Life is Random, Time is Not

Markov Decision Processes with Window Objectives

Thomas Brihaye (UMONS)
Youssouf Oualhadj (LACL – UPEC)

Florent Delgrange (UMONS & RWTH)
Mickael Randour (UMONS & FNRS)

May 29, 2019
UnRAVeL Seminar, RWTH Aachen University

Research supported by F.R.S.-FNRS (ManySynth)
Strategy synthesis

Finding good controllers for systems interacting with an environment

- **Game setting**: ensure a specified behavior against all possible strategies of the environment

- **Markov Decision Process (MDP) setting**:
  - environment **stochastic**
  - ensure a specified behavior with a sufficient probability

- **Classical objectives** reason about infinite runs *in their limit*

- **Window objectives in games** [CDRR15, BHR16]: *ensure a good behavior in a parametrized time frame all along the run*
  - *conservative approximations of classical objectives*

Aim of this talk

Introducing **window objectives** in the **stochastic** context
Example

- **Parity:** asks the **minimum priority seen infinitely often to be even**
  - $\Rightarrow$ canonical way of encoding $\omega$-regular properties
  - $\Rightarrow$ controller winning
Example

- **Parity:** asks the *minimum priority seen infinitely often to be even*

  - canonical way of encoding $\omega$-regular properties

  - controller winning

- **Window parity:** asks the *minimum priority seen within at most $\lambda > 0$ time steps to be even from each position of the infinite run*
Example

- **Parity:** asks the **minimum priority seen infinitely often to be even**
  - $\Rightarrow$ canonical way of encoding $\omega$-regular properties
  - $\Rightarrow$ controller winning

- **Window parity:** asks the **minimum priority seen within at most $\lambda > 0$ time steps to be even from each position of the infinite run**

- Every time $S_1$ is visited, there is a probability $> 0$ of not seeing the priority 0 before $\lambda$ steps $\left(\frac{1}{2^\lambda-1}\right)$
Example

- **Parity:** asks the **minimum priority seen infinitely often to be even**
  - canonical way of encoding $\omega$-regular properties
  - controller winning

- **Window parity:** asks the **minimum priority seen within at most $\lambda > 0$ time steps to be even from each position of the infinite run**
  - Every time $S_1$ is visited, there is a probability $> 0$ of not seeing the priority 0 before $\lambda$ steps ($\frac{1}{2^{\lambda-1}}$)
Example

- **Parity:** asks the **minimum priority seen infinitely often to be even**
  - canonical way of encoding \( \omega \)-regular properties
  - controller winning

![Diagram](image)

- **Window parity:** asks the **minimum priority seen within at most \( \lambda > 0 \) time steps to be even from each position of the infinite run**

- Every time \( S_1 \) is visited, there is a probability > 0 of not seeing the priority 0 before \( \lambda \) steps (\( \frac{1}{2^{\lambda-1}} \))
  - probability zero
Outline

1. Model
2. Objectives
   2.1 Classical long-run objectives
   2.3 Window objectives
   2.5 Decision problem
3. Fixed case
4. Prefix independent objectives
   4.1 The case of end-components
   4.3 Fixed and Bounded window
5. Extensions
Markov Decision Process (MDP)

An MDP $\mathcal{M} = (S, A, \Delta)$ is a tuple such that

- $S$ is the set of states of the system
- $A$ is the set of actions of the system
- $\Delta : S \times A \rightarrow \mathcal{D}(S)$ is the probability transition function
- $w : A \rightarrow \mathbb{Z}$ is a weight function
- $p : S \rightarrow \{0, 1, \ldots, d\}$ is a priority function ($d \leq |S| + 1$ w.l.o.g.)
Runs and strategies

- **Runs**: $\rho = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \ldots \xrightarrow{a_n} s_n \xrightarrow{\ldots} \in \text{Runs}(\mathcal{M})$ such that $\Delta(s_i, a_i)(s_{i+1}) > 0$

- **Strategy**: $\sigma$ chooses at each step an action
  - pure finite-memory strategies: choose actions according to a finite number of informations gathered in the past
  - pure memoryless strategies: $\sigma : S \rightarrow A$
Runs and strategies

- Fix a strategy $\sigma$
- $\rho \in \text{Runs}^\sigma(\mathcal{M}) \overset{\sim}{\Rightarrow} \text{runs compatible with } \sigma$
- Induce a *discrete-time Markov chain*: fully stochastic process $\mathcal{M}^\sigma$

$\overset{\sim}{\Rightarrow}$ **Event**: $E \subseteq \text{Runs}^\sigma(\mathcal{M})$

$\overset{\sim}{\Rightarrow}$ $\mathbb{P}^\sigma_{\mathcal{M}}[E]$: probability measure of the event $E$
Outline

1. Model

2. Objectives
   2.1 Classical long-run objectives
   2.3 Window objectives
   2.5 Decision problem

3. Fixed case

3.1 Reductions
3.3 Direct Fixed Window

4. Prefix independent objectives
   4.1 The case of end-components
   4.3 Fixed and Bounded window

5. Extensions
Mean-Payoff objective

- \( \text{MP}(\rho = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \ldots) = \lim \inf_{n \to \infty} \frac{1}{n} \cdot \sum_{i=0}^{n} a_i \)
  - Example: \( \forall n \in \mathbb{N}, \text{MP}((s_2 \xrightarrow{a_0})^n (s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^\omega) = \text{MP}((s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^\omega) = \frac{1}{2} \)
- MeanPayoff = \( \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{MP}(\rho) \geq 0 \} \)
Mean-Payoff objective

- $\text{MP}(\rho = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \ldots) = \lim \inf_{n \to \infty} \frac{1}{n} \cdot \sum_{i=0}^{n} a_i$
  
  - Example: $\forall n \in \mathbb{N}$, $\text{MP}((s_2 \xrightarrow{a_0})^n (s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^\omega) = \text{MP}((s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^\omega) = \frac{1}{2}$

- $\text{MeanPayoff} = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{MP}(\rho) \geq 0 \}$

- $\mathbb{P}^\sigma_{\mathcal{M},s_2}[\text{MeanPayoff}] = 1$
Parity objective

- Parity = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \min_{s \in \inf(\rho)} \rho(s) = 0 \text{ (mod 2)} \}
Parity objective

- Parity = \{ \rho \in \text{Runs}(\mathcal{M}) | \min_{s \in \inf(\rho)} \rho(s) = 0 \pmod{2} \}
- \mathcal{P}^{\sigma}_{\mathcal{M},s_2}[\text{Parity}] = 1
Window Objectives

- Runs that exhibit good behaviors within a configurable time frame
- Strengthen traditional objectives (correct behaviors at the limit)

\[ GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) | \text{good behavior in at most } \lambda \text{ steps from } S_0 \} \]

Window Mean-Payoff

\[ \rho = \]

- Positive sum in at most \( \lambda = 3 \) steps?
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

$\Rightarrow$ Make use of the **window formalism** to reason about **behaviors** in a given time bound $\lambda > 0$.

$$GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{good behavior in at most } \lambda \text{ steps from } S_0 \}$$

**Window Mean-Payoff**

- Positive sum in at most $\lambda = 3$ steps?
- Window of maximal size $\lambda = 3$
**Window Objectives**

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)
  
  Make use of the **window formalism** to reason about **behaviors** in a given time bound $\lambda > 0$.

$$GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{good behavior in at most } \lambda \text{ steps from } s_0 \}$$

**Window Mean-Payoff**

- Positive sum in at most $\lambda = 3$ steps?
- Window of maximal size $\lambda = 3$
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

\[ \Rightarrow \text{Make use of the window formalism to reason about behaviors in a given time bound } \lambda > 0. \]

\[ GW(\lambda) = \{ \rho \in \text{Runs}(M) \mid \text{good behavior in at most } \lambda \text{ steps from } S_0 \} \]

**Window Mean-Payoff**

- Positive sum in at most \( \lambda = 3 \) steps?
- Window of maximal size \( \lambda = 3 \)

\[ \rho = \begin{align*}
S_2 & \xrightarrow{a_0, -1} S_0 & \xrightarrow{a_0, -1} S_1 & \xrightarrow{a_1, 2} S_0 & \xrightarrow{a_0, -1} S_1 & \xrightarrow{a_1, 2} \cdots \\
S_0 & \xrightarrow{a_2, 5} S_2 & \\
S_1 & \xrightarrow{a_1, 2} S_1 & \\
\end{align*} \]
Window Objectives

- Runs that exhibit **good behaviors** within a configurable time frame
- **Strengthen traditional objectives** (correct behaviors at the limit)

\[ \Rightarrow \text{Make use of the window formalism to reason about behaviors in a given time bound } \lambda > 0. \]

\[ GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) | \text{good behavior in at most } \lambda \text{ steps from } s_0 \} \]

**Window Mean-Payoff**

- Positive sum in at most \( \lambda = 3 \) steps?
- Window of maximal size \( \lambda = 3 \)

\[ \rho = \begin{pmatrix} a_0, -1 & a_0, -1 & a_1, 2 & a_0, -1 \\ a_0, -1 & a_0, -1 & a_1, 2 & a_0, -1 \\ a_0, -1 & a_0, -1 & a_0, -1 & a_0, -1 \\ a_0, -1 & a_0, -1 & a_0, -1 & a_0, -1 \end{pmatrix} \]

\[ \Rightarrow \text{Good window of size } \lambda = 3 \]
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

→ Make use of the **window formalism** to reason about **behaviors** in a given time bound $\lambda > 0$.

$GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{good behavior in at most } \lambda \text{ steps from } s_0 \}$

**Window Mean-Payoff**

- Positive sum in at most $\lambda = 3$ steps?
- Window of maximal size $\lambda = 3$

$\rho = \begin{cases} s_2 \rightarrow s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \cdots \\ a_2, 5 \rightarrow a_0, -1 \rightarrow a_0, -1 \rightarrow a_1, 2 \rightarrow a_0, -1 \rightarrow a_1, 2 \rightarrow \cdots \end{cases}$

→ Good window of size $\lambda = 3$

→ $\rho \in GW_{mp}(3)$
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

\[ \Rightarrow \text{Make use of the window formalism to reason about behaviors in a given time bound } \lambda > 0. \]

\[ GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{good behavior in at most } \lambda \text{ steps from } s_0 \} \]

**Window Parity**

- Minimum priority is even in at most \( \lambda = 3 \) steps?

\[ \rho = \begin{cases} 1 & a_0 \\ 2 & a_0 \\ 0 & a_1 \\ 1 & a_0 \\ 0 & a_1 \\ \ldots \end{cases} \]
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**

- **Strengthen traditional objectives** (correct behaviors at the limit)

  ➞ Make use of the **window formalism** to reason about **behaviors** in a given time bound $\lambda > 0$.

  $GW(\lambda) = \{ \rho \in \text{Runs}(M) \mid \text{good behavior in at most } \lambda \text{ steps from } s_0 \}$

**Window Parity**

- Minimum priority is even in at most $\lambda = 3$ steps?

- Window of maximal size $\lambda = 3$

$$\rho = \begin{array}{ccc}
1 & 2 & 0 \\
\scriptstyle s_2 & a_0 & s_0 \\
1 & a_0 & a_0 \\
\scriptstyle s_0 & s_1 & a_1 \\
2 & 0 & 1 \\
\scriptstyle s_0 & s_0 & s_1 \\
0 & a_0 & a_1 \\
\end{array}$$
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

\[ \Rightarrow \text{Make use of the window formalism to reason about behaviors in a given time bound } \lambda > 0. \]

\[ GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) | \text{good behavior in at most } \lambda \text{ steps from } s_0 \} \]

**Window Parity**

- Minimum priority is even in at most \( \lambda = 3 \) steps?
- Window of maximal size \( \lambda = 3 \)

\[ \rho = \begin{array}{ccc}
1 & 2 & 0 \\
1 & a_0 & a_0 \\
1 & s_2 & s_0 \\
\end{array} \]

\[ \begin{array}{ccc}
1 & 0 & 1 \\
1 & 0 & a_1 \\
1 & s_0 & s_1 \\
\end{array} \]

\[ \begin{array}{ccc}
0 & 1 & 0 \\
0 & a_0 & a_1 \\
0 & s_1 & s_0 \\
\end{array} \]
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

\[ \Rightarrow \text{Make use of the window formalism to reason about behaviors in a given time bound } \lambda > 0. \]

\[ GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{good behavior in at most } \lambda \text{ steps from } s_0 \} \]

**Window Parity**

- Minimum priority is even in at most \( \lambda = 3 \) steps?
- Window of maximal size \( \lambda = 3 \)

\[ \rho = \begin{array}{c}
1 \\
2 \\
0 \\
1 \\
0 \\
\end{array} \]

\[ \text{min priority} = 1 \]
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

$\Rightarrow$ Make use of the **window formalism** to reason about **behaviors** in a given time bound $\lambda > 0$.

$$GW(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{good behavior in at most } \lambda \text{ steps from } s_0 \}$$

**Window Parity**

- Minimum priority is even in at most $\lambda = 3$ steps?
- Window of maximal size $\lambda = 3$

$\Rightarrow$ Good window of size $\lambda = 3$
Window Objectives

- Runs that exhibit **good behaviors** within a configurable **time frame**
- **Strengthen traditional objectives** (correct behaviors at the limit)

\[ \rightsquigarrow \text{Make use of the window formalism to reason about behaviors in a given time bound } \lambda > 0. \]

\[ \text{GW}(\lambda) = \{ \rho \in \text{Runs}(\mathcal{M}) \mid \text{good behavior in at most } \lambda \text{ steps from } s_0 \} \]

**Window Parity**

- Minimum priority is even in at most \( \lambda = 3 \) steps?
- Window of maximal size \( \lambda = 3 \)

\[ \rho = \begin{bmatrix} 1 & a_0 & a_0 & 1 & 1 \\ a_2 & a_0 & a_1 & a_0 & a_1 \\ s_2 & s_0 & s_1 & s_0 & s_1 \\ s_2 & s_0 & s_1 & s_0 & s_1 \\ s_2 & s_0 & s_1 & s_0 & s_1 \end{bmatrix} \]

\[ \rightsquigarrow \text{Good window of size } \lambda = 3 \]

\[ \rightsquigarrow \rho \in \text{GW}_{\text{par}}(3) \]
Window Objectives: Direct Fixed Window

- Fix a window size $\lambda > 0$
- **Direct Fixed Window objective**: $\text{DFW}(\lambda) \equiv \square \text{GW}(\lambda)$
- **Good Window** of maximal size $\lambda$ sliding along the run

$$\rho = S_2 \overset{a_0}{\rightarrow} (S_0 \overset{a_0}{\rightarrow} S_1 \overset{a_1}{\rightarrow})^\omega \in \text{DFW}(3)$$
Window Objectives: Direct Fixed Window

- Fix a window size $\lambda > 0$
- **Direct Fixed Window objective**: $\text{DFW}(\lambda) \equiv \square \text{GW}(\lambda)$
- Good Window of maximal size $\lambda$ sliding along the run

$$\rho = s_2 \xrightarrow{a_0} (s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^\omega \in \text{DFW}(3)$$
Window Objectives: Direct Fixed Window

- Fix a window size $\lambda > 0$
- **Direct Fixed Window objective:** $\text{DFW}(\lambda) \equiv \square \text{GW}(\lambda)$
- **Good Window** of maximal size $\lambda$ sliding along the run

$$\rho = s_2 \xrightarrow{a_0} (s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^\omega \in \text{DFW}(3)$$
Window Objectives: Direct Fixed Window

- Fix a window size $\lambda > 0$
- **Direct Fixed Window objective:** $DFW(\lambda) \equiv \Box GW(\lambda)$
- **Good Window** of maximal size $\lambda$ sliding along the run

$$\rho = s_2 \xrightarrow{a_0} (s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^\omega \in DFW(3)$$

---

**Inductive property**
Window Objectives: Direct Fixed Window

- Fix a window size $\lambda > 0$
- **Direct Fixed Window objective**: $\text{DFW}(\lambda) \equiv \Box \text{GW}(\lambda)$
- **Good Window** of maximal size $\lambda$ sliding along the run

$$\rho = S_2 \xrightarrow{a_0} (S_0 \xrightarrow{a_0} S_1 \xrightarrow{a_1})^\omega \in \text{DFW}(3)$$
Window Objectives: prefix independence

- Window objectives at the limit
- **Fixed Window objective**: $\text{FW}(\lambda) \equiv \Diamond \text{DFW}(\lambda) \equiv \Diamond \Box \text{GW}(\lambda)$
- **Bounded Window objective**: $\text{BW} \equiv \exists \lambda > 0, \text{FW}(\lambda)$

$$\rho = (s_2 \xrightarrow{a_0})^+ (s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1})^* \in \text{FW}(3) \cap \text{BW}$$
Threshold Probability Problem

Given

- an **MDP** $\mathcal{M}$ with state space $S$,
- a **maximal window size** $\lambda > 0$,
- an initial **state** $s \in S$,
- a **window objective** $\mathcal{O} \in \{\text{DFW}(\lambda), \text{FW}(\lambda), \text{BW}\}$ for mean-payoff or parity and
- a **probability threshold** $\alpha \in [0, 1] \cap \mathbb{Q}$,

decide if

$$\exists \sigma \quad \mathbb{P}_{\mathcal{M}, s}[\mathcal{O}] \geq \alpha$$
**Window Games**

\[ \exists \sigma_\bigcirc \forall \sigma_\square, \quad \rho = \text{Outcome}(s, \sigma_\bigcirc, \sigma_\square) \in \text{DFW}(\lambda) \]

- Existence of a uniform bound \( \lambda^* \) on the maximal window size
- \( \bigcirc \) is loosing for all \( \lambda \) since \( \square \) can choose \( a \rightarrow s \)

\[ \exists \sigma, \quad \mathbb{P}_s^\sigma[\text{DFW}(\lambda)] \geq \alpha \]

- no uniform bound on the maximal window size
- For \( \lambda > 1 \), \( \mathbb{P}_s[\text{DFW}(\lambda)] = 1 - \frac{1}{2^{\lambda-1}} \)
- \( \mathbb{P}_s[\text{DFW}(3)] = \Delta(s, a)(t) + \Delta(s, a)(s') \cdot \Delta(s, a)(t) = \frac{1}{2} + \frac{1}{4} = \frac{3}{4} \)
## Results overview

### Markov Decision Processes

<table>
<thead>
<tr>
<th>Model</th>
<th>Parity Complexity</th>
<th>Parity Memory</th>
<th>Mean-Payoff Complexity</th>
<th>Mean-Payoff Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFW</td>
<td>P-c. polynomial</td>
<td></td>
<td>EXPTIME/PSPACE-h.</td>
<td>pseudo-polynomial</td>
</tr>
<tr>
<td>FW</td>
<td></td>
<td>polynomial</td>
<td>P-c.</td>
<td>polynomial</td>
</tr>
<tr>
<td>BW</td>
<td>memoryless</td>
<td></td>
<td>NP ∩ coNP</td>
<td>memoryless</td>
</tr>
</tbody>
</table>

### Games [CDRR15, BHR16]

<table>
<thead>
<tr>
<th>Model</th>
<th>Parity Complexity</th>
<th>Parity Memory (P₁)</th>
<th>Mean-Payoff Complexity</th>
<th>Mean-Payoff Memory (P₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFW</td>
<td>P-c.</td>
<td>polynomial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FW</td>
<td></td>
<td>polynomial</td>
<td>P-c.</td>
<td>polynomial</td>
</tr>
<tr>
<td>BW</td>
<td>memoryless</td>
<td>NP ∩ coNP</td>
<td>memoryless</td>
<td></td>
</tr>
</tbody>
</table>
Outline

1. Model
2. Objectives
   2.1 Classical long-run objectives
   2.3 Window objectives
   2.5 Decision problem
3. Fixed case
4. Prefix independent objectives
   4.1 The case of end-components
   4.3 Fixed and Bounded window
5. Extensions
Strategies: memory requirements

- **Pure finite memory strategies** are sufficient for the threshold probability problem
- **Main tools:** natural reduction from
  - DFW to safety
  - FW to co-Büchi

~~ unfolding based on the maximal window size $\lambda$

- **Idea:** incorporate weights (resp. priorities) as well as the current number of steps in the state space of the MDP

### Mean-Payoff

\[ S \times \{0, 1, \ldots, \lambda\} \times \{-\lambda \cdot W, \ldots, 0\} \]

### Parity

\[ S \times \{0, 1, \ldots, \lambda\} \times \{0, 1, \ldots, d\} \]
Unfolding: example

∀σ in ℳ, ∃˜σ in ℳλ

MDP ℳ

∀σ in ℳ, ∃˜σ in ℳλ

∀σ in ℳ, ∃˜σ in ℳλ

ℳλ, unfolding of ℳ up to λ = 3
## Complexity and memory requirements

### DFW

<table>
<thead>
<tr>
<th>Model</th>
<th>Objectives</th>
<th>Fixed case</th>
<th>Prefix independent objectives</th>
<th>Extensions</th>
</tr>
</thead>
</table>

### Mean-Payoff

- **EXPTIME algorithm**
- **Pseudo-polynomial-memory optimal strategies**
- **PSPACE-hard [HK15]**
- **Pseudo-polynomial-memory strategies necessary**

### Parity

- **P algorithm**
- **Polynomial-memory optimal strategies**
- **P-hard [Bee80, Imm81]**
- **Polynomial-memory strategies necessary**

**Threshold probability problem**
Outline

1. Model
2. Objectives
   2.1 Classical long-run objectives
   2.3 Window objectives
   2.5 Decision problem
3. Fixed case
4. Prefix independent objectives
   4.1 The case of end-components
   4.3 Fixed and Bounded window
5. Extensions
End-component

Let $\mathcal{M}$ be an MDP, an \textit{end-component (EC)} is a strongly connected sub-MDP $\mathcal{C}$ of $\mathcal{M}$ formed by states and actions allowing to never leave $\mathcal{C}$.

- For any strategy $\sigma$, all runs $\rho$ compatible with $\sigma$ end up in an EC with probability one.

$\rightarrow$ the set of states and actions seen infinitely often in $\rho$ form an EC with probability one.

![Diagram of an MDP with end-components highlighted.](image-url)
End-component

Let $\mathcal{M}$ be an MDP, an **end-component (EC)** is a strongly connected sub-MDP $\mathcal{C}$ of $\mathcal{M}$ formed by states and actions allowing to never leave $\mathcal{C}$

- The number of ECs may be exponential in the size of $\mathcal{M}$
- An EC may have sub-ECs
- the union of two ECs with non-empty intersection is an EC

$\implies$ **Maximal end-component (MEC)** = ECs that cannot be extended

$\rightsquigarrow$ **MEC($\mathcal{M}$)** computable in polynomial time
MEC classification and Zero-one Law

- **Prefix independence** $\leadsto$ MECs analysis
- **Main result**: **MEC classification**
- strong link between ECs and 2 player games

$\leadsto$ 2 types of MECs: ✔ and ❌

Given an objective $\mathcal{O} \in \{FW(\lambda), BW\}$

- ✔ ∀s of $\mathcal{C}$ ∃$\sigma$, $P^\sigma_{\mathcal{C},s}[\mathcal{O}] = 1$
- ❌ ∀s of $\mathcal{C}$ ∀$\sigma$, $P^\sigma_{\mathcal{C},s}[\mathcal{O}] = 0$
Safe EC

An EC $C$ of state space $S_C$ is $\lambda$-safe iff $\forall s \in S_C$, $\exists \sigma \forall \rho \in \text{Runs}^\sigma(C), \rho \in \text{DFW}(\lambda)$

\[ \rightsquigarrow \text{Boils down to interpreting } C \text{ as a 2 player game} \]

- Determine if $C$ is safe?
  - $\rightarrow$ compute the winning set $\mathcal{W}_{\text{dfw}}$ of the DFW game version of $C$
    - $\mathcal{W}_{\text{dfw}} = S_C \implies C$ is safe
Safe EC

An EC $\mathcal{C}$ of state space $S_{\mathcal{C}}$ is $\lambda$-safe iff $\forall s \in S_{\mathcal{C}}$, $\exists \sigma \forall \rho \in \text{Runs}^\sigma(\mathcal{C})$, $\rho \in \text{DFW}(\lambda)$

~~ Boils down to interpreting $\mathcal{C}$ as a 2 player game

- $\exists \mathcal{C}$, $\lambda$-safe EC inside a super-EC $\mathcal{C}^*$
  
  $\rightarrow$ compute the winning set $\mathcal{W}_{\text{dfw}}$ of the DFW game version of $\mathcal{C}^*$
  
  $\mathcal{W}_{\text{dfw}} \neq \emptyset \implies \exists \lambda$-safe EC $\mathcal{C}$ inside $\mathcal{C}^*$
Good EC

An EC $\mathcal{C}$ is

- ✓ $\lambda$-good for $\lambda > 0$ if it contains a sub-EC $\mathcal{C}'$ which is $\lambda$-safe
- ✓ BW-good if it contains a sub-EC $\mathcal{C}'$ which is $\lambda$-safe for some $\lambda > 0$

<table>
<thead>
<tr>
<th>Model</th>
<th>Objectives</th>
<th>Fixed case</th>
<th>Prefix independent objectives</th>
<th>Extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>oo</td>
<td></td>
<td>oo</td>
<td>oo</td>
<td>o</td>
</tr>
</tbody>
</table>

- In all EC $\mathcal{C}$, $\exists \sigma^C_{\text{visit}}$ allowing to visit with probability one all states of $\mathcal{C}$
  \[ \Pr_{\sigma^C_{\text{visit}}} (\Diamond \mathcal{C}_{\text{safe}}) = 1 \]
- $\sigma^{\forall} = \text{combine } \sigma^C_{\text{visit}} \text{ and } \sigma_{\text{safe}}$
  \[ \Pr_{\sigma^{\forall}} [\Box] = 1 \text{ for } \Box \in \{\text{FW}(\lambda), \text{BW}\} \]
BW-goodness

- **Uniform bound \( \lambda^* \) in window games**: window size \( \lambda^* \) is sufficiently large so that the bounded version coincides with the fixed one
- Win DFW for \( \lambda^* \) inside \( \mathcal{C} \) \( \implies \mathcal{C} \) is BW-good ✔
- \( \lambda^* \) polynomial for window parity games
- \( \lambda^* \) pseudo-polynomial for window mean-payoff games

*Can we do better?*


**BW-goodness**

- **Uniform bound $\lambda^*$ in window games**: window size $\lambda^*$ is sufficiently large so that the bounded version coincides with the fixed one.
- Win DFW for $\lambda^*$ inside $C \implies C$ is BW-good ✓
- $\lambda^*$ polynomial for window parity games
- $\lambda^*$ pseudo-polynomial for window mean-payoff games

*Can we do better? Yes!* [CHH09, CDRR15, BHR16]

**Bad Window** =

- Mean-Payoff
  $\neg$Total Payoff = \{ $\rho = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots | \limsup_{n \to \infty} \sum_{i=0}^{n} w(a_i) < 0$ \}

- Parity
  co-Weak Parity = \{ $\rho = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots | \min_{i>0} [\rho(s_i)]$ is odd \}

- $\exists \lambda > 0$, $\text{DFW}(\lambda) \equiv \exists \lambda > 0$, $\square \text{GW}(\lambda) \equiv \neg \Diamond \text{Bad Window}$

- **Classification**: compute the safe region $\mathcal{W}_{\text{safe}}$ w.r.t. Bad Window of the 2PG version of $C$
  - $\mathcal{W}_{\text{safe}} \neq \emptyset \implies C$ is BW-good ✓ with $\sigma_{\text{safe}}$, a memoryless (safety) strategy
Bad EC?

- Is there a safe sub-EC inside the 2PG version of $C$?
- Fix any finite-memory strategy $\sigma$ inside $C$.
- The set of states and actions seen infinitely often form sub-ECs $C_{\text{sub}}$ with probability one.
- In these sub-ECs $C_{\text{sub}}$, $\exists \rho \notin \text{DFW}(\lambda)$ (otherwise $C_{\text{sub}}$ is safe).
- Extract a bad prefix $\rho^x$ (=bad window) in $\rho$.
- $\rho^x$ is repeated infinitely often with probability one in $C_{\text{sub}}$.
- $\forall C_{\text{sub}}, \mathbb{P}^{\sigma}_{C_{\text{sub}}}[\text{DFW}(\lambda)] = 0 \Rightarrow \mathbb{P}^\sigma_C[\emptyset] = 0 \quad \emptyset \in \{\text{FW}(\lambda), \text{BW}\}$
Summary: complexity and strategies

Given $\mathcal{O} \in \{FW(\lambda), BW\}$, 2 types of MECs: ✓ and ✗

✓ $\forall s \text{ of } \mathcal{C} \exists \sigma, \quad P^\sigma_{\mathcal{C},s}[\mathcal{O}] = 1$

✗ $\forall s \text{ of } \mathcal{C} \forall \sigma, \quad P^\sigma_{\mathcal{C},s}[\mathcal{O}] = 0$

<table>
<thead>
<tr>
<th>MEC classification</th>
<th>Mean-Payoff</th>
<th>Parity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed window (FW)</td>
<td><img src="image" alt="Fixed window (FW)" /></td>
<td></td>
</tr>
<tr>
<td>Bounded window (BW)</td>
<td><img src="image" alt="Bounded window (BW)" /></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>in P (2PG)</th>
<th>in P (2PG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed window (FW)</td>
<td><img src="image" alt="Fixed window (FW)" /></td>
<td><img src="image" alt="Fixed window (FW)" /></td>
</tr>
<tr>
<td>Bounded window (BW)</td>
<td><img src="image" alt="Bounded window (BW)" /></td>
<td><img src="image" alt="Bounded window (BW)" /></td>
</tr>
</tbody>
</table>

- pure polynomial finite-memory strategy $\sigma_\surd := \sigma^C_{\text{visit}} + \sigma_{\text{safe}}$
- pure memoryless strategy $\sigma_\surd := \sigma^C_{\text{visit}} + \sigma_{\text{safe}}$
Optimal strategies for FW and BW

\[ \sigma^{\text{max}} = \arg \max_{\sigma} P^{\sigma}[\Diamond \checkmark] \]

\[ \sigma^* := \sigma^{\text{max}} \left[ \Diamond \checkmark \right] + \sigma^{\checkmark} \left[ \checkmark \right] \]

until reaching \( \checkmark \) in \( P^{\text{max}} \)
Summary

Markov Decision Processes

<table>
<thead>
<tr>
<th></th>
<th>parity</th>
<th>mean-payoff</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>complexity</td>
<td>memory</td>
</tr>
<tr>
<td>DFW</td>
<td>P-c.</td>
<td>polynomial</td>
</tr>
<tr>
<td>FW</td>
<td>memoryless</td>
<td>NP ∩ coNP</td>
</tr>
<tr>
<td>BW</td>
<td>memoryless</td>
<td>memoryless</td>
</tr>
</tbody>
</table>

EXPTIME/PSpace-h.  pseudo-polynomial
P-c.  polynomial
Outline

1. Model
   2. Objectives
      2.1 Classical long-run objectives
      2.3 Window objectives
      2.5 Decision problem

2. Fixed case

3. Reductions
   3.1 Reductions
   3.3 Direct Fixed Window

4. Prefix independent objectives
   4.1 The case of end-components
   4.3 Fixed and Bounded window

5. Extensions
• **Expected window size**: strategies maintaining the best time bounds possible in their local environment
  
  \[
  \min_\sigma E^\sigma_{\mathcal{M}}[\lambda] = \min_\sigma \sum_{\lambda > 0}^{\infty} \lambda \cdot P^\sigma_{\mathcal{M}}[FW(\lambda) \setminus FW(\lambda - 1)]
  \]

  - **Refine the classification process**: identify best window size \(\lambda\) in each MEC by binary search
  - **Contract each good MEC and assign \(\lambda\) as entering weight**

• **Multi window objectives**:
  
  - **DFW**: extends the unfolding for multiple dimensions
  - **MEC classification**: games with multiple window objectives

• **Tool support**: implementation in **STORM**
  
  + DFW, FW and BW objectives for parity and mean-payoff
  + Efficient unfoldings for DFW
  + Window games for parity [BHR16] and mean-payoff [CDRR15]
  + Total payoff games: efficient pseudo-polynomial time algorithm with value iteration [BGHM14]
  + Weak parity games [CHH09]
  + Rich MEC classification methods
  + Strategy synthesis (export in .dot format)
References I


References II

