Beyond Value Iteration for Parity Games: Strategy Iteration with Universal Trees

Zhuan Khye Koh Georg Loho

THE LONDON SCHOOL OF ECONOMICS AND POLITICAL SCIENCE

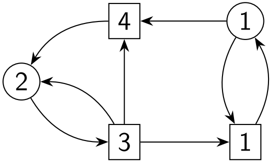
UNIVERSITY OF TWENTE.

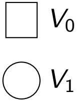
Overview

1. Parity game
2. Complexity of deciding the winner
3. Winning certificate from a universal tree
4. Value iteration
5. Strategy iteration

Parity game

Setting: A directed graph $G = (V, E)$ with partition $V = V_0 \sqcup V_1$, and a priority function $\pi : V \to \{1, 2, \ldots, d\}$.





- Nodes in V_0 and V_1 are owned by players Even and Odd respectively.

- A token is placed on a starting node $v \in V$. In every turn, the owner of the current node moves the token to an out-neighbour.
$\implies$ an infinite walk P (assume G is sinkless).

- If the highest priority occuring infinitely often in P is even, then Even wins. Otherwise, Odd wins.

Parity game

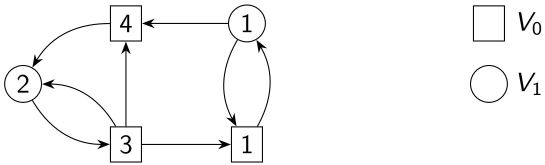
Positional Determinacy: Starting from any node $v \in V$, either Even or Odd can guarantee to win using a positional strategy.

• A (positional) strategy for Even is a function $\sigma : V_0 \to V$ such that $v\sigma(v) \in E$ for all $v \in V_0$. Its strategy subgraph is $G_\sigma = (V, E_\sigma)$ where

$$E_\sigma = \{v\sigma(v) : v \in V_0\} \cup \{vw \in E : v \in V_1\}.$$

A strategy τ for Odd and its strategy subgraph G_τ are defined similarly.

G



Problem: Given (G, π) and starting node $v \in V$, output the winner.

Parity game

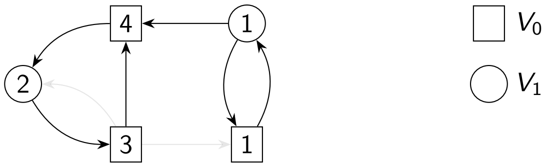
Positional Determinacy: Starting from any node $v \in V$, either Even or Odd can guarantee to win using a positional strategy.

• A (positional) strategy for Even is a function $\sigma : V_0 \to V$ such that $v\sigma(v) \in E$ for all $v \in V_0$. Its strategy subgraph is $G_\sigma = (V, E_\sigma)$ where

$$E_\sigma = \{v\sigma(v) : v \in V_0\} \cup \{vw \in E : v \in V_1\}.$$

A strategy τ for Odd and its strategy subgraph G_τ are defined similarly.

G_σ



Problem: Given (G, π) and starting node $v \in V$, output the winner.

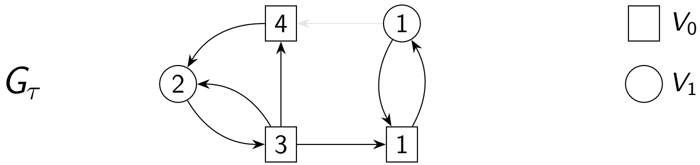
Parity game

Positional Determinacy: Starting from any node $v \in V$, either Even or Odd can guarantee to win using a positional strategy.

• A (positional) strategy for Even is a function $\sigma : V_0 \to V$ such that $v\sigma(v) \in E$ for all $v \in V_0$. Its strategy subgraph is $G_\sigma = (V, E_\sigma)$ where

$$E_\sigma = \{v\sigma(v) : v \in V_0\} \cup \{vw \in E : v \in V_1\}.$$

A strategy τ for Odd and its strategy subgraph G_τ are defined similarly.



Problem: Given (G, π) and starting node $v \in V$, output the winner.

Complexity of deciding the winner

- Belongs to **NP** $\cap$ **coNP**.

- Important in logic and verification, e.g., polynomial-time equivalent to the model-checking problem for modal μ-calculus.

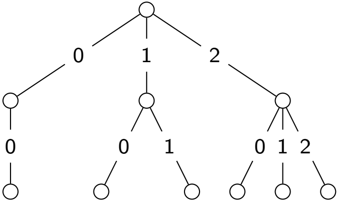
- Pre-2017 algorithms were exponential or subexponential time.

- Quasi-polynomial time [Calude, Jain, Khoussainov, Li, Stephan '17].

- Many other quasi-polynomial algorithms soon follow: [Fearnley, Jain, Keijzer, Schewe, Stephan, Wojtczak '17] [Gimbert, Ibsen–Jensen '17] [Jurdziński, Lazić '17] [Lehtinen '18] [Parys '19] [Lehtinen, Schewe, Wojtczak '19] [Daviaud, Jurdziński, Thejaswini '20] [Benerecetti, Dell'Erba, Mogavero, Schewe, Wojtczak '21].

- Most of them have been unified via the notion of a universal tree [Czerwiński, Daviaud, Fijalkow, Jurdziński, Lazić, Parys '19].

Ordered tree



$M = \{0, 1, 2\}$

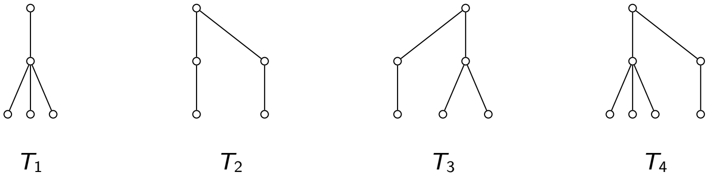
Def: Given a totally ordered set $(M, \leq)$, an ordered tree T is a prefix-closed set of tuples whose elements are drawn from M.

- Elements in M induce branching directions at every vertex $v \in V(T)$.
- The tuple corresponding to a vertex v is given by the root-v path.
- $\leq$ extends lexicographically to $V(T)$.
- $L(T) :=$ leaf set of T. Assume every leaf has the same depth.

Universal tree

• Given ordered trees T and T', T embeds into T' ($T \sqsubseteq T'$) if there exists an injective function $f : V(T) \to V(T')$ such that

1. $uv \in E(T) \implies f(u)f(v) \in E(T')$ (homomorphism)
2. $u \leq v \implies f(u) \leq f(v)$ (order-preserving)

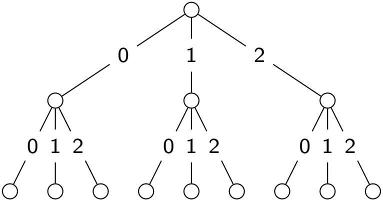


Def: An (ℓ, h)-universal tree is an ordered tree T' such that $T \sqsubseteq T'$ for all ordered trees T of height at most h and with at most ℓ leaves.

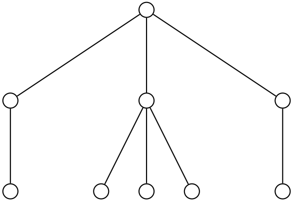
Thm: Every universal tree has at least quasi-polynomially many leaves. [Czerwiński et al. '19].

Examples of universal trees

- Perfect (ℓ, h)-universal tree:
  - Every leaf is an h-tuple of integers from $\{0, 1, \ldots, \ell - 1\}$.
  - Contains ℓ^h leaves.
- Succinct (ℓ, h)-universal tree [Jurdziński, Lazić '17]:
  - Every leaf is an h-tuple of binary strings with at most $\lfloor \log \ell \rfloor$ bits in total.
  - Contains $\ell^{\log h + O(1)}$ leaves.



A perfect (3,2)-universal tree.



A succinct (3,2)-universal tree.

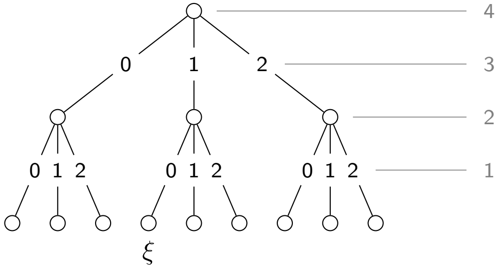
Winning certificate from a universal tree

- Let $\overline{L}(T) := L(T) \cup \{\top\}$, where $\top > v$ for all $v \in V(T)$.
- Given an instance (G, π) with $n = |V|$, a node labeling is a function $\mu : V \to \overline{L}(T)$ for some $(n, d/2)$-universal tree T.
- For a leaf $\xi \in L(T)$, we index its tuple by $(\xi_{d-1}, \xi_{d-3}, \ldots, \xi_1)$.

Intuition: records how many times an odd priority is encountered.

- For a priority p, the p-truncation of ξ is obtained by deleting the components with index less than p.

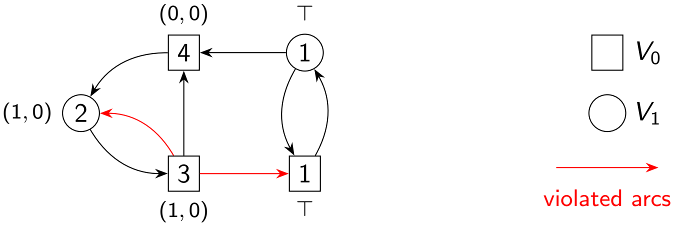
$$
\begin{aligned}
\xi &= (1, 0) \\
\xi|_1 &= (1, 0) \\
\xi|_2 &= (1) \\
\xi|_3 &= (1) \\
\xi|_4 &= ()
\end{aligned}
$$



Winning certificate from a universal tree

Def: A node labeling μ is feasible in G if Even has a strategy σ such that every arc vw in G_σ satisfies

- If $\pi(v)$ is even, then $\mu(v)|_{\pi(v)} \geq \mu(w)|_{\pi(v)}$.
- If $\pi(v)$ is odd, then $\mu(v)|_{\pi(v)} > \mu(w)|_{\pi(v)}$ or $\mu(v) = \mu(w) = \top$.

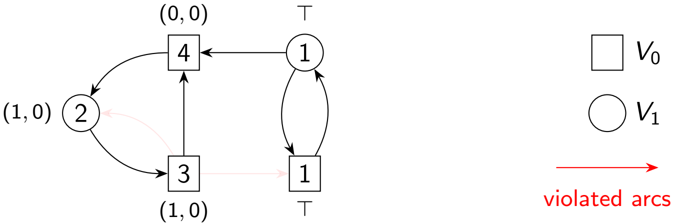


Thm: Let μ^* be a node labeling which is feasible in G and has minimal $\top$-support. Even wins from $v \in V \iff \mu^*(v) \neq \top$ [Jurdziński '00].

Winning certificate from a universal tree

Def: A node labeling μ is feasible in G if Even has a strategy σ such that every arc vw in G_σ satisfies

- If $\pi(v)$ is even, then $\mu(v)|_{\pi(v)} \geq \mu(w)|_{\pi(v)}$.
- If $\pi(v)$ is odd, then $\mu(v)|_{\pi(v)} > \mu(w)|_{\pi(v)}$ or $\mu(v) = \mu(w) = \top$.



Thm: Let μ^* be a node labeling which is feasible in G and has minimal $\top$-support. Even wins from $v \in V \iff \mu^*(v) \neq \top$ [Jurdziński '00].

Value iteration

Def: Given $\mu : V \to \overline{L}(T)$ and $vw \in E$, let lift(μ, vw) be the smallest element $\xi \in \overline{L}(T)$ such that $\xi \geq \mu(v)$ and vw is non-violated after setting $\mu(v)$ to ξ.

Value-Iteration(G, π, T)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. **while** μ is not feasible:

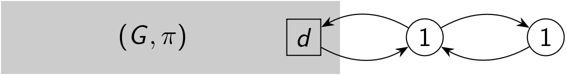
   $\mu(v) \leftarrow \min_{vw \in \delta^+(v)} \text{lift}(\mu, vw)$ for some node $v \in V_0$ whose outgoing arcs $\delta^+(v)$ are all violated **or**

   $\mu(v) \leftarrow \text{lift}(\mu, vw)$ for some violated arc $vw \in E$ where $v \in V_1$
3. **return** μ

- Returns the least fixed point of G in $\Theta(n|L(T)|)$ iterations.
- Also called the progress measure algorithm [Jurdziński '00].

Behaviour of value iteration

- Not robust against its worst-case runtime:



- If d is even, then the two additional nodes see every element in $\overline{L}(T)$.
  $\implies \Omega(|L(T)|)$ time.

Idea: Iterate over strategies instead of arcs:

- Fix a strategy τ for Odd.
- Update μ to the least fixed point of G_τ.
- Pivot to a "better" strategy τ' for Odd, and repeat.

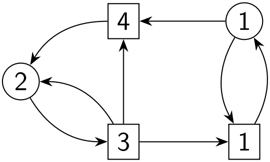
Impossibility result: The label set $\overline{L}(T)$ is not fit for strategy iteration [Ohlmann '22].

Strategy iteration

Strategy-Iteration(G, π, T, τ)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. $\mu \leftarrow$ least fixed point of G_τ which is at least μ
3. **while** μ is not feasible in G:

   Odd pivots to a strategy τ' by selecting violated arc(s)
   $\tau \leftarrow \tau'$
   $\mu \leftarrow$ least fixed point of G_τ which is at least μ
4. **return** μ

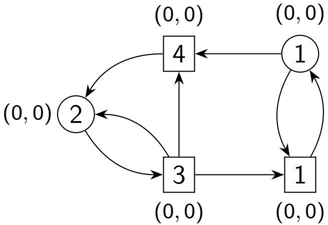


Strategy iteration

Strategy-Iteration(G, π, T, τ)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. $\mu \leftarrow$ least fixed point of G_τ which is at least μ
3. **while** μ is not feasible in G:

   Odd pivots to a strategy τ' by selecting violated arc(s)
   $\tau \leftarrow \tau'$
   $\mu \leftarrow$ least fixed point of G_τ which is at least μ
4. **return** μ

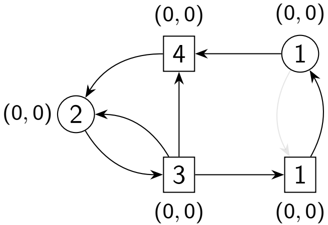


Strategy iteration

Strategy-Iteration(G, π, T, τ)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. $\mu \leftarrow$ least fixed point of G_τ which is at least μ
3. **while** μ is not feasible in G:

   Odd pivots to a strategy τ' by selecting violated arc(s)
   $\tau \leftarrow \tau'$
   $\mu \leftarrow$ least fixed point of G_τ which is at least μ
4. **return** μ

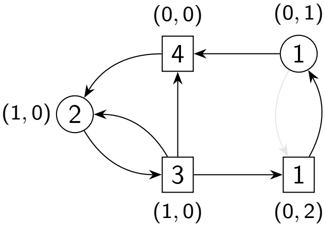


Strategy iteration

Strategy-Iteration(G, π, T, τ)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. $\mu \leftarrow$ least fixed point of G_τ which is at least μ
3. **while** μ is not feasible in G:

   Odd pivots to a strategy τ' by selecting violated arc(s)
   $\tau \leftarrow \tau'$
   $\mu \leftarrow$ least fixed point of G_τ which is at least μ
4. **return** μ

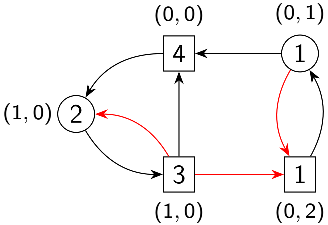


Strategy iteration

Strategy-Iteration(G, π, T, τ)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. $\mu \leftarrow$ least fixed point of G_τ which is at least μ
3. **while** μ is not feasible in G:

   Odd pivots to a strategy τ' by selecting violated arc(s)
   $\tau \leftarrow \tau'$
   $\mu \leftarrow$ least fixed point of G_τ which is at least μ
4. **return** μ



Strategy iteration

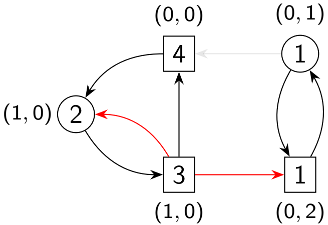
Strategy-Iteration(G, π, T, τ)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. $\mu \leftarrow$ least fixed point of G_τ which is at least μ
3. **while** μ is not feasible in G:

   Odd pivots to a strategy τ' by selecting violated arc(s)

   $\tau \leftarrow \tau'$

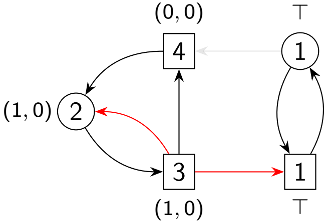
   $\mu \leftarrow$ least fixed point of G_τ which is at least μ
4. **return** μ



Strategy iteration

Strategy-Iteration(G, π, T, τ)

1. $\mu(v) \leftarrow \min L(T)$ for all $v \in V$
2. $\mu \leftarrow$ least fixed point of G_τ which is at least μ
3. **while** μ is not feasible in G:
   Odd pivots to a strategy τ' by selecting violated arc(s)
   $\tau \leftarrow \tau'$
   $\mu \leftarrow$ least fixed point of G_τ which is at least μ
4. **return** μ



Computing the least fixed point of $G_{\mathcal{T}}$

- Value iteration on the 1-player game $G_{\mathcal{T}}$ still takes $\Theta(|L(T)|)$ time.

Idea: Approach the least fixed point from above.

- Inspired by label-correcting (e.g. Bellman–Ford) and label-setting (e.g. Dijkstra) techniques from shortest path.

- We give an efficient method to compute the least fixed point of $G_{\mathcal{T}}$ for any universal tree T.

Running times for specific T

- $O(d(m + n \log n))$ for a perfect $(n, d/2)$-universal tree.
- $O(mn^2 \log n \log d)$ for a succinct $(n, d/2)$-universal tree.
- $O(mn^2 \log^3 n \log d)$ for a Strahler $(n, d/2)$-universal tree (introduced by [Daviaud, Jurdziński, Thejaswini '20]).

Conclusion

• We now have a strategy iteration framework for parity games that works with universal trees.

• Total running time is upper bounded by value iteration's running time.

Open question: Is there a subquasi-polynomial pivot rule using some universal tree?