Looking at Mean-Payoff and Total-Payoff through Windows

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

12.04.2013

Cassting kick-off meeting

IST AUSTRIA Institute of Science and Technology
ENS CACHAN
UMONS Université de Mons
ULB UNIVERSITÉ LIBRE DE BRUXELLES

Aim of this talk

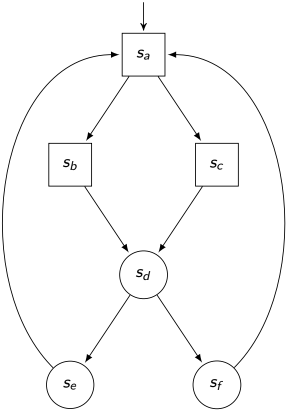
1. Overview of the situation for (multi) MP and TP games
   - ▷ No P algorithm known in one dimension
   - ▷ In multi dimensions, MP is coNP-complete
   - ▷ First contribution: **TP is undecidable in multi dimensions**

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**

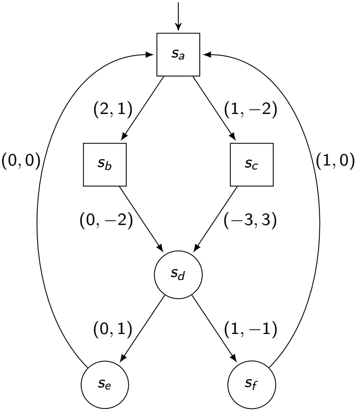
2. Introduction of **window objectives**
   - ▷ Conservative approximation of MP/TP
   - ▷ Break the complexity barriers
   - ▷ Algorithms, complexity and memory requirements
   - ▷ Several flavors of the objective

Turn-based games



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

Integer k-dim. payoff function



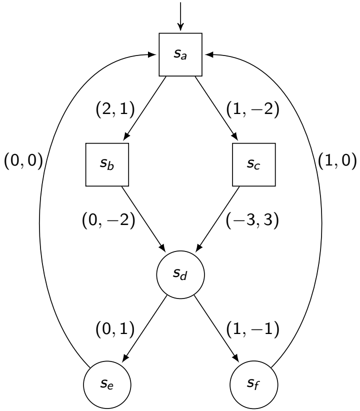
- $G = (S_1, S_2, E, \underline{k}, \underline{w})$
- $w : E \to \mathbb{Z}^k$
- Play $\pi = s_0 s_1 s_2 \ldots$
- *Total-payoff*

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

- *Mean-payoff*

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

TP and MP threshold problems



- **TP (MP) threshold problem**

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

  $$\exists ?\, \lambda_1 \in \Lambda_1 \text{ s.t. } \forall \lambda_2 \in \Lambda_2,$$
  $$\underline{\mathsf{TP}}(\mathsf{Outcome}_G(s_{\mathsf{init}}, \lambda_1, \lambda_2)) \geq v$$

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 $\cap$ coNP	mem-less		coNP-c. / NP $\cap$ coNP	infinite	mem-less
$\underline{\text{TP}}$ / $\overline{\text{TP}}$	NP $\cap$ coNP	mem-less		??	??	??

- ▷ See [EM79, Jur98, ZP96, GS09, CDHR10, VR11]
- ▷ 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
    $$\textbf{If } C_i == 0 \textbf{ do } \textit{this} \textbf{ else do } \textit{that}$$
- W.l.o.g. if the machine stops, it stops with both counters to zero

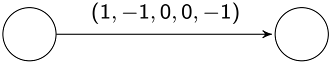
Encoding a 2CM in a 5-dim. TP game

- 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

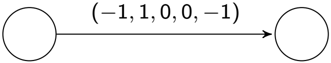
Key idea: after m steps, the TP has value $(v_1, -v_1, v_2, -v_2, -m)$ **iff** the 2CM counters have value (v_1, v_2)

Instructions

- Increment C_1

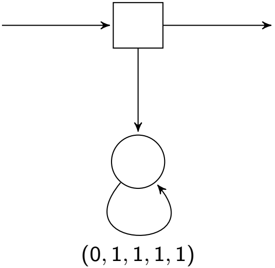


- Decrement C_1



Instructions

- Checking counter C_1 is non-negative

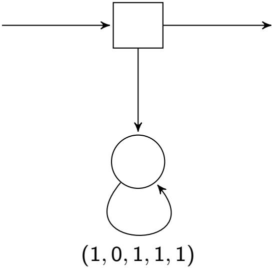


▷ If $\mathcal{P}_1$ cheats, he is doomed!

▷ Otherwise, $\mathcal{P}_2$ has no interest in retaliating.

Instructions

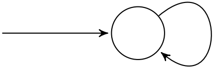
- Checking a zero test on C_1



- ▷ 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
  - ▷ Infimum vs. supremum
  - ▷ Describe what happens *at the limit*: no guarantee about a time frame

Motivations

- Classical MP and TP objectives have some drawbacks
  - ▷ Complexity issues
  - ▷ Infimum vs. supremum
  - ▷ Describe what happens *at the limit*: no guarantee about a time frame
- **Window objectives** consider what happens inside a *finite window sliding along a play*
  - ▷ 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 $= 4$

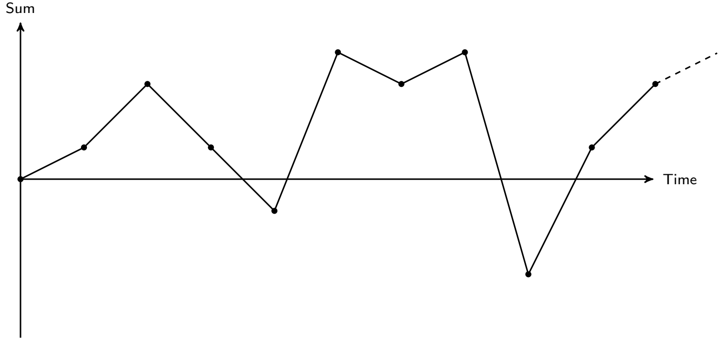


Illustration: WMP, threshold zero, maximal window $= 4$

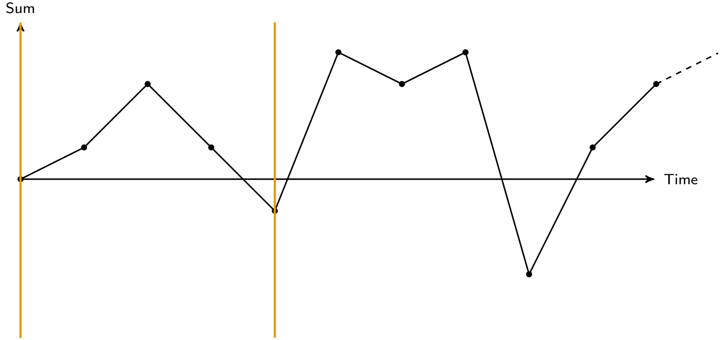


Illustration: WMP, threshold zero, maximal window = 4

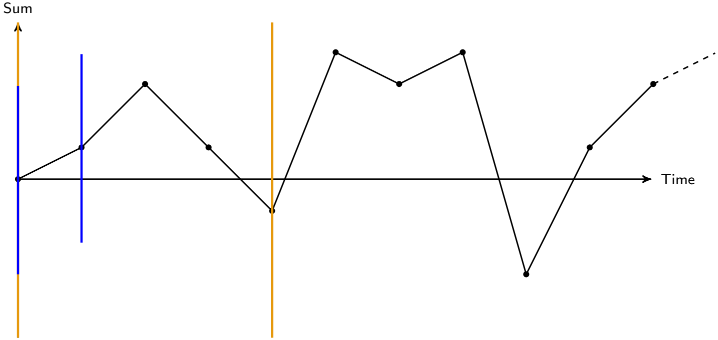


Illustration: WMP, threshold zero, maximal window = 4

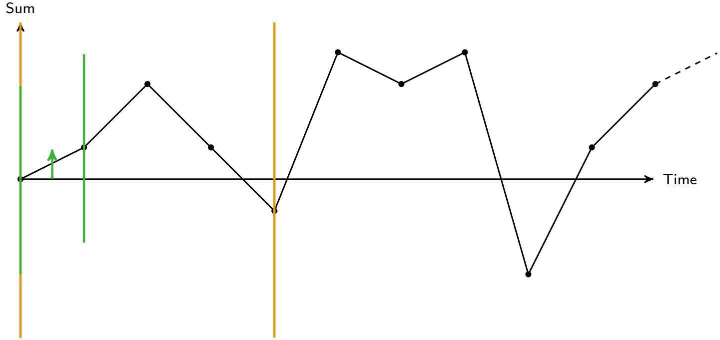


Illustration: WMP, threshold zero, maximal window $= 4$

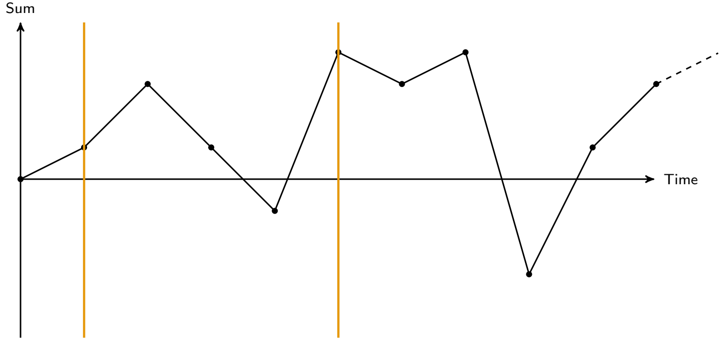


Illustration: WMP, threshold zero, maximal window = 4

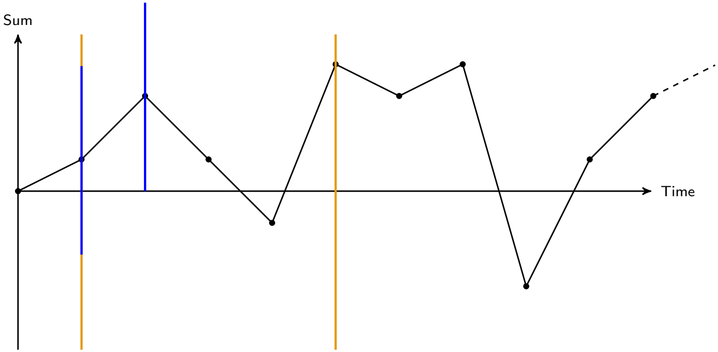


Illustration: WMP, threshold zero, maximal window = 4

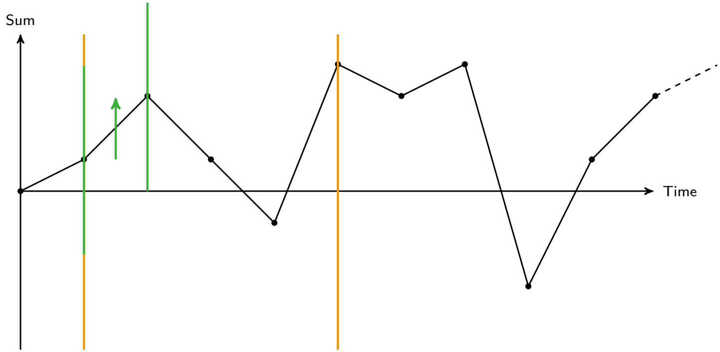


Illustration: WMP, threshold zero, maximal window = 4

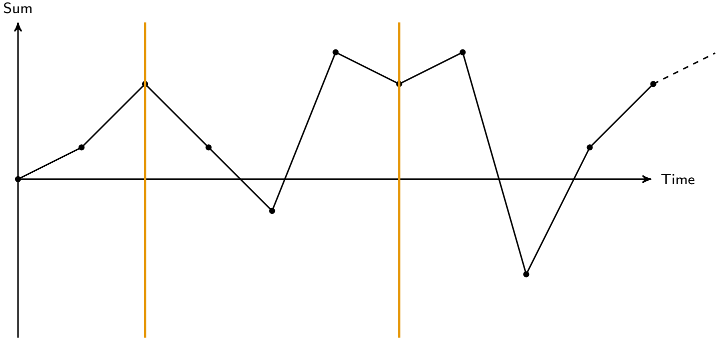


Illustration: WMP, threshold zero, maximal window = 4

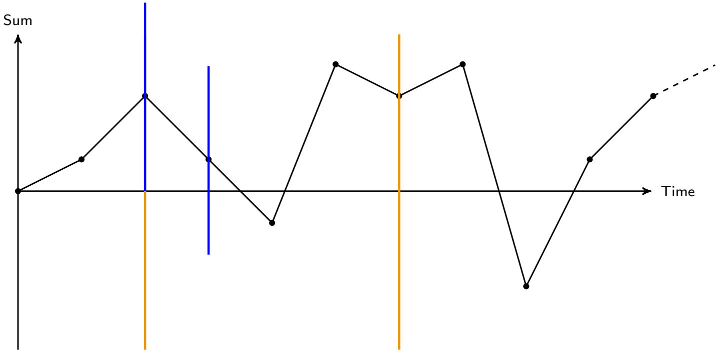


Illustration: WMP, threshold zero, maximal window $= 4$

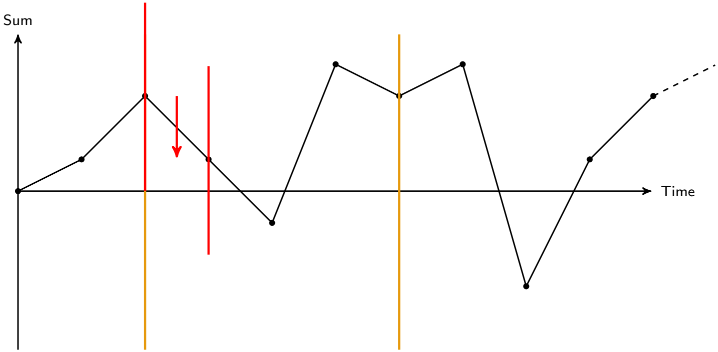


Illustration: WMP, threshold zero, maximal window = 4

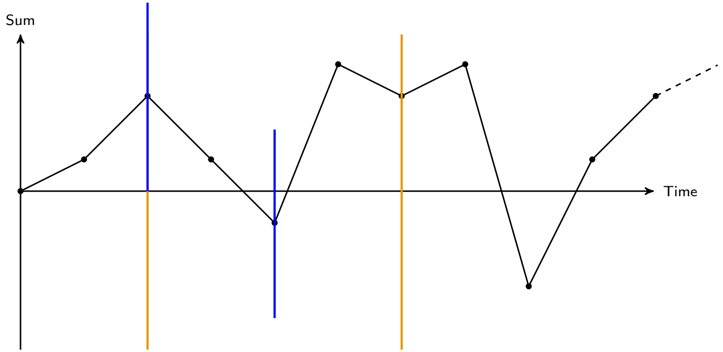


Illustration: WMP, threshold zero, maximal window = 4

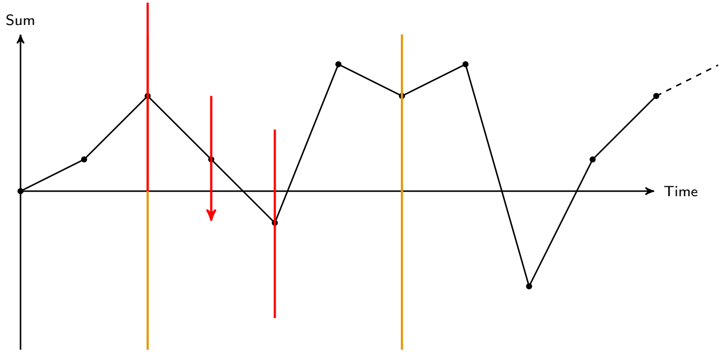


Illustration: WMP, threshold zero, maximal window = 4

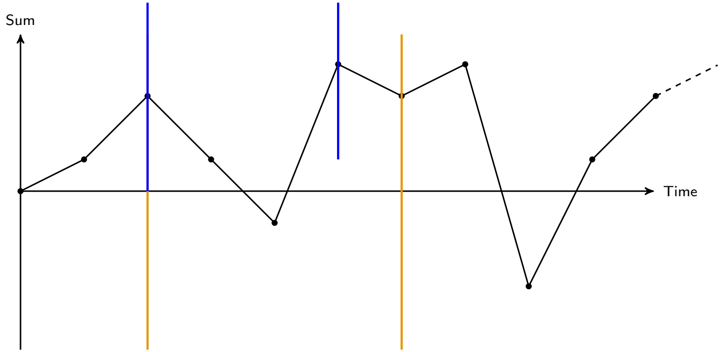
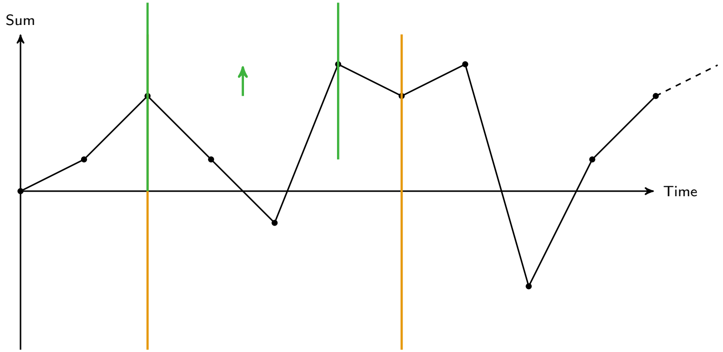


Illustration: WMP, threshold zero, maximal window = 4



Definitions

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

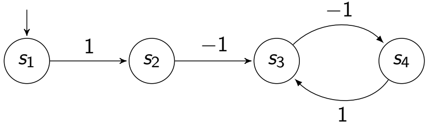
Definitions

- 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*: $\mathbf{BW} \equiv \Diamond \mathbf{DBW} \equiv \exists\, l_{\max},\ \mathbf{FW}(l_{\max})$

▷ A window *closes* when the sum becomes positive

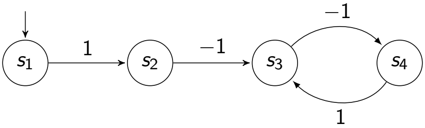
▷ A window is *open* if not yet closed

Examples

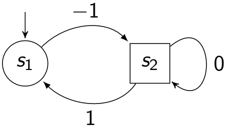


- $\triangleright$ **FW**(2) is satisfied, **DBW** is not, MP is satisfied.

Examples



▷ **FW**(2) is satisfied, **DBW** is not, MP is satisfied.



▷ MP is satisfied but none of the window objectives is.

Conservative approximation of MP ? (one-dim.)

The following are true

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

Results overview

<table>
<tr><td rowspan="2"></td><td colspan="3">one-dimension</td><td colspan="3">k-dimension</td></tr>
<tr><td>complexity</td><td>$\mathcal{P}_1$ mem.</td><td>$\mathcal{P}_2$ mem.</td><td>complexity</td><td>$\mathcal{P}_1$ mem.</td><td>$\mathcal{P}_2$ mem.</td></tr>
<tr><td>$\underline{\text{MP}}$ / $\overline{\text{MP}}$</td><td>NP $\cap$ coNP</td><td colspan="2">mem-less</td><td>coNP-c. / NP $\cap$ coNP</td><td>infinite</td><td>mem-less</td></tr>
<tr><td>$\underline{\text{TP}}$ / $\overline{\text{TP}}$</td><td>NP $\cap$ coNP</td><td colspan="2">mem-less</td><td>undec.</td><td>-</td><td>-</td></tr>
<tr><td>WMP: fixed polynomial window</td><td>P-c.</td><td colspan="2" rowspan="2">mem. req.
$\leq$ linear$(|S| \cdot l_{\max})$</td><td>PSPACE-h.
EXP-easy</td><td colspan="2" rowspan="2">exponential</td></tr>
<tr><td>WMP: fixed arbitrary window</td><td>P$(|S|, V, l_{\max})$</td><td>EXP-c.</td></tr>
<tr><td>WMP: bounded window problem</td><td>NP $\cap$ coNP</td><td>mem-less</td><td>infinite</td><td>NPR-h.</td><td>-</td><td>-</td></tr>
</table>

- $|S|$ 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.

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.**	-	-

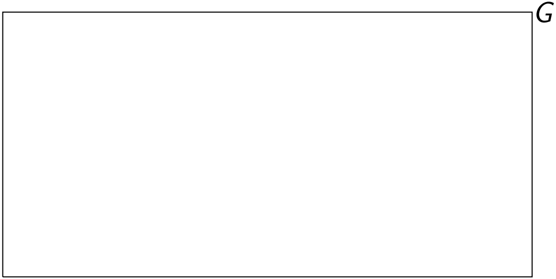
- $|S|$ the # of states, V the length of the binary encoding of weights, and $l_{\max}$ the window size.
- For one-dim. games with poly. windows, we are in P.
- No time to discuss everything. Focus.

High level sketch: top-down approach

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

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

▷ 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 $\mathbf{DFW}(l_{\max})$,

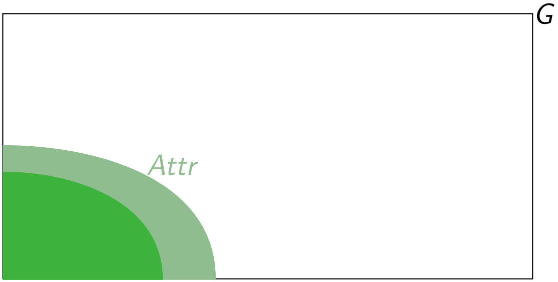
▷ Compute attractor, declare winning and recurse on subgame.

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 $\mathbf{DFW}(l_{\max})$,

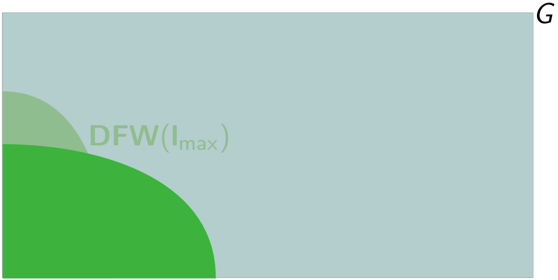
▷ Compute attractor, declare winning and recurse on subgame.

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.

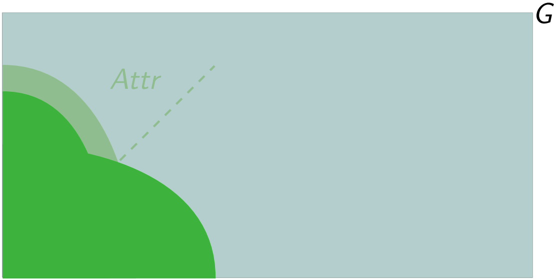


High level sketch: top-down approach

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

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

▷ Compute attractor, declare winning and recurse on subgame.

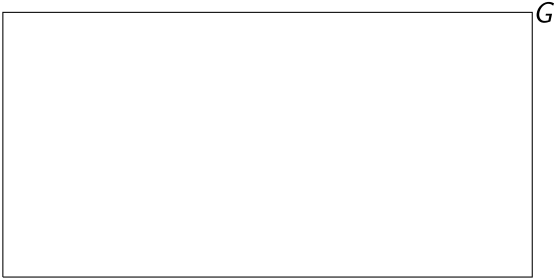


High level sketch: top-down approach

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

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

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly.

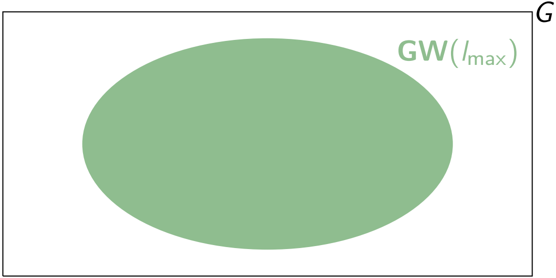


High level sketch: top-down approach

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

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

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly.

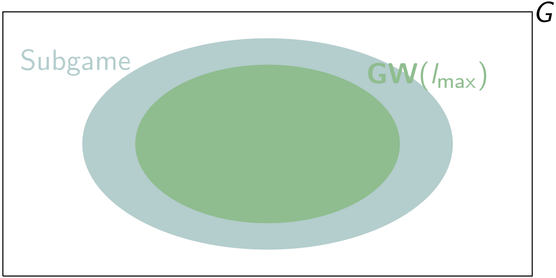


High level sketch: top-down approach

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

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

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly.

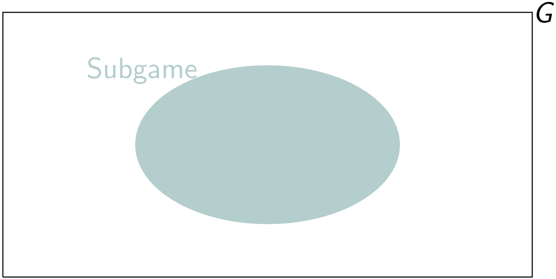


High level sketch: top-down approach

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

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

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly.

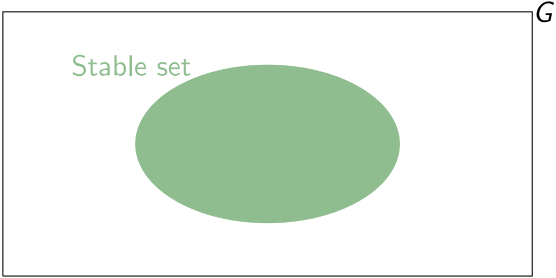


High level sketch: top-down approach

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

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

▷ Compute the stable set s.t. $\mathcal{P}_1$ can satisfy it repeatedly.



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.

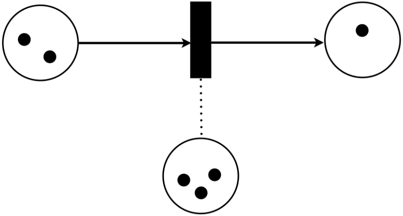
Approach

- 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

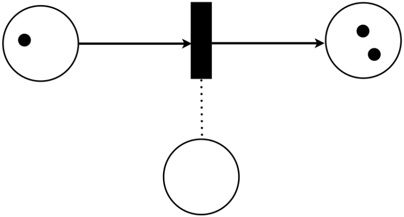
- Classic Petri net (places, tokens, transitions) with added *reset arcs*



▷ 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

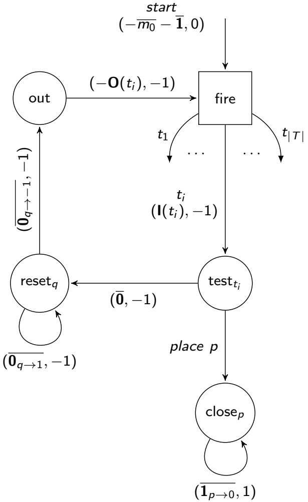
- Crux of the construction: encoding the markings
  - We use one dimension for each place
  - If a place p contains m tokens, then there will be an open window on dimension p with sum value $-m-1$
  - Hence **during a faithful simulation, all windows remain open** (you cannot consume tokens that do not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

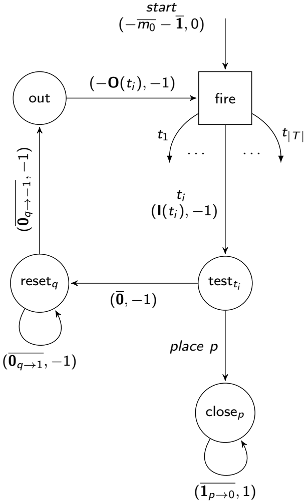
- $\mathcal{P}_2$ simulates the net
- $\mathcal{P}_1$ checks if he is faithful
- $\mathcal{P}_1$ wants to win the direct bounded window MP obj.
  - only able to do so if $\mathcal{P}_2$ cheats, i.e., if all runs terminate

The construction in a nutshell



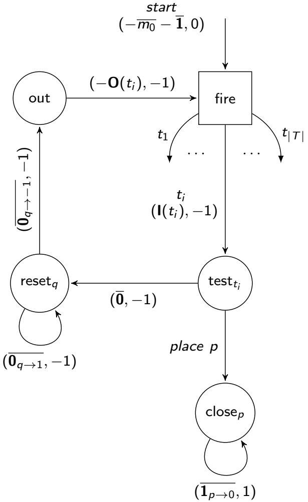
- ▷ The initial marking open corresponding windows in all places
- ▷ $\mathcal{P}_2$ chooses transitions to fire, which consume tokens
- ▷ $\mathcal{P}_1$ can branch or continue (and apply reset, then output)

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.
$\underline{\mathsf{MP}}$ / $\overline{\mathsf{MP}}$	NP $\cap$ coNP	mem-less		coNP-c. / NP $\cap$ coNP	infinite	mem-less
$\underline{\mathsf{TP}}$ / $\overline{\mathsf{TP}}$	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
- Provides timing guarantees
- Breaks the NP $\cap$ coNP barrier in one-dim. poly. window case
- Decidable approximation of TP in multi-dim. case
- *Open question*: is BW decidable in multi-dim. ?

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.

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

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

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

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.

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.