Energy and Mean-payoff Timed Games

Romain Brenguier, Franck Cassez, Jean-François Raskin

Energy and Mean-payoff Timed Games*

Romain Brenguier
Département d'informatique,
Université Libre de Bruxelles
(U.L.B), Belgium
brenguier@ulb.ac.be

Franck Cassez
NICTA,
Sydney, Australia
franck.cassez@nicta.com.au

Jean-François Raskin
Département d'informatique,
Université Libre de Bruxelles
(U.L.B), Belgium
jraskin@ulb.ac.be

ABSTRACT

In this paper, we study energy and mean-payoff timed games. The decision problems that consist in determining the existence of winning strategies in those games are undecidable, and we thus provide semi-algorithms for solving these strategy synthesis problems. We then identify a large class of timed games for which our semi-algorithms terminate and are thus complete. We also study in detail the relation between mean-payoff and energy timed games. Finally, we provide a symbolic algorithm to solve energy timed games and demonstrate its use on small examples using HyTech.

1. INTRODUCTION

Timed automata [1], respectively timed games [34, 20], are fundamental models to verify, respectively to synthesize controllers for, timed systems which have to enforce hard real-time constraints. Those models were introduced in the nineties and the underlying theory has since then been successfully implemented in efficient analysis tools such as Kronos [17] and UppAal [33] for verification, and UppAal-Tiga [5] for synthesis. The latter has been used to solve industrial case studies, e.g. [29, 21].

Recently, there has been an important research effort to lift verification and synthesis techniques from the Boolean case to the quantitative case, see [27] and references therein. More specifically, lots of progress has been made recently on zero-sum two-player games played on weighted graphs, in which edges are decorated with *costs* or *rewards*, see for example [15, 23, 24, 19], with the objective of setting up a framework for the synthesis of optimal controllers (see also [36] for applications in linear control systems). Important examples of such games are *mean-payoff* and *energy* games [15, 19, 24]. In those games, two players move a token along the edges of a weighted graph whose vertices are partitioned into vertices that belong to player 1, and player 2 respectively. In each round of the game, the player that owns the vertex with the token chooses an outgoing edge and target vertex to move the token to. By playing in such a way, the two players form an infinite path through the graph. Player 1 wins the *mean-payoff objective* if the long-run average of the edge-weights along this path is non-negative, and he wins the *energy objective*, if there exists a bound $c \in \mathbb{Z}$ such that the running sum of weights of the traversed edges along the infinite path never goes below c (this can model for example that the system never runs out of energy). As the games we consider are zero-sum, player 2 wins when he can enforce the complementary objectives. In the finite state case, the mean-payoff and energy objectives are *inter-reducible*, and this fact was used recently to provide algorithmic improvements to solve mean-payoff games [19].

Extensions of timed automata with costs and rewards have also been studied. In [3, 6], timed automata are extended with continuous variables that are used as *observers*, and allow for modeling accumulation of costs or rewards along executions. The main motivation for studying those extensions is to offer an extra modeling power while avoiding severe intractability of richer models like hybrid automata. Indeed, it has been shown that the reachability problem for weighted/priced timed automata remains decidable [3, 6], and more precisely PSPACE-c [10], while the reachability problem is undecidable already for the class of stopwatch automata [22] (a simple class of hybrid automata). Also the existence of executions in a weighted automaton that ensure a bound on the mean-payoff can also be decided in PSPACE [12]. In this paper, we consider timed extensions of the important classes of mean-payoff and energy games.

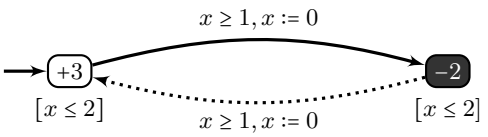

Figure 1: Turn-Based Energy Timed Game $\mathcal{A}$.

Example 1. Fig. 1 gives an example of an energy (turn-based) timed game. `Eve` (player 1) owns the left location and decides when to take the transition from left to right, while `Adam` (player 2) owns the right location and decides when to take the transition from right to left; x is a dense-time clock. Each transition resets the clock x, and when time elapses the energy level grows with derivative 3 in `Eve`'s location and decreases with derivative 2 in `Adam`'s location. In

*Work supported by ERC Starting Grant inVEST (279499).

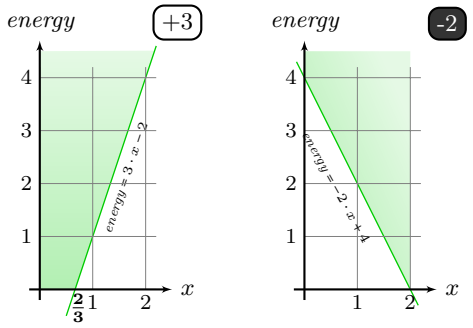

Figure 2: Winning Zones for Timed Game $\mathcal{A}$.

the remaining of this paper, we use the following conventions: plain (resp. dashed) arrows are `Eve`'s (resp. `Adam`'s) transitions; a location *invariant* is enclosed in brackets (e.g. $[x \leq 2]$) and must be satisfied by the valuation of the clocks when the control is in that location; when an edge weight is non zero, we attach it to the edge. Fig. 2, left, depicts the initial energy levels that are sufficient for `Eve` in order to have a winning strategy against any strategy of `Adam`, i.e. to ensure an infinite execution in which the energy level never falls below 0. For instance, if the infinite game starts in `Eve`'s location, then if the value of clock x is less than $\frac{2}{3}$, an initial energy level 0 is sufficient; for $x > \frac{2}{3}$, the initial energy level should be larger than or equal to $3 \cdot x - 2$. Similarly, the winning zone in `Adam`'s location is depicted on the right hand side of Fig. 2. The main purpose of this article is to propose algorithmic methods to compute such information.

Unfortunately, for weighted extensions of timed games, even the cost bounded reachability problem is undecidable [18], and we show here that both mean-payoff and energy games are undecidable. This is unfortunate as such cost extensions of timed games are very natural and well-suited to model optimality problems in embedded control [21]. Nevertheless, we believe that the undecidability result should not be an end to the story and we study in this paper *semi-algorithms* (completeness and/or termination is not guaranteed) to solve those two synthesis problems. We also identify a large class of timed games where our semi-algorithms are complete. To the best of our knowledge, there are the first positive results for those objectives on timed automata. There are related works in the literature but they apply to orthogonal classes of games, or to other objectives. Indeed, in [11], it is shown that mean-payoff games are decidable for O-minimal hybrid automata, this class is different from the one identified here as timed automata are not O-minimal hybrid automata. In [31], the authors study the average time per transition problem for turn-based timed games; their results do not apply to mean-payoff, nor to energy objectives.

Contributions. Our contribution is threefold. First, we study the relation between mean-payoff and energy timed games. As we already mentioned, in the finite state case, the mean-payoff and energy objectives are inter-reducible [19]: given a weighted game G, `Eve` wins the mean-payoff objective if, and only if, she wins the energy objective. We show here that the relationship between the two types of games is more complex in the timed case. We identify conditions under which it is possible to transfer winning strategies for one objective into winning strategies for the other objective, and we show that those conditions are also necessary. Those results are formalized by Thm. 1 and Thm. 2.

Second, Thm. 3 establishes the undecidability of the decision problems associated with energy and mean-payoff timed games. This result is unfortunate but not surprising (it was already conjectured in [9], see page 89). This negative result motivates the main contribution of this paper: we propose two semi-algorithms for synthesizing winning strategies. We first consider a *cycle forming game* (in the spirit of [8]) on the region graph associated with the underlying weighted timed game: the two players move a token on the region graph and the game is stopped as soon as a cycle is formed. In Sect. 3.5, we partition the set of simple cycles of the region graph into those that are *good* for `Eve`, those that are *good* for `Adam`, and those that are neither good for `Eve` nor for `Adam`. If the formed cycle belongs to the first set then `Eve` is declared winner of the cycle forming game, if the cycle belongs to the second then `Adam` is the winner, otherwise it is a draw. Thm. 4 establishes that if `Eve` wins the cycle forming game then she has also a winning strategy in associated energy games, and Thm. 5 proves a similar result for `Adam`. Then, we identify a class of weighted timed games, that we call *robust*, for which this reduction to the cycle forming game on the region graph is complete: in this case the good cycles for `Eve` and the good cycles for `Adam` partition the set of simple cycles of the region graph. This class covers the class of timed games where costs appear on edges only. Thm. 11 establishes the decidability of the membership problem for the class of robust weighted timed games.

Finally, as the cycle forming game is defined on the region graph, it does not lead to a practical algorithmic solution. This is why we propose in addition a symbolic semi-algorithm to solve energy timed games. In Thm. 16, we show that our symbolic algorithm is also complete on the class of *robust* weighted timed games. In order to show the feasibility of our approach, we have implemented this algorithm as a script for HyTech [28] and ran it on small examples.

Our main theorems and their relation with the different classes of games we consider are depicted in Fig. 3 and 4.

Structure of the paper. In Sect. 2, we define the mean-payoff and energy timed games. In Sect. 3, we develop semi-algorithms based on reductions to cycle games played on the region graph. In Sect. 4, we identify a class *robust games*, for which the reduction to cycle games is complete. In Sect. 5, we propose a symbolic semi-algorithm which is also complete for robust games.

2. PRELIMINARIES

In this section, we first recall the definition of concurrent games. Then we review a useful result from [30] that defines a canonical decomposition of infinite paths in a graph into *simple cycles*. Next, we introduce *weighted timed games*, the semantics of which is given in term of infinite concurrent games. Starting from the notion of *weight* (or cost/reward), we define *mean payoff* and *energy* objectives. We close the section with a study of the relationships that exist between

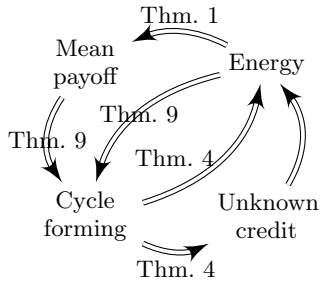

Figure 3: Winning strategies for Eve

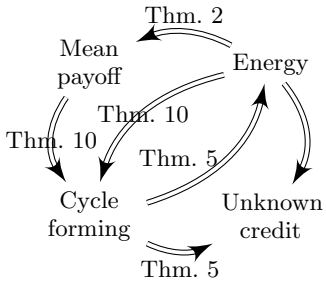

Figure 4: Winning strategies for Adam

mean-payoff and energy objectives in timed games.

We let $\mathbb{N}$ be the set of natural numbers, $\mathbb{Z}$ the set of integers, $\mathbb{R}$ the set of reals and $\mathbb{R}_+$ the set of non-negative reals.

2.1 Concurrent Games

Definition 1. ([2]) A *concurrent game* between two players $\mathtt{Eve}$ and $\mathtt{Adam}$ is a tuple $\mathcal{C} = \langle \mathtt{St}, \iota, \mathtt{Act}, \mathtt{Mov}, \mathtt{Tab}, \Omega \rangle$, where:

- $\mathtt{St}$ is the set of *states*;
- $\iota \in \mathtt{St}$ is the *initial* state;
- $\mathtt{Act}$ is the set of *actions*;
- $\mathtt{Mov} : \mathtt{St} \times \{\mathtt{Eve}, \mathtt{Adam}\} \mapsto 2^{\mathtt{Act}} \setminus \{\varnothing\}$ gives for a state and a player the set of *allowed actions*, we let $\mathtt{Mov}(s) = \{(a,b) \mid a \in \mathtt{Mov}(s, \mathtt{Eve}), b \in \mathtt{Mov}(s, \mathtt{Adam})\}$;
- $\mathtt{Tab} : \mathtt{St} \times \mathtt{Act} \times \mathtt{Act} \mapsto \mathtt{St}$ is the *transition function*;
- $\Omega \subseteq (\mathtt{St} \cdot (\mathtt{Act} \times \mathtt{Act}))^\omega$ is the *objective* for $\mathtt{Eve}$.

A concurrent game is *finite* if $\mathtt{St}$ and $\mathtt{Act}$ are finite. It is *turn-based* if in each state s, one of the players has only one allowed action.

A round of the game consists in $\mathtt{Eve}$ and $\mathtt{Adam}$ to choose independently and simultaneously some actions, say $a_\exists$ and $a_\forall$ respectively, such that $a_\exists \in \mathtt{Mov}(s, \mathtt{Eve})$ and $a_\forall \in \mathtt{Mov}(s, \mathtt{Adam})$. The pair $m = (a_\exists, a_\forall)$ is an allowed *move* i.e. $m \in \mathtt{Mov}(s)$. By playing finitely (resp. infinitely) many rounds from state $s \in \mathtt{St}$, the players build a finite (resp. infinite) *path*. Formally, a *path* is a finite or infinite sequence $s_0 \cdot m_0 \cdot s_1 \cdot m_1 \cdots s_k \cdot m_k \cdots$ of alternating states and moves, such that $\forall i \geq 0, m_i \in \mathtt{Mov}(s_i)$ and $s_{i+1} = \mathtt{Tab}(s_i, m_i)$. We also write a path as $s_0 \xrightarrow{m_0} s_1 \xrightarrow{m_1} \cdots$. The length, $|\rho|$, of an infinite path ρ is ∞ and the length of a finite path ρ with n moves (ending in s_n) is n and $\mathtt{last}(\rho) = s_n$. We write $\rho_n, n \leq |\rho|$, for s_n, the $n+1$-th state in ρ, and $\mathtt{first}(\rho) = \rho_0 = s_0$. Given a path $\rho = s_0 \xrightarrow{m_0} s_1 \xrightarrow{m_1} \cdots$, we write $\rho_{\leq n}, n \leq |\rho|$ for the prefix of ρ up to s_n that is the finite path $s_0 \xrightarrow{m_0} \cdots \xrightarrow{m_{n-1}} s_n$.

A *play* is an infinite path in $(\mathtt{St} \cdot (\mathtt{Act} \times \mathtt{Act}))^\omega$, and a *history* is a finite path in $(\mathtt{St} \cdot (\mathtt{Act} \times \mathtt{Act}))^* \cdot \mathtt{St}$. The set of plays from s is $\mathtt{Play}(\mathcal{C}, s)$ and $\mathtt{Play}(\mathcal{C}) = \mathtt{Play}(\mathcal{C}, \iota)$. A play is *winning* for $\mathtt{Eve}$ if it belongs to the objective Ω.

Definition 2. (Strategies) A *strategy* for $\mathtt{Eve}$ (resp. $\mathtt{Adam}$) is a function which associates with a history h an $\mathtt{Eve}$-action in $\mathtt{Mov}(\mathtt{last}(h), \mathtt{Eve})$ (resp. an $\mathtt{Adam}$-action). A pair of strategies $(\sigma_\exists, \sigma_\forall)$ forms a *strategy profile*. Given a strategy profile, its *outcome from state s*, written $\mathtt{Out}_s(\sigma_\exists, \sigma_\forall)$, is the unique play ρ such that: $\rho_0 = s$ and $\forall n \geq 0, \rho_{n+1} = \mathtt{Tab}(\rho_n, \sigma_\exists(\rho_{\leq n}), \sigma_\forall(\rho_{\leq n}))$. Given a strategy $\sigma_\exists$ of $\mathtt{Eve}$, its *outcomes from state s*, written $\mathtt{Out}_s(\sigma_\exists)$, are the set of plays ρ for which there is a strategy $\sigma_\forall$ of $\mathtt{Adam}$, such that $\rho = \mathtt{Out}_s(\sigma_\exists, \sigma_\forall)$. A strategy $\sigma_\exists$ of $\mathtt{Eve}$ is *winning*, if for all strategies $\sigma_\forall$ of $\mathtt{Adam}$, $\mathtt{Out}_\iota(\sigma_\exists, \sigma_\forall) \in \Omega$; strategy $\sigma_\forall$ of $\mathtt{Adam}$ is *winning*, if for all strategies $\sigma_\exists$ of $\mathtt{Eve}$, $\mathtt{Out}_\iota(\sigma_\exists, \sigma_\forall) \notin \Omega$.

2.2 Decomposition in Simple Cycles

In the sequel we will reduce energy and mean payoff games to games played on the region with *cycles* objectives. In this paragraph we recall the key results [30] related to the decomposition of a play into *simple cycles*. A history $h = s_0 \cdot m_0 \cdot s_1 \cdot m_1 \cdots s_n$ is a *cycle* if $s_0 = s_n, n \geq 1$. A *simple cycle* is a cycle such that for all i and j, $0 \leq i < j < n$, $s_i \neq s_j$. We write $\mathbb{C}(\mathcal{C})$ ($\mathbb{C}$ when $\mathcal{C}$ is clear from the context) for the set of simple cycles in the concurrent game $\mathcal{C}$.

Every history h of a finite game can be uniquely decomposed into a sequence of simple cycles, except for a finite part. The decomposition process maintains a *stack*, $\mathtt{st}(h)$, of distinct states and moves. We write the stack content $s_1 \cdot m_1 \cdot s_2 \cdot \cdots \cdot m_{n-1} \cdot s_n$ where s_1 is at the bottom of the stack and s_n the top. We use the notation $s \in \mathtt{st}(h)$ for $s \in \{s_1, s_2, \cdots, s_n\}$. The *decomposition*, $\mathtt{dec}(h)$, is a set of simple cycles. We define $\mathtt{dec}(h)$ and $\mathtt{st}(h)$ inductively as follows:

- for the single state history s, $\mathtt{dec}(s) = \varnothing$ and $\mathtt{st}(s) = s$.
- let $h' = h \cdot m \cdot s$, m a move, $s \in \mathtt{St}$, be a history.
  - If $s \in \mathtt{st}(h)$, and $\mathtt{st}(h) = \alpha \cdot s \cdot \beta$, then $\mathtt{st}(h') = pop(\mathtt{st}(h), |\beta|)$ and $\mathtt{dec}(h') = \mathtt{dec}(h) \cup \{s \cdot \beta \cdot m \cdot s\}$.
  - else $\mathtt{dec}(h') = \mathtt{dec}(h)$, $\mathtt{st}(h') = push(\mathtt{st}(h), m \cdot s)$.

Note that the stack always contains distinct elements, therefore only simple cycles are added to the decomposition. The elements in the stack from the bottom to the top, form a history $s_0 \cdot m_0 \cdot s_1 \cdot m_1 \cdots s_n$, where $n+1$ is the height of the stack. The decomposition of a play is the union of the decompositions of the finite prefixes of the play.

2.3 Weighted Timed Games

Let X be a finite set of variables called *clocks*. A *clock valuation* is a mapping $v : X \to \mathbb{R}_+$. We let $\mathbb{R}_+^X$ be the set of clock valuations over X. We let $\mathbf{0}_X$ be the *zero* valuation where all the clocks in X are set to 0 (we use $\mathbf{0}$ when X is clear from the context). Given $\delta \in \mathbb{R}_+$, $v + \delta$ denotes the valuation defined by $(v+\delta)(x) = v(x) + \delta$. We let $\mathfrak{C}(X)$ be the set of *convex constraints* on X which is the set of conjunctions of constraints of the form $x \bowtie c$ with $c \in \mathbb{N}$ and $\bowtie \in \{\leq, <, =, >, \geq\}$. Given a constraint $g \in \mathfrak{C}(X)$ and a valuation v, we write $v \vDash g$ if g is satisfied by v. Given $Y \subseteq X$ and a valuation v, $[Y \leftarrow 0]v$ is the valuation defined by $([Y \leftarrow 0]v)(x) = v(x)$ if $x \notin Y$ and $([Y \leftarrow 0]v)(x) = 0$ otherwise.

Definition 3. A *weighted timed game* [32] (WTG for short) is a tuple $\mathcal{T} = \langle \mathtt{L}, \ell_\iota, X, T_\exists, T_\forall, \mathtt{Inv}, \mathtt{w} \rangle$, where:

- $\mathtt{L}$ is the (finite) set of locations and ℓ_ι is the initial location;
- X is a finite set of clocks;
- $T_\exists, T_\forall \subseteq \mathtt{L} \times \mathfrak{C}(X) \times 2^X \times \mathtt{L}$ are the set of transitions belonging to `Eve` and `Adam` respectively, and we let $T = T_\exists \cup T_\forall$; An element of $T_\exists$ (resp. $T_\forall$) is an `Eve`-transition (resp. `Adam`-transition).
- $\mathtt{Inv}\colon \mathtt{L} \to \mathfrak{C}(X)$ defines the invariants of each location;
- $\mathtt{w}\colon \mathtt{L} \cup T \to \mathbb{Z}$ is a weight function assigning integer weights to locations and discrete transitions.

If, from each location, all the outgoing transitions belong to the same player, $\mathcal{T}$ is said *turn-based*.

Informally, a WTG is played as follows: a state of the game is a pair (ℓ, v) where ℓ is a location and v is a clock valuation such that $v \vDash \mathtt{Inv}(\ell)$. The game starts from the initial state $(\ell_\iota, \mathbf{0})$. From a state (ℓ, v), each player $p \in \{\mathtt{Eve}, \mathtt{Adam}\}$ chooses (independently) a *timed action* $a_p = (d_p, e_p)$ where $d_p \in \mathbb{R}_+$ and $e_p = (\ell, g, Y, \ell')$ is a p-transition. The intended meaning is that p wants to delay for d_p time units and then fire transition e_p. There are some restrictions on the possible choices of timed actions (d_p, e_p):

1. d_p must be compatible with the current state (ℓ, v) and location invariant, i.e. for all $0 \le d' \le d_p$, $v + d' \vDash \mathtt{Inv}(\ell)$;
2. e_p must be enabled after d_p time units, i.e. $v + d_p \vDash g$;
3. the target location's invariant must be satisfied when entering this location, i.e. $[Y \leftarrow 0](v + d_p) \vDash \mathtt{Inv}(\ell')$.

A timed action satisfying these restrictions is said *legal*. If from a given state, one player has no legal timed action to play (i.e. no discrete action is enabled in the future for this player), it plays a special action $\bot$. At each round of the game, players propose some actions, $a_\exists$ for `Eve`, and $a_\forall$ for `Adam`. Either

1. $a_\exists$ is a legal action for `Eve`;
2. or there are no legal actions for `Eve` and $a_\exists = \bot$.

Similarly for $a_\forall$. We assume that from any reachable state of the game, at least one player has a legal action, hence the pair $(\bot, \bot)$ is never proposed.

To determine the effect of a joint action, we select the player p that chooses the shortest delay d_p. In case both players choose the same delay, the convention is that `Adam` is selected (this is without loss of generality and other policies can be accommodated for). These informal game rules are formalized in the next section.

2.4 Semantics of Timed Games

Given a timed action $(d, e) \in \mathbb{R}_+ \times T$ with $e = (\ell, g, Y, \ell')$, a state (ℓ, v), the successor state in the WTG is (ℓ', v') if:

1. $\forall 0 \le \delta \le d, v + \delta \vDash \mathtt{Inv}(\ell)$;
2. and $v + \delta \vDash g$;
3. and $[Y \leftarrow 0](v + d) \vDash \mathtt{Inv}(\ell')$.

We denote this transition $(\ell, v) \xrightarrow{(d,e)} (\ell', v')$ which accounts for a combined delay transition of d time units followed by the discrete step firing edge e. The *duration* of this transition is $\mathtt{d}((\ell, v) \xrightarrow{(d,e)} (\ell', v')) = d$. Its *reward* (or *weight*) is $\mathtt{w}((\ell, v) \xrightarrow{(d,e)} (\ell', v')) = d \cdot \mathtt{w}(\ell) + \mathtt{w}(e)$.

Given an objective $\Omega \subseteq \Big((\mathtt{L} \times \mathbb{R}_+^X) \cdot ((\mathbb{R}_+ \times T_\exists) \times (\mathbb{R}_+ \times T_\forall))\Big)^\omega$, the semantics of the WTG $\mathcal{T}$ is the (infinite) concurrent game $\mathcal{C}(\mathcal{T}, \Omega) = (\mathtt{St}, \iota, \mathtt{Act}, \mathtt{Mov}, \mathtt{Tab}, \Omega)$ defined by:

- the set of states is $\mathtt{St} = \mathtt{L} \times \mathbb{R}_+^X$ and the initial state is $\iota = (\ell_\iota, \mathbf{0})$;
- the set of actions is $\mathtt{Act} = \mathtt{Act}_\exists \cup \mathtt{Act}_\forall$, where $\mathtt{Act}_\exists = \mathbb{R}_+ \times T_\exists$ are the actions for `Eve` and $\mathtt{Act}_\forall = \mathbb{R}_+ \times T_\forall$ are the actions for `Adam`;
- $\mathtt{Mov}(s, \mathtt{Eve}) \in \Big(2^{\mathtt{Act}_\exists} \setminus \{\varnothing\}\Big) \cup \{\bot\}$ is the set of legal actions for `Eve` in s if there is at least one, or $\{\bot\}$ otherwise; and $\mathtt{Mov}(s, \mathtt{Adam})$ is defined similarly.
  Given $(a_\exists, a_\forall) \in \mathtt{Mov}(s, \mathtt{Eve}) \times \mathtt{Mov}(s, \mathtt{Adam})$, we define $\mathtt{Mov}(a_\exists, a_\forall)$ as follows:
  - if $a_\exists = \bot$ (resp. $a_\forall = \bot$) then `Adam` (resp. `Eve`) is selected and $\mathtt{Mov}(a_\exists, a_\forall) = a_\forall$ (resp. $\mathtt{Mov}(a_\exists, a_\forall) = a_\exists$);
  - otherwise $a_\exists = (d_\exists, e_\exists)$ and $a_\forall = (d_\forall, e_\forall)$ and:
    1. if $d_\exists < d_\forall$, $\mathtt{Mov}(a_\exists, a_\forall) = a_\exists$;
    2. if $d_\forall \le d_\exists$ then $\mathtt{Mov}(a_\exists, a_\forall) = a_\forall$;
- Given two actions $a_\exists$ and $a_\forall$, $\mathtt{Tab}((\ell, v), a_\exists, a_\forall) = (\ell', v')$ if $(\ell, v) \xrightarrow{\mathtt{Mov}(a_\exists, a_\forall)} (\ell', v')$.

Let $h = s_0 \xrightarrow{a_1^\exists, a_1^\forall} s_1 \quad \cdots \quad s_{n-1} \xrightarrow{a_{n-1}^\exists, a_{n-1}^\forall} s_n \cdots$ be a finite or infinite path in $\mathcal{C}(\mathcal{T}, \Omega)$. The *duration* and *reward* of h are respectively:

$$\mathtt{d}(h) = \sum_{k=0}^{|h|-1} \mathtt{d}\left(s_k \xrightarrow{\mathtt{Mov}(a_k^\exists, a_k^\forall)} s_{k+1}\right)$$

$$\mathtt{w}(h) = \sum_{k=0}^{|h|-1} \mathtt{w}\left(s_k \xrightarrow{\mathtt{Mov}(a_k^\exists, a_k^\forall)} s_{k+1}\right)$$

A play ρ is said *non-Zeno* if $(d(\rho_{\le n}))_{n \in \mathbb{N}}$ is unbounded. A strategy σ is *immune from Zenoness* if all its outcomes are non-Zeno. A game is said to have *bounded transitions* if there is a bound D, such that for all states (ℓ, v), actions $a_\exists$, $a_\forall$: $\mathtt{d}\Big((\ell, v) \xrightarrow{\mathtt{Mov}(a_\exists, a_\forall)} \mathtt{Tab}((\ell, v), a_\exists, a_\forall)\Big) \le D$.

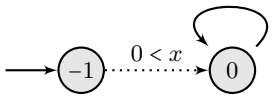

Figure 5: WTG $\mathcal{B}$.

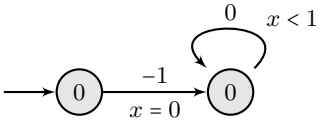

Figure 6: WTG $\mathcal{D}$.

2.5 Mean-payoff and Energy Objectives

The *mean payoff* (per time unit) of a play is defined as the long-run average of reward per time unit. Formally, the mean payoff of a play ρ is:

$$\mathtt{MP}(\rho) = \liminf_{n\to\infty} \frac{\mathtt{w}(\rho_{\leq n})}{\mathtt{d}(\rho_{\leq n})}.$$

Definition 4. We consider the following types of games:

- The *mean payoff game* $\mathcal{T}_{\mathtt{MP}}$ associated with a WTG $\mathcal{T}$, is the game played on $\mathcal{T}$ where the objective (for `Eve`) is to obtain a non-negative mean payoff: i.e. $\mathcal{T}_{\mathtt{MP}} = \mathcal{C}(\mathcal{T}, \Omega_{\mathtt{MP}})$ where $\Omega_{\mathtt{MP}} = \{\rho \in \mathtt{Play}(\mathcal{C}) \mid \mathtt{MP}(\rho) \geq 0\}$.
- Given an initial credit $c \geq 0$, the *c-energy game* $\mathcal{T}_{E(c)}$ associated to a WTG $\mathcal{T}$, is the game played on $\mathcal{T}$ where the objective $\Omega(c)$ is to maintain the reward of every prefix of every play above $-c$: i.e. $\mathcal{T}_{E(c)} = \mathcal{C}(\mathcal{T}, \Omega(c))$ where $\Omega(c) = \{\rho \in \mathtt{Play}(\mathcal{C}) \mid \forall n \in \mathbb{N}.\ c + \mathtt{w}(\rho_{\leq n}) \geq 0\}$.
- The *energy game* associated with a WTG $\mathcal{T}$, is the game $\mathcal{C}(\mathcal{T}, \Omega_E)$ where the objective is $\Omega_E = \cup_{c\geq 0}\Omega(c)$.

Decision problems. For each type of games, we define the associated decision problem:

- *Mean-payoff*: Given a mean payoff game $\mathcal{T}_{\mathtt{MP}}$, is there a winning strategy for `Eve` in $\mathcal{T}_{\mathtt{MP}}$?
- *c-energy*: Given a c-energy game $\mathcal{T}_{E(c)}$, is there a winning strategy for `Eve` in $\mathcal{T}_{E(c)}$?
- *Energy*: Given an energy game $\mathcal{T}_E$, is there a winning strategy for `Eve` in $\mathcal{T}_E$?

We also consider the following related problem:

- *Unknown initial credit*: Given a WTG $\mathcal{T}$, is there a credit c such that `Eve` has a winning strategy in $\mathcal{T}_{E(c)}$?

We also consider these problems from `Adam`'s point of view.

To conclude this section we study the relations between mean payoff and energy games and state that all decision problems we have defined are undecidable for WTG.

2.6 Relations Between Mean-payoff and Energy Objectives

Obviously, if for some c the c-energy game is won by `Eve`, then the energy game is also won. In the other direction, if `Adam` has a winning strategy for the energy game then it is also winning for any c-energy game. In the finite state case the problem of energy and unknown initial credit are equivalent [19]: if `Eve` has a winning strategy for the energy game she has a memoryless one, and there is a bound on the maximum energy consumed by the outcomes of that strategy. This is not the case in general for WTG as demonstrated by the WTG $\mathcal{B}$ of Fig. 5, for which `Eve` wins the mean-payoff game and the energy game, but no c-energy game.

While energy and mean-payoff objectives are inter-reducible in the finite state case [19], the relationships between the two classes of objectives, formalized in the next two theorems, is more subtle for weighted timed games.

THEOREM 1. *Let $\mathcal{T}$ be a WTG. If `Eve` has a winning strategy $\sigma_\exists$ in the energy game $\mathcal{T}_E$ and $\sigma_\exists$ is immune from Zenoness, then $\sigma_\exists$ is a winning strategy in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$.*

PROOF. Let $\sigma_\exists$ be a strategy for `Eve` in $\mathcal{T}_E$. Let $\sigma_\forall$ be a strategy for `Adam`, and $\rho = \mathtt{Out}_{(\ell_0,\mathbf{0})}(\sigma_\exists, \sigma_\forall)$. We know that $\exists c.\ \forall n.\ c + \mathtt{w}(\rho_{\leq n}) \geq 0$. As ρ is non-Zeno,

1. for n sufficiently large, $\mathtt{d}(\rho_{\leq n}) > 0$ and
2. $\lim_{n\to\infty} \mathtt{d}(\rho_{\leq n}) = \infty$.

Hence for n sufficiently large, $\frac{\mathtt{w}(\rho_{\leq n})}{\mathtt{d}(\rho_{\leq n})} \geq \frac{-c}{\mathtt{d}(\rho_{\leq n})}$ and the limits are in the same order, this means that $\mathtt{MP}(\rho) \geq 0$. Therefore $\sigma_\exists$ is also winning in the mean-payoff game. □

Example 2. The following example shows that if we do not have immunity from Zenoness, the property no longer holds. In the game of Fig. 6, any play is winning for `Eve` in the c-energy game if $c > 1$. However, the total delay of a play is always smaller or equal to 1, hence the mean-payoff is smaller than -1, which means that `Eve` is losing.

We let $\mathcal{T}^{+\delta}$ be the game $\mathcal{T}$ in which we increase the weights of all locations by $\delta \in \mathbb{R}$. Formally $\mathcal{T}^{+\delta}$ is the WTG $\langle \mathtt{L}, \ell_\iota, X, T_\exists, T_\forall, \mathtt{Inv}, \mathtt{w}_{+\delta}\rangle$, where:

1. $\mathtt{w}_{+\delta}(\ell) = \mathtt{w}(\ell) + \delta$ if $\ell \in \mathtt{L}$;
2. $\mathtt{w}_{+\delta}(t) = \mathtt{w}(t)$ if $t \in T$.

THEOREM 2. *Let $\mathcal{T}$ be a WTG. If there exists $\delta > 0$, such that `Adam` has a winning strategy $\sigma_\forall$ in the energy game $\mathcal{T}_E^{+\delta}$ which is immune from Zenoness, then $\sigma_\forall$ is a winning strategy in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$.*

PROOF. Let $\sigma_\forall$ be a strategy which is winning the energy game $\mathcal{T}_E^{+\delta}$ and immune from Zenoness. Let $\sigma_\exists$ be a strategy for `Eve` and ρ be the outcome $\mathtt{Out}_{(\ell_\iota,\mathbf{0})}(\sigma_\exists, \sigma_\forall)$. We know that $\forall c.\ \exists n.\ c + \mathtt{w}_{+\delta}(\rho_{\leq n}) < 0$, so $\forall c.\ \exists n.\ c + \mathtt{w}(\rho_{\leq n}) + \mathtt{d}(\rho_{\leq n})\cdot\delta < 0$. We define the sequence $(n_k)_{k\in\mathbb{N}}$ by $n_0 = 0$ and given n_k we

choose n_{k+1} such that writing $c_k = \max\{-\mathtt{w}(\rho_{\leq n}) - \mathtt{d}(\rho_{\leq n}) \cdot \delta \mid n \leq n_k\}$ we have that $c_k + \mathtt{w}(\rho_{\leq n_{k+1}}) + \mathtt{d}(\rho_{\leq n_{k+1}}) \cdot \delta < 0$. Notice that the sequence n_k is strictly increasing. Since $\sigma_\forall$ is immune from Zenoness, after for k big enough, $\mathtt{d}(\rho_{\leq n_k}) > 0$. We have that $\frac{\mathtt{w}(\rho_{\leq n_k})}{\mathtt{d}(\rho_{\leq n_k})} < \frac{-c_k}{\mathtt{d}(\rho_{\leq n_k})} - \delta$. Hence $\liminf_{n\to\infty} \frac{\mathtt{w}(\rho_{\leq n})}{\mathtt{d}(\rho_{\leq n})} \leq -\delta$. This means that $\mathtt{MP}(\rho) < 0$, therefore $\tau_\forall$ is also winning in the mean-payoff game $\mathcal{T}$. □

Example 3. The following example shows that if we do not add this δ to the weight of locations, the property no longer holds. In the game of Fig. 7, for any initial credit c, `Adam` wins the c-energy game $\mathcal{T}_{E(c)}$. However `Eve` has a winning strategy in the mean-payoff game $\mathcal{T}_{\mathtt{MP}}$. She has to choose a delay which increases fast enough so that the weight of the play is small compared to its duration. For instance, if at the n-th step of the game, she chooses to delay for n^2 time units, the average weight of the play will be greater than $-\frac{1}{n}$. Hence it converges towards 0 and the mean-payoff is 0. Notice that if we add a small positive δ to the weight of each location, then following the same strategy, `Eve` also wins the c-energy game $\mathcal{T}_{E(c)}$ for c greater than $\frac{1}{\delta}$.

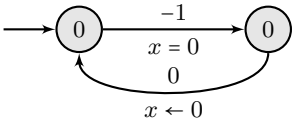

Figure 7: A WTG $\mathcal{T}$.

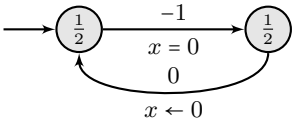

Figure 8: WTG $\mathcal{T}^{+\frac{1}{2}}$.

As already announced in the introduction, all the decisions problems that we have defined on weighted timed games are undecidable. The following theorem can be established using variants of techniques used in [18, 16]:

THEOREM 3. *The mean-payoff, c-energy, energy and unknown initial credit problems are undecidable for WTG.*

PROOF. The c-energy problem was already proven undecidable in [16] for one player (i.e. `Adam` does not control any transition). However in the one player case, [16] also proves that the unknown initial credit is in PSPACE. We will adapt the proof of undecidability of the c-energy problem for the other problems, making use of the extra player `Adam`.

We encode linear constraints for two clocks x and y which stay between 0 and 1 as illustrated by the example of Fig. 12.

A play entering the module $y \leq 2x$, in Fig. 11 has a reward of $2x - y$ each times it comes back to the first module $+x$. Hence it is winning for `Eve` for the energy objective only if $y \leq 2x$. It is easy to adapt this module to encode constraints of the type $a \cdot y \leq b \cdot x$ with $a, b \in \mathbb{N}$.

We will then use these linear constraints to encode a two-counter machine. We use two clocks x_1 and y_1 for counter c_1 and two clocks x_2 and y_2 for counter c_2. The value of counter c_1 is encoded by $\frac{1}{2^{x_1}}$ and c_2 by $\frac{1}{2^{x_2}}$. Clocks y_1 and y_2 are used to remember the value of clocks x_1 and x_2 before they are updated by increments and decrements so that we (i.e. `Adam`) can check that the operation was performed correctly.

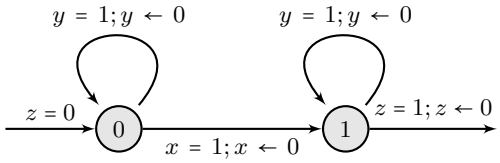

Figure 9: Module $+x$

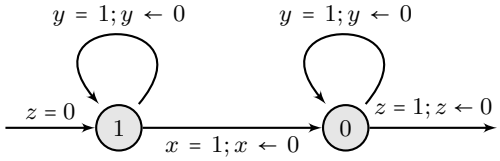

Figure 10: Module $+(1-x)$

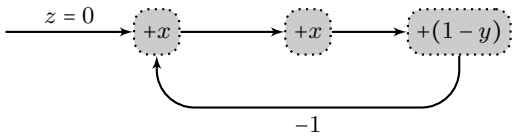

Figure 11: Module $y \leq 2x$

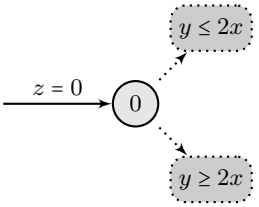

Figure 12: Game for checking $y = 2x$

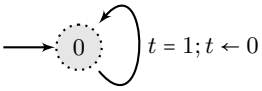

Figure 13: Accepting state q_f

Zero tests are easy to encode and we show how to increment a counter in Fig. 14. To decrement a counter, simply replace the module checking $2 \cdot x_1 = y_1$ in Fig. 14 by one checking $x_1 = 2 \cdot y_1$. Assume that we start in this module with $x_1 = y_1$. `Eve` has to choose when to reset x_1 to 0. In the second state `Adam` can choose to check if $x_1 = \frac{y_1}{2}$. Therefore `Eve` loses if she does not divide the valuation of x_1 by two, which means adding 1 to c_1. In the last state, we reset x_1 and y_1 at the same time so that the property $x_1 = y_1$ is satisfied when entering the next module.

Once the accepting state is reached, the weight of the play no longer decreases. Therefore if the machine terminates, there is a strategy for `Eve` that reaches this final state, and wins for the c-energy game where c is the number of execution steps of the machine times 2. This strategy is also winning for the mean payoff and the energy game.

Conversely if the machine does not terminate, either:

- the game stays in the states that encode the different instruction of the machine, like the one in Fig. 14, in that case there is a reward of −1 at each time step;

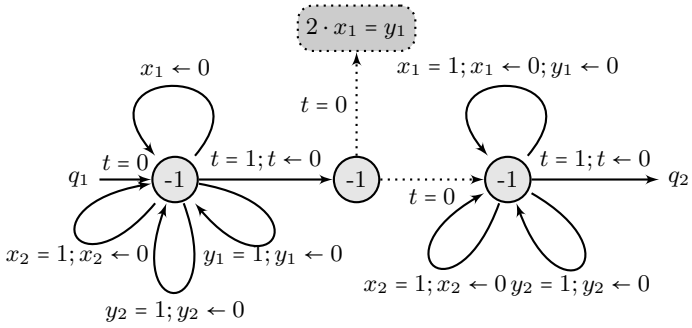

Figure 14: Module to encode an increment instruction $q_1 \xrightarrow{c_1++} q_2$

- or `Eve` has to "cheat" at some point which is the occasion for `Adam` to punish this by taking the edge going to a module that check the values of the clocks, like the one in Fig. 12.

Therefore `Eve` has no winning strategy in the energy game. Moreover the strategy of `Adam` we described is also winning in the mean-payoff game. □

3. A SEMI-PROCEDURE USING REGIONS

In this section, starting from the classical notion of regions [1], we define a finite concurrent game that exploits the relationship between timed paths and their projections in the region graph. We identify simple cycles in the region graph that are *good* for `Eve` (they roughly correspond to fragments of timed paths with *positive* reward), and others that are *good* for `Adam` (they roughly correspond to fragments of timed paths with *negative* reward). Thm. 4 tells us that if `Eve` can force to visit only her good cycles in the region graph then she has a winning strategy in the original energy timed game, and Thm. 5 is a symmetric result for `Adam`. To formalize those results, we need the notion of *quasi-path*: when we decompose a timed path according to simple cycles of the region graph, we introduce *jumps* inside regions because we remove a fragment of a timed path that starts inside a region and ends up in a possibly different state of the same region. Finally, we show how to solve the cycle forming game. This reduction is not complete: there are games which have winning strategies for `Eve` (or `Adam`) that our procedure will not find. However, we identify in Section 4 a class of games for which this reduction is complete.

3.1 Regions

We first recall the classical notion of *regions* [1]. If $k \in \mathbb{N}$, we write $\mathfrak{C}_k(X)$ for the set of constraints in $\mathfrak{C}(X)$ in which constants are integers within the interval $[\![0; k]\!]$. Let $\mathcal{T}$ be a WTG, and let $M = \max\{c \mid x \sim c \text{ is a constraint in } \mathcal{T}\}$. For $\delta \in \mathbb{R}_+$, we write $\lfloor \delta \rfloor$ the integral part of δ and $\mathrm{fr}(\delta)$ its fractional part. The equivalence relation $\equiv_{X,M}$ over $\mathbb{R}_+^X \times \mathbb{R}_+^X$ by $v \equiv_{X,M} v'$ if, and only if:

1. for all clocks $x \in X$, either $\lfloor v(x) \rfloor$ and $\lfloor v'(x) \rfloor$ are the same, or both $v(x)$ and $v(x)$ exceed M;
2. for all clocks $x, y \in X$ with $v(x) \leq M$ and $v(y) \leq M$, $\mathrm{fr}(v(x)) \leq \mathrm{fr}(v(y))$ if, and only if, $\mathrm{fr}(v'(x)) \leq \mathrm{fr}(v'(y))$;
3. for all clocks $x \in X$ with $v(x) \leq M$, $\mathrm{fr}(v(x)) = 0$ if, and only if, $\mathrm{fr}(v'(x)) = 0$;

This equivalence relation naturally induces a partition $\mathcal{R}_{X,M}$ of $\mathbb{R}_+^X$. We write $[v]_{X,M}$ ($[v]$ when X and M are fixed) for the equivalence class of $v \in \mathbb{R}_+^X$. An equivalence class is called a *region*. It is well known that this partition has the following properties:

1. it is compatible with the constraints in $\mathfrak{C}_M(X)$, i.e. for every $r \in \mathcal{R}_{X,M}$, and constraint $g \in \mathfrak{C}_M(X)$ either all valuations in r satisfy the clock constraint g, or no valuation in r satisfies it;
2. it is compatible with time elapsing, i.e. if there is $v \in r$ and $t \in \mathbb{R}_+$ such that $v + t \in r'$, then for all $v' \in r$ there is t' such that $v' + t' \in r'$;
3. it is compatible with resets, i.e. if $Y \subseteq X$ then if $[Y \leftarrow 0]r \cap r' \neq \varnothing$ then $[Y \leftarrow 0]r \subseteq r'$.

A region r is said to be *time-elapsing*, if for any $v \in r$ there is $t > 0$ such that $v + t \in r$. We write $\mathrm{Succ}(r)$ the *successors* of r by time elapsing, it is defined by $r' \in \mathrm{Succ}(r)$ if there is $v \in r$ and $t \geq 0$ such that $v + t \in r'$.

The number of region is bounded by $|X|! \cdot (4M + 4)^{|X|}$, note that this is exponential both in the number of clocks in X and in the size of the binary encoding of the maximal constant.

3.2 Region Game

Given an objective $\Omega \subseteq ((\mathtt{L} \times \mathcal{R}_{X,M}) \cdot \mathtt{Act})^\omega$, the *region game* associated with a WTG $\mathcal{T} = \langle \mathtt{L}, \ell_0, X, T_\exists, T_\forall, \mathtt{Inv}, \mathtt{w} \rangle$ is the concurrent game $\mathcal{R}(\mathcal{T}, \Omega) = \langle \mathtt{St}, \iota, \mathtt{Act}, \mathtt{Mov}, \mathtt{Tab}, \Omega \rangle$ where:

- $\mathtt{St} = \mathtt{L} \times \mathcal{R}_{X,M}$ and $\iota = (\ell_\iota, \mathbf{0})$ is the initial state;
- `Act` is the set of actions. They are either $\perp$ or of the form (r, e, a) where $r \in \mathcal{R}_{X,M}$, $e \in T_\exists \cup T_\forall$ is a transition, and $a \in \{\mathtt{head}; \mathtt{tail}\}$; intuitively, an action is a target region (abstract delay) and a discrete transition. The extra component in $\{\mathtt{head}; \mathtt{tail}\}$ is needed to determine who plays first when the two players choose the same abstract delay (target region).
- Let $s = (\ell, r)$. Action (r', e, a) belongs to $\mathtt{Mov}(s, \mathtt{Eve})$ (resp. $\mathtt{Mov}(s, \mathtt{Adam})$), if:
  1. $\exists (\ell, g, Y, \ell') \in T_\exists$ (resp. $T_\forall$);
  2. $r' \in \mathrm{Succ}(r)$;
  3. $r' \subseteq \mathtt{Inv}(\ell)$;
  4. $r' \cap g \neq \varnothing$;
  5. and $[Y \leftarrow 0]r' \subseteq \mathtt{Inv}(\ell')$.

  If there are no such action then only $\perp$ is allowed and this is the only situation in which $\perp$ is allowed.

- Let $s = (\ell, r)$ and $(r_\exists, e_\exists, a_\exists) \in \texttt{Mov}(s, \texttt{Eve})$, $(r_\forall, e_\forall, a_\forall) \in \texttt{Mov}(s, \texttt{Adam})$. $s' = \texttt{Tab}(s, (r_\exists, e_\exists, a_\exists), (r_\forall, e_\forall, a_\forall))$ is defined as follows:
  - if $r_\exists \neq r_\forall$, one region is a strict (time abstract) predecessor of the other (as they are both successors of r). If $r_\exists$ is a strict predecessor of $r_\forall$, Eve's action $(r_\exists, e_\exists, a_\exists)$ is selected and otherwise Adam's action $(r_\forall, e_\forall, a_\forall)$ is selected.
  - otherwise $r_\exists = r_\forall$ and two cases arise:
    1. either $r_\exists$ is not a time-elapsing region: in this case Adam's move is selected;
    2. or $r_\exists$ is a time-elapsing region; which move is selected then depends on the extra components $a_\exists, a_\forall \in \{\texttt{head}; \texttt{tail}\}$: if $a_\exists = a_\forall$ then Eve's move is selected and otherwise Adam's move is selected.

  Once an action (r', e, a) with $e = (\ell, g, Y, \ell')$ is selected, the resulting state is $s' = (\ell', r'')$ with $r'' = [Y \leftarrow 0]r'$.

Remark 1. *In case $\mathcal{T}$ is turn-based, then in each state of $\mathcal{R}(\mathcal{T}, \Omega)$ only one player has a choice. The region game can then be seen as a (classical) turn-based finite game.*

Example 4. We want to reduce the problem of finding winning strategies in a WTG to an equivalent problem in the region game. To illustrate the need for the extra component in the actions (i.e. $a \in \{\texttt{head}; \texttt{tail}\}$) consider the example of Fig. 15. In the WTG (left), Eve has no winning strategy to win the mean payoff game: Adam can always choose a delay shorter than her from ℓ_0 to enforce location ℓ_2. For the same reason Adam has also no winning strategy. In the region game (right), we abstract away from the actual delays Eve and Adam can propose: they have only one possible choice which is to propose to delay up-to region $0 < x < 1$. To reproduce the possibility that either Eve or Adam are able to propose the smallest delay, we use the choices of both players in $\{\texttt{head}, \texttt{tail}\}$.

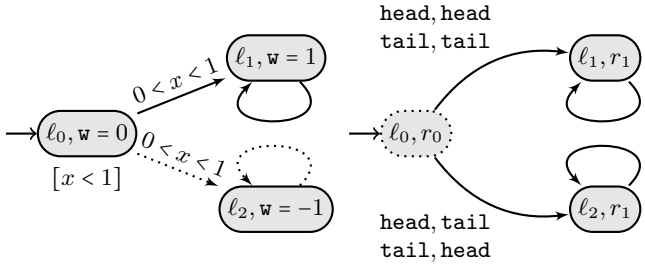

Figure 15: A WTG $\mathcal{T}$ and its associated region game.

3.3 Game Simulation

In this part, we show that the region game is a *game simulation* of the WTG.

Definition 5. Let $\mathcal{C}$ and $\mathcal{C}'$ be two games. A relation $\mathfrak{R}$ is a *game simulation between $\mathcal{C}$ and $\mathcal{C}'$* if for all states $s \in \texttt{St}, s' \in \texttt{St}'$ if $s \mathfrak{R} s'$ then:

$$\forall a_\exists \in \texttt{Mov}(s, \texttt{Eve}).\ \exists b_\exists \in \texttt{Mov}'(s', \texttt{Eve}).$$
$$\forall b_\forall \in \texttt{Mov}'(s', \texttt{Adam}).\ \exists a_\forall \in \texttt{Mov}(s, \texttt{Adam}).$$
$$\texttt{Tab}(s, a_\exists, a_\forall)\ \mathfrak{R}\ \texttt{Tab}'(s', b_\exists, b_\forall)$$

We say that $\mathcal{C}'$ *simulates* $\mathcal{C}$ if there exists such a relation $\mathfrak{R}$ with $\iota\ \mathfrak{R}\ \iota'$, where ι (resp. ι') is the initial state of $\mathcal{C}$ (resp. $\mathcal{C}'$). When $a_\exists$ and $b_\exists$ are witness of the above property we say that action $b_\exists$ *simulates action* $a_\exists$ in s. When the inverse relation is a game simulation between $\mathcal{C}'$ and $\mathcal{C}$, it is called a *game bisimulation*.

The following proposition shows that $\mathcal{R}(\mathcal{T}, \Omega)$ simulates $\mathcal{T}$.

Proposition 1. *The relation $\sqsubset$ defined by $(\ell, v) \sqsubset (\ell', r) \Leftrightarrow \ell = \ell' \wedge v \in r$ is a game bisimulation.*

We decompose the proposition in two lemmas.

Lemma 1. *The relation $\sqsubset$ is a simulation.*

Proof. Let $(\ell, v) \sqsubset (\ell, r)$, we have that $r = [v]$. Let $a_\exists \in \texttt{Mov}((\ell, v), \texttt{Eve})$. If $a_\exists = \bot$, then take $b_\exists = \bot$. Otherwise let $(d, e) = a_\exists$. We define $b_\exists$ to be $(r', e, \texttt{head})$ such that $v + d \in r'$. This action is allowed in $\mathcal{T}$, since r' is a successor of $[v]$ satisfying the invariant of ℓ, the guard of e and the invariant of the target of e after reset.

We now show that:

$$\forall b_\forall \in \texttt{Mov}'((\ell, r), \texttt{Adam}).\ \exists a_\forall \in \texttt{Mov}((\ell, v), \texttt{Adam}).$$
$$\texttt{Tab}'((\ell, r), b_\exists, b_\forall) \sqsubset \texttt{Tab}((\ell, v), a_\exists, a_\forall).$$

- If $b_\forall = \bot$ then we set $a_\forall = \bot$, as it is the only action allowed for Adam in (ℓ, v). $\texttt{Tab}'(b_\exists, \bot) = (r'[z \leftarrow 0], \ell'')$ where $b_\exists = (r', e, \texttt{head})$ and $e = (\ell, g, z, \ell'')$. By definition, $v + d \in r'$. Hence $v + d_\exists[z \leftarrow 0] \in r'[z \leftarrow 0]$ and $\texttt{Tab}((\ell, v), a_\exists, \bot) \in \texttt{Tab}'((\ell, r), b_\exists, \bot)$.
- If $b_\exists = \bot$ then $a_\exists = \bot$. Take $a_\forall = (d, e_\forall)$ such that $v + d \in r'$. By the same reasoning $\texttt{Tab}((\ell, v), \bot, a_\forall) \sqsubset \texttt{Tab}'((\ell, r), \bot, b_\forall)$.

Otherwise let $(r_\exists, e_\exists, a_\exists) = b_\exists$ and $(r_\forall, e_\forall, a_\forall) = b_\forall$. We set $a_\forall = (d_\forall, e_\forall)$ such that $v + d_\forall \in r_\forall$ and if $e_\exists$ is selected then $d_\exists < d_\forall$ and otherwise $d_\exists \geq d_\forall$. This is possible because if $e_\exists$ is selected then either $r_\forall$ is a (strict) successor of $r_\exists$ or $r_\forall = r_\exists$ and it is a time elapsing region; and if $e_\forall$ is selected either $r_\forall$ is a successor of $r_\exists$ or $r_\forall = r_\exists$. Then the same transition is applied, and therefore $\texttt{Tab}((\ell, v), a_\exists, a_\forall) \sqsubset \texttt{Tab}'((\ell, r), b_\exists, b_\forall)$. □

We define the inverse relation $\sqsupset$, by $a \sqsupset b \Leftrightarrow b \sqsubset a$.

Lemma 2. *The relation $\sqsupset$ is a simulation.*

PROOF. Let $(\ell, v) \sqsubset (\ell, r)$, we have that $r = [v]$. Let $a_\exists \in \mathtt{Mov}'((\ell, r), \mathtt{Eve})$. If $a_\exists = \bot$, then take $b_\exists = \bot$. Otherwise let $(r', e, a) = a_\exists$. We define $b_\exists$ to be (d, e) such that $v+d \in r'$. Such a d exists since r' has to be a successor of $[v]$. Moreover this action is allowed in $\mathcal{T}$.

We now show that:

$$\forall b_\forall \in \mathtt{Mov}((\ell, v), \mathtt{Adam}).\ \exists a_\forall \in \mathtt{Mov}'((\ell, r), \mathtt{Adam}).\ \mathtt{Tab}((\ell, v), b_\exists, b_\forall) \sqsubset \mathtt{Tab}'((\ell, r), a_\exists, a_\forall).$$

- If $b_\forall = \bot$ then we set $a_\forall = \bot$, as it is the only action allowed for $\mathtt{Adam}$ in s. $\mathtt{Tab}(b_\exists, \bot) = (v + d_\exists[Y \leftarrow 0], \ell'')$ where $b_\exists = (d_\exists, e)$ and $e = (\ell, g, Y, \ell'')$. By definition, $v + d_\exists \in r'$ where $a_\exists = (r', e, a)$. Hence $v + d_\exists[Y \leftarrow 0] \in r'[Y \leftarrow 0]$ and $\mathtt{Tab}((\ell, v), b_\exists, \bot) \sqsubset \mathtt{Tab}'((\ell, r), a_\exists, \bot)$.
- If $b_\exists = \bot$ then $a_\exists = \bot$. Take $a_\forall = (r', e_\forall, a)$ such that $v + d_\forall \in r'$. By the same reasoning than the first point $\mathtt{Tab}((\ell, v), \bot, b_\forall) \sqsubset \mathtt{Tab}'((\ell, r), \bot, a_\forall)$.

Otherwise let $(d_\exists, e_\exists) = b_\exists$ and $(d_\forall, e_\forall) = b_\forall$. We set $a_\forall = (r_\forall, e_\forall, a_\forall)$ such that $v + d_\forall \in r_\forall$ and if $e_\exists$ is selected then $a_\exists = a_\forall$ and otherwise $a_\exists = a_\forall$.

- If $[v + d_\exists]$ is not a successor of $[v + d_\forall]$, then $r_\exists < r_\forall$. Hence in both cases $e_\exists$ is selected.
- If $[v + d_\forall]$ is not a successor of $[v + d_\exists]$, then $r_\forall < r_\exists$. Hence in both cases $e_\forall$ is selected.

Otherwise $[v+d_\exists] = [v+d_\forall]$, then the player selected depends on whether $a_\exists = a_\forall$. Since $a_\forall$ have been chosen to make the player selected coincide, the same transition is applied. □

We say that a path $\rho' = s'_0 \xrightarrow{m'_0} s'_1 \xrightarrow{m'_1} \cdots$ simulates a play $\rho = s_0 \xrightarrow{m_0} s_1 \xrightarrow{m_1} \cdots$, if for all i, s'_i simulates s_i.

LEMMA 3. *Let $\mathcal{C}$ and $\mathcal{C}'$ be two games, s_0 a state of $\mathcal{C}$ and s'_0 a state of $\mathcal{C}'$. If $\mathcal{C}'$ simulates $\mathcal{C}$ and s'_0 simulates s_0 then for all strategies $\sigma_\exists$ in $\mathcal{C}$ there is a strategy $\sigma'_\exists$ in $\mathcal{C}'$ such that for all outcome ρ' of $\sigma'_\exists$ from s'_0 there is an outcome ρ of $\sigma_\exists$ from s_0 such that ρ' simulates ρ.*

PROOF. We define $\sigma'_\exists$ inductively. Assume we have defined it for all histories of length k, and that for all prefixes of length k of outcomes of $\sigma'_\exists$, there is a prefix of an outcome of $\sigma_\exists$ that is simulated by it. Given a history h' of length k, we choose h that is simulated by h', and choose $\sigma'_\exists(h')$ to simulate $\sigma_\exists(h)$. Now $\sigma'_\exists(h')$ simulates $\sigma_\exists(h)$, hence there exists an action of $\mathtt{Adam}$ such that $\mathtt{Tab}(h, \sigma_\exists(h), a_\forall)$ is simulated by $\mathtt{Tab}'(h, \sigma'_\exists(h), \sigma'_\forall(h'))$. This way the property will be verified for length $k + 1$.

This way, given $\rho' = \mathtt{Out}_s(\sigma'_\exists, \sigma'_\forall)$ for some strategy $\sigma'_\forall$, we can construct ρ outcome of $\sigma_\exists$ which is simulated by ρ' This shows that an outcome of $\sigma_\exists$ is simulated by ρ'. □

3.4 Quasi Paths

A *quasi path* in a WTG is a sequence of states and transitions $\rho = (\ell_0, v_0)\tau_0(\ell_1, v_1)\tau_1 \cdots \tau_{n-1}(\ell_n, v_n)$ such that for all $0 \leq i \leq n - 1$ either:

1. τ_i is a move $(a_\exists, a_\forall)$ and $(\ell_i, v_i) \xrightarrow{\mathtt{Mov}(a_\exists, a_\forall)} (\ell_{i+1}, v_{i+1})$;
2. or $\ell_i = \ell_{i+1}$ and $[v_i] = [v_{i+1}]$. In that case τ_i is called a *jump*. We will denote jumps by $\rightsquigarrow$.

A *quasi cycle* is a quasi path such that $(\ell_n, [v_n]) = (\ell_0, [v_0])$.

In order to extend the reward to quasi paths, we need a *weight function* $\delta : \mathtt{L} \times \mathcal{R}_{X,M} \mapsto \mathbb{R}$ which attributes a weight to jumps according to the region in which they happen. We define $\mathtt{w}_\delta$ the reward for each transition τ_i, depending on its type:

1. if τ_i is a move $(a_\exists, a_\forall)$ and $(\ell_i, v_i) \xrightarrow{\mathtt{Mov}(a_\exists, a_\forall)} (\ell_{i+1}, v_{i+1})$, then $\mathtt{w}_\delta((\ell_i, v_i)\tau_i(\ell_{i+1}, v_{i+1})) = d \cdot \mathtt{w}(\ell_i) + \mathtt{w}(e)$ where $(d, e) = \mathtt{Mov}(a_\exists, a_\forall)$;
2. otherwise $\tau_i = \rightsquigarrow$, then $\mathtt{w}_\delta((\ell_i, v_i) \rightsquigarrow (\ell_{i+1}, v_{i+1})) = \delta([\ell_i, v_i])$.

The reward for the quasi path ρ is then $\mathtt{w}_\delta(\rho) = \sum_{i<|\rho|} \mathtt{w}_\delta((\ell_i, v_i)\tau_i(\ell_{i+1},$

We define a projection from quasi paths to paths in the region game by forgetting jumps and projecting each state to its associated region. Formally, the projection π is defined inductively:

1. $\pi((\ell, v)) = (\ell, [v])$;
2. $\pi(h \rightsquigarrow (\ell, v)) = \pi(h)$;
3. $\pi\left(h \cdot (\ell, v) \xrightarrow{a_\exists, a_\forall} (\ell, v')\right) = \pi(h \cdot (\ell, v)) \xrightarrow{b_\exists, b_\forall} (\ell', [v'])$ where $a_p = (d_p, e_p)$ for $p \in \{\mathtt{Eve}, \mathtt{Adam}\}$, $b_\exists = ([v + d_\exists], e_\exists, \mathtt{head})$, and $b_\forall = ([v + d_\forall], e_\forall, \mathtt{head})$ if $d_\exists < d_\forall$ and $b_\forall = ([v + d_\forall], e_\forall, \mathtt{tail})$ otherwise.

It is naturally extended from histories to plays: ρ' is the projection of ρ if for all finite prefixes h of ρ, $\pi(h)$ is a prefix of ρ'. If h is a path in the region game, the path ρ is said *compatible* with h if $\pi(\rho) = h$, and we write $\gamma(h)$ for the set of path compatible with h.

We now associate to histories of WTG quasi paths by forgetting simple cycle like we did in the simple cycle decomposition. It is compatible with the stack of the projection on regions but contains more information.

Definition 6. (Timed Stack) The *timed stack* $\mathtt{tst}$ of a history is defined inductively as a quasi path. We set $\mathtt{tst}(\varepsilon) = \varepsilon$. Given a history $h' = h \xrightarrow{a_\exists, a_\forall} (\ell, v)$, and assuming $\mathtt{tst}$ has been defined for h:

1. if there is $i \leq |h|$ such that $\mathtt{tst}(h)_i = (\ell, v')$ with $v' \in [v]$, then we consider the greatest such i and set $\mathtt{tst}(h') = \mathtt{tst}(h)_{\leq i} \rightsquigarrow (\ell, v)$;

2. otherwise $\mathtt{tst}(h') = \mathtt{tst}(h) \xrightarrow{a_\exists, a_\forall} (\ell, v)$.

Notice that the timed stack can contain several states of the same region, hence i is not always unique. However, the states of the same region are all linked by jumps.

LEMMA 4. *For any history h, $\mathtt{tst}(h)$ is compatible with $\mathtt{st}(\pi(h))$.*

PROOF. We prove this by induction on the size of h. This is obvious when h only contains one state. We now assume the property is true for h and all its prefixes. We write $\mathtt{st}(\pi(h)) = r_0 \xrightarrow{m_0} \cdots \xrightarrow{m_{k-1}} r_k$. Consider now some history $h \xrightarrow{a_\exists, a_\forall} s$.

- if for all $i \in [\![0, k]\!]$, $s \notin r_i$, then by induction hypothesis on h, there is no index i such that h_i belongs to the same region than s. Then $\mathtt{tst}(h \xrightarrow{a_\exists, a_\forall} s) = \mathtt{tst}(h) \xrightarrow{a_\exists, a_\forall} s$ is a quasi path that is compatible with $\mathtt{st}(\pi(h \xrightarrow{a_\exists, a_\forall} s))$.
- otherwise there is i, n' such that $\mathtt{st}(\pi(h_{\leq n'})) = r_0 \xrightarrow{m_0} \cdots r_i$ and $s \in r_i$. $\mathtt{tst}(h \xrightarrow{a_\exists, a_\forall} s) = \mathtt{tst}(h)_{\leq i} \rightsquigarrow s = \mathtt{tst}(h_{\leq n'}) \rightsquigarrow s$. Since $\pi(s) = r_i$, $\mathtt{st}(\pi(h \xrightarrow{a_\exists, a_\forall} s) = r_0 \xrightarrow{m_0} \cdots r_i$. By induction hypothesis, $\mathtt{tst}(h_{\leq n'})$ is compatible with $\mathtt{st}(\pi(h_{\leq n'}))$, hence $\pi(\mathtt{tst}(h_{\leq n'}) \rightsquigarrow s) = r_0 \xrightarrow{m_0} \cdots r_i$. Which shows that $\mathtt{tst}(h \xrightarrow{a_\exists, a_\forall} s)$ is compatible with $\mathtt{st}(\pi(h \xrightarrow{a_\exists, a_\forall} s))$.

□

3.5 Reduction to the Region Game

Given a weight function $\delta : \mathtt{L} \times \mathcal{R}_{X,M} \mapsto \mathbb{R}$, we will write $\mathbb{C}^+_\delta$ for the set of simple cycles in the region game that only correspond to quasi cycles rewarding more than δ if δ is positive and more than 0 otherwise. Formally[1]:

$$\mathbb{C}^+_\delta = \{c \in \mathbb{C} \mid \forall \rho \in \gamma(c).\ \mathtt{w}_\delta(\rho) \geq \max\{\delta(\mathtt{first}(c)), 0\}\}.$$

Given a real number $\varepsilon > 0$, we write $\mathbb{C}^{-\varepsilon}_\delta$ for the simple cycles of the region game that correspond to quasi cycles with weight lower than δ and lower than $-\varepsilon$. Formally[2]:

$$\mathbb{C}^{-\varepsilon}_\delta = \{c \in \mathbb{C} \mid \forall \rho \in \gamma(c).\ \mathtt{w}_\delta(\rho) \leq \min\{\delta(\mathtt{first}(c)), -\varepsilon\}\}.$$

The winning condition of the region game will be given by *cycle objectives*. The intuition behind the definition of these objectives, is that if Eve can force the play to see only cycles with positive reward (i.e. in $\mathbb{C}^+_\delta$), the accumulated weight will be positive, except for a finite part. Which means she is winning the c-energy game, if c is big enough to cover the loss of energy in this finite part.

In the region game, we will consider the cases where the objective of Eve is given by $\Omega^+_\delta = \{\rho \mid \mathtt{dec}(\rho) \subseteq \mathbb{C}^+_\delta\}$. That is, she wins for plays whose decomposition in simple cycles only contains positive cycles.

We will prove in the next lemma that the remaining finite part in the cycle decomposition cannot have arbitrary big weight.

Given δ, we write δ^+ (resp. δ^-) the weight function δ where we replaced all the negative (resp. positive) value by 0, i.e. $\delta^+(r) = \max\{0, \delta(r)\}$ (resp. $\delta^-(r) = \min\{0, \delta(r)\}$).

LEMMA 5. *Let $\rho \in \mathtt{Play}(\mathcal{T})$, if $\mathtt{dec}(\pi(\rho)) \subseteq \mathbb{C}^+_\delta$ then there is a bound $W \in \mathbb{Z}$ such that for all n, $w_{\delta^+}(\mathtt{tst}(\rho_{\leq n})) \geq W$.*

Similarly if $\mathtt{dec}(\pi(\rho)) \subseteq \mathbb{C}^{-\varepsilon}_\delta$ then there is a bound $W \in \mathbb{Z}$ such that for all n, $w_{\delta^-}(\mathtt{tst}(\rho_{\leq n})) \leq W$.

Note that this is trivial when the game has only bounded transitions, but non-obvious in the general case.

PROOF. The prove the result for $\mathbb{C}^+_\delta$; the result for $\mathbb{C}^{-\varepsilon}_\delta$ can be proved the same way.

First notice that in the quasi paths of the form $\mathtt{tst}(\rho_{\leq n})$, the number of transitions that are not jumps is bounded, because the same region cannot appear twice unless it follows a jump. Moreover all the jumps have positive rewards (w.r.t. $\mathtt{w}_{\delta^+}$) since all the values of δ^+ are greater or equal to 0. The only solution left to have weights arbitrarily far below 0, is that there is a sequence of transitions in ρ whose weights grow arbitrarily big negatively. We write $\rho = (\ell_0, v_0) \xrightarrow{m_0} (\ell_1, v_1) \xrightarrow{m_1} \ldots$.

Toward a contradiction, assume

$$\forall i.\ \exists n_i.\ \mathtt{w}((\ell_{n_i}, v_{n_i}) \xrightarrow{m_{n_i}} (\ell_{n_i+1}, v_{n_i+1})) \leq -i. \qquad (1)$$

Then since the number of region is finite, there is a transition in the region game $(\ell, r) \xrightarrow{m} (\ell', r')$, such that $\pi\left(\rho_{n_i} \xrightarrow{m_{n_i}} \rho_{n_i+1}\right) = (\ell, r) \xrightarrow{m} (\ell', r')$ for an infinite number of i's. Since $(\ell, r) \xrightarrow{m} (\ell', r')$ is appears more than once in the projection of ρ, there is a simple cycle $c \in \mathtt{dec}(\pi(\rho))$ such that $(\ell, r) \xrightarrow{m} (\ell', r')$ appears in c, i.e. $c = c_1 \cdot (\ell, r) \xrightarrow{m} (\ell', r') \cdot c_2$ for some path $c_1 \cdot (\ell, r)$ and $(\ell', r') \cdot c_2$ in the region game. Let $\rho' \in \gamma(c)$, $\rho' = \rho^1 \cdot (\ell, v) \xrightarrow{m'} (\ell', v') \cdot \rho^2$ with $\pi\left((\ell, v) \xrightarrow{m'} (\ell', v')\right) = (\ell, r) \xrightarrow{m} (\ell', r')$. Then $\rho'' = \rho^1 \cdot (\ell, v) \rightsquigarrow (\ell_{m_p}, v_{m_p}) \xrightarrow{m_p} (\ell_{m_p+1}, v_{m_p+1}) \rightsquigarrow (\ell', v') \cdot \rho^2 \in \gamma(c)$, where we take m_p to be such that $(\ell_{m_p}, v_{m_p}) \xrightarrow{m_p} (\ell_{m_p+1}, v_{m_p+1})$ has weight strictly smaller than $-\mathtt{w}(\rho^1 \cdot (\ell, v)) - \delta(\ell, r) - \delta(\ell', r') - \mathtt{w}((\ell', v') \cdot \rho^2) + \delta(\mathtt{first}(c))$, this is possible because of hypothesis (1). We then have that $\mathtt{w}(\rho'') < \delta(\mathtt{first}(c))$. Since $\rho'' \in \gamma(c)$, this contradicts the fact that all the quasi cycle compatible with c are greater than $\delta(\mathtt{first}(c))$. □

[1]Note, that this definition is inductive: as a jump in the region (ℓ, r) gives a reward of $\delta(\ell, r)$, we make sure that a (quasi)-cycle on that region always provides a reward larger than or equal to this value.

[2]Note, that the definition for the *good* cycles of Adam is symmetric but slightly stronger as we require that the weight of (quasi)-cycles to be ε-bounded away from zero.

Given a WTG $\mathcal{T}$, let $W_T = \min_{t\in T}\{\mathtt{w}(t)\} \cup \{0\}$ and $W_{\mathtt{L}} = \min_{\ell\in \mathtt{L}}\{\mathtt{w}(\ell)\} \cup \{0\}$.

THEOREM 4. *Let $\mathcal{T}$ be a WTG, if* **Eve** *has a winning strategy in $\mathcal{R}(\mathcal{T}, \Omega^+_\delta)$ then:*

1. *she has a winning strategy $\tau_\exists$ in the energy game $\mathcal{T}_E$;*
2. *if $\mathcal{T}$ has bounded transitions, $\tau_\exists$ is a winning strategy in the energy game $\mathcal{T}_{E(c)}$ for the initial credit $c = |\mathtt{L} \times \mathcal{R}_{X,M}| \cdot (W_{\mathtt{L}} \cdot D + W_T)$;*
3. *if $\tau_\exists$ is immune from Zenoness, then it is winning in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$.*

PROOF. Let $\sigma_\exists$ be a winning strategy of **Eve** for the condition given by $\mathbb{C}^+_\delta$ from $(\ell_\iota, [v_0])$. Thanks to Lem. 2 and 3 there is a strategy $\tau_\exists$ such that for all outcome ρ of $\tau_\exists$, $\pi(\rho)$ is an outcome of $\sigma_\exists$.

We show that for all outcomes ρ of $\tau_\exists$, there is a value c_0 such that all the prefixes of ρ have a weight greater than $-c_0$. Let $\rho = (\ell_0, v_0) \xrightarrow{m_0} (\ell_1, v_1) \xrightarrow{m_1} \cdots$ be an outcome of $\tau_\exists$. We write $(d_n, e_n) = \mathtt{Mov}(m_n)$. We show by induction over n that $\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le n})) \le \mathtt{w}(\rho_{\le n})$. This is obviously true if $n = 0$. Assuming this is true for some $n \ge 0$, we show this is true for $n+1$.

- if $\pi(\rho_{n+1})$ does not appear in the stack $\mathtt{st}(\pi(\rho_{\le n}))$, then:
$$\begin{aligned}&\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le n+1}))\\ &\quad= \mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_n)) + d_n \cdot \mathtt{w}(\ell_n) + \mathtt{w}(e_n)\\ &\quad\le \mathtt{w}(\rho_{\le n}) + d_n \cdot \mathtt{w}(\ell_n) + \mathtt{w}(e_n) \text{ (by induction hyp.)}\\ &\quad\le \mathtt{w}(\rho_{\le n+1})\end{aligned}$$
- otherwise there is i such that $[\rho_i] = [\rho_{n+1}]$ and $\mathtt{tst}(\rho_{\le n}) = \mathtt{tst}(\rho_{\le i}) \xrightarrow{m'_0} s_1 \xrightarrow{m'_1} \cdots s_j$ and $s_j = \rho_n$. $\mathtt{dec}(\pi(\rho_{\le n+1})) = \mathtt{dec}(\pi(\rho_{\le i})) \cup \{\pi(\rho_i \xrightarrow{m'_0} s_1 \xrightarrow{m'_1} \cdots s_j \xrightarrow{m_n} \rho_{n+1})\}$ so $\pi(\rho_i \xrightarrow{m'_0} \cdots s_j \xrightarrow{m_n} \rho_{n+1})$ is in $\mathtt{dec}(\pi(\rho))$. Since $\sigma_\exists$ is a winning strategy, $\pi(\rho)$ is a winning play in the region game, therefore $\pi(\rho_i \xrightarrow{m'_0} \cdots s_j \xrightarrow{m_n} \rho_{n+1}) \in \mathbb{C}^+_\delta$. $\rho_i \xrightarrow{m'_0} \cdots s_j \xrightarrow{m_n} \rho_{n+1} \in \gamma(\pi(\rho_i \xrightarrow{m'_0} \cdots s_j \xrightarrow{m_n} \rho_{n+1}))$ so:
$$\mathtt{w}_\delta(\rho_i \xrightarrow{m'_0} \cdots s_j \xrightarrow{m_n} \rho_{n+1}) \ge \delta^+([\rho_i]) \qquad (2)$$
Now we can write:
$$\begin{aligned}&\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le n+1}))\\ &= \mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le i}) \rightsquigarrow \rho_{n+1})\\ &= \mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le i})) + \delta^+([\rho_{n+1}])\\ &\le \mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le i})) + \mathtt{w}_\delta(\rho_i \xrightarrow{m'_0} \cdots s_j \xrightarrow{m_n} \rho_{n+1}) \text{ by equation (2)}\\ &\le \mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le i}) \xrightarrow{m'_0} \cdots \xrightarrow{m'_{j-1}} \rho_n) + \mathtt{w}(\rho_n \xrightarrow{m_n} \rho_{n+1})\\ &\le \mathtt{w}(\rho_{\le n}) + \mathtt{w}(\rho_n \xrightarrow{m_n} \rho_{n+1}) \text{ by induction hypothesis}\\ &\le \mathtt{w}(\rho_{\le n+1})\end{aligned}$$

Now notice that in the quasi paths of the form $\mathtt{tst}(\rho_{\le n})$, the number of transitions that are not jumps is bounded, because the same region cannot appear twice unless it follows a jump. Moreover all the jumps have positive rewards since all the value of δ^+ are greater or equal to 0.

In the case where $\mathcal{T}$ has bounded transition, the total weight of the play is greater or equal to $|\mathtt{L} \times \mathcal{R}_{X,M}| \cdot (W_{\mathtt{L}} \cdot D + W_T)$. Therefore the strategy $\tau_\exists$ is winning the energy game $\mathcal{T}_{E(c)}$ for the initial credit $c = |\mathtt{L} \times \mathcal{R}_{X,M}| \cdot (W_{\mathtt{L}} \cdot D + W_T)$.

In the other case, Lem. 5 shows that there is a bound W such that for all n, $\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le n})) \ge W$. Hence for all n, $\mathtt{w}(\rho_{\le n}) \ge W$, and therefore ρ is in $\Omega(-W)$. This being true for all outcomes of $\tau_\exists$, $\tau_\exists$ is a winning strategy in the energy game.

Point 3 in Thm. 4 is a direct consequence of point 1 Thm. 4 and Thm. 1. □

REMARK 2. *We made the hypothesis that there exists a bound on the duration of transitions in order to get the result for the unknown initial credit. Consider the example of Fig. 5. In this game,* **Eve** *is winning in the region game for Ω^+_δ, and therefore by Thm. 4 she also wins in the energy game $\mathcal{T}_E$, by Thm. 1 she also wins the mean-payoff game $\mathcal{T}_{\mathtt{MP}}$ if we consider a strategy that is immune from Zenoness. However, the transition going out of ℓ_0 can be taken by* **Adam** *at any moment, its duration is not bounded. Indeed, whatever the initial credit is,* **Adam** *can force a play which costs more than this credit, by delaying the transition for long enough. Therefore* **Eve** *has no winning strategy for any fixed initial credit and the answer to the unknown initial credit problem is negative.*

We now consider the objective for **Eve**: $\Omega^{-\varepsilon}_\delta = \{\rho \mid \mathtt{dec}(\rho) \notin \mathbb{C}^{-\varepsilon}_\delta\}$. That is, she wins if the decomposition in simple cycle contains at least on simple cycle that is not below $-\varepsilon$.

THEOREM 5. *Let $\mathcal{T}$ be a WTG, if* **Adam** *has a winning strategy in $\mathcal{R}(\mathcal{T}, \Omega^{-\varepsilon}_\delta)$ then*

1. *he has a winning strategy in the energy game $\mathcal{T}_E$;*
2. *if $\mathcal{T}$ has bounded transitions, then* **Adam** *has a winning strategy in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$.*

PROOF. The proof is very similar to the proof of Thm. 4. The main difference is that instead of proving that for all outcomes ρ of $\tau_\exists$, and all n, $\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\le n})) \le \mathtt{w}(\rho_{\le n})$, we prove that for all outcomes ρ of $\tau_\forall$, and all n:
$$\mathtt{w}(\rho_{\le n}) \le \mathtt{w}_0(\mathtt{tst}(\rho_{\le n})) - \left(\frac{n - ||\mathtt{tst}(\rho_{\le n})||}{m+1} - 1\right) \cdot \varepsilon.$$
where $\mathtt{w}_0$ assign weight 0 to all regions, $m = |\mathtt{L}| \cdot |\mathcal{R}_{X,M}|$, and $||q||$ is the number of distinct elements in the timed stack.

This is proven by induction, it holds for $n = 0$ and assuming it holds for $n \ge 0$, we write $\rho = (\ell_0, v_0) \xrightarrow{m_0} (\ell_1, v_1) \xrightarrow{m_1} \ldots$ and $(d_n, e_n) = \mathtt{Mov}(m_n)$:

- if $\pi(\rho_{n+1})$ does not appear in the stack $\mathtt{st}(\pi(\rho_{\leq n}))$, then:

$$\begin{aligned}
&\mathtt{w}(\rho_{\leq n+1})\\
&\quad= \mathtt{w}(\rho_n) + d_n \cdot \mathtt{w}(\ell_n) + \mathtt{w}(e_n)\\
&\quad\leq \mathtt{w}_0(\mathtt{tst}(\rho_n)) - \left(\tfrac{n-\|\mathtt{tst}(\rho_{\leq n})\|}{m+1} - 1\right)\cdot\varepsilon\\
&\qquad + d_n \cdot \mathtt{w}(\ell_n) + \mathtt{w}(e_n)\\
&\quad\leq \mathtt{w}_0(\mathtt{tst}(\rho_{n+1})) - \left(\tfrac{n-\|\mathtt{tst}(\rho_{\leq n})\|}{m+1} - 1\right)\cdot\varepsilon\\
&\quad\leq \mathtt{w}_0(\mathtt{tst}(\rho_{n+1})) - \left(\tfrac{n+1-\|\mathtt{tst}(\rho_{\leq n+1})\|}{m+1} - 1\right)\cdot\varepsilon
\end{aligned}$$

- otherwise there is i such that $[\rho_i] = [\rho_{n+1}]$ and $\mathtt{tst}(\rho_{\leq n}) = \mathtt{tst}(\rho_{\leq i}) \xrightarrow{m_1} s_1 \cdots s_j$ and $s_j = \rho_n$.

$$\begin{aligned}
&\mathtt{w}(\rho_{\leq n+1})\\
&\quad= \mathtt{w}(\rho_n) + d_n \cdot \mathtt{w}(\ell_n) + \mathtt{w}(e_n)\\
&\quad\leq \mathtt{w}_0(\mathtt{tst}(\rho_n)) - \left(\tfrac{n-\|\mathtt{tst}(\rho_{\leq n})\|}{m+1} - 1\right)\cdot\varepsilon\\
&\qquad + d_n \cdot \mathtt{w}(\ell_n) + \mathtt{w}(e_n)\\
&\quad\leq \mathtt{w}_0\big(\mathtt{tst}(\rho_i \xrightarrow{t_1} \ldots \xrightarrow{t_{j-1}} \rho_n)\big) + d_n \cdot \mathtt{w}(\ell_n)\\
&\qquad + \mathtt{w}(e_n) - \left(\tfrac{n-\|\mathtt{tst}(\rho_{\leq n})\|}{m+1} - 1\right)\cdot\varepsilon\\
&\quad\leq \mathtt{w}_0(\mathtt{tst}(\rho_i)) - \varepsilon - \left(\tfrac{n-\|\mathtt{tst}(\rho_{\leq n})\|}{m+1} - 1\right)\cdot\varepsilon\\
&\quad\leq \mathtt{w}_0(\mathtt{tst}(\rho_i)) - \varepsilon - \left(\tfrac{n+1-\|\mathtt{tst}(\rho_{\leq n+1})\|}{m+1}\right)\cdot\varepsilon\\
&\quad\leq \mathtt{w}_0(\mathtt{tst}(\rho_{n+1})) - \left(\tfrac{n+1-\|\mathtt{tst}(\rho_{\leq n+1})\|}{m+1} - 1\right)\cdot\varepsilon
\end{aligned}$$

We then use Lem. 5 to conclude that $\tau_\forall$ is a winning strategy.

Point 2 in Thm. 5 is a direct consequence of point 1 of Thm. 5 and Thm. 2. □

3.6 Solving the Region Game

We now show how to solve a finite game with objective of the form $\Omega = \{\rho \mid \forall c \in \mathtt{dec}(\rho).c \in \mathbb{C}^W\}$ or $\Omega = \{\rho \mid \exists c \in \mathtt{dec}(\rho).c \in \mathbb{C}^W\}$. We can then apply this technique to solve the region game. To do so we unravel the game, and stop as soon as a cycle is formed. The play is then winning if the cycle formed belong to $\mathbb{C}^W$. This technique is adapted from [8].

Definition 7. Let $\mathcal{G} = \langle \mathtt{St}, \iota, \mathtt{Act}, \mathtt{Mov}, \mathtt{Tab}, \Omega\rangle$ be a concurrent game with $\Omega = \{\rho \mid \forall c \in \mathtt{dec}(\rho).c \in \mathbb{C}^W\}$ or $\Omega = \{\rho \mid \exists c \in \mathtt{dec}(\rho).c \in \mathbb{C}^W\}$. The unraveling of $\mathcal{G}$, written $\mathcal{U}(\mathcal{G})$, is the tuple $\langle \mathtt{St}', \iota, \mathtt{Act}, \mathtt{Mov}', \mathtt{Tab}', \Omega'\rangle$:

- the set of states is $\mathtt{St}' = \{h \in (\mathtt{St}\cdot(\mathtt{Act}\times\mathtt{Act}))^*\cdot\mathtt{St} \mid \forall i,j \neq i.\ h_i \neq h_j\} \cup \{\uparrow,\downarrow\}$, the set of histories of the original game where all states appear at most once; with the addition of a winning state $\uparrow$ and a loosing state $\downarrow$ for Eve;
- $\mathtt{Mov}'(h,p) = \mathtt{Mov}(\mathtt{last}(h),p)$;
- for an history h and a move $(a_\exists, a_\forall)$, let $s = \mathtt{Tab}(\mathtt{last}(h), a_\exists, a_\forall)$:
  1. if s does not appear in h then $\mathtt{Tab}'(h, a_\exists, a_\forall) = h \xrightarrow{a_\exists, a_\forall} s$;
  2. otherwise, let i be such that $h_i = s$, and $c = h_{\geq i} \xrightarrow{a_\exists, a_\forall} s$ (notice that such a i is unique): if c belongs to $\mathbb{C}^W$ then $\mathtt{Tab}'(h, a_\exists, a_\forall) = \uparrow$ and otherwise $\mathtt{Tab}'(h, a_\exists, a_\forall) = \downarrow$.

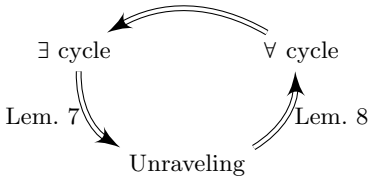

Figure 16: Winning strategies of Eve.

  Then, from $\uparrow$ and $\downarrow$, there are only self loops, thus $\mathtt{Tab}'(x, a_\exists, a_\forall) = x$ for $x \in \{\uparrow,\downarrow\}$.
- the objective is to reach $\uparrow$, i.e. $\Omega' = (\mathtt{St}'\cdot(\mathtt{Act}\times\mathtt{Act}))^*\cdot(\uparrow\cdot(\mathtt{Act}\times\mathtt{Act}))^\omega$.

THEOREM 6. *Let $\mathcal{G}$ be a concurrent game and $\mathcal{U}(\mathcal{G})$ its unraveling. Then Eve has a winning strategy in the unraveled game $\mathcal{U}(\mathcal{G})$ if, and only if, she has a winning strategy in $\mathcal{G}$.*

In the following we will call objectives of the form $\{\rho \mid \forall c \in \mathtt{dec}(\rho).\ c \in \mathbb{C}^W\}$ *universal cycle objective* and objectives of the $\{\rho \mid \exists c \in \mathtt{dec}(\rho).\ c \in \mathbb{C}^W\}$ *existential cycle objective.*

We show the following lemma wich implies Thm. 6.

LEMMA 6. *Let $\mathcal{G}$ be a concurrent game and $\mathcal{U}(\mathcal{G})$ its unraveling. The three following properties are equivalent:*

1. *Eve has a winning strategy in the unraveled game $\mathcal{U}(\mathcal{G})$;*
2. *Eve has a winning strategy in $\mathcal{G}$ for the existential cycle objective;*
3. *Eve has a winning strategy in $\mathcal{G}$ for the universal cycle objective;*

The implication from universal cycle objective to the existential one is obvious. We decompose the remaining of the proof in two lemmas as represented in Fig. 16.

LEMMA 7. *If Eve has a winning strategy in $\mathcal{G}$ for the existential cycle condition $\mathbb{C}^W$, she has a winning strategy in the unraveled game $\mathcal{U}(\mathcal{G})$.*

PROOF. We will show that if Eve has no winning strategy in $\mathcal{U}(\mathcal{G})$, then she has none in $\mathcal{G}$, by considering the attractor of $\uparrow$. The idea is that if the stack of the play in $\mathcal{G}$ is not in this attractor, then Eve cannot enforce the next simple cycle in the decomposition to be positive. The *attractor* of $\uparrow$, written $\mathtt{Attr}_\exists(\uparrow)$ is defined as $\mathtt{Attr}_\exists(\uparrow) = \{h \in \mathtt{St}^* \mid \exists\sigma_\exists.\ \forall\sigma_\forall.\ \exists n.\ \mathtt{Out}_s(\sigma_\exists,\sigma_\forall)_n = \uparrow\}$. This attractor set can be constructed iteratively, consider $\mathtt{Attr}^0_\exists(\uparrow) = \uparrow$ and $\mathtt{Attr}^{k+1}_\exists(\uparrow) = \{h \in \mathtt{St}^* \mid \exists a_\exists.\ \forall a_\forall.\ h\cdot\mathtt{Tab}(\mathtt{last}(h), a_\exists, a_\forall) \in \mathtt{Attr}^k_\exists(\uparrow)\}$, a fixed point is reached after $n \leq |\mathtt{St}|$ number of steps, and $\mathtt{Attr}^n_\exists(\uparrow) = \mathtt{Attr}_\exists(\uparrow)$. Assuming that Eve has no winning strategy in $\mathcal{U}(\mathcal{G})$, the initial state is not in the attractor.

Fix a strategy $\sigma_\exists$ in $\mathcal{G}$, let h be a history in $\mathcal{G}$ and $l = \mathtt{last}(h)$. Notice that if $\mathtt{st}(h) \notin \mathtt{Attr}_\exists(\uparrow)$ then there is an

action $a_\forall$ such that writing $s = \mathtt{Tab}(\ell, \sigma_\exists(h), a_\forall)$, we have that $\mathtt{st}(h \xrightarrow{\sigma_\exists(h), a_\forall} s) \notin \mathtt{Attr}_\exists(\uparrow)$ and if there is i such that $\mathtt{st}(h)_i = s$, then $\mathtt{st}(h)_{\geq i} \xrightarrow{\sigma_\exists(h), a_\forall} s \in \mathbb{C}^W$. We construct a strategy $\sigma_\forall$, that for all history h such that $\mathtt{st}(h) \notin \mathtt{Attr}_\exists(\uparrow)$, chose an action $a_\forall$ witness of the previous property.

It is easy to show by induction that the outcome of $(\sigma_\exists, \sigma_\forall)$ stays outside the attractor: this is because the initial state is outside of it and when a cycle is formed, the stack comes back to a prefix of the current one, and since (by induction hypothesis) along the history no state of the attractor has been seen, we are still outside the attractor. Therefore in the decomposition of the outcome, all the simple cycles are in $\mathbb{C} \setminus \mathbb{C}^W$. This contradicts the fact that the strategy of Eve is winning in $\mathcal{G}$ for the existential cycle condition. □

LEMMA 8. *If Eve has a winning strategy in the unraveling of $\mathcal{G}$ from s, she has a winning strategy in $\mathcal{G}$ for the universal cycle condition $\mathbb{C}^W$.*

PROOF. Let $\nu_\exists$ be a winning strategy in the unraveling. We define a strategy $\sigma_\exists$ in the original game by simply forgetting cycles. It therefore plays according to the element that are in the stack of the decomposition process: $\sigma_\exists(h) = \nu_\exists(\mathtt{st}(h))$. Let ρ be an outcome of $\sigma_\exists$, we show that it is winning. Let c be a cycle appearing in the decomposition in simple cycles of ρ. Let k be the first index such that $\mathtt{last}(\mathtt{dec}(\rho_{\leq k})) = c$.

We write $\mathtt{st}(\rho_{\leq k-1}) = s_0 \xrightarrow{m_0} \cdots s_k$, and we will prove by induction over $n \in [\![0, k]\!]$ that the sequence of states in the stack forms a history compatible with $\nu_\exists$. For the case s_0 this is simply because the initial state will never disappear from the stack. We now assume that the property holds until some index $n \in [\![0, k-1]\!]$. We write i the last occurrence of s_n in ρ before position k, that is $i = \max\{j \leq k-1 \mid \rho_j = s_n\}$.

We show that $\rho_{i+1} = s_{n+1}$. Since $i+1 \leq k$, state ρ_{i+1} is added to the stack and will only disappear before index k if a cycle is formed with some state s_j with $j \leq n+1$:

- if $j < n$ then s_n is also removed from the stack and since it does not appear after i this contradicts the fact that s_n is still present on the stack of $\rho_{\leq k}$;
- $j = n$ contradicts the fact that i was the last appearance of s_n before position k;
- otherwise $j = n+1$, and in that case, the same state ρ_{i+1} is still on top of the stack once the cycle has been removed.

Therefore $s_{n+1} = \rho_{i+1}$.

Now, since ρ is an outcome of $\sigma_\exists$, there is some $a_\forall$ such that $\mathtt{Tab}(\rho_i, \sigma_\exists(\rho_{\leq i}), a_\forall) = \rho_{i+1}$. By definition, $\sigma_\exists(\rho_{\leq i}) = \nu_\exists(\mathtt{st}(\rho_{\leq i}))$ and then:

$$\begin{aligned}\mathtt{Tab}'(s_0 \xrightarrow{m_0} \cdots s_n, \nu_\exists(s_0 \xrightarrow{m_0} \cdots s_n), a_\forall) &= \mathtt{Tab}(s_n, \sigma_\exists(\rho_{\leq i}), a_\forall)\\ &= \rho_{i+1}\\ &= s_{n+1}\end{aligned}$$

Hence $s_0 \xrightarrow{m_0} \cdots s_{n+1}$ is an outcome of $\nu_\exists$.

This proves that $\mathtt{st}(\rho_{\leq k-1})$ is an outcome of $\nu_\exists$. We now have to complete the cycle. There is $a_\forall$ such that:

$$\mathtt{Tab}(\rho_{k-1}, \sigma_\exists(\rho_{\leq k-1}), a_\forall) = \rho_k.$$

And ρ_k appears in $\mathtt{st}(\rho_{\leq k-1})$ to form the cycle c. Since $\nu_\exists(\mathtt{st}(\rho_{\leq k-1})) = \sigma_\exists(\rho_{\leq k-1})$, we have that:

$$\mathtt{Tab}'(\mathtt{st}(\rho_{\leq k-1}), \nu_\exists(\mathtt{st}(\rho_{\leq k-1})), a_\forall) = \begin{cases} \uparrow & \text{if } c \in \mathbb{C}^W \\ \downarrow & \text{otherwise} \end{cases}.$$

Since $\nu_\exists$ is a winning strategy the final state is necessarily $\uparrow$, meaning that c belongs to $\mathbb{C}^W$. This shows that $\sigma_\exists$ is a winning strategy in $\mathcal{G}$. □

We are now ready to give an algorithm that solves game with cycle objectives. Its complexity depends on the way we represent the cycle objective, as an explicit representation can be exponentially larger than the size of the game. To be as general as possible, we assume that we have an algorithm to tell us if a cycle is in $\mathbb{C}^W$, which works using space at most $p(|\mathtt{St} + \mathtt{Act}|)$ where p is a function from $\mathbb{N}$ to $\mathbb{N}$.

THEOREM 7. *Given a concurrent game $\mathcal{G}$ with (existential or universal) cycle objective $\mathbb{C}^W$, we can decide in space $\mathcal{O}(|\mathtt{St}| \cdot |\mathtt{Act}|^2) + p(|\mathtt{St} + \mathtt{Act}|)$ whether Eve has a winning strategy.*

PROOF. The algorithm proceeds by an exploration of the unraveling. We give an inductive procedure win. Given a state of the unraveling h:

- if h contains a cycle c, we ask if $c \in \mathbb{C}^W$ and answer **true** if it does and **false** otherwise; this is done in space $p(|\mathtt{St} + \mathtt{Act}|)$;
- otherwise we successively compute win for each history $h \cdot s$ with $s = \mathtt{Tab}(\mathtt{last}(h), a_\exists, a_\forall)$ where $a_\exists \in \mathtt{Mov}(\mathtt{last}(h), \mathtt{Eve})$ and $a_\forall \in \mathtt{Mov}(\mathtt{last}(h), \mathtt{Adam})$, and keep only the result each time; now we check that:

  $$\exists a_\exists.\ \forall a_\forall.\ \mathsf{win}\,(h \cdot \mathtt{Tab}(\mathtt{last}(h), a_\exists, a_\forall))\,;$$

  This can be done in space $\mathcal{O}(|\mathtt{Act}|^2)$.

The depth of the computation is at most $|\mathtt{St}| + 1$ and each step is done using space bounded by $\mathcal{O}(|\mathtt{Act}|^2)$, except at the leafs where we use space $p(|\mathtt{St} + \mathtt{Act}|)$. Hence our algorithm works in space $\mathcal{O}(|\mathtt{St}| \cdot |\mathtt{Act}|^2) + p(|\mathtt{St} + \mathtt{Act}|)$. □

THEOREM 8. *Given a finite concurrent game $\mathcal{G}$ with objective $\Omega = \{\rho \mid \forall c \in \mathtt{dec}(\rho).c \in \mathbb{C}^W\}$ or $\Omega = \{\rho \mid \exists c \in \mathtt{dec}(\rho).c \in \mathbb{C}^W\}$ where $\mathbb{C}^W$ is given by an automaton, deciding if Eve has a winning strategy is* PSPACE*-complete.*

We assume that the set of cycles is given by an (non-deterministic) automaton $\mathcal{A}$ whose language is $\mathbb{C}^W$.

PROOF. We use the algorithm of Thm. 7. Determining if a cycle is accepted by $\mathcal{A}$ can be done in polynomial space by doing a subset construction on the fly. The problem is therefore in PSPACE.

The hardness proof is an encoding of QBF. Assume we are given a formula $\phi = \exists x_1.\ \forall x_2.\ \ldots \forall x_n.\ C_1 \wedge \cdots C_m$ with for all j, $C_j = (l_{j,1} \vee l_{j,2} \vee l_{j,3})$ and $l_{i,j}$ is of the form x_k or $\neg x_k$ for some k. We describe a game $\mathcal{G}$. In state s_0, `Eve` chooses to go to either state x_1 or state $\neg x_1$, then we go to s_1 and `Adam` chooses to go to either x_2 or $\neg x_2$, and we repeat this until s_n, then `Adam` chooses a clause C_j and we come back to s_0. This forms a cycle $s_0 \cdot l_1 \cdot s_1 \cdot l_2 \cdots l_n \cdot s_n \cdot C_j \cdot s_0$ where $l_i = x_i$ or $\neg x_i$. Notice that in this proof we only write states of the path and not moves, this is in order to simplify the notations. The objective $\mathbb{C}^W$ for `Eve` is given as the language:

$$\bigcup_{j \in [\![1,m]\!]} \bigcup_{i \in [\![1,3]\!]} \mathtt{St}^* \cdot l_{i,j} \cdot \mathtt{St}^* \cdot C_j.$$

This can be described by a non-deterministic automaton of polynomial size. Then `Eve` has a winning strategy in $\mathcal{G}$ if, and only if, the formula ϕ is valid. □

4. ROBUST GAMES

The reduction to the cycle forming game in the region graph is complete when there exists a weight function δ, that partitions the set of simple cycles of the region into good ones for `Eve` and good ones for `Adam`.

Definition 8. (Robust game) A WTG is said *δ-robust* if $\mathbb{C} = \mathbb{C}^+_\delta \cup \mathbb{C}^{-\varepsilon}_\delta$ for some ε. We simply call a WTG *robust* when there exists $\delta : \mathcal{R}_{X,M} \mapsto \mathbb{R}$ such that it is δ-robust.

REMARK 3. *Note that a WTG $\mathcal{G}$ where all the costs are discrete (i.e. $\forall \ell \in L.\ w(\ell) = 0$) is robust for $\delta = 0$ and $\varepsilon < 1$. The results of this section implies decidability of the energy problem and mean payoff problem for this class.*

REMARK 4. *If $\mathcal{T}$ is robust, then the leafs of $\mathcal{U}(\mathcal{R}(\mathcal{T}))$ are partitioned between winning for `Eve` and winning for `Adam`. If, in addition, this game is turn-based, then it is determined. By Thm. 4 and Thm. 10, we can conclude that if $\mathcal{T}$ is robust and turn-based then the energy game $\mathcal{T}_E$ is determined.*

Now, we establish that in a robust WTG, we can decide if `Eve` can win the energy game. This is a consequence of the following theorem which complements Thm. 4. A symmetric result also holds for `Adam`.

THEOREM 9. *Let $\mathcal{T}$ be a δ-robust WTG:*

1. *if `Eve` has a winning strategy in the energy game $\mathcal{T}_E$ then she has a winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega^+_\delta))$;*
2. *if $\mathcal{T}$ has bounded transitions and `Eve` has a winning strategy in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$ then she has a winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega^+_\delta))$.*

PROOF. Let $\tau_\exists$ be a strategy in $\mathcal{T}$. Thanks to Lem. 1 and 3, there is a strategy $\sigma_\exists$ such that for any outcome ρ of $\sigma_\exists$ there is an outcome ρ' of $\tau_\exists$ such that $\pi(\rho') = \rho$.

We will prove that `Eve` has no winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega^+_\delta))$, then she has none in the energy game $\mathcal{T}_E$; and in the case where $\mathcal{T}$ has bounded transitions she has no winning strategy in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$ either.

Assume that `Eve` has no winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega^+_\delta))$, according to Lem.7, she has no winning strategy for the existential cycle objective $\mathbb{C}^+_\delta$. Therefore there is an outcome ρ of $\tau_\exists$, such that $\mathtt{dec}(\pi(\rho))$ only contains cycles of $\mathbb{C} \setminus \mathbb{C}^+_\delta$. As $\mathcal{T}$ is robust, $\mathbb{C} \setminus \mathbb{C}^+_\delta = \mathbb{C}^{-\varepsilon}_\delta$, and each cycle decreases the weight of at least ε. Therefore for all index n:

$$\mathtt{w}(\rho'_{\leq n}) \leq \mathtt{w}_{\delta^+}(\mathtt{tst}(\rho'_{\leq n})) - \varepsilon \cdot \left(\frac{n}{|\mathrm{L}| \cdot |\mathcal{R}_{X,M}|} - 1 \right).$$

Thanks to Lem. 5, $\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\leq n}))$ is bounded, so the weight of ρ decreases infinitely low. This contradicts the fact that $\tau_\exists$ is a winning strategy in $\mathcal{T}_E$.

Assuming now that $\mathcal{T}$ has bounded transitions, the bound D is such that:

$$\frac{\mathtt{w}(\rho_{\leq n})}{\mathtt{d}\rho_{\leq n}} \leq \frac{\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\leq n}))}{\mathtt{d}\rho_{\leq n}} - \varepsilon \cdot \left(\frac{1}{D \cdot |\mathrm{L}| \cdot |\mathcal{R}_{X,M}|} - \frac{1}{n} \right).$$

Thanks to Lem. 5, $\frac{\mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\leq n}))}{\mathtt{d}(\rho_{\leq n})} + \frac{\varepsilon}{n}$ converges to 0. The mean payoff is therefore strictly below 0, this contradicts that $\tau_\exists$ is a winning strategy for the mean payoff game $\mathcal{T}_{\mathtt{MP}}$. □

THEOREM 10. *Let $\mathcal{T}$ be a δ-robust WTG:*

1. *if `Adam` has a winning strategy in the energy game $\mathcal{T}_E$, then he has a winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega^{-\varepsilon}_\delta))$;*
2. *if $\mathcal{T}$ has bounded transitions and `Adam` has a winning strategy in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$, then he has a winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega^{-\varepsilon}_\delta))$.*

PROOF. Similarly to Thm. 9, we assume that `Adam` has no winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega^{-\varepsilon}_\delta))$, and prove that he has none in the energy game $\mathcal{T}_E$; and in the case where $\mathcal{T}$ has bounded transitions he has no winning strategy in the mean payoff game $\mathcal{T}_{\mathtt{MP}}$ either. Let $\tau_\forall$ be a strategy of `Adam`. Using the same arguments than for Thm. 9, there is an outcome ρ of $\tau_\forall$ such that $\mathtt{dec}(\pi(\rho))$ only contains cycles of $\mathbb{C}^+_\delta$. We therefore have that for all n, $\mathtt{w}(\rho_{\leq n}) \geq \mathtt{w}_{\delta^-}(\mathtt{tst}(\rho_{\leq n}))$. We showed in Lem. 5 that $\mathtt{w}_{\delta^-}(\mathtt{tst}(\rho_{\leq n}))$ cannot go arbitrarily far below 0. Hence $\tau_\forall$ is not winning the energy game $\mathcal{T}_E$.

Now, if $\mathcal{T}$ has bounded transitions, $\frac{\mathtt{w}(\rho_{\leq n})}{\mathtt{d}(\rho_{\leq n})} \geq \mathtt{w}_{\delta^+}(\mathtt{tst}(\rho_{\leq n})) \cdot \frac{D}{n}$. The mean payoff is therefore greater or equal to 0. This means that $\tau_\forall$ is not winning the mean payoff game $\mathcal{T}_M$. □

Given a weighted timed game, it is decidable whether this game is robust or not.

THEOREM 11. *The membership problem for the class of robust game is decidable.*

To solve the membership problem, we encode the condition on simple cycles as a formula of the theory of real closed fields, for which there is a decision procedure in EXPSPACE [7]. The size of the formula we obtain can be doubly exponential, and therefore our algorithm is 3EXPSPACE.

THEOREM 12. *Deciding if a WTG $\mathcal{T}$ is robust can be done in* 3EXPSPACE.

For a weight function δ, we will denote the weight $\delta(\ell, r)$ by the variable $\delta_{\ell,r}$.

Given a simple cycle $c = (\ell_0, r_0) \xrightarrow{(r^0_\exists, e^0_\exists, a^0_\exists), (r^0_\forall, e^0_\forall, a^0_\forall)} (\ell_1, r_1) \xrightarrow{p_1} \cdots (\ell_n, r_n)$ of the region game, a quasi path $h \in \gamma(c)$ is of the form $(\ell_0, v^0_0) \rightsquigarrow \cdots \rightsquigarrow (\ell_0, v^{j_0}_0) \xrightarrow{(d^0_\exists, e^0_\exists), (d^0_\forall, e^0_\forall)} (\ell_1, v^0_1) \cdots (\ell_n, v^{j_n}_n)$. Where for all $i \in [\![1, n]\!]$, j_i is the number of jumps in the region r_i, and for all $j \in [\![0, j_i]\!]$, v^j_i is in the region r_i. We write $(d_i, e_i) = \texttt{Mov}((d^0_\exists, e^0_\exists), (d^0_\forall, e^0_\forall))$.

We have the constraint on the number of jumps:

$$\psi_i = j_i \geq 0 \wedge \left(v^0_i(x) \neq v^{j_i}_i(x) \implies j_i \geq 1\right).$$

The fact that the correct transition is selected at each step is expressed by:

$$\varsigma_i = \begin{cases} d^i_\exists < d^i_\forall \wedge d_i = d^i_\exists & \text{if } \texttt{Eve} \text{ is selected in } (r^i_\exists, e^i_\exists, a^i_\exists), (r^i_\forall, e^i_\forall, a^i_\forall) \\ d^i_\exists < d^i_\forall \wedge d_i = d^i_\exists & \text{otherwise} \end{cases}$$

The transition is correctly applied if $[z \leftarrow 0]v^{j_i}_i + d_i = v^1_{i+1}$. This is expressed by:

$$\xi_i = \bigwedge_{x \in X} \begin{cases} v^{j_i}_i(x) + d_i = v^1_{i+1}(x) & \text{if } x \notin z \\ v^1_{i+1}(x) = 0 & \text{otherwise} \end{cases}$$

Let $g \in \mathfrak{C}(X)$, it is of the form $x_1 \bowtie c_1 \wedge \cdots \wedge x_k \bowtie c_k$. The fact that a valuation $v + d$ ($v \in \mathbb{R}^X$ and $d \in \mathbb{R}$) satisfies the guard g is expressed by the formula $\phi(v, d, g) = v_{x_1} + d \bowtie c_1 \wedge \cdots \wedge v_{x_k} + d \bowtie c_k$. Each transition is allowed if $v^{j_i}_i + d_i \vDash \texttt{Inv}(\ell_i)$, $v^{j_i}_i + d_i \vDash g$ and $v^0_{i+1} + d_i \vDash \texttt{Inv}(\ell_{i+1})$. This is expressed by the conjunction:

$$\phi_i = \phi\left(v^{j_i}_i, d_i, \texttt{Inv}(\ell_i)\right) \wedge \phi\left(v^{j_i}_i, d_i, g\right) \wedge \phi\left(v^0_{i+1}, 0, \texttt{Inv}(\ell_{i+1})\right).$$

The weight of this path is expressed by the polynomial $p_c = j_1 \cdot \delta_{\ell_1, r_1} + d_1 \cdot \texttt{w}(\ell_1) + \texttt{w}(e_1) + \cdots + j_n \cdot \delta_{\ell_n, r_n}$.

To each region is associated set of constraints on clocks that are satisfied exactly by the valuation belonging to that region. We can translate them to a formula $\chi(r, v)$. The fact that the history is compatible with c is expressed by for all i and $j \in [\![1, j_i]\!]$: $\chi(r_i, v^j_i)$. Note that for $j \neq 1, j_i$ the variables v^j_i have not appeared yet and it is therefore unnecessary to put constraint on these. We therefore write

$$\chi_i = \chi(r_i, v^0_i) \wedge \chi(r_i, v^{j_i}_i) \wedge \chi(r_i, v^{j_i}_i) \wedge \chi(r^i_\exists, v^{j_i} + d^i_\exists) \wedge \chi(r^i_\forall, v^{j_i} + d^i_\forall).$$

The following formula characterizes the quasi paths that are compatible with c: $\kappa_c = \left(\bigwedge_{i \in [\![1, n-1]\!]} \psi_i \wedge \phi_i \wedge \varsigma_i \wedge \xi_i \wedge \chi_i\right)$

The algorithm proceeds as follows: we look at all the possible partition of $\mathbb{C}$ into two sets G^+ and G^- there are a triply exponential number of possible partitions; G^+ and G^- potentially correspond to $\mathbb{C}^+_\delta$ and $\mathbb{C}^{-\varepsilon}_\delta$ for some δ and ε; we consider the formula:

$$\exists_{\ell \in \mathsf{L}, r \in \mathcal{R}_{X,M}} \delta_{\ell,r}.\ \exists \varepsilon.$$
$$\bigwedge_{c \in G^+} \left(\forall_{i \in [\![1, |c|]\!]} d_i, j_i.\ \forall_{x \in X} v^0_i(x), v^{j_i}_i(x).\ (\kappa_c \implies p_c \geq 0) \right)$$
$$\bigwedge_{c \in G^-} \left(\forall_{i \in [\![1, |c|]\!]} d_i, j_i.\ \forall_{x \in X} v^0_i(x), v^{j_i}_i(x).\ (\kappa_c \implies p_c < -\varepsilon) \right)$$

we then check that the formula has a solution. The size of the formula is at most doubly exponential, the algorithm we described is therefore 3EXPSPACE.

THEOREM 13. *If δ and ε are given then deciding if $\mathcal{T}$ is δ, ε-robust can be done in* 2EXPSPACE.

PROOF. This is because in that case, the formula we considered can be translated to an existential formula, which has size doubly exponential in the input. We know that the existential theory of the reals is decidable in PSPACE [4]. Therefore the global algorithm is 2EXPSPACE. □

PROPOSITION 2. *Deciding if a game is δ, ε-robust is* co-NEXP*-hard.*

PROOF. We show hardness of the problem by a reduction from the Succinct Hamilton Cycle problem: Given a Boolean circuit C with $2N$ inputs, does the graph on $2N$ nodes with edge relation encoded by C have a Hamiltonian cycle? This problem is known to be NEXP-complete [35].

We will construct a WTG of polynomial size, which is robust if, and only, if there is no Hamiltonian cycle in the succinctly represented graph. Thus showing that the problem is co-NEXP-hard

Variables representing nodes of the graph are encoded by clocks $x_1 \ldots x_N$ and $y_1 \ldots y_N$. Variables x_i being for the current node and y_i for the successor node, i.e. x_i are the first N inputs of the circuit and y_i the N last.

We encode the circuit by having two clocks g_i and $\bar{g_i}$ for each gate g_i. They represent whether the gate output should be true: in that case $g_i = 1$ and $\bar{g_i} = 0$; in the other case $g_i = 0$ and $\bar{g_i} = 1$.

We assume the gates are ordered $g_1, g_2, \ldots, g_G$ such that if $i < j$ implies that g_i does not take g_j as input, it is always possible to find such an order as Boolean circuits are acyclic.

Gates are encoded as represented in Fig. 21 for a $\wedge$-gate. For the $\wedge$-gate, if both inputs, from g_1 and g_2 are true, then the valuation of the clocks is such that $g_1 = g_2 = 1$. The first transition can be taken, and g_3 is set to 1. This encodes the fact that the output of gate g_3 is positive.

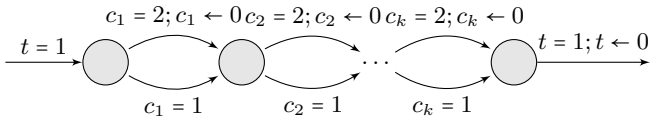

Figure 17: Module $X \leftarrow 1 - X$. Where $X = \{c_1, \ldots, c_k\}$.

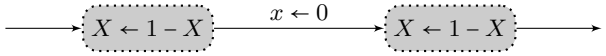

Figure 18: Module $x \leftarrow 1$.

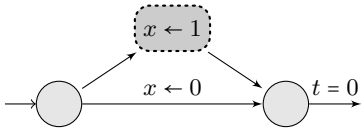

Figure 19: Module `Choose`(x).

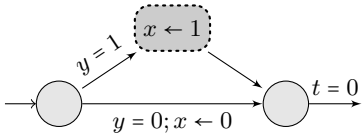

Figure 20: Module $x \leftarrow y$.

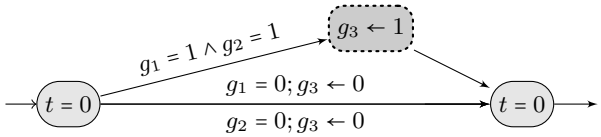

Figure 21: Encoding of a $\wedge$-gate g_3, which takes input from gates g_1 and g_2

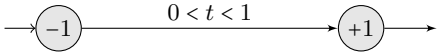

Figure 22: Module `MIX` introducing weight around 0

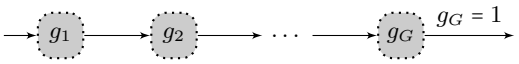

Figure 23: Module `Edge` encoding the edge relation.

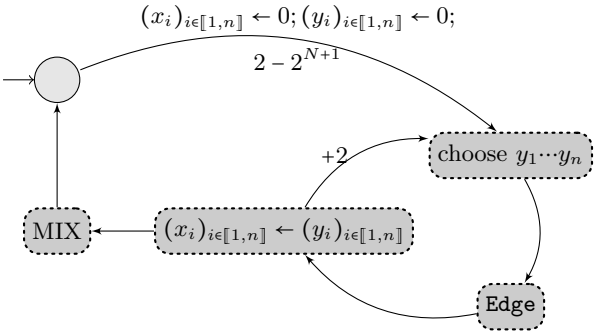

Figure 24: Global WTG encoding the succinct graph.

Module `Edge`, represented in Fig. 23, encode the edge relation. A path can traverse this module only if there is an edge between the nodes encoded by the valuation of $x_1, \cdots, x_n$ and $y_1, \cdots, y_n$.

Notice that in the definition of WTG, we assumed that there always is a player that can play an action, which is not the case here, but we can add an edge in each state to go to a sink state. This does not change the fact that the game is robust, but we do not include this edges to make the presentation clearer.

The global game, represented in Fig. 24, is played as follows:

1. we start with a weight of $2-2^{N+1}$ and the initial state i is encoded by a valuation of 0 for all variables $x_1, \cdots, x_n$;
2. `Eve` can choose a successor j by choosing the valuations of $y_1, \cdots, y_n$;
3. the play can traverse the module `Edge` only if there is a edge between nodes i and j;
4. we replace the valuation of $x_1, \cdots, x_n$ by that of $y_1, \cdots, y_n$, i.e. we now consider that j is the current state;
5. either we go back to step 2 with a reward of 2 or to the module `MIX`.

If there is a Hamiltonian path in the graph succinctly represented, then following it, until j is the initial state, will give a weight of 0. Then entering `MIX`, can give simple cycle that are either positive or negative, while all traverse the same regions. Hence the game is not robust for any δ and ε.

If there is a simple cycle in the region game, that can correspond to both positive and negative cycle, then: it goes through the module `MIX` since it is the only place where the weight can depend on the valuations; it must have been $2^N - 1$ times through the `Edge` module, so that we can have a weight around 0; the valuations each time must have encoded different states, since otherwise this cycle is not a simple one. Thus it describes an Hamiltonian path. □

Finally, we can characterize the complexity of deciding the energy problem for robust weighted timed games:

THEOREM 14. *The energy problem for robust games is in* EXPSPACE *and is* EXP*-hard.*

PROOF SKETCH. The algorithm proceeds by constructing the region game and then solving it using the algorithm of Thm. 8. This is correct because of Thm. 4, 6 and 9. □

PROOF. We consider the region game, which has at most exponential size. Deciding if a simple cycle is winning for `Eve` is easy since we only need to check one arbitrary path in the WTG whose projection on the regions corresponds to the simple cycle we consider. If its weight is greater or equal to 0, the cycle is in $\mathbb{C}^+_\delta$. This can be done using polynomial space (w.r.t. the size of the simple cycle). Using the algorithm of Thm. 7 we obtain a global procedure which is in EXPSPACE. EXP-hardness already holds for reachability objectives in timed games. □

5. FIXPOINT ALGORITHM

While the reduction to the cycle forming game in the region graph is elegant and allows us to identify a large and natural class of weighted timed games with decidable properties, this reduction does not lead directly to a practical semi-algorithm. In this section, we design a valuation iteration algorithm that can be implemented symbolically using polyhedra, which can be executed on any weighted timed game, and find winning strategies for `Eve` when it terminates successfully. We also show that termination is guaranteed on robust weighted timed games.

5.1 Value Iteration Algorithm

Our value iteration algorithm is an adaptation of the solution for the finite state case described in [19] to the setting of WTG. It computes successive approximations of the minimal energy level that `Eve` needs to win the energy game.

Essentially, the semi-algorithm is based on the iteration of an operator that computes successive approximations of the energy level/credit that is necessary for `Eve` to maintain the energy level positive for k rounds in the energy timed game, where k increases along with the iterations. Most importantly, our algorithm is parameterized by a value $c \in \mathbb{N}$, that represents a maximal energy level that we want to track: if the energy level necessary to stay alive from a given state (l, v) for k rounds is larger than c then (ℓ, v) is considered as loosing. This is a sound approximation when looking for winning strategies. This parameter is important to enforce termination of the analysis. If a fixed point is reached, then it contains enough information to identify winning states (those that are not mapped to $+\infty$ by the operator) and construct winning strategies. If the analysis is negative (no winning strategy found) then this value can be increased. Furthermore, we show that for robust weighted timed games, there is a finite value c which is computable and sufficient to detect winning strategies for `Eve`.

In this section, we assume the WTG is fixed and transitions, weight functions, etc ... refer to this game. Given $c \in \mathbb{N}$, we write $a \ominus_c b$ for $\max(0, a-b)$ if $a - b \le c$ and $+\infty$ otherwise. Notice that for all $a, b \in \mathbb{R} \cup \{+\infty\}$:

$$a - b \le a \ominus_c b \tag{3}$$

In the sequel we consider mappings in $\mathcal{S} = [\mathtt{St} \mapsto \mathbb{R}_+ \cup \{+\infty\}]$ that associate with each state an element in $\mathbb{R}_+ \cup \{+\infty\}$. Given $f, g \in \mathcal{S}$, we write $f \preceq g$ if $\forall s \in \mathtt{St}, f(s) \le g(s)$.

Given $c \in \mathbb{N}$, the operator $\mathtt{lift}_c : \mathcal{S} \mapsto \mathcal{S}$ is defined by:

$$\forall s \in \mathtt{St}, \mathtt{lift}_c(f)(s) = \inf_{a_\exists} \sup_{a_\forall} \left\{ f(s') \ominus_c \mathtt{w}(s \xrightarrow{a_\exists, a_\forall} s') \right\}$$

REMARK 5. *The operator* $lift_c$ *is monotonic, i.e.* $f \preceq g \implies lift_c(f) \preceq lift_c(g)$.

We let $f_0^c : \mathtt{St} \mapsto \mathbb{R}_+ \cup \{+\infty\}$ be the mapping defined by $\forall s \in \mathtt{St}, f_0^c(s) = 0$. We then inductively define f_{k+1}^c, for $k \ge 0$ to be $\mathtt{lift}_c(f_k^c)$. Notice that the sequence $(f_n^c)_{n \in \mathbb{N}}$ forms an increasing sequence. Thus, $f_n^c(s)$ represents the initial energy level that is needed by `Eve` to keep the energy level positive for n steps from s. If more than c is needed then $f_n^c(s)$ is set to be $+\infty$ (c being the maximal energy level that we want to track). This is formalized in Lem. 9.

LEMMA 9. *For all state* s*, index* n*, credit* c*,* $\varepsilon > 0$*:*

$$f_n(s) \ge -\sup_{\sigma_\exists} \inf_{\sigma_\forall} \left\{ w_{/c}(\mathit{Out}_s(\sigma_\exists, \sigma_\forall)_{\le n}) \right\}.$$

where for a history $h = h_0 \xrightarrow{m_0} h_1 \cdots \xrightarrow{m_n} h_n$*,* $w_{/c}(h) = -\infty$ *if* $\exists i.\ w(h_i \xrightarrow{m_i} h_{i+1}) < -c - \varepsilon$ *and* $w(h)$ *otherwise.*

PROOF. We prove the property by induction over n, it is obvious for $n = 0$. Now assuming this holds for n we show it for $n+1$. Let $\varepsilon > 0$, we define a strategy $\sigma_\exists$. We select $\sigma_\exists(s)$ that is $\frac{\varepsilon}{2}$ close to the optimal of

$$\inf_{a_\exists} \sup_{a_\forall} \left\{ f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{a_\exists, a_\forall} s') \right\}$$

if this value is different from $+\infty$ (otherwise the result is obvious). Hence for all action $a_\forall$:

$$f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') \le f_{n+1}^c(s) + \frac{\varepsilon}{2}$$

$$f_n^c(s') - \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') \le f_{n+1}^c(s) + \frac{\varepsilon}{2} \text{ (by remark (3))}$$

$$\mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') \ge f_n^c(s') - f_{n+1}^c(s) - \frac{\varepsilon}{2}$$

where $\mathtt{Tab}(s, \sigma_\exists(s), a_\forall) = s'$.

For the remaining of the proof, we introduce the notation $\sigma_\exists \circ (s \xrightarrow{m} s')$ for the strategy that associate to a history h, the action $\sigma_\exists(s \xrightarrow{m} h)$. We select $\sigma_\exists \circ (s \to s')$ to be at most $\frac{\varepsilon}{2}$ below the optimal of

$$\sup_{\sigma_\exists} \inf_{\sigma_\forall} \left\{ \mathtt{w}_{/c}(\mathtt{Out}_{s'}(\sigma_\exists, \sigma_\forall)_{\le n}) \right\}$$

Consider now a strategy $\sigma_\forall$ and ρ the outcome of $\sigma_\exists$ and $\sigma_\forall$. If there is i such that $\mathtt{w}\left(\rho_i \xrightarrow{\sigma_\exists(\rho_{\le i}), \sigma_\forall(\rho_{\le i})} \rho_{i+1}\right)$, then:

- if $i = 0$, then

$$\inf_{a_\exists} \sup_{a_\forall} \left\{ f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{a_\exists, a_\forall} s') \right\} > -c$$

because we chose $\sigma_\exists(s)$ to be $\frac{\varepsilon}{2}$ close to the optimal. This means that $f_{n+1}^c(s) = +\infty$ and the property is verified;

- otherwise $i > 0$, because we chose $\sigma_\exists \circ (s \to s')$ to be close to the optimal, and using the induction hypothesis, $f_n^c(s') = +\infty$. This shows

$$\sup_{a_\forall} \left\{ f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') \right\} = +\infty$$

Since $\sigma_\exists(s)$ was also selected to be close to the optimal, this means that $f_{n+1}^c(s) = +\infty$.

In the other case:

$$\begin{aligned}
w_{|c}(\rho_{\leq n+1}) &= \mathtt{w}(\rho_{\leq n+1}) \\
&= \mathtt{w}(s \xrightarrow{\sigma_\exists(s), \sigma_\forall(s)} s') \\
&\quad + \mathtt{Out}_{s'}(\sigma_\exists \circ (s \xrightarrow{\sigma_\exists(s), \sigma_\forall(s)} s')_{\leq n}, \sigma_\forall \circ (s \xrightarrow{\sigma_\exists(s), \sigma_\forall(s)} s')) \\
&\geq f_n^c(s') - f_{n+1}^c(s) - \frac{\varepsilon}{2} \\
&\quad + \sup_{\sigma_\exists} \inf_{\sigma_\forall} \left\{ \mathtt{w}_{/c}\left(\mathtt{Out}_{s'}(\sigma_\exists, \sigma_\forall)_{\leq n}\right) \right\} - \frac{\varepsilon}{2} \\
&\geq f_n^c(s') - f_{n+1}^c(s) - \frac{\varepsilon}{2} - f_n^c(s') - \frac{\varepsilon}{2} \text{ (induction hypothesis)} \\
&\geq -f_{n+1}^c(s) - \varepsilon
\end{aligned}$$

This means that

$$\inf_{\sigma_\forall}(\mathtt{w}_{/c}(\mathtt{Out}_s(\sigma_\exists, \sigma_\forall)_{\leq n})) \geq -f_{n+1}^c(s) - \varepsilon$$

and since we have such a strategy $\sigma_\exists$ for any $\varepsilon > 0$:

$$\sup_{\sigma_\forall} \inf_{\sigma_\forall}(\mathtt{w}_{/c}(\mathtt{Out}_s(\sigma_\exists, \sigma_\forall)_{\leq n})) \geq -f_{n+1}^c(s)$$

Which proves the property for f_{n+1}. □

Lemma 10. *For all state s, index n, credit c, $\varepsilon > 0$:*

$$-\sup_{\sigma_\exists} \inf_{\sigma_\forall} \left\{ w_{/c}^+(\mathtt{Out}_s(\sigma_\exists, \sigma_\forall)_{\leq n}) \right\} \geq f_n(s).$$

where for a history $h = h_0 \xrightarrow{m_0} h_1 \cdots \xrightarrow{m_{n-1}} h_n$, $w_{/c}^+(h) = -\infty$ if $\exists i, j.\ w(h_i \xrightarrow{m_i} h_{i+1} \xrightarrow{m_{i+1}} \cdots h_j) < -c + \varepsilon$ and $w(h)$ otherwise.

Proof. The proof is quite similar to that Lem. 9, we do it by induction over n and the property is obvious for $n = 0$. Let $\sigma_\exists$ be a strategy of Eve and $\varepsilon > 0$. In order to prove the property for rank $n+1$, we construct a strategy $\sigma_\forall$ such that $\mathtt{w}_{/c}^+(\mathtt{Out}_s(\sigma_\exists, \sigma_\forall)_{\leq n+1}) \leq -f_{n+1}^c + \varepsilon$.

Let $a_\forall$ be $\frac{\varepsilon}{2}$ close to the optimal of

$$\sup_{a_\forall} \left\{ f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') \right\}$$

if it is different from $+\infty$ and such that $f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') = +\infty$ otherwise. Let $s' = \mathtt{Tab}(s, \sigma_\exists(s), a_\forall)$ and $\sigma'_\forall$ be such that:

$$\mathtt{w}_{/c}^+\left(\mathtt{Out}_{s'}(\sigma_\exists \circ (s \xrightarrow{\sigma_\exists(s), a_\forall} s'), \sigma'_\forall)_{\leq n}\right) \geq -f_n^c(s') - \frac{\varepsilon}{2};$$

if $f_n^c(s') \neq +\infty$ and such that

$$\mathtt{w}_{/c}^+\left(\mathtt{Out}_{s'}(\sigma_\exists \circ (s \xrightarrow{\sigma_\exists(s), a_\forall} s'), \sigma'_\forall)_{\leq n}\right) = -\infty$$

otherwise. This is possible thanks to the induction hypothesis.

We define strategy $\sigma_\forall$ by:

$$\sigma_\forall = \begin{cases} s & \mapsto a_\forall \\ s \xrightarrow{\sigma_\exists(s), a_\forall} h & \mapsto \sigma'_\forall(h) \end{cases}$$

Let $h = h_0 \xrightarrow{m_0} h_1 \cdots \xrightarrow{m_n} h_{n+1} = \mathtt{Out}_s(\sigma_\exists, \sigma_\forall)_{\leq n+1}$. If there is i, j such that $\mathtt{w}(h_i \xrightarrow{m_i} h_{i+1} \xrightarrow{m_{i+1}} \cdots h_j) < -c$ then $\mathtt{w}_{/c}^+(h) = -\infty$ and the property is obviously verified. Otherwise, lets look at the weight of the outcome:

$$\begin{aligned}
\mathtt{w}(h) &= \mathtt{w}(h_0 \xrightarrow{m_0} h_1) + \mathtt{w}(h_{\geq 1}) \\
&= \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') + \mathtt{w}_{/c}^+(h_{\geq 1})
\end{aligned}$$

If $f_n^c(s') = +\infty$ then $\mathtt{w}_{/c}^+(h_{\geq 1}) = +\infty$, which contradicts the fact that there is no i, j such that $\mathtt{w}(h_i \xrightarrow{m_i} h_{i+1} \xrightarrow{m_{i+1}} \cdots h_j) < -c$. Therefore $f_n^c(s') \neq +\infty$ and:

$$\mathtt{w}(h) \geq \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') - f_n^c(s') - \frac{\varepsilon}{2}$$

- If $f_n^c(s') - \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') \geq c$ then $\mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') + \mathtt{w}(h_{\geq 1}) \geq c - \frac{\varepsilon}{2}$. Hence $\mathtt{w}_{/c}^+(h) = +\infty$.
- Otherwise

$$\begin{aligned}
\mathtt{w}(h) &\geq -\left(f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{\sigma_\exists(s), a_\forall} s') \right) - \frac{\varepsilon}{2} \\
&\geq -f_{n+1}^c(s) - \varepsilon
\end{aligned}$$

□

Theorem 15 (Correctness). *If there exists $n \geq 0$ such that $f_{n+1} = f_n$, and $f_n^c(\iota) \neq +\infty$, then for any $\varepsilon > 0$, Eve has a winning strategy for the c'-energy game $\mathcal{T}_{E(c')}$ with initial credit $c' = f_n^c(\iota) + \varepsilon$.*

Proof. We define a strategy $\sigma_\exists$ for Eve, that for a history h and a state s, select an action $a_\exists$ that is $\frac{\varepsilon}{2^{|h|+1}}$ close to the optimal. That is $\sigma_\exists(h \cdot s) = a_\exists$ such that:

$$\sup_{a_\forall} \left\{ f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{a_\exists, a_\forall} s') \right\} \leq f_n^c(s) + \frac{\varepsilon}{2^{|h|+1}}.$$

This is possible since $f_n^c(s) = f_{n+1}^c(s) = \inf_{a_\exists} \sup_{a_\forall} \left\{ f_n^c(s') \ominus_c \mathtt{w}(s \xrightarrow{a_\exists, a_\forall} s') \right\}$.

Consider now an outcome $\rho = \rho_0 \xrightarrow{m_0} \rho_1 \xrightarrow{m_1} \cdots$ of $\sigma_\exists$ such that $\rho_0 = s$. We show by induction that for all k, $f_n^c(\rho_0) + \varepsilon \cdot (1 - \frac{1}{2^k}) \geq f_n^c(\rho_k) - \mathtt{w}(\rho_{\leq k})$. This is true for $k = 0$ since

$\mathtt{w}(\rho_{\leq 0}) = 0$. Assume now that the property holds for some integer k. By the definition of $\sigma_\exists$, we have:

$$f_n^c(\rho_k) + \frac{\varepsilon}{2^{k+1}} \geq f_n^c(\rho_{k+1}) \ominus_c \mathtt{w}(\rho_k \xrightarrow{m_k} \rho_{k+1})$$
$$f_n^c(\rho_k) + \frac{\varepsilon}{2^{k+1}} \geq f_n^c(\rho_{k+1}) - \mathtt{w}(\rho_k \xrightarrow{m_k} \rho_{k+1}) \text{ (by rem. (3))}$$
$$f_n^c(\rho_k) + \frac{\varepsilon}{2^{k+1}} \geq f_n^c(\rho_{k+1}) - (\mathtt{w}(\rho_{\leq k+1}) - \mathtt{w}(\rho_{\leq k}))$$
$$f_n^c(\rho_k) - \mathtt{w}(\rho_{\leq k}) + \frac{\varepsilon}{2^{k+1}} \geq f_n^c(\rho_{k+1}) - \mathtt{w}(\rho_{\leq k+1})$$

Using the induction hypothesis, we obtain:

$$f_n^c(\rho_0) + \varepsilon \cdot (1 - \frac{1}{2^k}) + \frac{\varepsilon}{2^{k+1}} \geq f_n^c(\rho_{k+1}) - \mathtt{w}(\rho_{\leq k+1})$$
$$f_n^c(\rho_0) + \varepsilon \cdot (1 - \frac{1}{2^{k+1}}) \geq f_n^c(\rho_{k+1}) - \mathtt{w}(\rho_{\leq k+1})$$

Which proves the property for $k+1$. Hence for all k, $f_n^c(\rho_0) + \varepsilon + \mathtt{w}(\rho_{\leq k}) \geq 0$. This shows that from s, $\sigma_\exists$ is a winning strategy for the initial credit $f_n^c(s)$. □

Remark 6. *Note that winning strategies do not necessarily exist for the initial credit $f_n^c(s)$ we compute for each state as this is an* infimum. *However, given any $\varepsilon > 0$, ε-optimal strategies i.e. winning strategies with initial credit $f_n^c(s) + \varepsilon$ are guaranteed to exist. This is illustrated in Fig. 25 where for all $\varepsilon > 0$, `Eve` can win for the initial credit ε, but she cannot win for (limit) initial credit* 0.

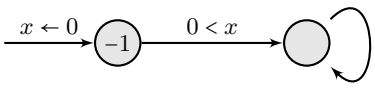

Figure 25: WTG with no optimal strategy.

In the case of robust game, we show that we can stop the algorithm after a finite number of iterations.

Theorem 16 (Termination). *Let $\mathcal{T}$ be a WTG and c a fixed credit. If $\mathcal{T}$ is δ, ε-robust and $f_{n_0}^c(\ell_\iota, \mathbf{0}) \neq +\infty$ for $n_0 \geq \left(\frac{c \cdot (|L| \cdot |\mathcal{R}_{X,M}| + 1)}{\varepsilon} + 1\right) \cdot |L| \cdot |\mathcal{R}_{X,M}|$ then `Eve` has a winning strategy in the energy game.*

Proof. We show in fact, the equivalent property that if `Eve` has no winning strategy for the energy game $\mathcal{T}$, then $f_{n_0}^c(\ell_\iota, \mathbf{0}) = +\infty$.

We use the correspondence with the region game. Assuming `Eve` has no winning strategy for the energy game $\mathcal{T}$, by Thm. 4, she has no winning strategy in $\mathcal{R}(\mathcal{T}, \Omega_\delta^+)$, and by Lem. 8, she has no winning strategy in $\mathcal{U}(\mathcal{R}(\mathcal{T}, \Omega_\delta^+))$. According to Lem.7 she has no winning strategy in $\mathcal{R}(\mathcal{T})$ for the existential cycle objective $\mathbb{C}_\delta^+$. Once a strategy $\sigma_\exists$ for `Eve` is fixed, `Adam` has a strategy $\sigma_\forall$ such that the simple cycles in the decomposition of the projection of the outcome are never in $\mathbb{C}_\delta^+$. Since the game is robust, all these simple cycles have a weight smaller than $-\varepsilon$.

We have that:

$$\sup_{\sigma_\exists} \left\{ \mathtt{w}_{/c}(\mathtt{Out}_s(\sigma_\exists, \sigma'_\forall)_{\leq n}) \right\} \leq -\varepsilon \cdot \left(\frac{n}{|\mathtt{L}| \cdot |\mathcal{R}_{X,M}|} - 1 \right) + c \cdot |\mathtt{L}| \cdot |\mathcal{R}_{X,M}|.$$

Let $n_0 \geq \left(\frac{c \cdot (|L| \cdot |\mathcal{R}_{X,M}| + 1)}{\varepsilon} + 1\right) \cdot |\mathtt{L}| \cdot |\mathcal{R}_{X,M}|$. Using Lem. 9 we have that $f_{n_0}(s) > c$, hence $f_{n_0}^c(s) = +\infty$. □

If moreover the game has bounded transition, the algorithm is complete.

Theorem 17 (Completeness for robust games). *Let $\mathcal{T}$ be a robust WTG, which has bounded transitions, and $c = |L \times \mathcal{R}_{X,M}| \cdot (W_L \times D + W_T) + 1$. `Eve` has a winning strategy in the energy game $\mathcal{T}_{E(c)}$ if, and only if, $f_{n_0}^c \neq +\infty$.*

Proof. Thanks to Thm. 9, we know that in such a game, if `Eve` has a winning strategy, then a credit of $c-1$ is enough to win. Lem. 10, ensures that f_c^n will never go higher than c in the initial state. Reciprocally Thm. 16, ensures that if `Eve` has no winning strategy, then $+\infty$ is reached within n_0 steps. □

5.2 Symbolic Algorithm

We have implemented the previous value iteration algorithm in HyTech [28]. The implementation is based on the symbolic *controllable timed predecessors* operator defined in [13] and first implemented in HyTech for cost optimal reachability games [14]. The choice of HyTech compared to state-of-the-art hybrid systems' analyzers like PHAVer [25] or SpaceEx [26] is motivated by the fact that HyTech has a built-in script language in which we can define the symbolic *controllable* predecessors operator easily. The symbolic algorithm/program in HyTech for the example of Fig. 1 is given in Appendix E. The result of the computation for the value iteration algorithm with $c = 4$ is depicted on Fig. 2 and show the winning zones for `Eve` in locations `Eve` and `Adam`.

The value iteration algorithm is implemented as the iterative computation of the fixpoint of a safety *hybrid* game. The hybrid game has a special variable E, the energy variable which is the only variable that is not a clock. Each location ℓ of the original WTG has a counterpart location $\ell_\mathcal{H}$ in the hybrid game. If $w(\ell) = k \in \mathbb{Z}$ then the derivative of E in $\ell_\mathcal{H}$ is given by $\frac{dE}{dt} = k$; each discrete transition (ℓ, g, z, ℓ') of the WTG also has a counterpart transition $(\ell, g, z \wedge E := E + k, \ell')$ if $w(\ell, g, z, \ell') = k$.

A state of the hybrid game is thus defined by $((\ell, v), E)$ where (ℓ, v) is a state of the original WTG. The existence of a winning strategy for the c-energy game $\mathcal{T}$ is reduced to the existence of a winning strategy in the associated hybrid game for the safety objective $E \geq 0$ (in each location.) Let $Safe = \{((\ell, v), E) \mid 0 \leq E \leq c\}$, where the upper bound c is the one used in the `lift` function from subsection 5.1. We define the winning states of the safety hybrid game as the greatest fixpoint of:

$$X = Safe \cap \mathtt{Pred}_t(\mathtt{Up}(\mathtt{cPred}(X)), \mathtt{uPred}(\overline{X}))$$

where `cPred` (controllable predecessor), `uPred` (uncontrollable predecessor), $\mathtt{Pred}_t$ (temporal predecessor) are defined

as in [13, 14] and

$$\mathtt{Up}(Y) = \{((\ell, v), e') \mid \exists((\ell, v), e) \in Y \wedge e' \geq e\}.$$

The `Up` operator captures in our symbolic implementation the role of the bound c in the $\mathtt{lift}_c$ operator: indeed, while the set X_i contains only triples $((\ell, v), e)$ where $e \leq c$, it is clear that we must include in X_{i+1} triples $((\ell', v'), e')$ from which `Eve` can force in one round the upward closure (for the energy level) of safe states in i steps. This is because if `Eve` can win from (ℓ, v) with a given energy level then she can win from that state with any greater energy level.

Example 5. In Fig. 26, plain (resp. dashed) arrows are controllable (resp. uncontrollable) edges. In location ℓ_0, no task is scheduled and the (battery) energy is recharging at rate +3. There is a background task B to be run at least every 2 t.u. if the other task has not arrived (and actually running in location ℓ_1) and a sporadic task S (interrupt) that can happen any time. The task B can be scheduled from location ℓ_0 (this is controllable) and we can stop to run it after at least 1 t.u. (measured by clock x.) The background task B has less priority than S and if S happens it is scheduled and B preempted. If we schedule the background task B, we are rewarded by +2 energy units. In locations ℓ_0, ℓ_1, the sporadic task S can occur (uncontrollable) and in this case it must be scheduled (going to ℓ_2) which consumes energy at rate $-\alpha$. The execution time of S is at most 1 t.u. (measured by clock x) and successive occurrences of S must be separated by at least 2 t.u. (measured by clock y.)

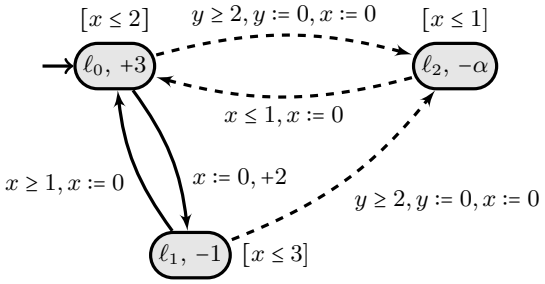

Figure 26: Scheduling Example

On this example, our symbolic algorithm terminates. If $\alpha = 3$, the HyTech program (Appendix E) gives a minimal initial energy level of 3 to be able to win the game (notice that we start with $y = 2$ and thus the sporadic task can arrive at the initial instant.) The optimal strategy from the point of view of `Adam` is to trigger the sporadic task S as often as possible. While a winning strategy for `Eve` (scheduler) is to wait in location ℓ_0 as long as possible. If the sporadic task arrives again, it is not before 1 t.u. and thus we are rewarded by at least 3 energy credits. If the sporadic does not occur before $x = 2$, we get 6 energy credits, and we can switch to ℓ_1 which increases energy by 2. This ensures winning the energy game, see Fig. 27 for a graphical representation of the winning region. The set of winning state computed by the HyTech program can be used to determine the minimal initial credit for each possible initial state: for example $energy \geq 3$ is necessary for the initial condition $x = 0, y = 2$.)

Now assume $\alpha = 4$. The previous strategy is not winning any more. However, the result of the HyTech program is now: $energy > 4$ (for $x = 0, y = 2$ as initial state.) In this case, while the minimal (infimum) initial credit is 4, there is no strategy realizing this value; the game cannot be won with an initial credit of 4 but rather with any value strictly above 4. Note that this information is collected by our symbolic algorithm but not by the operator $\mathtt{lift}_c$ as this operator is defined using inf, sup operators.

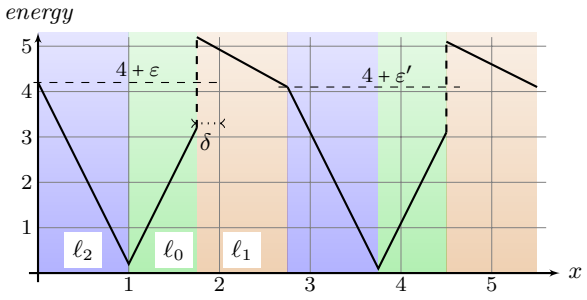

Figure 27: Winning Strategy for `Eve`, $\alpha = 4$.

6. REFERENCES

[1] R. Alur and D. L. Dill. A theory of timed automata. *TCS*, 126(2):183–235, 1994.

[2] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, Sept. 2002.

[3] R. Alur, S. La Torre, and G. J. Pappas. Optimal paths in weighted timed automata. In *HSCC*, volume 2034 of *LNCS*. Springer, 2001.

[4] S. Basu, R. Pollack, and M.-F. Roy. Existential theory of the reals. *Algorithms in Real Algebraic Geometry*, pages 505–532, 2006.

[5] G. Behrmann, A. Cougnard, A. David, E. Fleury, K. G. Larsen, and D. Lime. Uppaal-tiga: Time for playing games! In *CAV*, volume 4590 of *LNCS*. Springer, 2007.

[6] G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. W. Vaandrager. Minimum-cost reachability for priced timed automata. In *HSCC*, volume 2034 of *LNCS*. Springer, 2001.

[7] M. Ben-Or, D. Kozen, and J. Reif. The complexity of elementary algebra and geometry. *Journal of Computer and System Sciences*, 32(2):251–264, 1986.

[8] H. Björklund, S. Sandberg, and S. G. Vorobyov. Memoryless determinacy of parity and mean payoff games: a simple proof. *Theoretical Computer Science*, 310(1-3):365–378, 2004.

[9] P. Bouyer. From qualitative to quantitative analysis of timed systems – Mémoire d'habilitation à diriger des recherches, 2009.

[10] P. Bouyer, T. Brihaye, V. Bruyère, and J.-F. Raskin. On the optimal reachability problem of weighted timed automata. *Formal Methods in System Design*, 31(2), 2007.

[11] P. Bouyer, T. Brihaye, M. Jurdzinski, R. Lazic, and M. Rutkowski. Average-price and reachability-price games on hybrid automata with strong resets. In *FORMATS*, volume 5215 of *LNCS*. Springer, 2008.

[12] P. Bouyer, E. Brinksma, and K. G. Larsen. Staying

alive as cheaply as possible. In *HSCC*, volume 2993 of *LNCS*. Springer, 2004.

[13] P. Bouyer, F. Cassez, E. Fleury, and K. G. Larsen. Optimal Strategies in Priced Timed Game Automata. In *FSTTCS 2004*, pages 148–160, 2004.

[14] P. Bouyer, F. Cassez, E. Fleury, and K. G. Larsen. Synthesis of Optimal Strategies Using HyTech. *Electronic Notes in Theoretical Computer Science*, 119(1):11–31, 2005.

[15] P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey, and J. Srba. Infinite runs in weighted timed automata with energy constraints. In *FORMATS*, volume 5215 of *LNCS*. Springer, 2008.

[16] P. Bouyer, K. G. Larsen, and N. Markey. Lower-bound constrained runs in weighted timed automata. In *QEST'12*, pages 128–137, London, UK, Sept. 2012. IEEE Computer Society Press.

[17] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, and S. Yovine. Kronos: A model-checking tool for real-time systems. In *CAV*, volume 1427 of *LNCS*. Springer, 1998.

[18] T. Brihaye, V. Bruyère, and J.-F. Raskin. On optimal timed strategies. In *FORMATS*, volume 3829 of *LNCS*. Springer, 2005.

[19] L. Brim, J. Chaloupka, L. Doyen, R. Gentilini, and J.-F. Raskin. Faster algorithms for mean-payoff games. *Formal methods in system design*, 38(2):97–118, 2011.

[20] F. Cassez, A. David, E. Fleury, K. G. Larsen, and D. Lime. Efficient on-the-fly algorithms for the analysis of timed games. In *CONCUR*, volume 3653 of *LNCS*. Springer, 2005.

[21] F. Cassez, J. J. Jessen, K. G. Larsen, J.-F. Raskin, and P.-A. Reynier. Automatic synthesis of robust and optimal controllers - an industrial case study. In *HSCC*, volume 5469 of *LNCS*. Springer, 2009.

[22] F. Cassez and K. G. Larsen. The impressive power of stopwatches. In *CONCUR*, volume 1877 of *LNCS*. Springer, 2000.

[23] K. Chatterjee, L. Doyen, and T. A. Henzinger. A survey of stochastic games with limsup and liminf objectives. In *ICALP*, volume 5556 of *LNCS, Part II*. Springer, 2009.

[24] K. Chatterjee, L. Doyen, T. A. Henzinger, and J.-F. Raskin. Generalized mean-payoff and energy games. In *FSTTCS*, volume 8 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010.

[25] G. Frehse. PHAVer: algorithmic verification of hybrid systems past HyTech. *STTT*, 10(3):263–279, 2008.

[26] G. Frehse, C. L. Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. SpaceEx: Scalable Verification of Hybrid Systems. In *CAV*, volume 6806 of *LNCS*, pages 379–395. Springer, 2011.

[27] T. Henzinger. Quantitative reactive modeling and verification. *Computer Science Research and Development*, page 14, 2013.

[28] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: A model checker for hybrid systems. *STTT*, 1(1-2):110–122, 1997.

[29] J. J. Jessen, J. I. Rasmussen, K. G. Larsen, and A. David. Guided controller synthesis for climate controller using Uppaal Tiga. In *FORMATS*, volume 4763 of *LNCS*. Springer, 2007.

[30] M. Jurdziński. Deciding the winner in parity games is in UP and in co-UP. *Information Processing Letters*, 68(3):119–124, 1998.

[31] M. Jurdzinski and A. Trivedi. Average-time games. In *FSTTCS*, volume 2 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2008.

[32] S. La Torre, S. Mukhopadhyay, and A. Murano. Optimal-reachability and control for acyclic weighted timed automata. In *IFIP TCS*, volume 2, pages 485–497, 2002.

[33] K. G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a nutshell. *STTT*, 1(1-2):134–152, 1997.

[34] O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems (an extended abstract). In *STACS*, pages 229–242, 1995.

[35] C. H. Papadimitriou and M. Yannakakis. A note on succinct representations of graphs. *Information and Control*, 71(3):181–185, 1986.

[36] P. Tabuada. Controller synthesis for bisimulation equivalence. *Systems & Control Letters*, 57(6):443–452, 2008.

APPENDIX
A. HYTECH PROGRAMS

```
-- constant definition for c in lift_c
define(MAX,4)
var x: clock; -- regular clock
energy: analog; -- the energy variable
c,u: discrete; -- used to indicate type of move
               -- if c is changed -> control action
               -- if u is changed -> uncontrollable action
t,t1: analog; -- these variabes are not part of the model
               -- we just need them to do some existential
               -- quantification to compute other sets
automaton H
synclabs: ;
 initially l0 & x=0;
-- note that Hytech does not use the constraint x ≥ 0 when x
-- is a clock. This is why we add it in the invariant
loc l0: while x>=0 & x<=2 wait {denergy=3,dt=-1,dt1=-1}
   -- one controllable transition -c changed-
   when x>=1 do {u'=u,c'=1-c,x'=0} goto l1;

loc l1: while x>=0 & x<=2 wait {denergy=-2,dt=-1,dt1=-1}
   -- one uncontrollable transition -u changed-
   when x>=1 do {u'=1-u,c'=c,x'=0} goto l0;
end

var init_reg,safe, -- set of states
    STOP,          -- set of states from which time is stopped
    uPreX,uPrebarX,cPreX, -- variables in the CPre operator
    X,Y,Z: region ; -- variables used in computation

-- first define the initial and winning regions
init_reg := loc[H]=l0 & x=0 & energy>=1 ;
-- safe states truncated upto MAX
safe := energy>=0 & energy<=MAX ;
-- states from which time cannot elapse
STOP := ~(hide t,c,u in t>0 & c=0 & u=0 &
          pre(True & t=0 & c=0 & u=0) endhide);

-- compute the fixpoint by of lift_c
X := iterate X from safe using
{
-- uncontrollable predecessors of the complement of X
uPrebarX := hide t,u in t=0 & u=0 & pre(~X & u=1 & t=0) endhide;
-- controllable predecessors of X
cPreX := hide t,c in t=0 & c=0 & pre(X & t=0 & c=1) endhide ;
-- uncontrollable predecessors leading to winning states
uPreX := hide t,u in t=0 & u=0 & pre(X & u=1 & t=0) endhide;
--
Z := (cPreX | (uPreX & ~uPrebarX & STOP)) ;
-- compute Up(Z) in Y
Y := hide t1 in
       ((hide energy in Z & t1 >= energy endhide) & energy = t1)
     endhide;
-- set from which we can reach Y while
-- avoiding uPreBarX all along (works only if time deterministic !!)
X := safe & hide t in
               (hide c,u in t>=0 & c=0 & u=0 & pre(Y & t=0 & c=0 & u=0)
                endhide) &
               ~(hide t1 in
                      (hide c,u in
                         t1>=0 & t1<=t & c=0 & u=0 &
                         pre(uPrebarX & t1=0 & c=0 & u=0)
                          endhide)
                    endhide)
               endhide;
};

-- we win if init_reg is included in Up(X)
Y := hide t1 in (
        (hide energy in X & t1 >= energy endhide) & energy = t1 )
    endhide;
if init_reg<=Y then
   prints "[INFO] Eve can win energy game!";
   prints "[INFO] Set of winning states is:" ;
   print Y;
   prints "[INFO] winning states at init region:" ;
   print Y & init_reg;
   prints "[INFO] Minimum initial credit needed";
   print omit all locations hide x in Y &
                            (hide energy in init_reg endhide)
                           endhide;
else
   prints "[INFO] Could not prove that Eve can win energy game";
   prints "[INFO] Try to increase max constant";
endif;
```

Figure 28: HyTech Program to Solve Energy Game of Fig. 1.

```
-- Example for energy games
-- Two tasks: one background task B that can be run or not
-- One sporadic S, that can occur not too often, and with a bounded
-- execution time

-- constant definition
define(MAX,10)
define(interT2,2)

var
x,y: clock;
energy: analog; -- the energy variable
c,u: discrete;  -- used to indicate that
                -- if c is changed -> control action
                -- if u is changed -> uncontrollable action
t,t1: analog;   -- these variables are not part of the model
                -- we just need them to some existential quantification
                -- to compute controllable time predecessors
automaton H
synclabs: ;
 initially  l0 & x=0;
-- note that Hytech does not use the constraint x>=0 when x
-- is a clock. This is why we add it in the invariant.

-- recharging mode (not more than 2 t.u.)
loc l0: while x>=0 & x<=2 wait {denergy=3,dt=-1,dt1=-1}
  -- one controllable transition -c changed- start running T1
  when True do {energy'=energy+2,u'=u,c'=1-c,x'=0} goto l1;
  -- one uncontrollable transition -u changed- arrival of T2
  when y>=interT2 do {energy'=energy,u'=1-u,c'=c,y'=0,x'=0} goto l2;

-- T1 running for at most 3 t.u.
loc l1: while x>=0 & x<=3 wait {denergy=-1,dt=-1,dt1=-1}
        -- one controllable transition -c changed- stop running T1
        when x>=1 do {energy'=energy,u'=u,c'=1-c,x'=0} goto l0;
        -- one uncontrollable transition -u changed- arrival of T2
        when y>=interT2 do {energy'=energy,u'=1-u,c'=c,y'=0,x'=0} goto l2;

-- T2 running for at most 1 t.u.
loc l2: while x>=0 & x<=1 wait {denergy=-4,dt=-1,dt1=-1}
    when x<=1 do {energy'=energy,u'=1-u,c'=c,x'=0} goto l0;
end

var init_reg,safe, -- set of states
    STOP,              -- set of states from which time is stopped
    uPreX,uPrebarX,cPreX, -- variables in the CPre operator
    X,Y,Z: region ; -- variables used in computation

-- rest is similar to previous program
```

Figure 29: HyTech Program to Solve Energy Game of Fig. 26.