Games with Window Quantitative Objectives

Mickael Randour (LSV - CNRS & ENS Cachan)

Based on joint work with Krishnendu Chatterjee (IST Austria), Laurent Doyen (LSV - CNRS & ENS Cachan) and Jean-François Raskin (ULB).

25.02.2015 - Frontiers of Formal Methods 2015
General context: strategy synthesis in quantitative games

1. How complex is it to decide if a winning strategy exists?
2. How complex such a strategy needs to be? Simpler is better.
3. Can we synthesize one efficiently?

⇒ Depends on the winning objective.
Aim of this talk

- **New family of quantitative objectives**, based on mean-payoff (MP) and total-payoff (TP).
- Convince you of its *advantages* and *usefulness*.
- No technical stuff but feel free to check the full paper!
  - arXiv [CDRR13a]: abs/1302.4248
  - Conference version in ATVA’13 [CDRR13b], full version to appear in Information and Computation [CDRR15].
Classical MP and TP games

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

\[ \text{MP}(\pi) = \liminf_{n \to \infty} \frac{1}{n} \text{TP}(\pi(n)) \]
Classical MP and TP games

\[ TP(\pi) = \lim_{n \to \infty} \inf \sum_{i=0}^{i=n-1} W(s_i, s_{i+1}) \]

\[ MP(\pi) = \lim_{n \to \infty} \frac{1}{n} TP(\pi(n)) \]
Classical MP and TP games

\[
\text{MP}(\pi) = \lim_{n \to \infty} \inf \sum_{i=0}^{i=n-1} w(s_i, s_{i+1})
\]

\[
\text{TP}(\pi) = \lim_{n \to \infty} \frac{1}{n} \text{TP}(\pi(n))
\]
Classical MP and TP games

\[ TP(\pi) = \liminf_{n \to \infty} \sum_{i=0}^{i=n-1} w(s_i, s_{i+1}) \]

\[ MP(\pi) = \liminf_{n \to \infty} \frac{1}{n} TP(\pi(n)) \]
Classical MP and TP games

\[ \text{TP}(\pi) = \lim_{n \to \infty} \inf \sum_{i=0}^{i=n-1} w(s_i, s_{i+1}) \]

\[ \text{MP}(\pi) = \lim_{n \to \infty} \inf \frac{1}{n} \text{TP}(\pi(n)) \]
Classical MP and TP games

\[ TP(\pi) = \lim_{n \to \infty} \inf \sum_{i=0}^{i=n-1} w(s_i, s_{i+1}) \]

\[ MP(\pi) = \lim_{n \to \infty} \inf \frac{1}{n} TP(\pi(n)) \]
Classical MP and TP games

\[ TP(\pi) = \lim_{n \to \infty} \inf \sum_{i=0}^{i=n-1} w(s_i, s_{i+1}) \]

\[ MP(\pi) = \lim_{n \to \infty} \inf \frac{1}{n} TP(\pi(n)) \]

Then, \((2, 5, 2)^\omega\)
What do we know?

<table>
<thead>
<tr>
<th></th>
<th>one-dimension</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MP / MP</td>
<td>NP ∩ coNP</td>
<td>mem-less</td>
<td>coNP-c.</td>
<td>NP ∩ coNP</td>
<td>infinite</td>
</tr>
<tr>
<td>TP / TP</td>
<td>NP ∩ coNP</td>
<td>mem-less</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
</tbody>
</table>

What about multi total-payoff?

<table>
<thead>
<tr>
<th></th>
<th>one-dimension</th>
<th></th>
<th></th>
<th></th>
<th>$k$-dimension</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>complexity</td>
<td>$\mathcal{P}_1$ mem.</td>
<td>$\mathcal{P}_2$ mem.</td>
<td></td>
<td>complexity</td>
<td>$\mathcal{P}_1$ mem.</td>
<td>$\mathcal{P}_2$ mem.</td>
</tr>
<tr>
<td>$\text{MP} / \overline{\text{MP}}$</td>
<td>$\text{NP} \cap \text{coNP}$</td>
<td>mem-less</td>
<td></td>
<td>$\text{coNP-c.} / \text{NP} \cap \text{coNP}$</td>
<td>infinite</td>
<td>mem-less</td>
<td></td>
</tr>
<tr>
<td>$\text{TP} / \overline{\text{TP}}$</td>
<td>$\text{NP} \cap \text{coNP}$</td>
<td>mem-less</td>
<td></td>
<td></td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
</tbody>
</table>

- TP and MP look very similar in one-dimension
  - TP $\sim$ refinement of MP $= 0$

- Is it still true in multi-dimension?
What about multi total-payoff?

<table>
<thead>
<tr>
<th></th>
<th>one-dimension</th>
<th></th>
<th>k-dimensional</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>complexity</td>
<td>$P_1$ mem.</td>
<td>$P_2$ mem.</td>
</tr>
<tr>
<td>$MP / \overline{MP}$</td>
<td>$NP \cap coNP$</td>
<td>mem-less</td>
<td></td>
</tr>
<tr>
<td>$TP / \overline{TP}$</td>
<td>$NP \cap coNP$</td>
<td>mem-less</td>
<td></td>
</tr>
</tbody>
</table>

Unfortunately, no!

It would be nice to have... a **decidable** objective with the same flavor (some sort of approx.)
Is the complexity barrier breakable?

<table>
<thead>
<tr>
<th></th>
<th>one-dimension</th>
<th>k-dimensional</th>
</tr>
</thead>
<tbody>
<tr>
<td>complexity</td>
<td>$\mathcal{P}_1$ mem.</td>
<td>$\mathcal{P}_2$ mem.</td>
</tr>
<tr>
<td>$\mathsf{MP} / \overline{\mathsf{MP}}$</td>
<td>$\mathsf{NP} \cap \mathsf{coNP}$</td>
<td>mem-less</td>
</tr>
<tr>
<td>$\mathsf{TP} / \overline{\mathsf{TP}}$</td>
<td>$\mathsf{NP} \cap \mathsf{coNP}$</td>
<td>mem-less</td>
</tr>
</tbody>
</table>

- $\mathsf{NP} \cap \mathsf{coNP}$
- $\mathsf{mem-less}$
- $\mathsf{Undec.}$
- $\mathsf{infinite}$
- $\mathsf{mem-less}$

$\triangleright$ P membership for the one-dimension case is a long-standing open problem!

**It would be nice to have...**

an approximation decidable in **polynomial time**
Do we really want to play eternally?

<table>
<thead>
<tr>
<th></th>
<th>one-dimension</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P_1$ mem.</td>
<td>$P_2$ mem.</td>
<td></td>
</tr>
<tr>
<td>complexity</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$P_1$ mem.</th>
<th>$P_2$ mem.</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$MP / \overline{MP}$

$NP \cap coNP$

mem-less

$coNP-c. / NP \cap coNP$

infinite

mem-less

$MP$ and $TP$ give no timing guarantee: the “good behavior” occurs at the limit...

Sure, in one-dim., memoryless strategies suffice and provide bounds on cycles, but what if we are given an arbitrary play?

It would be nice to have...

a quantitative measure that specifies timing requirements
Window objectives: key idea

- **Window** of fixed size **sliding** along a play
  \(\sim\) defines a local finite horizon

- Objective: see a **local** \(MP \geq 0\) before hitting the end of the window
  \(\sim\) needs to be verified at every step
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window $= 4$
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window $= 4$
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window $= 4$
Window MP, threshold zero, maximal window = 4
Window MP, threshold zero, maximal window = 4
Multiple variants

- Given $l_{\text{max}} \in \mathbb{N}_0$, good window $GW(l_{\text{max}})$ asks for a positive sum in at most $l_{\text{max}}$ steps (one window, from the first state)

- *Direct Fixed Window*: $DFW(l_{\text{max}}) \equiv \Box GW(l_{\text{max}})$

- *Fixed Window*: $FW(l_{\text{max}}) \equiv \diamondsuit DFW(l_{\text{max}})$

- *Direct Bounded Window*: $DBW \equiv \exists l_{\text{max}}, DFW(l_{\text{max}})$

- *Bounded Window*: $BW \equiv \diamondsuit DBW \equiv \exists l_{\text{max}}, FW(l_{\text{max}})$
Multiple variants

- Given $l_{\text{max}} \in \mathbb{N}_0$, *good window* $GW(l_{\text{max}})$ asks for a positive sum in at most $l_{\text{max}}$ steps (one window, from the first state)

- **Direct Fixed Window**: $DFW(l_{\text{max}}) \equiv \Box GW(l_{\text{max}})$

- **Fixed Window**: $FW(l_{\text{max}}) \equiv \Diamond DFW(l_{\text{max}})$

- **Direct Bounded Window**: $DBW \equiv \exists l_{\text{max}}, DFW(l_{\text{max}})$

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

**Conservative approximations in one-dim.**

Any window obj. $\Rightarrow BW \Rightarrow MP \geq 0$

$BW \iff MP > 0$
Results overview

<table>
<thead>
<tr>
<th></th>
<th>one-dimension</th>
<th>k-dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>complexity</td>
<td>$P_1$ mem.</td>
</tr>
<tr>
<td>$\text{MP} / \text{MP}$</td>
<td>$\text{NP} \cap \text{coNP}$</td>
<td>mem-less</td>
</tr>
<tr>
<td>$\text{TP} / \overline{\text{TP}}$</td>
<td>$\text{NP} \cap \text{coNP}$</td>
<td>mem-less</td>
</tr>
<tr>
<td>WMP: fixed polynomial window</td>
<td>$P$-c.</td>
<td>mem. req.</td>
</tr>
<tr>
<td>WMP: fixed arbitrary window</td>
<td>$P(</td>
<td>S</td>
</tr>
<tr>
<td>WMP: bounded window problem</td>
<td>$\text{NP} \cap \text{coNP}$</td>
<td>mem-less</td>
</tr>
</tbody>
</table>

$|S|$ the # of states, $V$ the length of the binary encoding of weights, and $l_{\text{max}}$ the window size.
Results overview: advantages

<table>
<thead>
<tr>
<th></th>
<th>one-dimension</th>
<th>k-dimensional</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>complexity</td>
<td>$P_1$ mem.</td>
</tr>
<tr>
<td>MP / MP</td>
<td>NP $\cap$ coNP</td>
<td>mem-less</td>
</tr>
<tr>
<td>TP / TP</td>
<td>NP $\cap$ coNP</td>
<td>mem-less</td>
</tr>
<tr>
<td>WMP: fixed polynomial window</td>
<td>$P$-c.</td>
<td>mem req.</td>
</tr>
<tr>
<td></td>
<td>$P(</td>
<td>S</td>
</tr>
<tr>
<td>WMP: fixed arbitrary window</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WMP: bounded window problem</td>
<td>NP $\cap$ coNP</td>
<td>mem-less</td>
</tr>
</tbody>
</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 $P$.
- For multi-dim. games with fixed windows, we are **decidable**.
- Window objectives provide **timing guarantees**.
Taste of the proofs ingredients

- For those who like it technical, we use
  - 2CMs [Min61],
  - membership problem for APTMs [CKS81],
  - countdown games [JSL08],
  - generalized reachability [FH10],
  - reset nets [DFS98, Sch02, LNO+08],
  - ...

- *Open question*: is bounded window decidable in multi-dim.?
Check the full version on arXiv!  

**Thanks!**

Do not hesitate to discuss with us!
References I

V. Bruyère, E. Filiot, M. Randour, and J.-F. Raskin. 
Meet your expectations with guarantees: Beyond worst-case synthesis in quantitative games. 

Generalized mean-payoff and energy games. 

K. Chatterjee, L. Doyen, M. Randour, and J.-F. Raskin. 
Looking at mean-payoff and total-payoff through windows. 

K. Chatterjee, L. Doyen, M. Randour, and J.-F. Raskin. 
Looking at mean-payoff and total-payoff through windows. 

K. Chatterjee, L. Doyen, M. Randour, and J.-F. Raskin. 
Looking at mean-payoff and total-payoff through windows. 
Information and Computation, 2015. 
To appear.

Alternation. 
References II

K. Chatterjee, M. Randour, and J.-F. Raskin.
Strategy synthesis for multi-dimensional quantitative objectives.

K. Chatterjee, M. Randour, and J.-F. Raskin.
Strategy synthesis for multi-dimensional quantitative objectives.

C. Dufourd, A. Finkel, and P. Schnoebelen.
Reset nets between decidability and undecidability.

A. Ehrenfeucht and J. Mycielski.
Positional strategies for mean payoff games.

N. Fijalkow and F. Horn.
The surprising complexity of generalized reachability games.

Games through nested fixpoints.

H. Gimbert and W. Zielonka.
When can you play positionally?
M. Jurdziński, J. Sproston, and F. Laroussinie.
Model checking probabilistic timed automata with one or two clocks.

M. Jurdziński.
Deciding the winner in parity games is in UP ∩ co-UP.

Nets with tokens which carry data.

M.L. Minsky.
Recursive unsolvability of Post’s problem of “tag” and other topics in theory of Turing machines.

P. Schnoebelen.
Verifying lossy channel systems has nonprimitive recursive complexity.

Y. Velner and A. Rabinovich.
Church synthesis problem for noisy input.

U. Zwick and M. Paterson.
The complexity of mean payoff games on graphs.
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
  - $P_2$ 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
  - \( P_2 \) 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 \( P_2 \) needs infinite memory