **Lemma 11.** *Let \tilde{x} and $\tilde{y}_1, \dots, \tilde{y}_k$ denote the initial values of clocks x and y_1, \dots, y_k respectively. Provided that all clocks in $\{x\} \cup Y_1 \cup Y_2$ have values in $[0, 2)$ upon entering $\text{Control}_M(x, Y_1, Y_2)$, the cumulative weight of this submodule is $\left| M\tilde{x} - \sum_{i=1}^k \tilde{y}_i \right|$. Moreover, upon exiting, all clocks (aside from t) have recovered their initial values.*



■ **Figure 4** The Updates module resets the value of each clock in Z to its counterpart in X_{Min} .

► **Corollary 12.** For $\mu \in X_{\text{Min}}$ and $z \in Z$, let $\tilde{\mu}$ and \tilde{z} respectively denote the initial values of clocks μ and z upon entering module Pay_a . Assuming all clocks in $X_{\text{Min}} \cup Z$ have initial values in $[0, 2)$ upon entering Pay_a , then all clocks have recovered their initial values upon exiting Pay_a , and the cumulative weight has increased by

$$\sum_{j=1}^{\ell} \left| M\tilde{\mu}_j - \sum_{i|q_i \in \text{in}_a(q_j)} \tilde{z}_i \right| + \left| M\tilde{\mu}_F - \sum_{i|q_i \in F} \sum_{j|q_j \in \text{in}_a(q_i)} \tilde{z}_j \right|.$$

By symmetry, an entirely similar assertion holds for module Pay_b .

Both statements follow by inspection, making use of the previous assertions laid out in this section.

Our last module updates the values of clocks in Z to agree with their counterparts in X_{Min} , thereby erasing any gaps created by errors in Min's guesses, and setting the stage for a fresh cycle to play out; see Fig. 4.

The following is immediate:

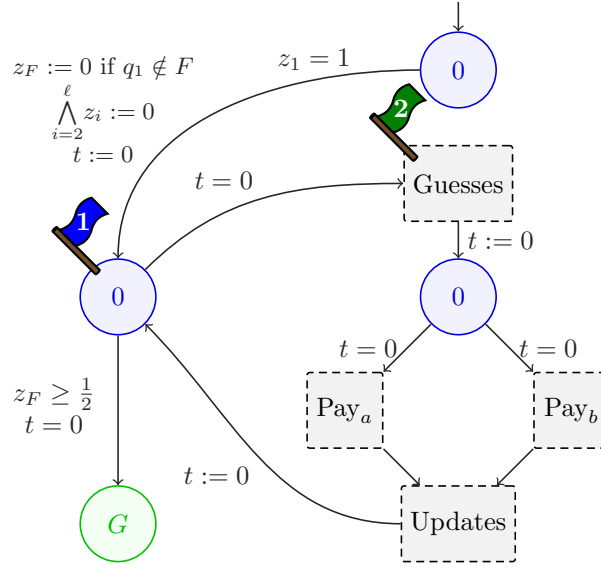
► **Lemma 13.** Provided that all initial values of clocks in $\{x\} \cup Y$ lie in $[0, 2)$ upon entering module $\text{Update}(x, x', Y)$, upon exiting all variables in $\{x\} \cup Y$ have recovered their initial values, x' agrees with x , and the cumulative weight remains unchanged.

Likewise, assuming all initial values of clocks in $X_{\text{Min}} \cup Z$ lie in $[0, 2)$, module Updates preserves the values of all clocks in X_{Min} , does not alter the cumulative weight, and ensures that every clock in Z has the same value as its counterpart in X_{Min} upon exit.

3.4 The Reduction

Recall that we are given a probabilistic automaton $\mathcal{A} = (Q, q_1, F, A_a, A_b)$ with set of states $Q = \{q_1, \dots, q_\ell\}$, over alphabet $\{a, b\}$, with the property that every non-zero state transition carries probability exactly $1/M$ for some $M \in \mathbb{N}$. We are moreover promised that either \mathcal{A} accepts some word with probability at least $1/2$, or \mathcal{A} accepts no word with probability exceeding $1/10$.

Our corresponding weighted timed game \mathcal{G} is depicted in Fig. 5. As noted earlier, the convenient use of the half-integral constant $1/2$ in one of the clock constraints is easily circumvented if desired.



■ **Figure 5** The weighted timed game \mathcal{G} . The start location sits at the top, and all clocks have value 0 in the initial configuration. All three blue circles have null weight (or rate), and likewise all transitions appearing in the drawing carry null weight. The clock constraint $t = 0$ on edges forces an immediate transition to the next location. The goal state (in green, bottom left) is designated by the letter G . In order to reach it, clock z_f must have value at least $1/2$ in the preceding location.

► **Proposition 14.** *Let \mathcal{A} and \mathcal{G} be as above. Then:*

- *If there is a word $w \in \{a, b\}^*$ such that $\mathcal{A}(w) \geq 1/2$, then the value of \mathcal{G} is exactly 0.*
- *If, for all words $w \in \{a, b\}^*$, $\mathcal{A}(w) \leq 1/10$, then the value of \mathcal{G} is at least $0.4M$.*

Proof. Consider a run of \mathcal{G} in which word $w = w_1 \dots w_n \in \{a, b\}^*$ has been played. Let $k \in \{1, \dots, n\}$, and for $i \in \{1, \dots, \ell\}$, let $\widetilde{z}_{i,k}$ and $\widetilde{z}_{F,k}$ be the respective values of clocks z_i and z_F upon exiting location for the k th time, and let $\widetilde{\mu}_{i,k}$ and $\widetilde{\mu}_{F,k}$ be the respective values of clocks μ_i and μ_F upon exiting module for the k th time.

By the lemmas and corollaries from the previous section, we have, for all i and k ,

$$\widetilde{\mu}_{i,k} = \widetilde{z}_{i,k+1}, \text{ and likewise } \widetilde{\mu}_{F,k} = \widetilde{z}_{F,k+1}. \quad (*)$$

Let us also introduce the following expressions:

$$E_{i,k} = \widetilde{z}_{i,k} - \mathbb{P}_{w_1 \dots w_{k-1}}(q_i) \quad \text{and} \quad E_{F,k} = \widetilde{z}_{F,k} - \mathbb{P}_{w_1 \dots w_{k-1}}(F),$$

$$\varepsilon_{i,k} = \widetilde{\mu}_{i,k} - \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \frac{\widetilde{z}_{j,k}}{M} \quad \text{and} \quad \varepsilon_{F,k} = \widetilde{\mu}_{F,k} - \sum_{i|q_i \in F} \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \frac{\widetilde{z}_{j,k}}{M}.$$

Intuitively, $E_{i,k}$ is the absolute (compound) error on the probability of being in state q_i after $k - 1$ iterations, and $\varepsilon_{i,k}$ is the marginal error on this probability upon reading letter w_k . Finally, we let cost_k be the maximum cumulative weight that Player Max can achieve upon exiting state for the k th time. For $k \in \{1, \dots, n\}$ and $i \in \{1, \dots, \ell\}$, we have:

23:10 Inapproximability in Weighted Timed Games

$$\begin{aligned}
E_{i,k+1} &= \widetilde{z_{i,k+1}} - \mathbb{P}_{w_1 \dots w_k}(q_i) \\
&= \widetilde{\mu_{i,k}} - \mathbb{P}_{w_1 \dots w_k}(q_i) && \text{(by (*))} \\
&= \varepsilon_{i,k} + \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \frac{\widetilde{z_{j,k}}}{M} - \mathbb{P}_{w_1 \dots w_k}(q_i) \\
&= \varepsilon_{i,k} + \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \frac{\widetilde{z_{j,k}}}{M} - \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \frac{\mathbb{P}_{w_1 \dots w_{k-1}}(q_j)(A_{w_k})_{i,j}}{M} \\
&= \varepsilon_{i,k} + \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \frac{\widetilde{z_{j,k}}}{M} - \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \frac{\mathbb{P}_{w_1 \dots w_{k-1}}(q_j)}{M} \\
E_{i,k+1} &= \varepsilon_{i,k} + \frac{1}{M} \sum_{j|q_j \in \text{in}_{w_k}(q_i)} E_{j,k}. && \text{(†)}
\end{aligned}$$


$$\text{Similarly: } E_{F,k+1} = \varepsilon_{F,k} + \frac{1}{M} \sum_{q_i \in F} \sum_{j|q_j \in \text{in}_{w_k}(q_i)} E_{j,k}. \quad (\star)$$

Let us now consider the following properties for $k \in \{1, \dots, n+1\}$:

$$\mathcal{P}(k) : \quad \sum_{i=1}^{\ell} |E_{i,k}| \leq \sum_{i=1}^{\ell} \sum_{m=1}^{k-1} |\varepsilon_{i,m}|$$

$$\mathcal{Q}(k) : \quad \text{cost}_k = M \sum_{i=1}^{\ell} \sum_{m=1}^{k-1} |\varepsilon_{i,m}| + M \sum_{m=1}^{k-1} |\varepsilon_{F,m}|.$$

We prove both properties by induction.

- The base case is $k = 1$. Since only the start location and location  have been visited, we have $\widetilde{z_{1,1}} = 1$, $\widetilde{z_{i,1}} = 0$ for $2 \leq i \leq \ell$, and $\widetilde{z_{F,1}} = 1$ if $q_1 \in F$, and $\widetilde{z_{F,1}} = 0$ otherwise. On the other hand, $\mathbb{P}_\lambda(q_1) = 1$, $\mathbb{P}_\lambda(q_i) = 0$ for $2 \leq i \leq \ell$, and $\mathbb{P}_\lambda(F) = 1$ if $q_1 \in F$, and $\mathbb{P}_\lambda(F) = 0$ otherwise.

Therefore $E_{i,1} = 0$ for $1 \leq i \leq \ell$ and $E_{F,1} = 0$. Likewise, $\text{cost}_1 = 0$, whence $\mathcal{P}(1)$ and $\mathcal{Q}(1)$ hold.

- Let $k \in \{1, \dots, n\}$, and assume that both $\mathcal{P}(k)$ and $\mathcal{Q}(k)$ hold. Thanks to Cor. 12, we have:

$$\begin{aligned}
\text{cost}_{k+1} &= \text{cost}_k + \sum_{j=1}^{\ell} \left| M \widetilde{\mu_{j,k}} - \sum_{i|q_i \in \text{in}_{w_k}(q_j)} \widetilde{z_{i,k}} \right| + \left| M \widetilde{\mu_{F,k}} - \sum_{i|q_i \in F} \sum_{j|q_j \in \text{in}_{w_k}(q_i)} \widetilde{z_{j,k}} \right| \\
&= \text{cost}_k + M \left(\sum_{i=1}^{\ell} |\varepsilon_{i,k}| + |\varepsilon_{F,k}| \right) \\
&= M \sum_{i=1}^{\ell} \sum_{m=1}^k |\varepsilon_{i,m}| + M \sum_{m=1}^k |\varepsilon_{F,m}|, \text{ as required.}
\end{aligned}$$

$$\begin{aligned}
\text{Also } \sum_{i=1}^{\ell} |E_{i,k+1}| &\leq \sum_{i=1}^{\ell} |\varepsilon_{i,k}| + \frac{1}{M} \sum_{i=1}^{\ell} \sum_{j|q_j \in \text{in}_{w_k}(q_i)} |E_{j,k}| && \text{(by †)} \\
&\leq \sum_{i=1}^{\ell} |\varepsilon_{i,k}| + \frac{1}{M} \sum_{i,j|(A_{w_k})_{i,j}=1/M} |E_{j,k}| \\
\sum_{i=1}^{\ell} |E_{i,k+1}| &\leq \sum_{i=1}^{\ell} |\varepsilon_{i,k}| + \frac{1}{M} \sum_{i=1}^{\ell} \sum_{j|q_j \in \text{out}_{w_k}(q_i)} |E_{i,k}|.
\end{aligned}$$

By our assumption on \mathcal{A} , each state has exactly M non-null outgoing transitions for each letter. Therefore

$$\sum_{i=1}^{\ell} |E_{i,k+1}| \leq \sum_{i=1}^{\ell} |\varepsilon_{i,k}| + \frac{1}{M} \sum_{i=1}^{\ell} M |E_{i,k}|.$$

Applying $\mathcal{P}(k)$,

$$\sum_{i=1}^{\ell} |E_{i,k+1}| \leq \sum_{i=1}^{\ell} |\varepsilon_{i,k}| + \sum_{i=1}^{\ell} \sum_{m=1}^{k-1} |\varepsilon_{i,m}| = \sum_{i=1}^{\ell} \sum_{m=1}^k |\varepsilon_{i,m}|,$$

and therefore $\mathcal{P}(k+1)$ holds, concluding the induction step.

Now if w is such that $\mathcal{A}(w) = \mathbb{P}_{w_1 \dots w_k}(F) \geq 1/2$, Player Min need only correctly set each clock μ_i to its expected value in every iteration, so that, for all $i \in \{1, \dots, \ell\}$ and all $k \in \{1, \dots, n+1\}$, we have $\varepsilon_{i,k} = 0$ and $\varepsilon_{F,k} = 0$. By $\mathcal{Q}(n+1)$, the value of \mathcal{G} is at most 0. Since it is easily seen that the value of \mathcal{G} cannot be negative, it must indeed be precisely 0.

If, on the other hand, $\mathcal{A}(w) \leq 1/10$, then in order for Min to reach the goal state after playing w , it is necessary to have $E_{F,n+1} \geq 1/2 - 1/10 = 0.4$. Using (\star) ,

$$\begin{aligned} 0.4 &\leq \varepsilon_{F,n+1} + \frac{1}{M} \sum_{q_i \in F} \sum_{j|q_j \in \text{in}_{w_n}(q_i)} E_{j,n+1} \\ &\leq |\varepsilon_{F,n+1}| + \frac{1}{M} \sum_{q_i \in F} \sum_{j|q_j \in \text{in}_{w_n}(q_i)} |E_{j,n+1}| \\ &\leq |\varepsilon_{F,n+1}| + \frac{1}{M} \sum_{i,j|(A_{w_n})_{i,j}=1/M \wedge q_i \in F} |E_{j,n+1}|. \end{aligned}$$

Recall that each state of \mathcal{A} has exactly M non-null outgoing transitions for each letter, and thus at most M w_n -labelled outgoing transitions to a final state. We then get

$$0.4 \leq |\varepsilon_{F,n+1}| + \sum_{j=1}^{\ell} |E_{j,n+1}|.$$

Using $\mathcal{P}(n+1)$, $0.4 \leq |\varepsilon_{F,n+1}| + \sum_{i=1}^{\ell} \sum_{m=1}^n |\varepsilon_{i,m}| \leq \sum_{i=1}^{\ell} \sum_{m=1}^n |\varepsilon_{i,m}| + \sum_{m=1}^n |\varepsilon_{F,m}|$,

whence, using $\mathcal{Q}(n+1)$, $0.4M \leq \text{cost}_{n+1}$.

This is true for any word played. Therefore if no word is accepted by \mathcal{A} with probability exceeding 0.1, the value of \mathcal{G} must be at least $0.4M$, as claimed. \blacktriangleleft

Our main result now immediately follows:

► **Theorem 1.** *Given a two-player, turn-based, weighted timed game with integer weights, the problem of approximating its value arbitrarily closely is computationally unsolvable.*

References

- 1 Rajeev Alur, Mikhail Bernadsky, and P. Madhusudan. Optimal reachability for weighted timed games. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP'04)*, pages 122–133, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.
- 2 Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994. URL: <https://www.sciencedirect.com/science/article/pii/0304397594900108>, doi:10.1016/0304-3975(94)90010-8.
- 3 Rajeev Alur, Salvatore La Torre, and George J. Pappas. Optimal paths in weighted timed automata. *Theoretical Computer Science*, 318(3):297–322, 2004. URL: <https://www.sciencedirect.com/science/article/pii/S0304397503005838>, doi:10.1016/j.tcs.2003.10.038.
- 4 Gerd Behrmann, Ansgar Fehnker, Thomas Hune, Kim Larsen, Paul Pettersson, Judi Romijn, and Frits Vaandrager. Minimum-cost reachability for priced time automata. In *HSCC01: Hybrid Systems: Computation and Control*, pages 147–161, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.
- 5 Patricia Bouyer, Thomas Brihaye, and Nicolas Markey. Improved undecidability results on weighted timed automata. *Information Processing Letters*, 98(5):188–194, 2006. URL: <https://www.sciencedirect.com/science/article/pii/S0020019006000652>, doi:10.1016/j.ipl.2006.01.012.
- 6 Patricia Bouyer, Franck Cassez, Emmanuel Fleury, and Kim G. Larsen. Optimal strategies in priced timed game automata. In *FSTTCS 2004: Foundations of Software Technology*

- and *Theoretical Computer Science*, pages 148–160, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
- 7 Patricia Bouyer, Samy Jaziri, and Nicolas Markey. On the Value Problem in Weighted Timed Games. In Luca Aceto and David de Frutos Escrig, editors, *Proceeding of the 26th International Conference on Concurrency Theory (CONCUR 2015)*, volume 42 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 311–324, Dagstuhl, Germany, 2015. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CONCUR.2015.311>, doi:10.4230/LIPIcs.CONCUR.2015.311.
 - 8 Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On optimal timed strategies. In *Formal Modeling and Analysis of Timed Systems*, pages 49–64, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
 - 9 Damien Busatto-Gaston, Benjamin Monmege, and Pierre-Alain Reynier. Symbolic Approximation of Weighted Timed Games. In Sumit Ganguly and Paritosh Pandya, editors, *Proceeding of the 38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2018)*, volume 122 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:16, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.FSTTCS.2018.28>, doi:10.4230/LIPIcs.FSTTCS.2018.28.
 - 10 Damien Busatto-Gaston, Benjamin Monmege, and Pierre-Alain Reynier. Optimal controller synthesis for timed systems. *Log. Methods Comput. Sci.*, 19(1), 2023.
 - 11 A. Condon and R.J. Lipton. On the complexity of space bounded interactive proofs. In *30th Annual Symposium on Foundations of Computer Science*, pages 462–467, 1989. doi:10.1109/SFCS.1989.63519.
 - 12 Nathanaël Fijalkow. Undecidability results for probabilistic automata. *ACM SIGLOG News*, 4(4):10–17, nov 2017. doi:10.1145/3157831.3157833.
 - 13 Marcin Jurdziński and Ashutosh Trivedi. Reachability-time games on timed automata. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming (ICALP'07)*, pages 838–849, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
 - 14 Salvatore La Torre, Supratik Mukhopadhyay, and Aniello Murano. *Optimal-Reachability and Control for Acyclic Weighted Timed Automata*, pages 485–497. Springer US, Boston, MA, 2002. doi:10.1007/978-0-387-35608-2_40.
 - 15 Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence*, 147(1):5–34, 2003. Planning with Uncertainty and Incomplete Information. URL: <https://www.sciencedirect.com/science/article/pii/S0004370202003788>, doi:10.1016/S0004-3702(02)00378-8.
 - 16 Marvin L. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall Series in Automatic Computation. Prentice-Hall, 1967.
 - 17 Günter Rote. Probabilistic finite automaton emptiness is undecidable. <https://page.mi.fu-berlin.de/rote/Kram/PFA-undecidable.pdf>, 2023. Accessed: 2024-02-20.