Looking at Mean-Payoff and Total-Payoff through Windows

K. Chatterjee (IST Austria) L. Doyen (ENS Cachan)
M. Randour (UMONS) J.-F. Raskin (ULB)

Paris - 16.12.2013

LIAFA verification seminar

Aim of this talk

1. Overview of the situation for (multi) MP and TP games
   - No P algorithm known in one dimension
   - In multi dimensions, MP is coNP-complete
   - First contribution: **TP is undecidable in multi dimensions**
   - No timing guarantee

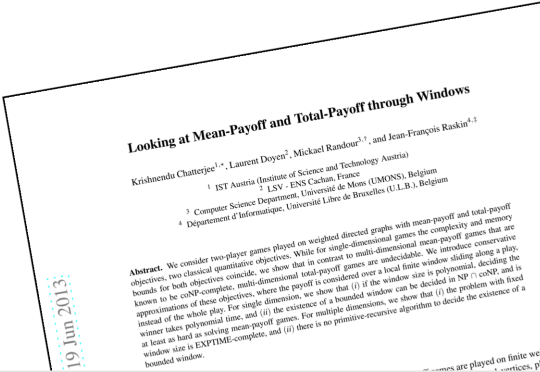
Aim of this talk

1. Overview of the situation for (multi) MP and TP games
   - No P algorithm known in one dimension
   - In multi dimensions, MP is coNP-complete
   - First contribution: **TP is undecidable in multi dimensions**
   - No timing guarantee

2. Introduction of **window objectives**
   - Conservative approximation of MP/TP
   - Break the complexity barriers
   - Specifies timing requirements
   - Algorithms, complexity and memory requirements
   - Several flavors of the objective

Advertisement

Full paper available on arXiv: abs/1302.4248



19 Jun 2013

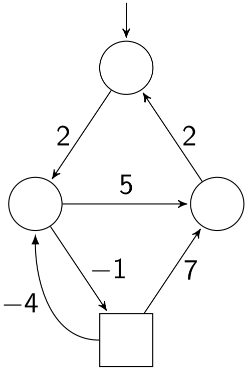
Looking at Mean-Payoff and Total-Payoff through Windows

Krishnendu Chatterjee[1,*], Laurent Doyen[2], Mickael Randour[3,†], and Jean-François Raskin[4,‡]

[1] IST Austria (Institute of Science and Technology Austria)
[2] LSV - ENS Cachan, France
[3] Computer Science Department, Université de Mons (UMONS), Belgium
[4] Département d'Informatique, Université Libre de Bruxelles (U.L.B.), Belgium

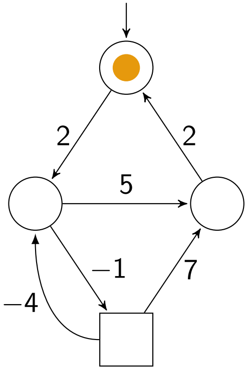
Abstract. We consider two-player games played on weighted directed graphs with mean-payoff and total-payoff objectives, two classical quantitative objectives. While for single-dimensional games the complexity and memory bounds for both objectives coincide, we show that in contrast to multi-dimensional mean-payoff games that are known to be coNP-complete, multi-dimensional total-payoff games are undecidable. We introduce conservative approximations of these objectives, where the payoff is considered over a local finite window sliding along a play, instead of the whole play. For single dimension, we show that (*i*) if the window size is polynomial, deciding the winner takes polynomial time, and (*ii*) the existence of a bounded window can be decided in NP ∩ coNP, and is at least as hard as solving mean-payoff games. For multiple dimensions, we show that (*i*) the problem with fixed window size is EXPTIME-complete, and (*ii*) there is no primitive-recursive algorithm to decide the existence of a bounded window.

MP and TP games



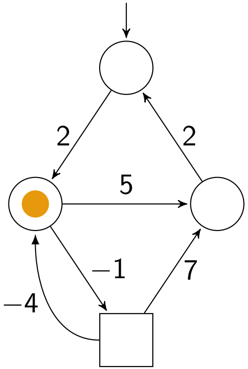
- $G = (S_1, S_2, E, w)$
- $S = S_1 \cup S_2, S_1 \cap S_2 = \emptyset, E \subseteq S \times S$, $w \colon E \to \mathbb{Z}$
- $\mathcal{P}_1$ states = ◯
- $\mathcal{P}_2$ states = □
- Plays, prefixes, **pure** strategies.

MP and TP games



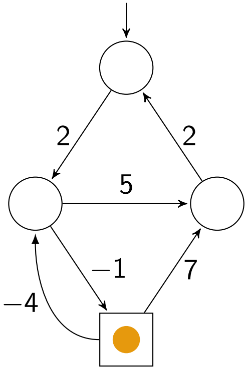
- $\mathrm{TP}(\pi) = \liminf_{n\to\infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})$
- $\mathrm{MP}(\pi) = \liminf_{n\to\infty} \frac{1}{n} \mathrm{TP}(\pi(n))$

MP and TP games

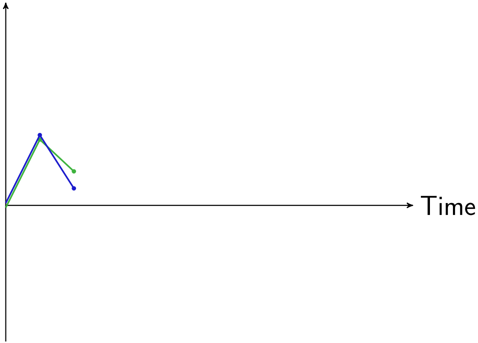


- $\underline{\text{TP}}(\pi) = \liminf_{n\to\infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})$
- $\underline{\text{MP}}(\pi) = \liminf_{n\to\infty} \frac{1}{n} \text{TP}(\pi(n))$

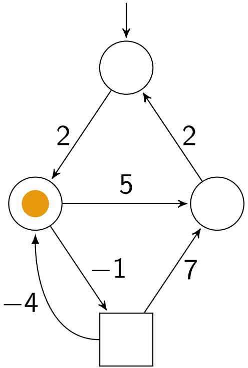
MP and TP games



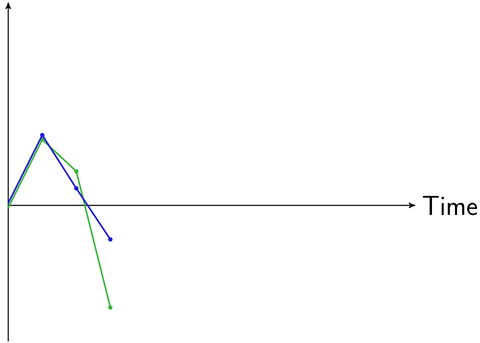
- $\underline{\text{TP}}(\pi) = \liminf_{n\to\infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})$
- $\underline{\text{MP}}(\pi) = \liminf_{n\to\infty} \frac{1}{n} \text{TP}(\pi(n))$



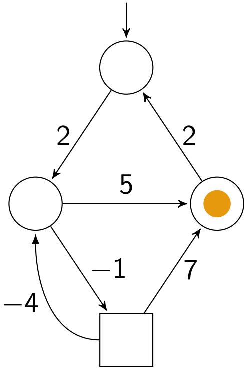
MP and TP games



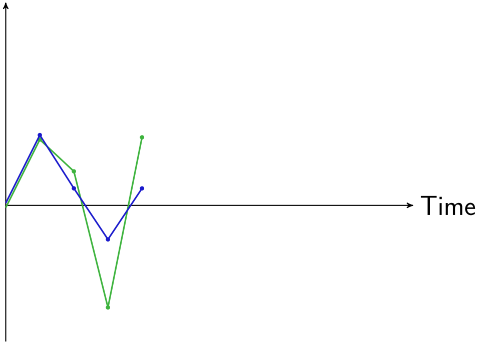
- $\mathrm{TP}(\pi) = \liminf_{n\to\infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})$
- $\mathrm{MP}(\pi) = \liminf_{n\to\infty} \frac{1}{n} \mathrm{TP}(\pi(n))$



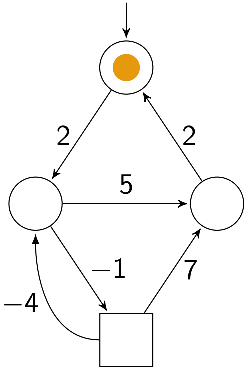
MP and TP games



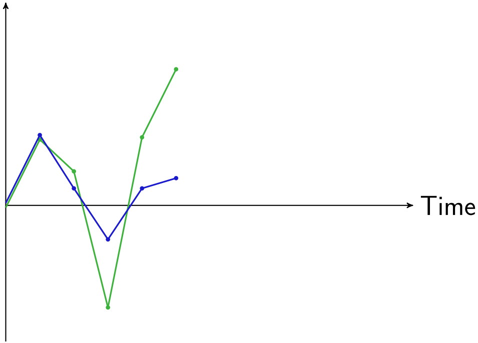
- $\text{TP}(\pi) = \liminf_{n\to\infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})$
- $\text{MP}(\pi) = \liminf_{n\to\infty} \frac{1}{n} \text{TP}(\pi(n))$



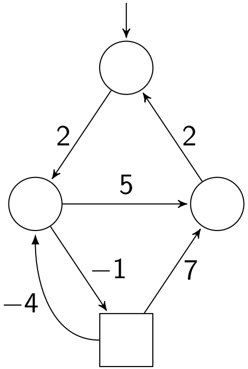
MP and TP games



- $\underline{\mathrm{TP}}(\pi) = \liminf_{n\to\infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})$
- $\underline{\mathrm{MP}}(\pi) = \liminf_{n\to\infty} \frac{1}{n}\mathrm{TP}(\pi(n))$

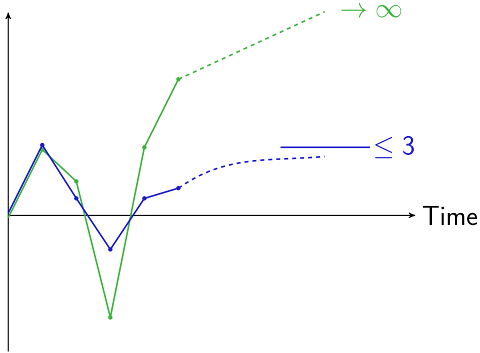


MP and TP games

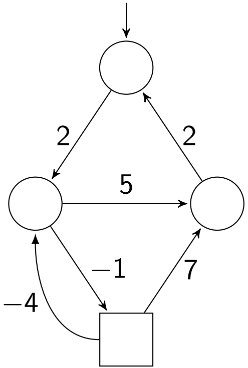


Then, $(2, 5, 2)^\omega$

- $\underline{\text{TP}}(\pi) = \liminf_{n\to\infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})$
- $\underline{\text{MP}}(\pi) = \liminf_{n\to\infty} \frac{1}{n} \text{TP}(\pi(n))$



MP and TP games



Then, $(2, 5, 2)^\omega$

- **TP (MP) threshold problem**

  Given $v \in \mathbb{Q}$ and $s_{\mathsf{init}} \in S$,

  $\exists ?\, \lambda_1 \in \Lambda_1$ s.t. $\forall \lambda_2 \in \Lambda_2$,

  $\underline{\mathsf{TP}}(\mathsf{Outcome}_G(s_{\mathsf{init}}, \lambda_1, \lambda_2)) \geq v$

Known results

	one-dimension			k-dimension		
	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.
$\underline{\text{MP}}$ / $\overline{\text{MP}}$	NP ∩ coNP	mem-less		coNP-c. / NP ∩ coNP	infinite	mem-less
$\underline{\text{TP}}$ / $\overline{\text{TP}}$	NP ∩ coNP	mem-less		??	??	??

- Long tradition of study. Non-exhaustive selection: [EM79, ZP96, Jur98, GZ04, GS09, CDHR10, VR11, CRR12]
- k-dimension: weights = integer vectors
- *No known polynomial time algorithm for one-dimension*
- *No result on multi-dimension total-payoff*

Multi-dimension TP games are undecidable

Theorem

The threshold problem for infimum and supremum total-payoff objectives is **undecidable** in multi-dimension games, for five dimensions.

Multi-dimension TP games are undecidable

Theorem

The threshold problem for infimum and supremum total-payoff objectives is **undecidable** in multi-dimension games, for five dimensions.

▷ Reduction from the **halting problem for 2CMs** [Min61]

Two-counter machines

- Finite set of instructions
- Two counters C_1 and C_2 taking values $(v_1, v_2) \in \mathbb{N}^2$
- Instructions:
  - ▷ Increment
    $$C_i ++$$
  - ▷ Decrement
    $$C_i --$$
  - ▷ Zero test and branch accordingly

    If $C_i == 0$ **do** *this* **else do** *that*
- W.l.o.g. if the machine stops, it stops with both counters to zero

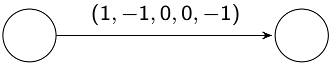
Encoding a 2CM in a 5-dim. TP game

- TP objective (inf or sup) of threshold $(0, 0, 0, 0, 0)$
- $\mathcal{P}_1$ must simulate faithfully
- $\mathcal{P}_2$ retaliates if $\mathcal{P}_1$ cheats
- At the end, $\mathcal{P}_1$ wins the TP game **iff** the 2CM stops

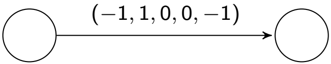
Key idea: after m steps, the TP has value $(v_1, -v_1, v_2, -v_2, -m)$ **iff** the 2CM counters have value (v_1, v_2)

Instructions

- Increment C_1

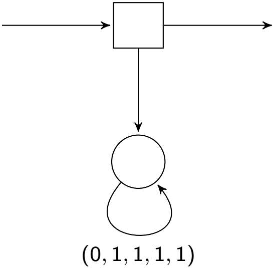


- Decrement C_1



Instructions

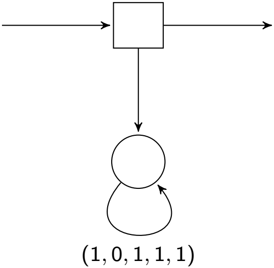
- Checking counter C_1 is non-negative



- ▷ If $\mathcal{P}_1$ cheats, he is doomed!
- ▷ Otherwise, $\mathcal{P}_2$ has no interest in retaliating.

Instructions

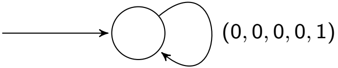
- Checking a zero test on C_1



- ▷ If $\mathcal{P}_1$ cheats, he is doomed!
- ▷ Otherwise, $\mathcal{P}_2$ has no interest in retaliating.

Halting

- If the 2CM halts (with counters to zero w.l.o.g.)



▷ Thanks to the fifth dim., $\mathcal{P}_1$ wins only if the machine halts.

The case is closed

	one-dimension			k-dimension		
	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.
$\underline{\text{MP}}$ / $\overline{\text{MP}}$	NP $\cap$ coNP	mem-less		coNP-c. / NP $\cap$ coNP	infinite	mem-less
$\underline{\text{TP}}$ / $\overline{\text{TP}}$	NP $\cap$ coNP	mem-less		**Undec.**	-	-

Motivations

- Classical MP and TP objectives have some drawbacks
  - Complexity issues
    - P membership for the one-dim. case is a long-standing open problem
    - TP undecidable in k-dim.
  - Infimum vs. supremum
  - **no timing guarantee**: the "good behavior" occurs at the limit...

Window objectives: key idea

- **Window** of fixed size **sliding** along a play
  $\rightsquigarrow$ defines a local finite horizon
- Objective: see a **local** $MP \geq 0$ *before hitting the end* of the window
  $\rightsquigarrow$ needs to be verified at *every* step

Window objectives: key idea

- **Window** of fixed size **sliding** along a play
  $\rightsquigarrow$ defines a local finite horizon
- Objective: see a **local** *MP* ≥ 0 *before hitting the end* of the window
  $\rightsquigarrow$ needs to be verified at *every* step

▷ Conservative approximation of MP/TP

▷ Intuition: local deviations from the threshold must be compensated in a parametrized # of steps

▷ Variety of results and algorithms

Illustration: WMP, threshold zero, maximal window = 4

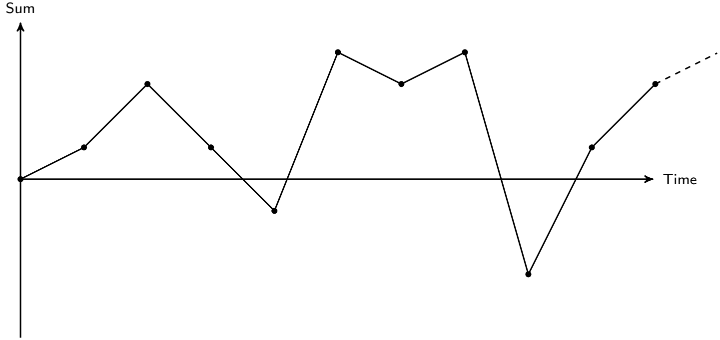


Illustration: WMP, threshold zero, maximal window $= 4$

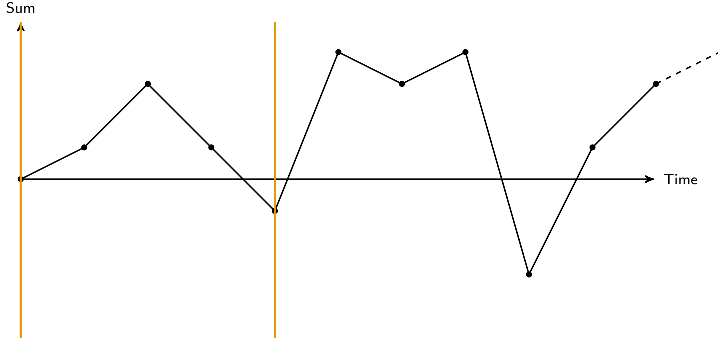


Illustration: WMP, threshold zero, maximal window = 4

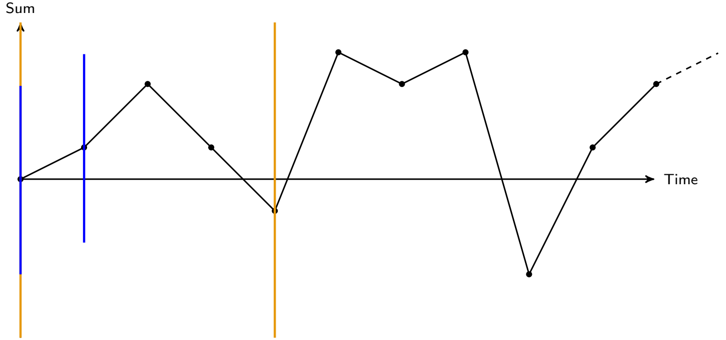


Illustration: WMP, threshold zero, maximal window = 4

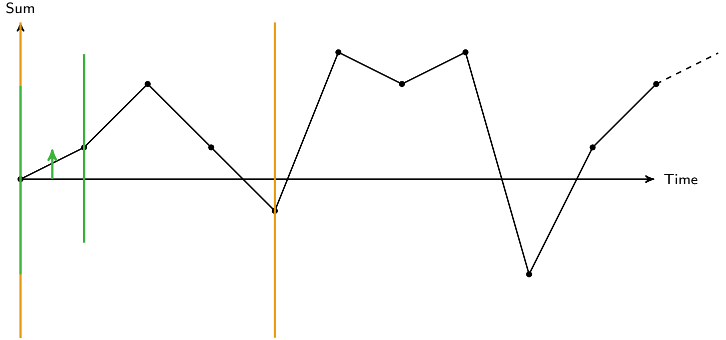


Illustration: WMP, threshold zero, maximal window = 4

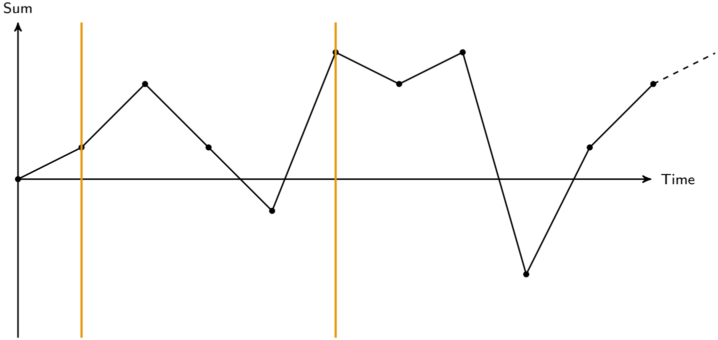


Illustration: WMP, threshold zero, maximal window = 4

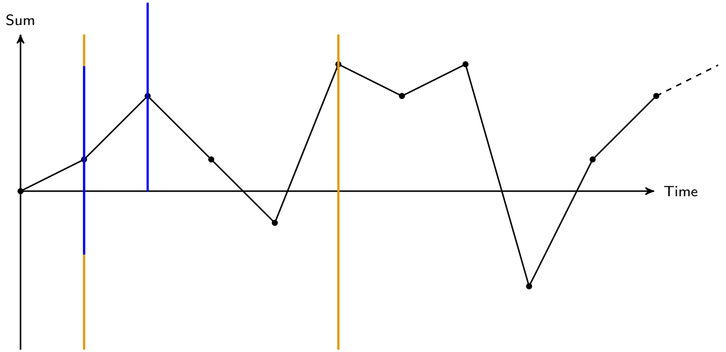


Illustration: WMP, threshold zero, maximal window = 4

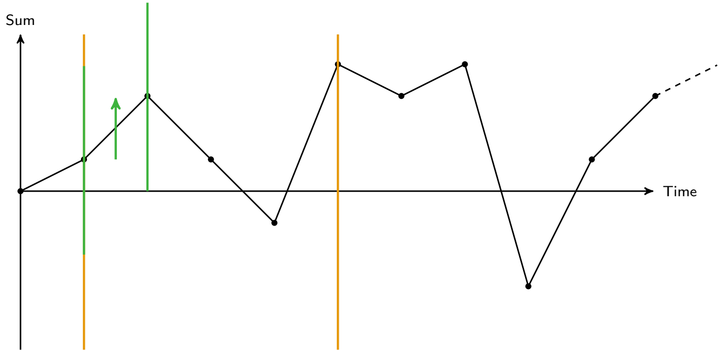


Illustration: WMP, threshold zero, maximal window = 4

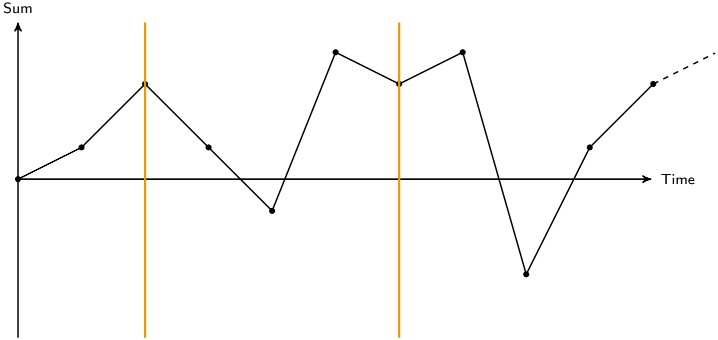


Illustration: WMP, threshold zero, maximal window = 4

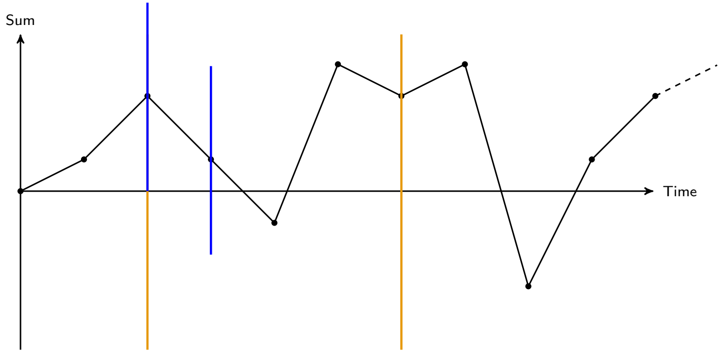


Illustration: WMP, threshold zero, maximal window = 4

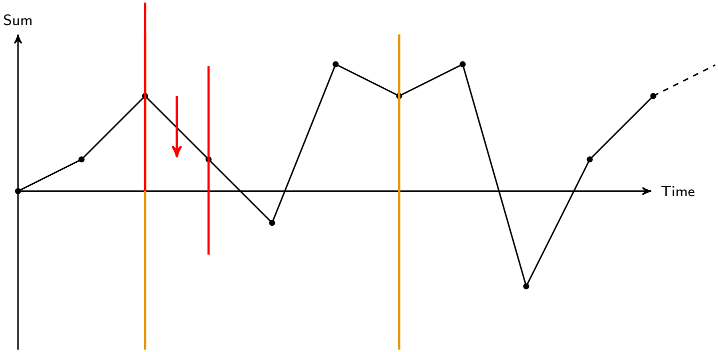


Illustration: WMP, threshold zero, maximal window = 4

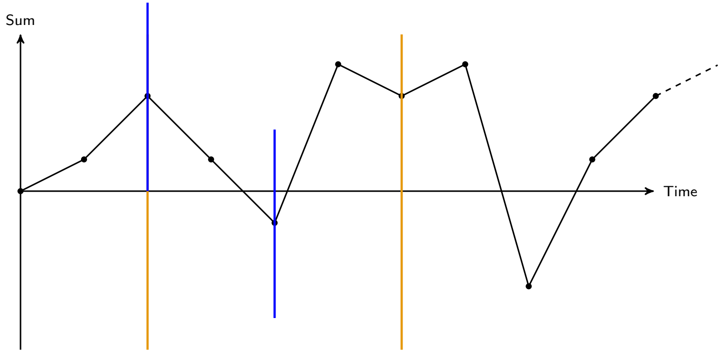


Illustration: WMP, threshold zero, maximal window = 4

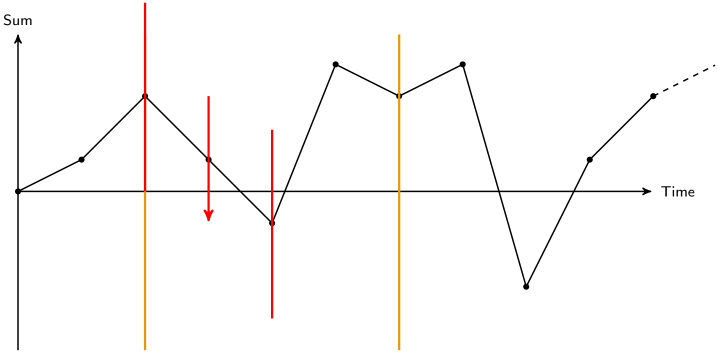


Illustration: WMP, threshold zero, maximal window = 4

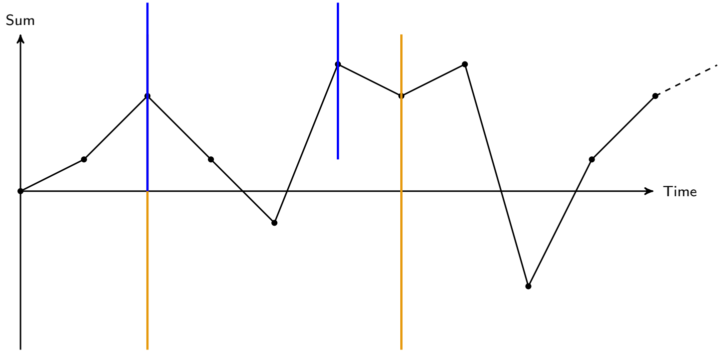
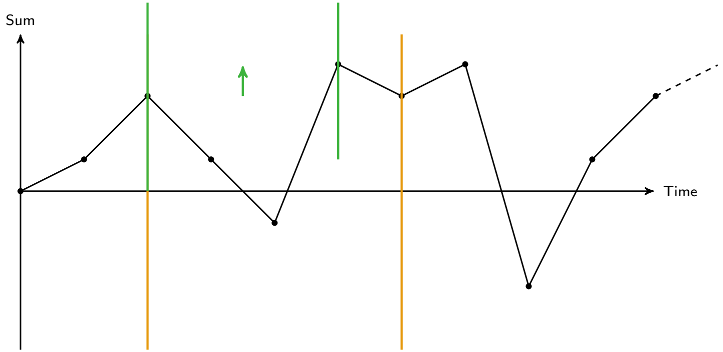


Illustration: WMP, threshold zero, maximal window = 4



Multiple variants

- Given $l_{\max} \in \mathbb{N}_0$, *good window* $\mathbf{GW}(l_{\max})$ asks for a positive sum in at most $l_{\max}$ steps (one window, from the first state)
- *Direct Fixed Window*: $\mathbf{DFW}(l_{\max}) \equiv \Box\mathbf{GW}(l_{\max})$
- *Fixed Window*: $\mathbf{FW}(l_{\max}) \equiv \Diamond\mathbf{DFW}(l_{\max})$
- *Direct Bounded Window*: $\mathbf{DBW} \equiv \exists\, l_{\max},\ \mathbf{DFW}(l_{\max})$
- *Bounded Window*: $\mathbf{BW} \equiv \Diamond\mathbf{DBW} \equiv \exists\, l_{\max},\ \mathbf{FW}(l_{\max})$

Multiple variants

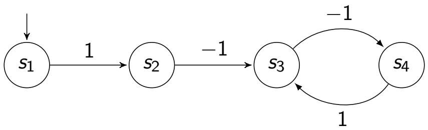
- Given $l_{\max} \in \mathbb{N}_0$, *good window* $\mathbf{GW}(l_{\max})$ asks for a positive sum in at most $l_{\max}$ steps (one window, from the first state)
- *Direct Fixed Window*: $\mathbf{DFW}(l_{\max}) \equiv \Box\mathbf{GW}(l_{\max})$
- *Fixed Window*: $\mathbf{FW}(l_{\max}) \equiv \Diamond\mathbf{DFW}(l_{\max})$
- *Direct Bounded Window*: $\mathbf{DBW} \equiv \exists\, l_{\max},\ \mathbf{DFW}(l_{\max})$
- *Bounded Window*: $\mathbf{BW} \equiv \Diamond\mathbf{DBW} \equiv \exists\, l_{\max},\ \mathbf{FW}(l_{\max})$

▷ Nice properties: monotonicity in $l_{\max}$, prefix-independence

▷ A window *closes* when the sum becomes positive

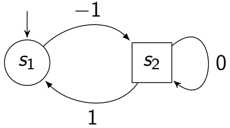
▷ A window is *open* if not yet closed

Example 1



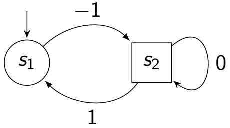
- MP is satisfied
  - ▷ the cycle is non-negative
- **FW**(2) is satisfied
  - ▷ thanks to prefix-independence
- **DBW** is not
  - ▷ the window opened in s_2 never closes

Example 2



- MP is satisfied
  - ▷ all simple cycles are non-negative
- *but* none of the window objectives is
  - ▷ $\mathcal{P}_2$ can force opening windows and delay their closing for as long as he wants (but not forever due to prefix-independence)

Example 2



- MP is satisfied
  - ▷ all simple cycles are non-negative
- *but* none of the window objectives is
  - ▷ $\mathcal{P}_2$ can force opening windows and delay their closing for as long as he wants (but not forever due to prefix-independence)

BW vs. MP

- BW asks for timing guarantees which cannot be enforced here
- Observe that $\mathcal{P}_2$ **needs infinite memory**

Conservative approximation of MP (one-dim.)

The following are true

$$\text{Any window obj.} \Rightarrow \mathbf{BW} \Rightarrow \text{MP} \geq 0$$
$$\mathbf{BW} \Leftarrow \text{MP} > 0$$

Results overview

	one-dimension			k-dimension		
	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.
$\underline{\text{MP}}$ / $\overline{\text{MP}}$	NP $\cap$ coNP	mem-less		coNP-c. / NP $\cap$ coNP	infinite	mem-less
$\underline{\text{TP}}$ / $\overline{\text{TP}}$	NP $\cap$ coNP	mem-less		**undec.**	-	-
WMP: fixed polynomial window	**P-c.**	**mem. req.** $\leq$ **linear**$(\lvert S\rvert \cdot l_{\max})$		**PSPACE-h.** **EXP-easy**	**exponential**	
WMP: fixed arbitrary window	**P**$(\lvert S\rvert, V, l_{\max})$			**EXP-c.**		
WMP: bounded window problem	**NP $\cap$ coNP**	**mem-less**	**infinite**	**NPR-h.**	-	-

- $|S|$ the # of states, V the length of the binary encoding of weights, and $l_{\max}$ the window size.

Results overview

	one-dimension			k-dimension		
	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.
$\underline{\text{MP}}$ / $\overline{\text{MP}}$	NP $\cap$ coNP	mem-less		coNP-c. / NP $\cap$ coNP	infinite	mem-less
$\underline{\text{TP}}$ / $\overline{\text{TP}}$	NP $\cap$ coNP	mem-less		**undec.**	-	-
WMP: fixed polynomial window	**P-c.**	**mem. req.** $\leq$ **linear**$(\lvert S\rvert \cdot l_{\max})$		**PSPACE-h.** **EXP-easy**	**exponential**	
WMP: fixed arbitrary window	**P**$(\lvert S\rvert, V, l_{\max})$			**EXP-c.**		
WMP: bounded window problem	**NP $\cap$ coNP**	**mem-less**	**infinite**	**NPR-h.**	-	-

- $\lvert S\rvert$ the # of states, V the length of the binary encoding of weights, and $l_{\max}$ the window size.
- For one-dim. games with poly. windows, we are in **P**
- For multi-dim. games with fixed windows, we are **decidable**
- Window obj. provide **timing guarantees**

Results overview

	one-dimension			k-dimension		
	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.
$\underline{\text{MP}}$ / $\overline{\text{MP}}$	NP $\cap$ coNP	mem-less		coNP-c. / NP $\cap$ coNP	infinite	mem-less
$\underline{\text{TP}}$ / $\overline{\text{TP}}$	NP $\cap$ coNP	mem-less		**undec.**	-	-
WMP: fixed polynomial window	**P-c.**	**mem. req.** $\leq$ **linear**($\lvert S\rvert \cdot l_{\max}$)		**PSPACE-h.** **EXP-easy**	**exponential**	
WMP: fixed arbitrary window	**P**($\lvert S\rvert, V, l_{\max}$)			**EXP-c. (1/2)**		
WMP: bounded window problem	**NP $\cap$ coNP**	**mem-less**	**infinite**	**NPR-h.**	-	-

- $|S|$ the # of states, V the length of the binary encoding of weights, and $l_{\max}$ the window size.

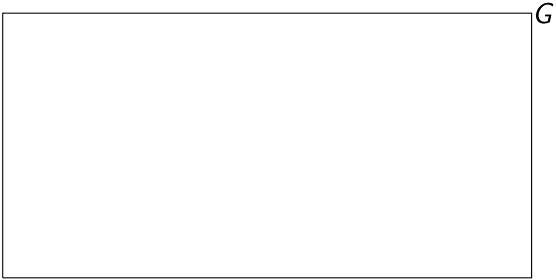
- No time to discuss everything. Focus.

High level sketch: top-down approach

- $\mathbf{FW}(l_{\max}) \equiv \Diamond \mathbf{DFW}(l_{\max})$

▷ Assume we can compute $\mathbf{DFW}(l_{\max})$,

▷ Compute attractor, declare winning and recurse on subgame.



High level sketch: top-down approach

- $\mathbf{FW}(l_{\max}) \equiv \Diamond \mathbf{DFW}(l_{\max})$

▷ Assume we can compute $\mathbf{DFW}(l_{\max})$,

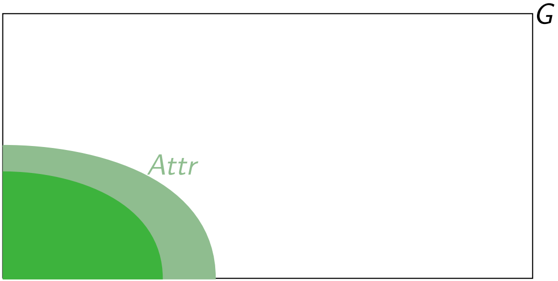
▷ Compute attractor, declare winning and recurse on subgame.

High level sketch: top-down approach

- $\mathbf{FW}(l_{\max}) \equiv \Diamond \mathbf{DFW}(l_{\max})$

▷ Assume we can compute $\mathbf{DFW}(l_{\max})$,

▷ Compute attractor, declare winning and recurse on subgame.



High level sketch: top-down approach

- $\mathbf{FW}(l_{\max}) \equiv \Diamond \mathbf{DFW}(l_{\max})$

▷ Assume we can compute $\mathbf{DFW}(l_{\max})$,

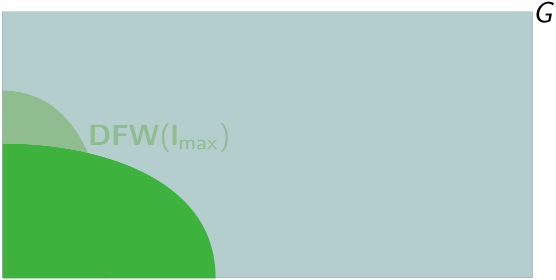
▷ Compute attractor, declare winning and recurse on subgame.

High level sketch: top-down approach

- $\mathbf{FW}(l_{\max}) \equiv \Diamond \mathbf{DFW}(l_{\max})$

▷ Assume we can compute $\mathbf{DFW}(l_{\max})$,

▷ Compute attractor, declare winning and recurse on subgame.

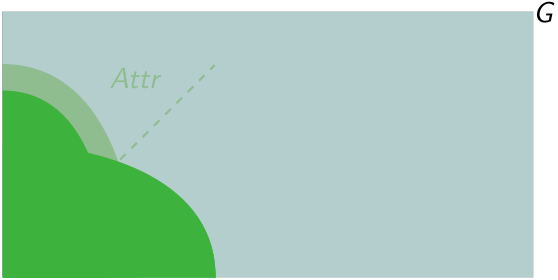


High level sketch: top-down approach

- $\mathbf{FW}(l_{\max}) \equiv \Diamond \mathbf{DFW}(l_{\max})$

▷ Assume we can compute $\mathbf{DFW}(l_{\max})$,

▷ Compute attractor, declare winning and recurse on subgame.

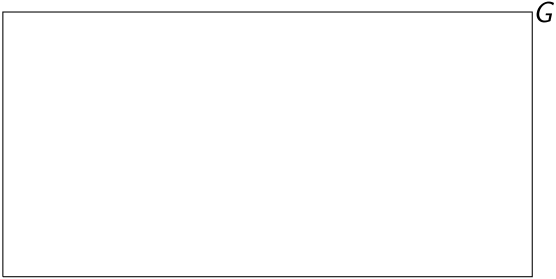


High level sketch: top-down approach

- $\mathbf{DFW}(l_{\max}) \equiv \Box\mathbf{GW}(l_{\max})$

▷ Assume we can compute $\mathbf{GW}(l_{\max})$,

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly (sufficient thanks to the *inductive property of windows*).

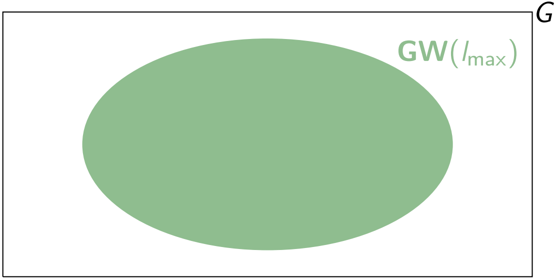


High level sketch: top-down approach

- $\mathbf{DFW}(l_{\max}) \equiv \Box\mathbf{GW}(l_{\max})$

▷ Assume we can compute $\mathbf{GW}(l_{\max})$,

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly (sufficient thanks to the *inductive property of windows*).



High level sketch: top-down approach

- $\mathbf{DFW}(l_{\max}) \equiv \Box\mathbf{GW}(l_{\max})$

▷ Assume we can compute $\mathbf{GW}(l_{\max})$,

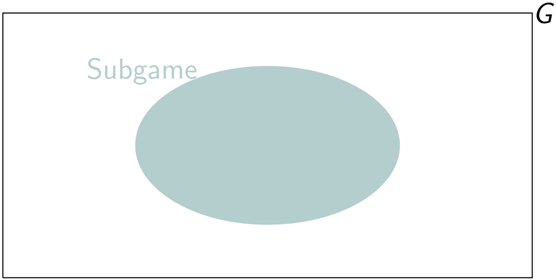
▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly (sufficient thanks to the *inductive property of windows*).

High level sketch: top-down approach

- $\mathbf{DFW}(l_{\max}) \equiv \Box\mathbf{GW}(l_{\max})$

▷ Assume we can compute $\mathbf{GW}(l_{\max})$,

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly (sufficient thanks to the *inductive property of windows*).

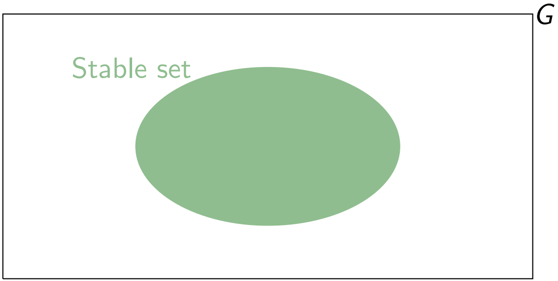


High level sketch: top-down approach

- $\mathbf{DFW}(l_{\max}) \equiv \Box\mathbf{GW}(l_{\max})$

▷ Assume we can compute $\mathbf{GW}(l_{\max})$,

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly (sufficient thanks to the *inductive property of windows*).



High level sketch: top-down approach

- **GW**($l_{\max}$)

▷ Simply compute the best sum achievable in at most $l_{\max}$ steps and check if positive.

High level sketch: top-down approach

- **GW**($l_{\max}$)

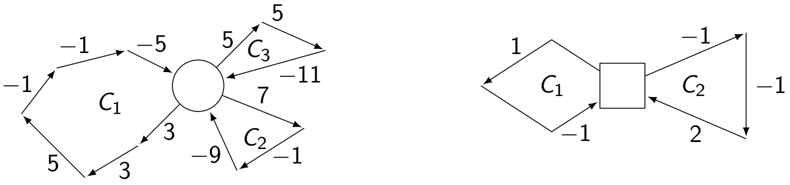
▷ Simply compute the best sum achievable in at most $l_{\max}$ steps and check if positive.

- Finally,

Theorem

In two-player one-dimension games,
(a) the fixed arbitrary window MP problem is decidable in time polynomial in the size of the game and the window size,
(b) **the fixed polynomial window MP problem is P-complete**,
(c) both players require memory, and memory of size linear in the size of the game and the window size is sufficient.

Memory is necessary for both players



$(C_1 C_2 C_3)^\omega, l_{\max} = 4$ $(C_1 C_2)^\omega, l_{\max} = 3$

Choices are based on

- ▷ the # of steps remaining to close the window,
- ▷ the amount to compensate.

EXPTIME algorithm

Winning plays for the FW objective:

- from some point on, on all dimensions, all opening windows are closed within $l_{\max}$ steps
- the closing may be asynchronous

EXPTIME algorithm

Winning plays for the FW objective:

- from some point on, on all dimensions, all opening windows are closed within $l_{\max}$ steps
- the closing may be asynchronous

Basically, winning = seeing only a finite number of *bad windows*

- ▷ reduction to an exponentially larger co-Büchi game
- ▷ EXPTIME membership and exponential upper bounds on memory follow

From FW($l_{\max}$) to a co-Büchi game

For each dimension, bookkeeping of

- the amount to compensate to close the window,
- the remaining # of steps to close it.

When a window closes on dim. t, we reset

- ▷ the amount to zero,
- ▷ the # of steps to $l_{\max}$.

From FW(l_{max}) to a co-Büchi game

For each dimension, bookkeeping of

- the amount to compensate to close the window,
- the remaining # of steps to close it.

When a window closes on dim. t, we reset

- ▷ the amount to zero,
- ▷ the # of steps to l_{max}.

Key elements

- $S \rightsquigarrow S \times (\{-l_{max} \cdot W, \ldots, 0\} \times \{1, \ldots, l_{max}\})^k$
- *bad states* representing windows not closing in time
- co-Büchi objective asks they are visited only finitely often

EXPTIME-hardness for 2 dim. and arbitrary weights

Reduction from **countdown games**.

- Weighted graph $(\mathcal{S}, \mathcal{T})$, with $\mathcal{S}$ the set of states and $\mathcal{T} \subseteq \mathcal{S} \times \mathbb{N}_0 \times \mathcal{S}$ the transition relation.
- Configurations (s, c), $s \in \mathcal{S}$, $c \in \mathbb{N}$.
- Game starts in (s_{init}, c_0).

EXPTIME-hardness for 2 dim. and arbitrary weights

Reduction from **countdown games**.

- Weighted graph $(\mathcal{S}, \mathcal{T})$, with $\mathcal{S}$ the set of states and $\mathcal{T} \subseteq \mathcal{S} \times \mathbb{N}_0 \times \mathcal{S}$ the transition relation.
- Configurations (s, c), $s \in \mathcal{S}$, $c \in \mathbb{N}$.
- Game starts in (s_{init}, c_0).
- Transitions from a configuration (s, c) performed as follows:
  1. $\mathcal{P}_1$ chooses a duration d, $0 < d \leq c$ such that there exists $t = (s, d, s') \in \mathcal{T}$ for some $s' \in \mathcal{S}$,
  2. $\mathcal{P}_2$ chooses a state $s' \in \mathcal{S}$ such that $t = (s, d, s') \in \mathcal{T}$,
  3. the game advances to $(s', c - d)$.

EXPTIME-hardness for 2 dim. and arbitrary weights

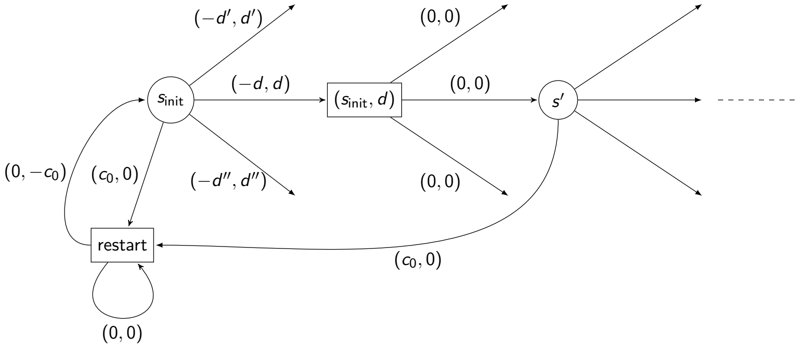
Reduction from **countdown games**.

- Weighted graph $(\mathcal{S}, \mathcal{T})$, with $\mathcal{S}$ the set of states and $\mathcal{T} \subseteq \mathcal{S} \times \mathbb{N}_0 \times \mathcal{S}$ the transition relation.
- Configurations (s, c), $s \in \mathcal{S}$, $c \in \mathbb{N}$.
- Game starts in (s_{init}, c_0).
- Transitions from a configuration (s, c) performed as follows:
  1. $\mathcal{P}_1$ chooses a duration d, $0 < d \leq c$ such that there exists $t = (s, d, s') \in \mathcal{T}$ for some $s' \in \mathcal{S}$,
  2. $\mathcal{P}_2$ chooses a state $s' \in \mathcal{S}$ such that $t = (s, d, s') \in \mathcal{T}$,
  3. the game advances to $(s', c - d)$.
- Terminal configurations reached whenever no legitimate move is available. $\mathcal{P}_1$ wins iff $(s, 0)$.

Deciding the winner is EXPTIME-complete [JSL08].

From CD games to FW

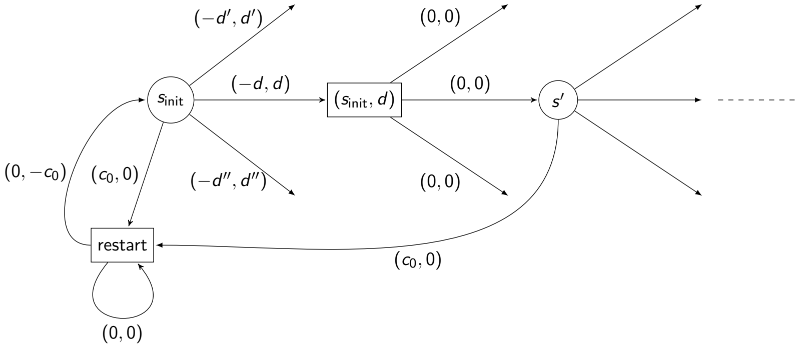
$(\mathcal{S}, \mathcal{T})$, $(s_{\mathsf{init}}, c_0) \rightsquigarrow G$, $k = 2$, $l_{\mathsf{max}} = 2 \cdot c_0 + 2$



- Two dimensions used to store the counter and its opposite
- $\mathcal{P}_1$ chooses durations and $\mathcal{P}_2$ chooses transitions of the CDG

From CD games to FW

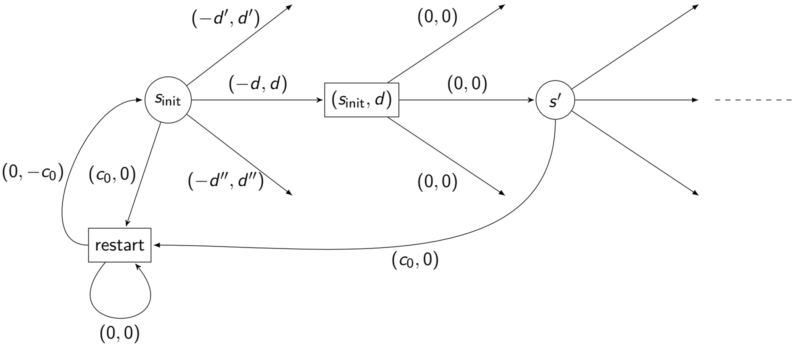
$(\mathcal{S}, \mathcal{T})$, $(s_{\mathsf{init}}, c_0) \rightsquigarrow G$, $k = 2$, $l_{\max} = 2 \cdot c_0 + 2$



- ▷ $\mathcal{P}_1$ can branch to restart at any time
- ▷ There, $\mathcal{P}_2$ can delay the closing of open windows then restart

From CD games to FW

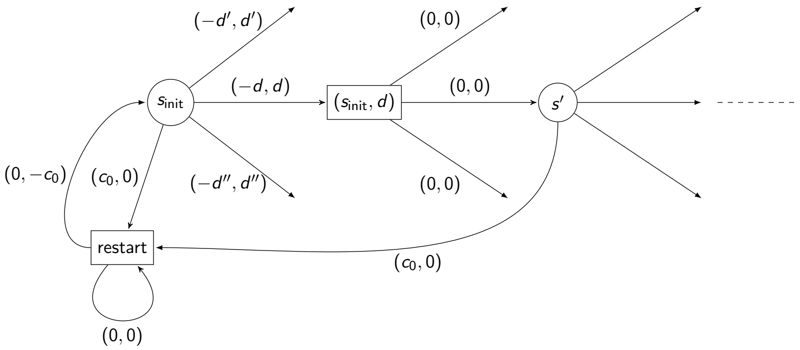
$(\mathcal{S}, \mathcal{T}), (s_{\text{init}}, c_0) \rightsquigarrow G,\ k = 2,\ l_{\max} = 2 \cdot c_0 + 2$



- ▷ To close the window on the 2nd dim., $\mathcal{P}_1$ has to accumulate *at least* c_0 before branching
- ▷ To be safe on the 1st, he must accumulate *at most* c_0

From CD games to FW

$(\mathcal{S}, \mathcal{T})$, $(s_{\text{init}}, c_0) \rightsquigarrow G$, $k = 2$, $l_{\max} = 2 \cdot c_0 + 2$



▷ $\mathcal{P}_1$ wins for FW iff he reaches *exactly* c_0, i.e., iff he can reach a terminal configuration $(s, 0)$ in the CDG

Other results

The multi-dim. FW problem is also

- EXPTIME-hard for weights $\{-1, 0, 1\}$ and arbitrary dimensions
  - ▷ membership problem for APTMs [CKS81]
- PSPACE-hard even for polynomial windows
  - ▷ generalized reachability games [FH10]
  - ▷ also induces that exponential memory is necessary (sufficient thanks to co-Büchi reduction)

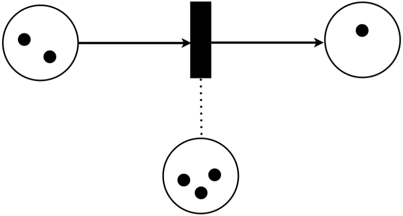
Approach

- We prove **non-primitive recursive**[1] **(NPR) hardness**
- Reduction from the termination problem in **reset nets** (Petri nets with reset arcs) [Sch02]

[1] Cf. Ackermann function

Reset nets

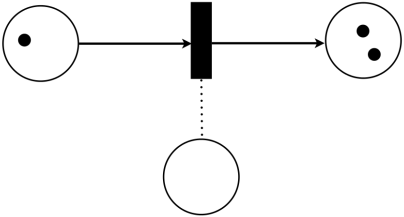
- Classic Petri net (places, tokens, transitions) with added *reset arcs*



▷ Transitions may empty a place from all its tokens

Reset nets

- Classic Petri net (places, tokens, transitions) with added *reset arcs*



▷ Transitions may empty a place from all its tokens

▷ Given an initial marking, the *termination problem* asks if there exists an infinite sequence of transitions that can be fired

From reset nets to **direct** bounded window games

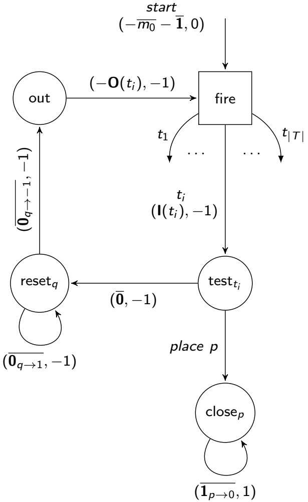
- Crux of the construction: encoding the markings
  - We use one dimension for each place
  - If a place p contains m tokens, then there will be an open window on dimension p with sum value $-m-1$
  - Hence **during a faithful simulation, all windows remain open** (you cannot consume tokens that do not exist)

From reset nets to **direct** bounded window games

- Crux of the construction: encoding the markings
  - We use one dimension for each place
  - If a place p contains m tokens, then there will be an open window on dimension p with sum value $-m-1$
  - Hence **during a faithful simulation, all windows remain open** (you cannot consume tokens that do not exist)

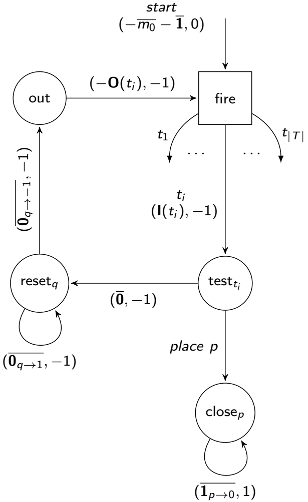
- $\mathcal{P}_2$ simulates the net
- $\mathcal{P}_1$ checks if he is faithful
- $\mathcal{P}_1$ wants to win the direct bounded window MP obj.
  - only able to do so if $\mathcal{P}_2$ cheats, i.e., if all runs terminate

The construction in a nutshell



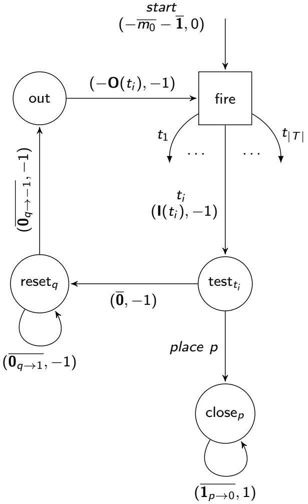
- ▷ The initial marking open corresponding windows in all places
- ▷ $\mathcal{P}_2$ chooses transitions to fire, which consume tokens
- ▷ $\mathcal{P}_1$ can branch or continue (and apply reset, then output)

The construction in a nutshell



- If no infinite execution exists, at some point, $\mathcal{P}_2$ must choose a transition without the needed tokens on some place p
- The window closes on dimension p
- By branching $\mathcal{P}_1$ can close all other windows and ensure winning

The construction in a nutshell



- ▷ If $\mathcal{P}_1$ branches while $\mathcal{P}_2$ is honest, one window stays open forever and he loses
- ▷ The additional dimension ensures that $\mathcal{P}_1$ leaves the reset state

Extension to bounded window objective

- More involved construction

Theorem

In two-player multi-dimension games, the bounded window mean-payoff problem is non-primitive recursive hard.

A new family of objectives

	one-dimension			k-dimension		
	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.	complexity	$\mathcal{P}_1$ mem.	$\mathcal{P}_2$ mem.
$\underline{\text{MP}}$ / $\overline{\text{MP}}$	NP $\cap$ coNP	mem-less		coNP-c. / NP $\cap$ coNP	infinite	mem-less
$\underline{\text{TP}}$ / $\overline{\text{TP}}$	NP $\cap$ coNP	mem-less		**undec.**	-	-
WMP: fixed polynomial window	**P-c.**	**mem. req.** $\leq$ **linear**$(\lvert S\rvert \cdot l_{\max})$		**PSPACE-h.** **EXP-easy**	**exponential**	
WMP: fixed arbitrary window	**P**$(\lvert S\rvert, V, l_{\max})$			**EXP-c.**		
WMP: bounded window problem	**NP $\cap$ coNP**	**mem-less**	**infinite**	**NPR-h.**	-	-

- ▷ Conservative approximation of MP/TP
- ▷ For one-dim. games with poly. windows, we are in **P**
- ▷ For multi-dim. games with fixed windows, we are **decidable**
- ▷ Window obj. provide **timing guarantees**
- ▷ *Open question*: is BW decidable in multi-dim. ?

Check the full version on arXiv! abs/1302.4248

Thanks!

Do not hesitate to discuss with us!

References I

K. Chatterjee, L. Doyen, T.A. Henzinger, and J.-F. Raskin.
Generalized mean-payoff and energy games.
In Proc. of FSTTCS, LIPIcs 8, pages 505–516. Schloss Dagstuhl - LZI, 2010.

K. Chatterjee, L. Doyen, M. Randour, and J.-F. Raskin.
Looking at mean-payoff and total-payoff through windows.
In Proc. of ATVA, LNCS 8172, pages 118–132. Springer, 2013.

A.K. Chandra, D. Kozen, and L.J. Stockmeyer.
Alternation.
J. ACM, 28(1):114–133, 1981.

K. Chatterjee, M. Randour, and J.-F. Raskin.
Strategy synthesis for multi-dimensional quantitative objectives.
In Proc. of CONCUR, LNCS 7454, pages 115–131. Springer, 2012.

C. Dufourd, A. Finkel, and P. Schnoebelen.
Reset nets between decidability and undecidability.
In Proc. of ICALP, LNCS 1443, pages 103–115. Springer, 1998.

A. Ehrenfeucht and J. Mycielski.
Positional strategies for mean payoff games.
Int. Journal of Game Theory, 8(2):109–113, 1979.

N. Fijalkow and F. Horn.
The surprizing complexity of generalized reachability games.
CoRR, abs/1010.2420, 2010.

References II

T. Gawlitza and H. Seidl.
Games through nested fixpoints.
In Proc. of CAV, LNCS 5643, pages 291–305. Springer, 2009.

H. Gimbert and W. Zielonka.
When can you play positionally?
In Proc. of MFCS, LNCS 3153, pages 686–697. Springer, 2004.

M. Jurdziński, J. Sproston, and F. Laroussinie.
Model checking probabilistic timed automata with one or two clocks.
Logical Methods in Computer Science, 4(3), 2008.

M. Jurdziński.
Deciding the winner in parity games is in UP $\cap$ co-UP.
Inf. Process. Lett., 68(3):119–124, 1998.

R. Lazic, T. Newcomb, J. Ouaknine, A.W. Roscoe, and J. Worrell.
Nets with tokens which carry data.
Fundam. Inform., 88(3):251–274, 2008.

M.L. Minsky.
Recursive unsolvability of Post's problem of "tag" and other topics in theory of Turing machines.
The Annals of Mathematics, 74(3):437–455, 1961.

P. Schnoebelen.
Verifying lossy channel systems has nonprimitive recursive complexity.
Inf. Process. Lett., 83(5):251–261, 2002.

References III

Y. Velner and A. Rabinovich.
Church synthesis problem for noisy input.
In Proc. of FOSSACS, LNCS 6604, pages 275–289. Springer, 2011.

U. Zwick and M. Paterson.
The complexity of mean payoff games on graphs.
Theoretical Computer Science, 158:343–359, 1996.