Parity Games
Tree Arena
and
Zermelo's theorem
Tree arenas

A branch = a play in the game tree
Example of a game tree
Tree arenas

- A (finite) game arena $A=(S_1,S_2,E,s_{\text{init}})$ is a (finite) tree arena if the underlying directed graph is a rooted tree. In a tree arena, the set $\text{Leaves} \subseteq S_1 \cup S_2$ are the states that have no successor.

- A game is played on a tree arena by moving a token from the root to a leaf where the game ends. As usual, the player that owns the state on which the token lies moves the token.

- A strategy for Player $j$, $j\in\{1,2\}$, in a tree arena is a function from states to states: $\lambda_j : S_j \to S$. Note that this definition is general enough as in a tree, a state uniquely identifies its unique history.

- The winning objectives in a tree arena are defined by partitioning the leaves into $\text{Leaves}_1$ that are winning for Player 1 and $\text{Leaves}_2$ that are winning for Player 2 (with this definition, the game is zero-sum).
Zermelo's theorem
Determinacy for finite trees

**Theorem [Zermelo 1923]**. In a game tree, either Player 1 has a strategy at the root to force a leaf in $\text{Leaves}_1$ or Player 2 has a strategy at the root to force a leaf in $\text{Leaves}_2 = \text{Leaves}\backslash\text{Leaves}_1$.

This theorem is a consequence of the **backward induction principle**.
Zermelo's theorem

Zermelo's theorem states that at the root of the tree

*either* Player 1 has a strategy to force a *green leaf (Leaves$_1$)*,

*or* Player 2 has a strategy to force a *red leaf (Leaves$_2$)*.
Proof by induction on the depth of the tree.

Each state of the tree can be labelled:

- in **green** if Player 1 can force green leaves from there, and
- in **red** if Player 2 can force red leaves from there.
Backward induction principle

- **Base case:** \text{Leaves}_1 \text{ are winning for Player 1, and } \text{Leaves}_2 \text{ are winning for Player 2 (and so losing for Player 1).}

- **Induction:**
  
  - if \( s \in S_1 \) and there exists one of the child \( s' \) of \( s \) that is winning for Player 1 then \( s \) is winning for Player 1, otherwise it is winning for Player 2
  
  - if \( s \in S_2 \) and all children \( s' \) of \( s \) are winning for Player 1 then \( s \) is winning for Player 1, otherwise it is winning for Player 2
Zermelo's theorem

**Base case**: tree=one leaf. Trivial.
Zermelo's theorem

Induction.

The state is green if there exists one green successor

Player 1’s state
Zermelo's theorem
Zermelo's theorem

Induction.

The state is red if all successors are red
Zermelo's theorem

Induction.
Zermelo's theorem

Induction.

The state is red if there exists one red successor.
Zermelo's theorem

Induction.
Zermelo's theorem

Induction.

The state is green if all successors are green.
Zermelo's theorem

Induction.
Zermelo and Chess game
Zermelo and Chess game

Finite tree:
- stop the game if we observe three times the same board configuration (draw)
Application of Zermelo’s theorem to Chess

• Corollary of Zermelo's theorem:

  in Chess we know that either the white or black can force a win or a draw, and so he/she is theoretically unbeatable.
First Cycle Unfolding
(from a graph to a tree)
Unfolding - an example

Unfolding of A starting in 1
First cycle unfolding

Let $A=(S_1, S_2, E, s_{init})$ be a two-player game arena (in which $E$ is assumed to be total). Its **first cycle unfolding** is the tree arena $\text{FCU}(A)=(\Pi_1, \Pi_2, B, n_{init})$ where:

- $\Pi_1$ is the set of prefixes $\pi=s_0s_1...s_n$ of initial plays in $A$ s.t. $\text{last}(\pi) \in S_1$ and $\pi(0..k-1)$ does **not** contain a cycle

- $\Pi_2$ is the set of prefixes $\pi=s_0s_1...s_n$ of initial plays in $A$ s.t. $\text{last}(\pi) \in S_2$ and $\pi(0..k-1)$ does **not** contain a cycle

- Notations: $\Pi = \Pi_1 \cup \Pi_2$, we identify elements of $\Pi$ with prefixes of initial plays in $A$, and so we write $\text{last}(\pi)$ to denote the last state of the prefix which is an element of $S$

- $B=\{ (\pi, \pi') \in \Pi \times \Pi \mid \pi'=\pi.s \land (\text{last}(\pi), s) \in E \}$

- The states without successor in the tree are $\text{Leaves}=\{ \pi \in \Pi \mid \neg \exists \pi': B(\pi, \pi') \}$
  
  =\{ $\pi$ | $\pi$ contains a cycle $\}$. 

Simple cycles

- A **simple cycle** $c = s_0 \ s_1 \ldots \ s_k$ in the game arena $A$ is a finite path in $A$ such that $s_0 = s_k$ (cycle) and for all $i, j$ s.t. $0 \leq i < j < k$, we have that $s_i \neq s_j$ (i.e. $c$ does not contain additional cycles).

- Let $\pi$ be a leaf in $\text{FCU}(A)$, then $\pi = \pi_1.s.\pi_2.s$ where $s.\pi_2.s$ is a **simple cycle**. We write $\text{cycle}(\pi)$ to denote $s.\pi_2.s$.

- Note that the **length** of any branch in this unfolding is bounded by $|S| + 1$, i.e. the number of states in the two-player game arena + 1.
Strategies in $\text{FCU}(A)$

- A strategy $\lambda_j$, $j \in \{1, 2\}$, in $\text{FCU}(A)$ is a function $\lambda_j : \Pi_j \backslash \text{Leaves} \rightarrow \Pi$ s.t. for all $\pi \in \Pi_j$: $(\pi, \lambda_j(\pi)) \in B$

- The outcome of $\lambda_j$ is the set of states $\pi \in \Pi$ that are compatible with $\lambda_j$:
  \[
  \text{Outcome}_{\text{FCU}(A)}(\lambda_j) = \{ \pi \in \Pi | \forall \pi_1, \pi_2 : \pi_1 \leq \pi, \pi_2 \leq \pi \text{ s.t. } B(\pi_1, \pi_2), \pi_1 \in \Pi_j : \pi_2 = \lambda_j(\pi_1) \}\]

- Given a set $\text{Leaves}_j$ of good leaves for Player $j$, we say that $\lambda_j$ is winning for Player $j$ if $\text{Outcome}_{\text{FCU}(A)}(\lambda_j) \cap \text{Leaves} \subseteq \text{Leaves}_j$

- Corollary of Zermelo's theorem: if there is no winning strategy $\lambda_1$ for Player 1 then Player 2 has a winning strategy for $\text{Leaves}_2 = \text{Leaves} \backslash \text{Leaves}_1$. 
Strategies in **FCU**(A)
In this unfolding:
- Player 1 has a strategy to force \{1,3\}
- Player 1 has no strategy to force \{1\}
which is equivalent to say that Player 2 has a strategy to force \{3,4\}
Cycle decomposition of a history

• Let $\pi$ be a history in the two-player game arena $A=(S_1,S_2,E,s_{init})$.

• The cycle decomposition of $\pi$, $\text{dec}(\pi)=(C(\pi),\beta(\pi))$ is a decomposition of $\pi$ into a sequence simple cycles $C(\pi)$ and a acyclic part $\beta(\pi)$ —called the residue, and it is defined inductively as follows:

  • for a single state history $\pi=s$, $\text{dec}(\pi)=(\emptyset,s)$;

  • for a history $\pi'=\pi.s$:

    • if $s \in \beta(\pi)$ and $\beta(\pi)=\pi_1.s.\pi_2$, then $C(\pi')=C(\pi);\{s.\pi_2.s\}$ and $\beta(\pi')=\pi_1.s$

    • if $s \not\in \beta(\pi)$ then $C(\pi')=C(\pi)$ and $\beta(\pi')=\beta(\pi).s$
Cycle decomposition

\[ \pi = s_0 e_0 s_1 e_1 s_2 e_2 s_3 e_3 s_4 e_4 s_5 e_5 s_6 e_6 s_7 e_7 \ldots \]

Residue
\[ \beta(\pi[..0])=s_0 \]
\[ \beta(\pi[..1])=s_0s_1 \]
\[ \beta(\pi[..2])=s_0s_1s_2 \]
\[ \beta(\pi[..3])=s_0s_1s_2s_3 \]
\[ \beta(\pi[..4])=s_0s_1s_2s_3s_4 \]
\[ \beta(\pi[..5])=s_0s_1s_2s_3s_4s_5 \]
\[ \beta(\pi[..6])=s_0s_1s_2s_5s_6 \]
\[ \beta(\pi[..7])=s_0s_6s_7 \]

Cycles
\[ C(\pi[..0])=\{\} \]
\[ C(\pi[..1])=\{\} \]
\[ C(\pi[..2])=\{\} \]
\[ C(\pi[..3])=\{\} \]
\[ C(\pi[..4])=\{\} \]
\[ C(\pi[..5])=\{s_3s_4s_5\} \]
\[ C(\pi[..6])=\{s_3s_4s_5,s_1s_2s_5s_6\} \]
\[ C(\pi[..7])=\{s_3s_4s_5,s_1s_2s_5s_6\} \]

“Stack decomposition”
Property of the cycle decomposition of a play

Lemma (residues). For all plays \( \rho \in \text{Play}(A) \), let

\[ S = \bigcup_{i \in \mathbb{N}} \beta(\rho(0..i)), \]

i.e. the set of states that appear in residues along the cycle decomposition of \( \rho \), then

\[ S = \text{visit}(\rho), \]

i.e. this set of states is exactly equal to the set of states that appears along \( \rho \).
Property of the cycle decomposition of a play

**Lemma (cycles).** For all plays $\rho \in \text{Play}(A)$, let $C$ be the set of cycles $c$ such that there exist infinitely many positions $i \geq 0$ in $\rho$ such that $c$ is added to $C(\rho(0..i))$, then the following equality holds $\inf(\rho) = \bigcup_{c \in C} \text{visit}(c)$.

**Proof.** Assume that $s \in \bigcup_{c \in C} \text{visit}(c)$ then there are infinitely many simple cycles along $\rho$ in which $s$ appears and so $s \in \inf(\rho)$. Now, assume that $s \in \inf(\rho)$, then between any two consecutive positions in which $s$ appear along $\rho$, a simple cycle that contains $s$ is added to the decomposition. As there are only finitely many simple cycle in the arena $A$, there is $c \in C$ such that $s \in \text{visit}(c)$, and so $s \in \bigcup_{c \in C} \text{visit}(c)$. QED.
Transfer of strategies from FCU to game arena

• Let $\lambda_j$ be a strategy for Player j in $\text{FCU}(A)$

• We associate to $\lambda_j$ the strategy $\lambda_j^*$ in the game arena as follows:

  for all initial histories $h \in \text{Hist}_j(A)$,

  $$\lambda_j^*(h) = \text{last}(\lambda_j(\beta(h)))$$

→ we play like in the tree up to a leaf. From there, we continue to play as if we were at the ancestor that starts the simple cycle at that leaf, and so on...
Transfer of strategies
An example

We play in the graph as in the tree!
When a leaf is reached we continue from the ancestor.
Properties of $\lambda_j$ and $\lambda_j^*$

**Lemma (residue transfer).** The set of states in $\text{FCU}(A)$ that are compatible with strategy $\lambda_j$ is equal to the set of states that are visited by outcomes of $\lambda_j^*$ in $A$, i.e.

\[
\{ s \in S \mid \exists \rho \in \text{Outcome}_A(\lambda_j^*). \exists i \geq 0: s = \rho(i) \} = \{ \text{last}(\pi) \mid \pi \in \Pi \text{ is compatible with } \lambda_j \}
\]

**Proof.** This is a consequence of the residues lemma.

**Corollary.** If $T \subseteq S$ is a set of states such that for all branches $b \in \text{Outcome}_{\text{FCU}(A)}(\lambda_j)$, $\text{visit}(b) \cap T \neq \emptyset$ then for all $\rho \in \text{Outcome}_A(\lambda_j^*)$: $\text{visit}(\rho) \cap T \neq \emptyset$.

**Corollary.** If $T \subseteq S$ is a set of states such that for all branches $b \in \text{Outcome}_{\text{FCU}(A)}(\lambda_j)$, $\text{visit}(b) \subseteq T$ then for all $\rho \in \text{Outcome}_A(\lambda_j^*)$: $\text{visit}(\rho) \subseteq T$. 
Properties of $\lambda_j$ and $\lambda_j^*$

**Lemma (cycle transfer).** The set of cycles in $\text{FCU}(A)$ that are compatible with strategy $\lambda_j$ is equal to the set of cycles that are visited by outcomes of $\lambda_j^*$ in $A$, i.e.

$$\{ \pi \in \mathcal{C}(A) \mid \exists i \geq 0. \exists \rho \in \text{Outcome}_A(\lambda_j^*), 0 \leq i < j : \pi = \rho(i..j) \}$$

$$= \{ \text{cycle}(\pi) \mid \pi \in \Pi: \pi \text{ is compatible with } \lambda_j \}$$

**Proof.** This is a consequence of the cycles lemma.

**Corollary.** If $T \subseteq S$ is a set of states such that for all branches $b \in \text{Outcome}_{\text{FCU}(A)}(\lambda_j)$, $\text{visit}(\text{cycle}(b)) \subseteq T$ then for all $\rho \in \text{Outcome}_A(\lambda_j^*)$: $\text{inf}(\rho) \subseteq T$. 
Applications of the first cycle unfolding
A solution to Büchi games through first cycle unfolding

- We first present a simple solution for Büchi games using first cycle unfolding

- Let \( A = (S_1, S_2, E, s_{\text{init}}, \text{Büchi}(T)) \) be a two-player turn based arena with winning objective \( \text{Büchi}(T) \).

- We consider \( \text{FCU}(A) = (\Pi_1, \Pi_2, B, \pi_{\text{init}}) \) and define \( \text{Leaves}_1 \) as the set of leaves \( \pi \) such that \( \text{visit}(\text{cycle}(\pi)) \cap T \neq \emptyset \).
Unfolding - an example

We want to decide if Player 1 has a strategy to force an infinite number of visits to \{3,4\} (Büchi objective)
A reduction to first cycle unfolding

**Theorem.** Player 1 has a winning strategy in $A=(S_1,S_2,E,s_{\text{init}},\text{Büchi}(T))$ iff Player 1 has a winning strategy in $\text{FCU}(A)=(\Pi_1,\Pi_2,B,\pi_{\text{init}},\text{Leaves}_1)$.

**Proof.** Assume that $\lambda_1$ is winning in $\text{FCU}(A)$. By the *cycle lemma*, we know that the set of cycles that are compatible with $\lambda_1^*$ in arena $A$ is exactly equal to the set of simple cycles that are compatible with $\lambda_1$ in $\text{FCU}(A)$. As $\lambda_1$ is winning for $\text{Leaves}_1$, we know that all those cycles are such that $\text{visit}(\text{cycle}(\pi)) \cap T \neq \emptyset$. So, $\lambda_1^*$ is winning in the game arena.

Assume that there is no $\lambda_1$ which is winning in $\text{FCU}(A)$. By Zermelo's theorem, there is $\lambda_2$ which is winning in $\text{FCU}(A)$ for Player 2, i.e. $\lambda_2$ forces leaves $\text{FCU}(A)$ whose cycles are s.t. $\text{visit}(\text{cycle}(\pi)) \cap T = \emptyset$. By the *cycle lemma*, we know that the set of cycles that are compatible with $\lambda_2^*$ in arena $A$ are cycles s.t. $\text{visit}(\text{cycle}(\pi)) \cap T = \emptyset$. And so $\lambda_2^*$ is a winning strategy for Player 2 in the game arena for the $\text{Win}_2=\text{coBüchi}(T)$, the complement of the objective of Player 1. So we conclude that there is no winning strategies for Player 1 in $A$ for $\text{Büchi}(T)$. 
Parity games
Decision problem

Given a two-player game arena with a parity objective $A=(S_1, S_2, E, s_{init}, \text{parity}(p))$, decide if Player 1 has a winning strategy in this game, i.e. if there exists a strategy $\lambda_1$ for Player 1 such that $\text{Outcome}(\lambda_1) \subseteq \text{parity}(p)$.

Recall that:

$\text{parity}(p)=\{ \rho \in S^\omega | \min\{ p(s) | s \in \inf(\rho) \} \text{ is even} \}$
\textbf{parity}(p) = \{ \rho \in S^\omega \mid \min \{ p(s) \mid s \in \text{inf}(\rho) \} \text{ is even} \}

if leave is defined by cycle c
the color \textcolor{green}{green} if the minimal color of c is even
the color \textcolor{red}{red} if the minimal color of c is odd
A reduction to first cycle unfolding

• Let \( A = (S_1, S_2, E, s_{init}, \text{parity}(p)) \) be a two-player turn based arena with a parity winning objective for Player 1 defined by \( \text{parity}(p) \), and consider \( \text{FCU}(A) = (\Pi_1, \Pi_2, B, \pi_{init}) \) where \( \text{Leaves}_1 \) is defined as the set of \( \{ \pi \mid \min \{ p(s) \mid s \in \text{visit}(\text{cycle}(\pi)) \} \text{ is even} \} \).

• **Theorem.** Player 1 has a winning strategy in the game \( A = (S_1, S_2, E, s_{init}, \text{parity}(p)) \) if and only if Player 1 has a winning strategy in the tree \( \text{FCU}(A) = (\Pi_1, \Pi_2, B, \pi_{init}, \text{Leaves}_1) \).

  **Proof.**

  • Assume that \( \lambda_1 \) is winning in \( \text{FCU}(A) \). By the **cycle lemma**, we know that the set of cycles that are compatible with \( \lambda_1^* \) in arena \( A \) is exactly equal to the set of cycles that are compatible with \( \lambda_1 \) in \( \text{FCU}(A) \). As \( \lambda_1 \) is winning for \( \text{Leaves}_1 \), we know that all those cycles are such that \( \min \{ p(s) \mid s \in \text{visit}(\text{cycle}(\pi)) \} \) is even. Then the minimal parity that appears infinitely often in an outcome compatible with \( \lambda_1^* \) must be even and so winning for \( \text{parity}(p) \) in \( A \).

  • Assume that there is no \( \lambda_1 \) is winning in \( \text{FCU}(A) \). By Zermelo thereom, there is \( \lambda_2 \) is winning in \( \text{FCU}(A) \) for player 2, i.e. \( \lambda_2 \) forces cycles such that \( \min \{ p(s) \mid s \in \text{visit}(\text{cycle}(\pi)) \} \) is odd. By the **cycle lemma**, we know that the set of cycles that are compatible with \( \lambda_2^* \) in arena \( A \) is such that \( \min \{ p(s) \mid s \in \text{visit}(\text{cycle}(\pi)) \} \) is odd. An so \( \lambda_2^* \) is a winning strategy for Player 2 in the game arena, and we conclude that there is no winning strategies for Player 1 in \( A \) for \( \text{parity}(p) \).
Corollaries of the first cycle reduction

- **Theorem.** Parity games are finite memory determined, i.e. each player has a finite memory strategy to enforce victory from winning states.

- **Theorem.** Parity games can be solved in $O(2^n)$. 
Attractor-based algorithm for solving parity games due to Zielonka

By induction on the number of states in the game arena.
Attractor-based algorithm due to Zielonka

if \( \langle 1 \rangle \Box \Diamond p^{-1}(0) \) from every state in \( S \) then clearly Player 1 wins the parity game from everywhere in \( S \)
Attractor-based algorithm due to Zielonka

if not, i.e. $S \not\equiv \langle 1 \rangle \square \Diamond p^{-1}(0)$, then we consider $\text{Attr}_1(p^{-1}(0))$ and the sub-arena $A[S \setminus \text{Attr}_1(p^{-1}(0))]$ (this arena contains at least one less state)
Attractor-based algorithm due to Zielonka

**Observation**: to win from a state in $S \setminus \text{Attr}_1(p^{-1}(0))$, Player 2 should be able to do it without entering infinitely often through $\text{Attr}_1(p^{-1}(0))$. 

\[ A[S \setminus \text{Attr}_1(p^{-1}(0))] \]
Attractor-based algorithm due to Zielonka

**Recursion:** Compute the winning regions $W_1'$ and $W_2'$ in the sub-arena $A[S \setminus \text{Attr}_1(p^{-1}(0))]$. If $W_2' = \emptyset$ then $W_1 = S$, else the states of $W_2'$ are clearly winning for Player 2 in the original arena, and so Player 1 should never enter them nor $\text{Attr}_2(W_2')$. So Player 1 should try to win in the sub-arena $A[S \setminus \text{Attr}_2(W_2')]$. The set of winning state $W_1''$ and $W_2''$ in this sub-arena can compute recursively.
Attractor-based algorithm due to Zielonka

Proc. $\text{PGSolver}$

\[
\text{if } \text{Attr}_1(p^{-1}(0)) = S \\
\text{then return } W_1 = S \text{ and } W_2 = \emptyset \\
\text{else } (W_1', W_2') = \text{PGSolver}(A[S \setminus \text{Attr}_1(p^{-1}(0))]) \\
\text{if } W_2' = \emptyset \\
\text{then return } W_1 = S \text{ and } W_2 = \emptyset \\
\text{else } (W_1'', W_2'') = \text{PGSolver}(A[S \setminus \text{Attr}_2(W_2')]) \\
\text{return } W_1 = W_1'' \text{ and } W_2 = S \setminus W_1''
\]
Corollary of this attractor-based algorithm

**Theorem.** Parity games are memoryless determined.

**Proof.** We reason by induction on the number of states and makes the following observations.

Proc. **PGSolver**

\[
\begin{align*}
\text{if } \text{Attr}_1(p^{-1}(0)) &= S \\
\text{then return } W_1 &= S \text{ and } W_2 = \emptyset \\
\text{else } (W_1', W_2') &= \text{PGSolver}(A[S\text{\textbackslash Attr}_1(p^{-1}(0))]) \\
&\quad \text{if } W_2' = \emptyset \\
&\quad \text{then return } W_1 = S \text{ and } W_2 = \emptyset \\
&\quad \text{else } (W_1'', W_2'') &= \text{PGSolver}(A[S\text{\textbackslash Attr}_2(W_2')]) \\
&\quad \text{return } W_1 = W_1'' \text{ and } W_2 = S \backslash W_1''
\end{align*}
\]
Corollary of this attractor-based algorithm

**Theorem.** Parity games are memoryless determined.

**Proof.** We reason by induction on the number of states and makes the following observations.

Proc. **PGSolver**

\[
\begin{align*}
\text{if } \text{Attr}_1(p^{-1}(0)) &= S \\
\text{then return } W_1 &= S \text{ and } W_2 = \emptyset \\
\text{else } (W_1', W_2') &= \text{PGSolver}(A[S\setminus \text{Attr}_1(p^{-1}(0))]) \\
\text{if } W_2' &= \emptyset \\
\text{then return } W_1 &= S \text{ and } W_2 = \emptyset \\
\text{else } (W_1'', W_2'') &= \text{PGSolver}(A[S\setminus \text{Attr}_2(W_2')]) \\
\text{return } W_1 &= W_1'' \text{ and } W_2 = S\setminus W_1''
\end{align*}
\]

in this case the results holds as Player 1 has a **memoryless winning** strategy for the **Büchi** objective \( \langle 1 \rangle \square \diamond p^{-1}(0) \) from all states in \( S \)
Corollary of this attractor-based algorithm

**Theorem.** Parity games are memoryless determined.

**Proof.** We reason by induction on the number of states and makes the following observations.

Proc. \texttt{PGSolver}

\begin{align*}
\text{if } \text{Attr}_1(p^{-1}(0)) &= S \\
\text{then return } W_1 &= S \text{ and } W_2 = \emptyset \\
\text{else } (W_1',W_2') &= \text{PGSolver}(A\backslash \text{Attr}_1(p^{-1}(0))) \\
&\quad \text{if } W_2' = \emptyset \\
&\quad \text{then return } W_1 = S \text{ and } W_2 = \emptyset \\
&\quad \text{else } (W_1'',W_2'') &= \text{PGSolver}(A\backslash \text{Attr}_2(W_2')) \\
&\quad \quad \text{return } W_1 = W_1'' \text{ and } W_2 = S \backslash W_1''
\end{align*}

In this case the results holds as Player 1 has, by induction hypothesis, a memoryless winning strategy in the sub-arena \( A[S\backslash \text{Attr}_1(p^{-1}(0))] \) and play it when the game is in this sub-arena.

Furthermore, whenever the game enters the set \( \text{Attr}_1(p^{-1}(0)) \) then Player 1 plays a \textbf{memoryless} attractor strategy.
Corollary of this attractor-based algorithm

**Theorem.** Parity games are memoryless determined.

**Proof.** We reason by induction on the number of states and makes the following observations.

Proc. **PGSolver**

```plaintext
if Attr₁(p⁻¹(0))=S
then return W₁=S and W₂=∅
else (W₁’,W₂’)=PGSolver(A[S\Attr₁(p⁻¹(0))]
  if W₂’=∅
    then return W₁=S and W₂=∅
  else (W₁”’,W₂”’)=PGSolver(A[S\Attr₂(W₂’)])
    return W₁=W₁” and W₂=S\W₁”
```

In this last case, by **induction hypothesis** and properties of **attractor strategies**, Player 2 has a **memoryless strategy** in Attr₂(W₂’). Whenever the game enters this region, Player 2 plays this strategy. And by **induction hypothesis**, Player 1 and Player 2 have **memoryless winning strategies** in W₁” and W₂” respectively.
Exercices

From the arguments developed during the presentation of the attractor-based algorithm:

• write a complete and detailed proof of correctness

• write a complete and detailed proof of the existence of memoryless strategies for both players in parity games
Qualitative games


Benjamin Aminof, Sasha Rubin: First Cycle Games. SR 2014: 83-90