Mini-course on algorithmic aspects of stochastic games and related models

清華大學 交叉信息研究院
Tsinghua University Institute for Interdisciplinary Information Sciences

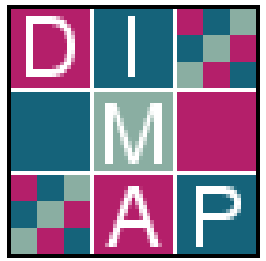

Marcin Jurdziński (University of Warwick)
Peter Bro Miltersen (Aarhus University)
Uri Zwick (武熠) (Tel Aviv University)

Oct. 31 – Nov. 2, 2011

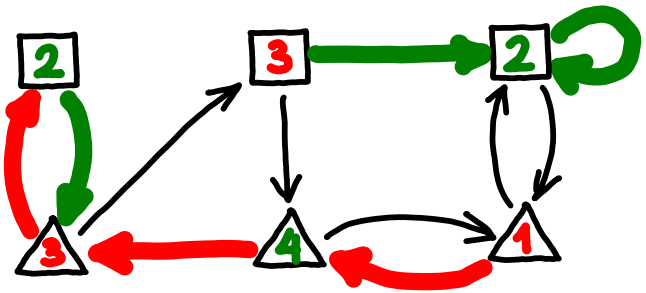
Day 2
Tuesday, November 1

Marcin Jurdziński

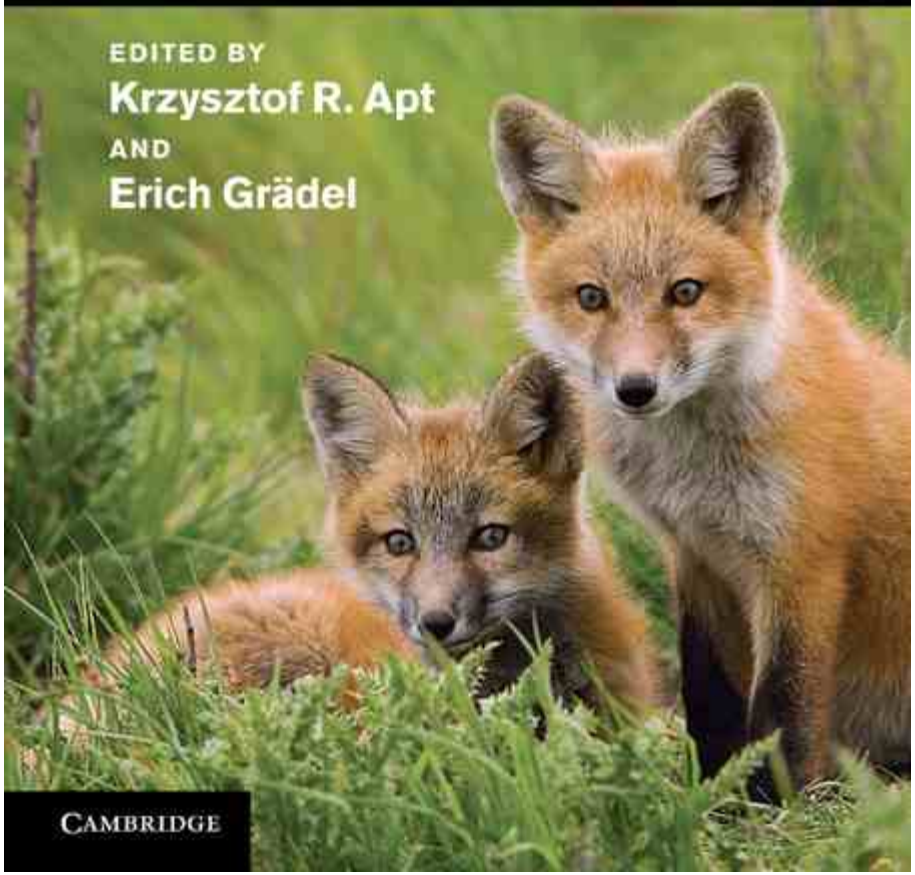
(University of Warwick)



Parity Games



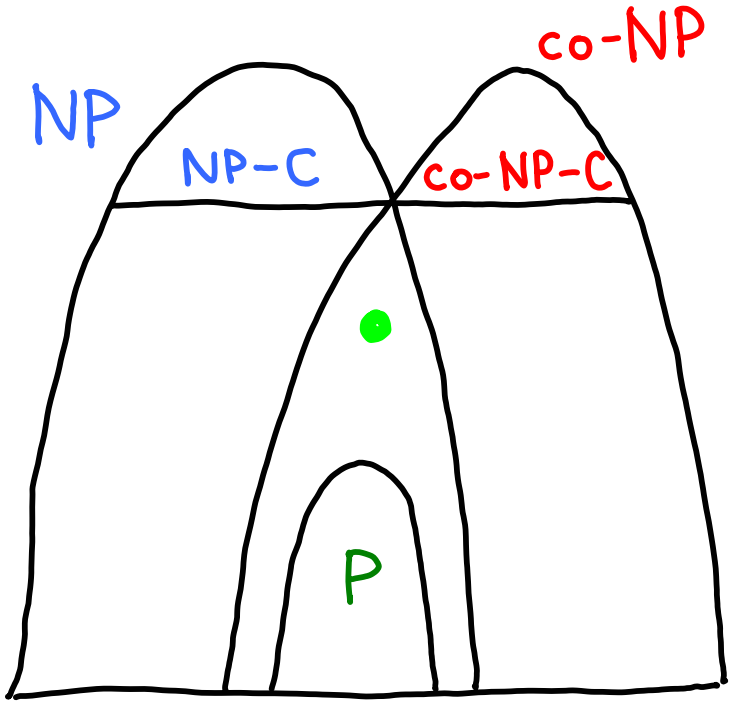
Lectures in Game Theory for Computer Scientists



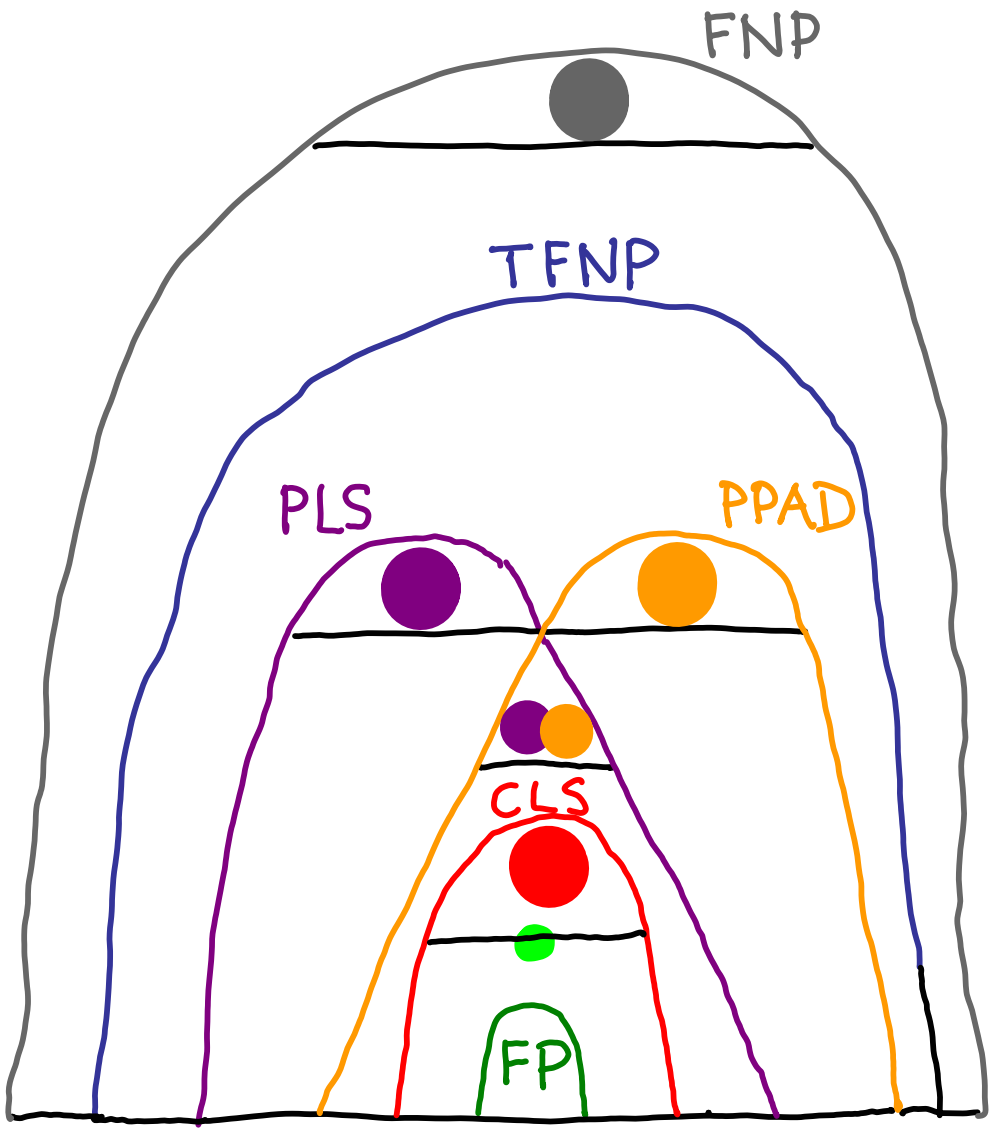
EDITED BY
Krzysztof R. Apt
AND
Erich Grädel

CAMBRIDGE

How difficult is it to find a winning strategy?



COMPUTING OPTIMAL STRATEGIES



PLAN

1. Motivating example
2. Games on graphs
3. Reachability / safety games
4. Büchi / co-Büchi games
5. Parity games
6. Better algorithms for parity games
7. Search complexity of computing optimal strategies

Boolean formula evaluation

State s:

p	0
q	1
r	1

Property φ: $(r \vee p) \wedge (\neg p \wedge (q \vee \neg r))$

$$s \stackrel{?}{\models} \varphi$$

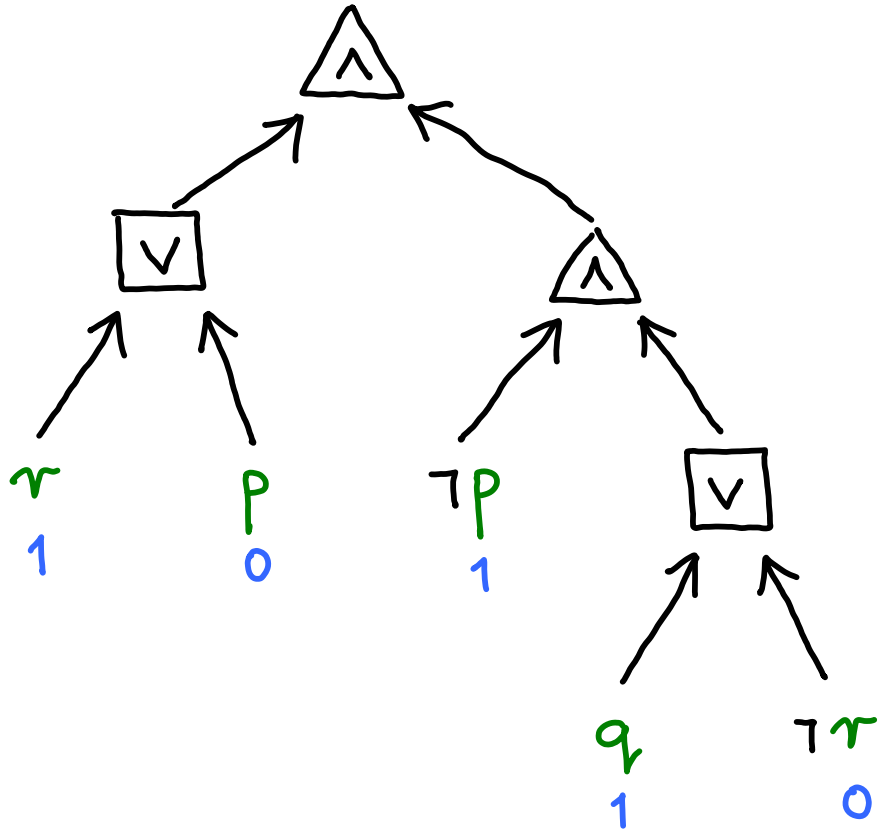
Boolean formula evaluation

State s:

p	0
q	1
r	1

Property φ: $(r \vee p) \wedge (\neg p \wedge (q \vee \neg r))$

$C_{s,\varphi}$:



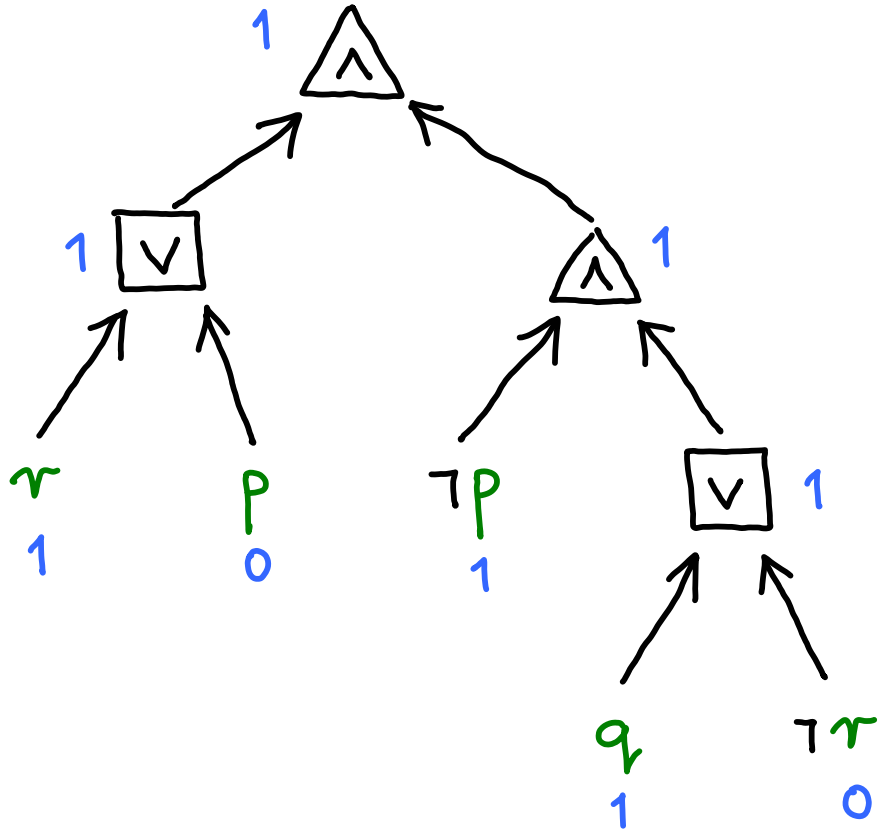
Boolean formula evaluation

State s:

p	0
q	1
r	1

Property φ: $(r \vee p) \wedge (\neg p \wedge (q \vee \neg r))$

$\mathcal{C}_{s,\varphi}$:



Fact

$s \models \varphi$

iff

the value of circuit $\mathcal{C}_{s,\varphi}$ is 1

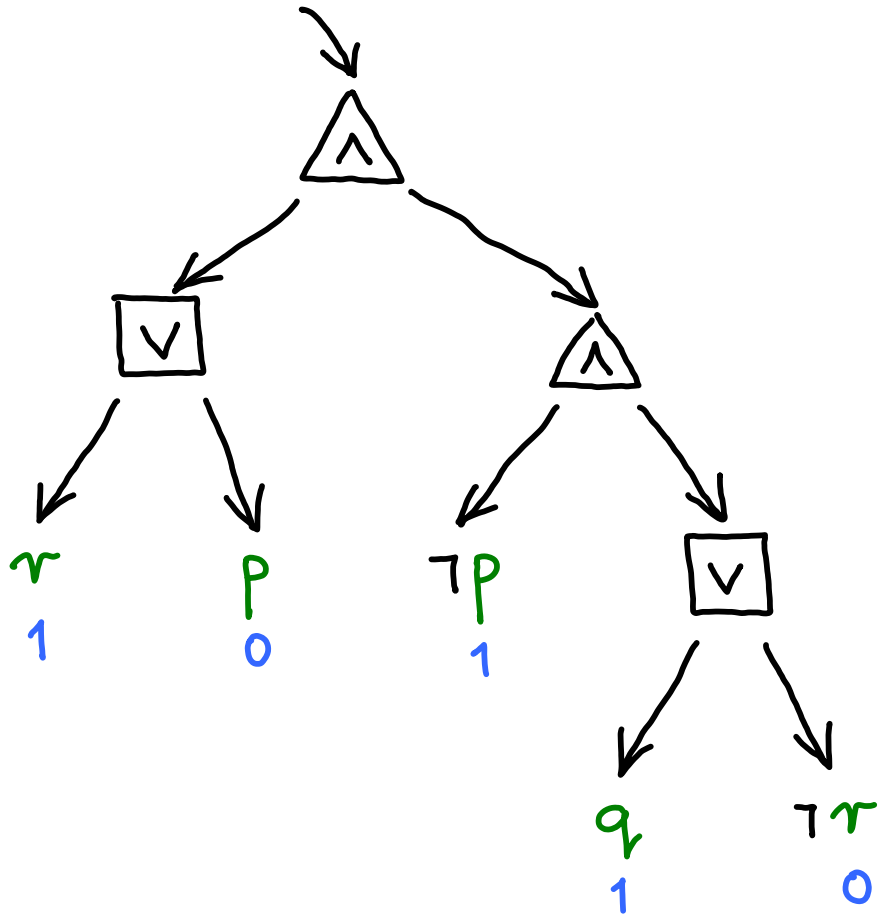
Boolean formula evaluation: Game

State s:

p	0
q	1
r	1

Property φ: $(r \vee p) \wedge (\neg p \wedge (q \vee \neg r))$

$\mathcal{G}_{s,\varphi}$:



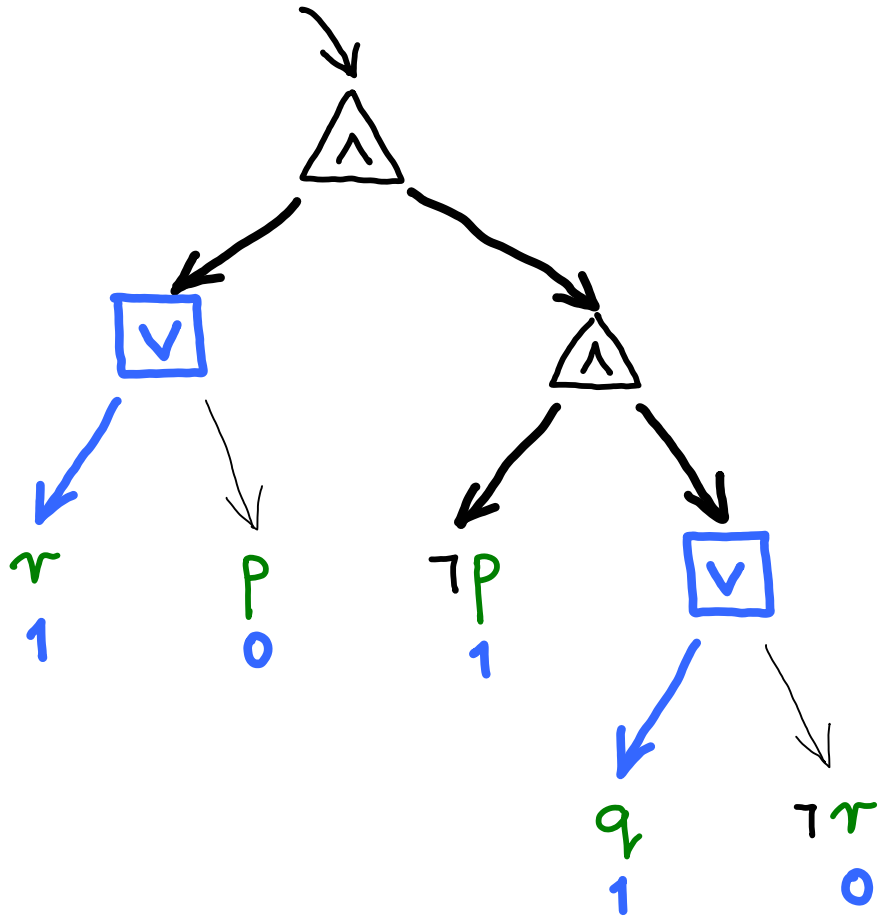
Boolean formula evaluation: Game

State s:

p	0
q	1
r	1

Property φ: $(r \vee p) \wedge (\neg p \wedge (q \vee \neg r))$

$\mathcal{G}_{s,\varphi}$:



Fact

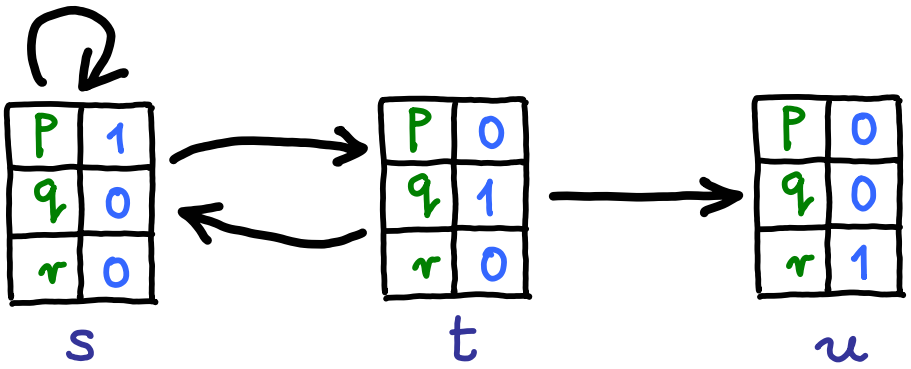
$s \models \varphi$

iff

∨ has a **strategy** to reach 1 in $\mathcal{G}_{s,\varphi}$

Temporal formula evaluation

(Model of a) program :



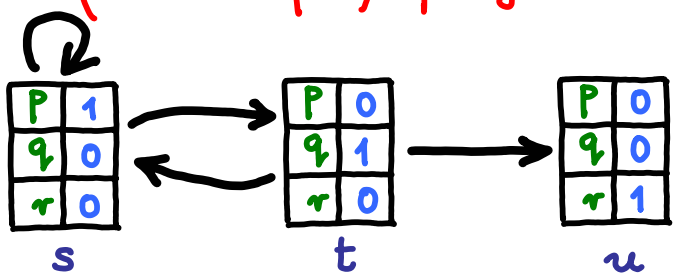
CTL property φ :

$$E((p \vee q) \, U \, r)$$

$$s \stackrel{?}{\models} \varphi$$

Temporal formula evaluation: Game

(Model of a) program:



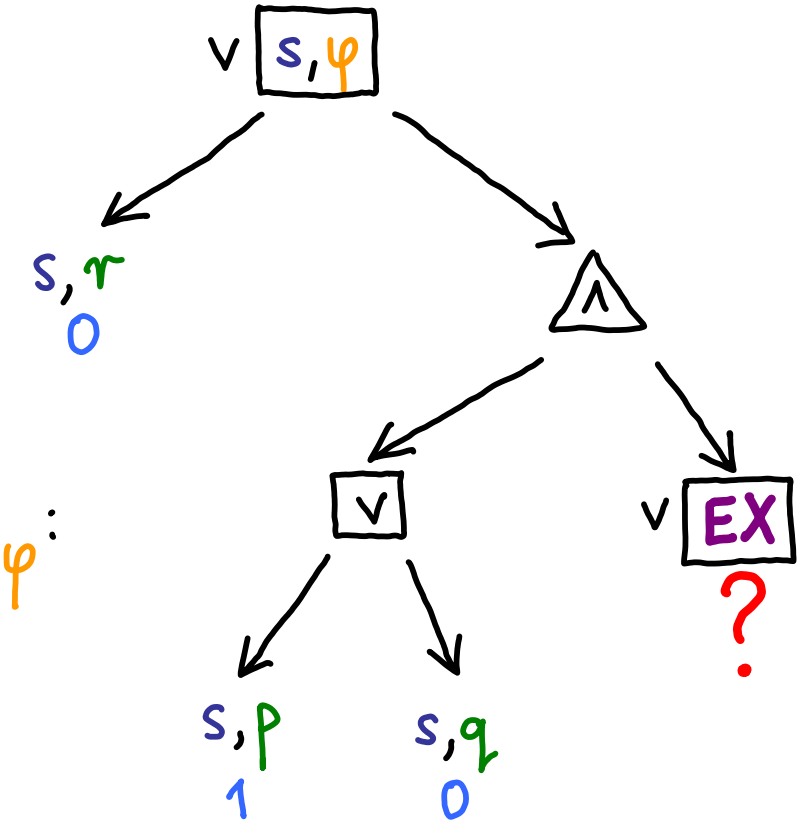
CTL property φ:

φ: $E((p \vee q) U r)$

$\equiv$

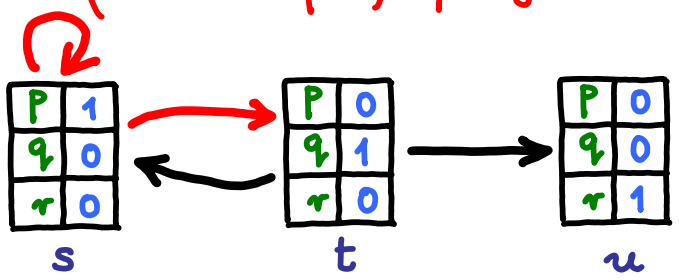
$r \vee ((p \vee q) \wedge EX\varphi)$

$G_{s,\varphi}$:



Temporal formula evaluation: Game

(Model of a) program:



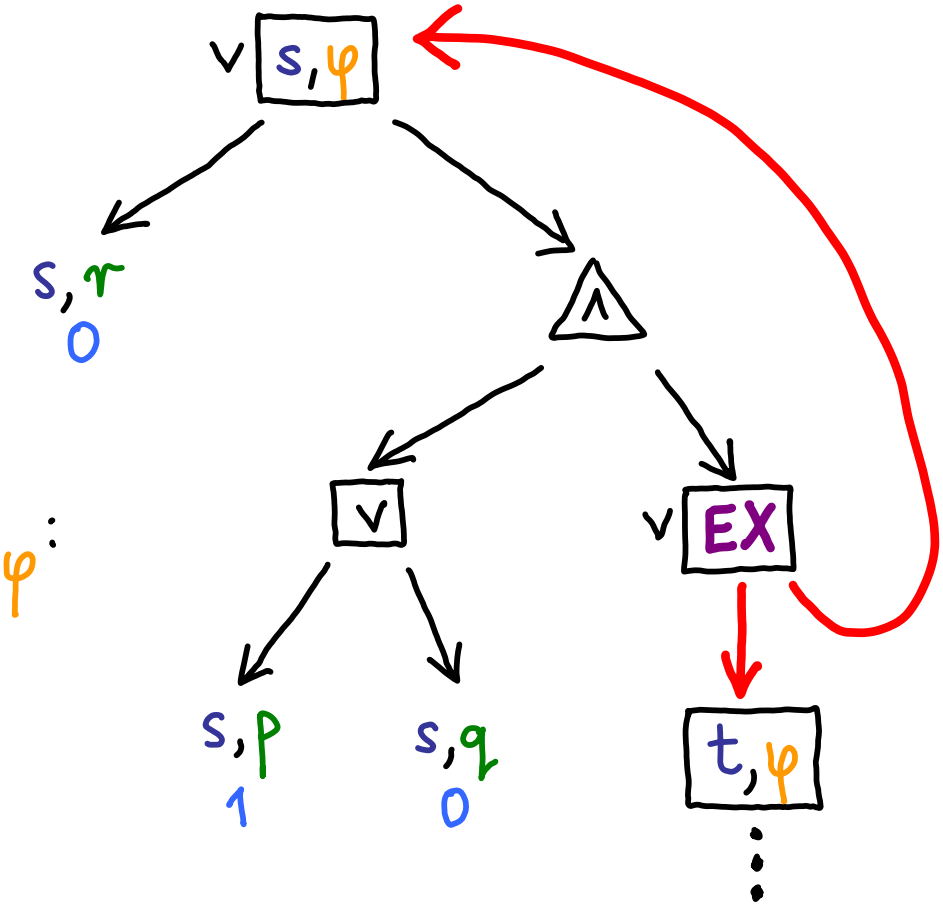
CTL property φ:

φ: $E((p \vee q) U r)$

$\Vert\!\Vert$

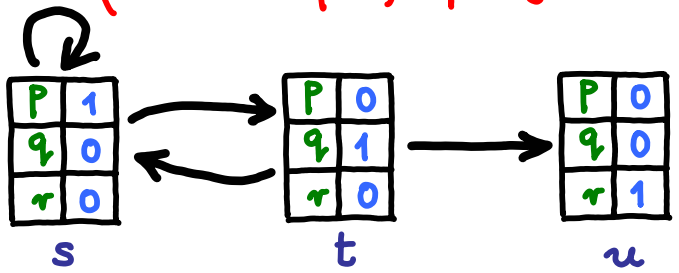
$r \vee ((p \vee q) \wedge EX\,\varphi)$

$\mathcal{G}_{s,\varphi}$:



Temporal formula evaluation: Game

(Model of a) program :

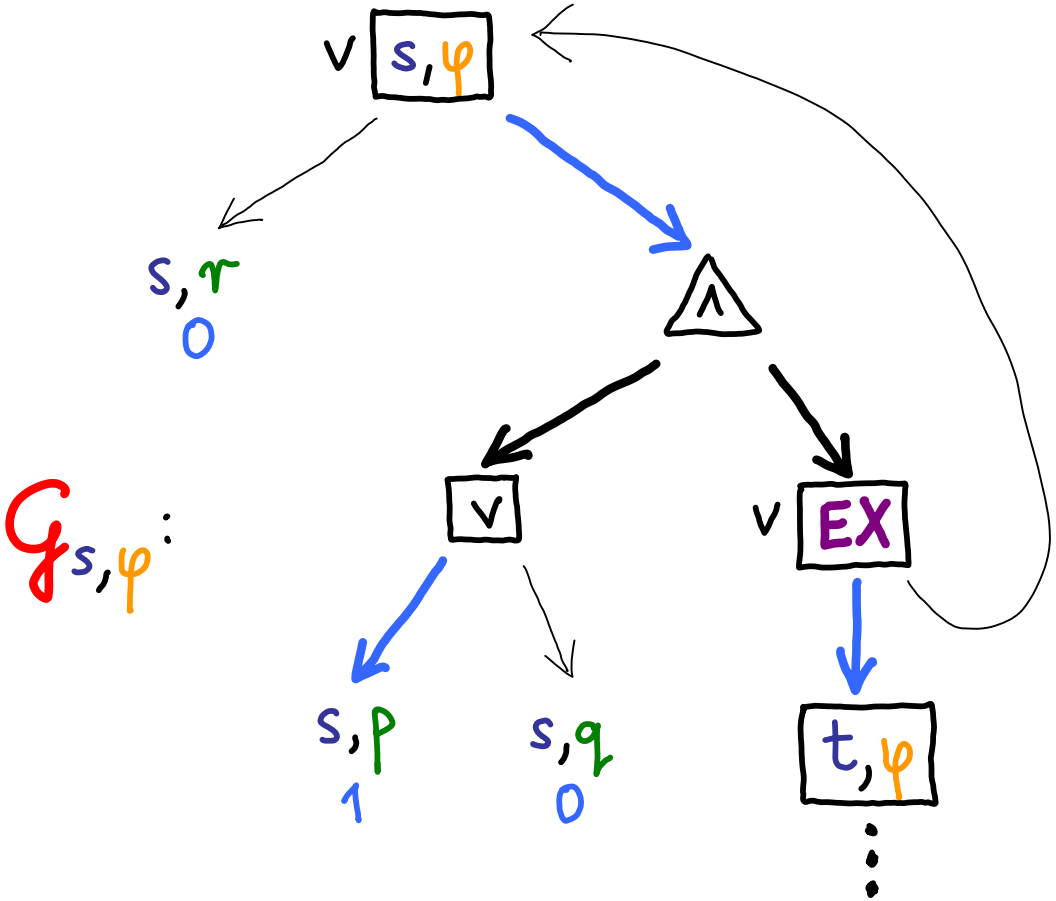


CTL property φ:

φ : $E((p \vee q) U r)$

$\equiv$

$r \vee ((p \vee q) \wedge EX\varphi)$



Thm

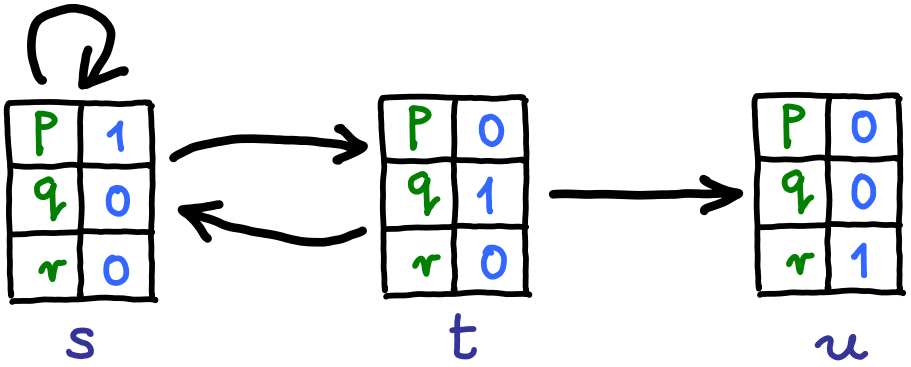
$s \models \varphi$

iff

∨ has a strategy to reach 1 in $G_{s,\varphi}$

Fixpoint formula evaluation

(Model of a) program :



μ-calculus property:

φ : $\boxed{\mu Z . \psi(Z)}$

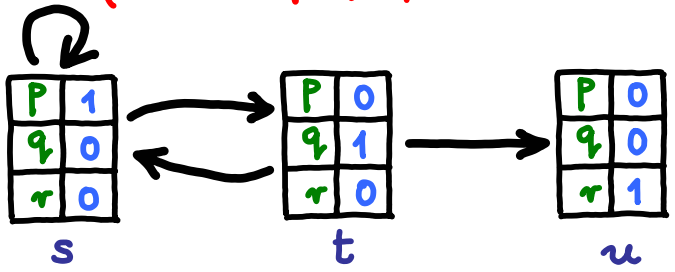
$\equiv$

$\psi(\varphi)$

$$s \stackrel{?}{\models} \varphi$$

Fixpoint formula evaluation

(Model of a) program:

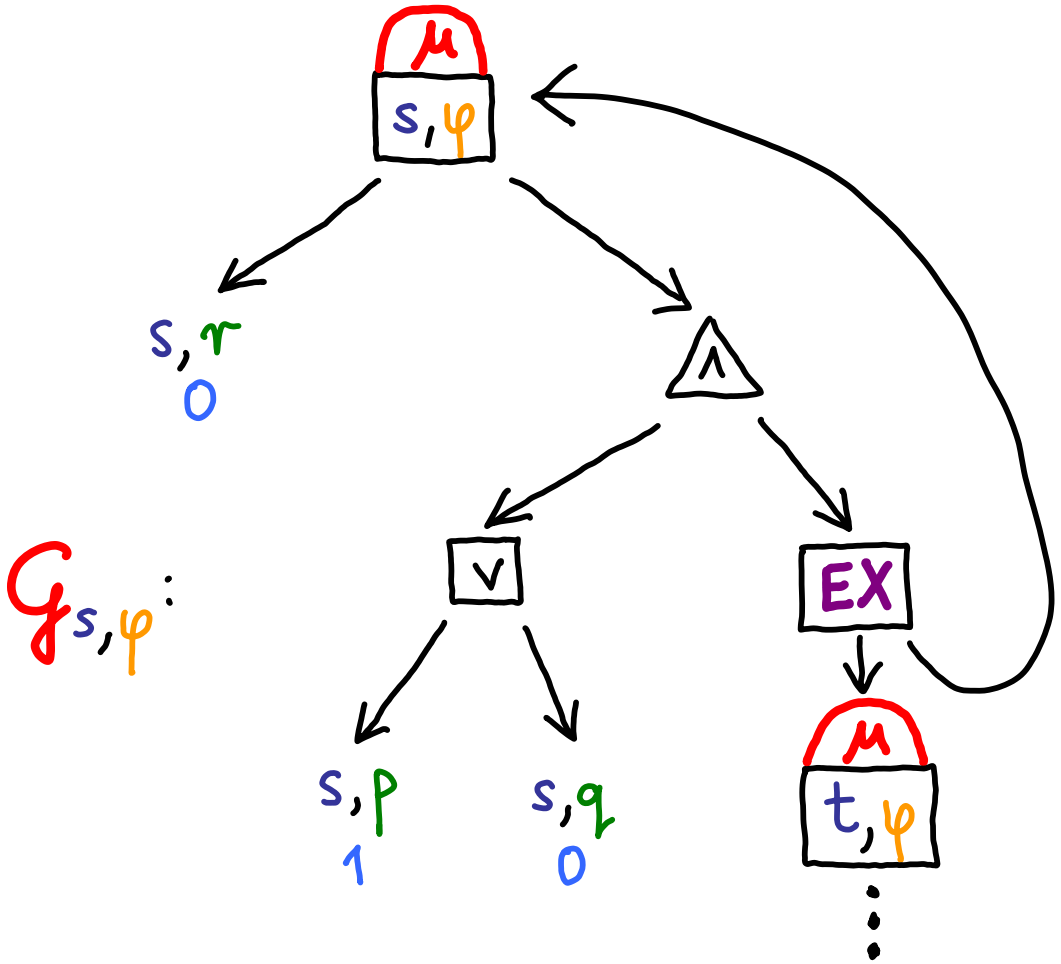


μ-calculus property φ:

$\varphi: \boxed{\mu Z.\, \psi(Z)}$

$\equiv$

$\psi(\varphi)$



Thm

$s \models \varphi$

iff

∨ has a **strategy** in $G_{s,\varphi}$

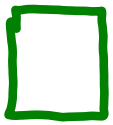
- to reach 1, or
- outermost infinitely occuring fixpoint subformula is **not** a μ (i.e., least fixpoint)

PLAN

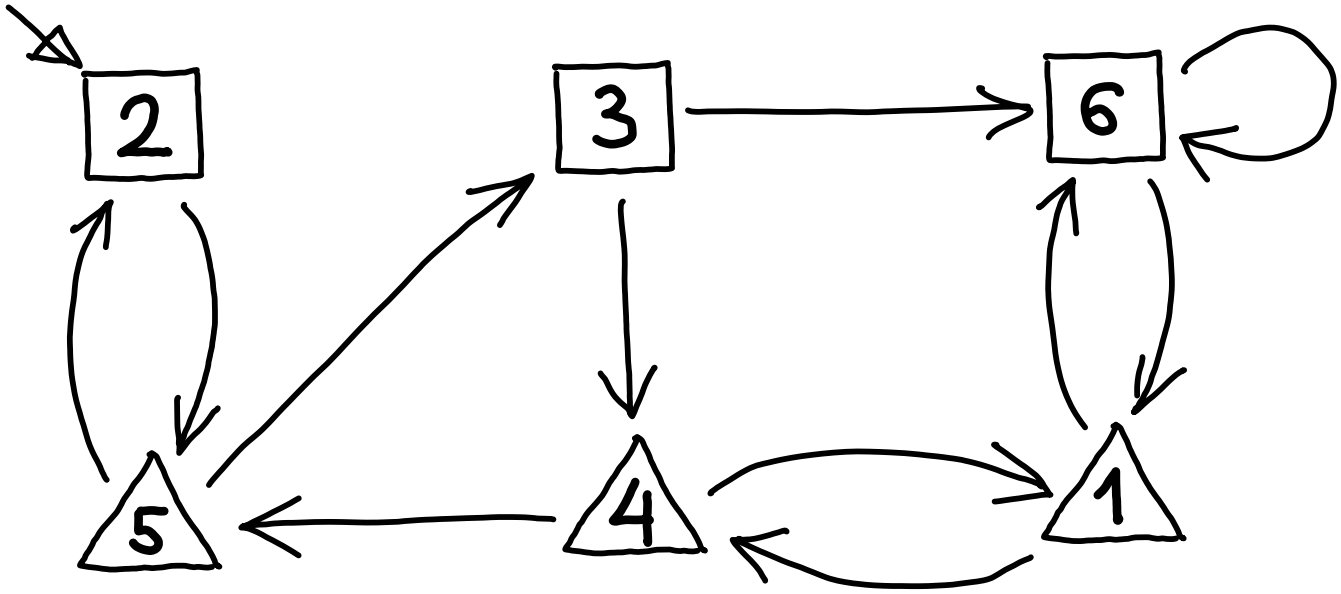
1. Motivating example
2. Games on graphs
3. Reachability / safety games
4. Büchi / co-Büchi games
5. Parity games
6. Better algorithms for parity games
7. Search complexity of computing optimal strategies

GAMES ON GRAPHS: GAME GRAPH

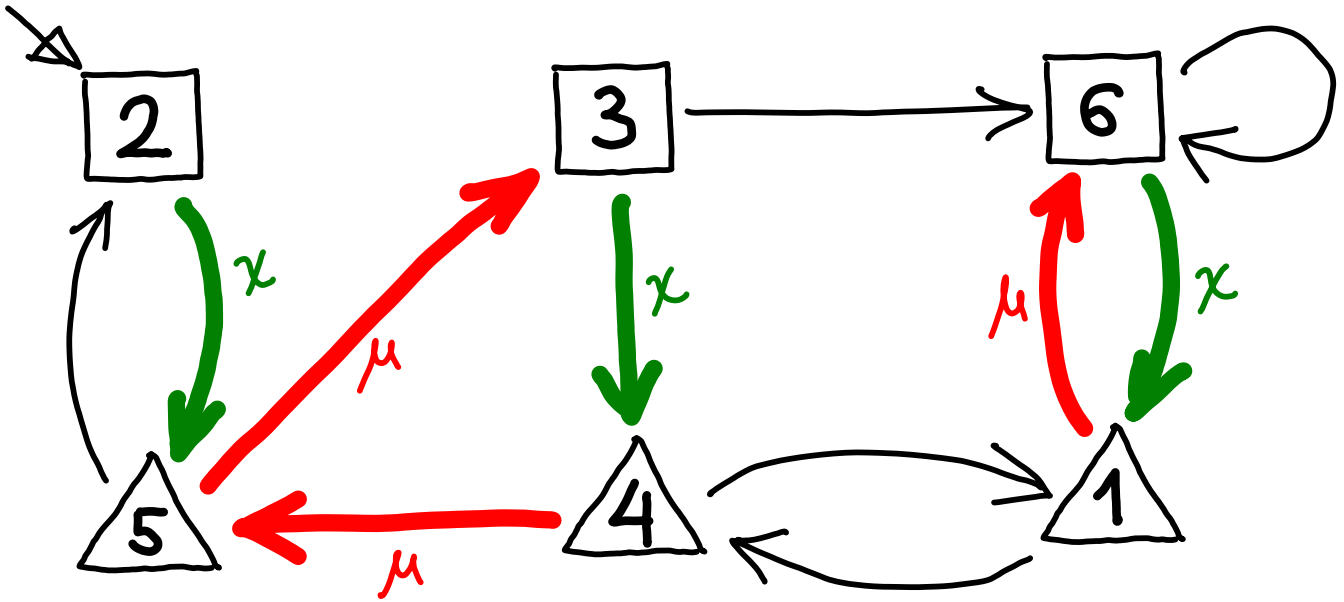
Players:



$n = |V|$ vertices, $m = |E|$ edges



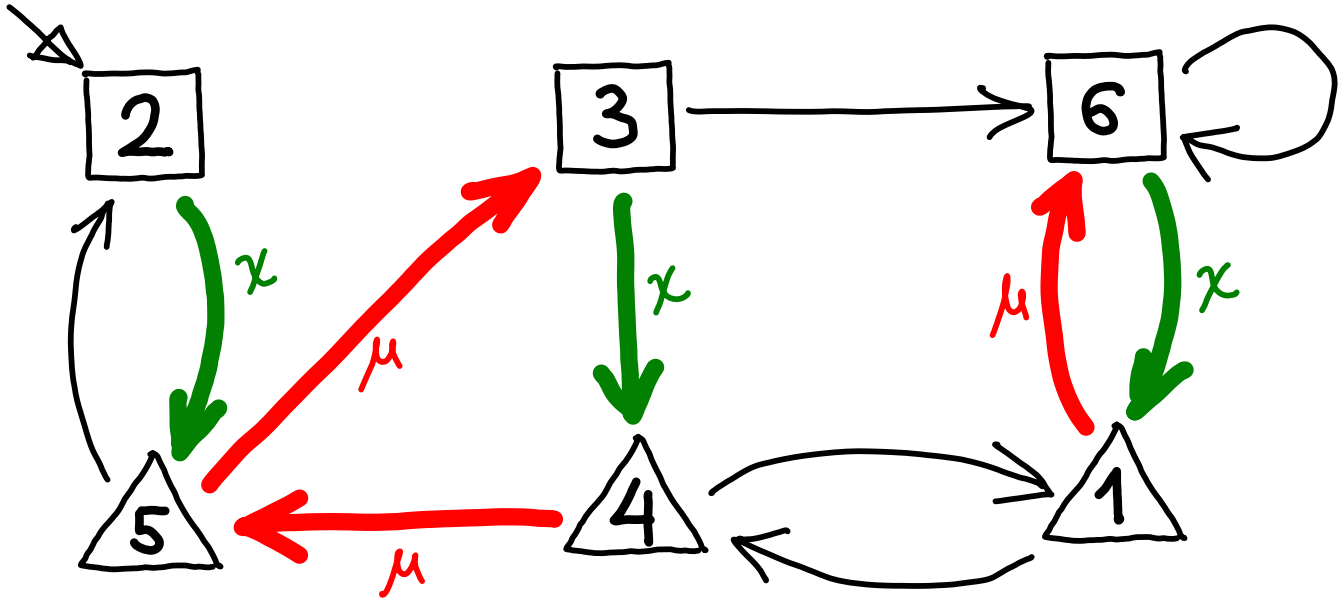
Games on graphs: positional strategies



Positional **strategy** for □: $\chi : V_{\square} \to V$

Positional **strategy** for △: $\mu : V_{\triangle} \to V$

Games on graphs: plays



$$\mathrm{Play}(2,\chi,\mu) = (2,5,3,4,5,(3,4,5)^{\omega}) \in V^{\omega}$$

GAMES ON GRAPHS: OBJECTIVES

Objectives: $W_\Box \subseteq V^\omega$

$W_\Delta = V^\omega \setminus W_\Box$

GAMES ON GRAPHS: OBJECTIVES

Objectives: $W_\square \subseteq V^\omega$

$W_\Delta = V^\omega \setminus W_\square$

Examples of objectives:

- Reachability: $W_\square = \{(v_0, v_1, v_2, \ldots) : v_i \in T \text{ for some } i \in \mathbb{N}\}$
- Safety: $W_\square = \{(v_0, v_1, v_2, \ldots) : v_i \in S \text{ for all } i \in \mathbb{N}\}$

where $T, S \subseteq V$

Games on graphs: winning strategies

$\chi \in \Sigma_{\square}$ is a **winning strategy** for $\square$ from $v \in V$,

if $\boxed{Play(v, \chi, \mu) \in W_{\square}, \quad \text{for all } \mu \in \Sigma_{\Delta}}$

$\mu \in \Sigma_{\Delta}$ is a **winning strategy** for Δ from $v \in V$,

if $\boxed{Play(v, \chi, \mu) \in W_{\Delta}, \quad \text{for all } \chi \in \Sigma_{\square}}$

GAMES ON GRAPHS: ALGORITHMIC PROBLEM

Given: game graph $(V = V_\Box \uplus V_\triangle, E)$

starting vertex $v \in V$

objective $W_\Box \subseteq V^\omega$

answer: if $\Box$ has a winning strategy from v

Games on graphs: Determinacy

A game $(V = V_\square \uplus V_\triangle, E; W_\square)$ is **determined**

if for every starting vertex $v \in V$

- $\square$ has a winning strategy from v, **or**
- $\triangle$ has a winning strategy from v

MARTIN'S DETERMINACY THEOREM

THM [MARTIN 1975]

Every game $\left(V = V_{\square} \uplus V_{\triangle}, E ; W_{\square}\right)$,

such that $W_{\square} \subseteq V^{\omega}$ is a Borel set,

is determined

GAMES ON GRAPHS: ALGORITHMIC PROBLEM

Given: game graph $(V = V_\square \uplus V_\triangle, E)$

objective $W_\square \subseteq V^\omega$

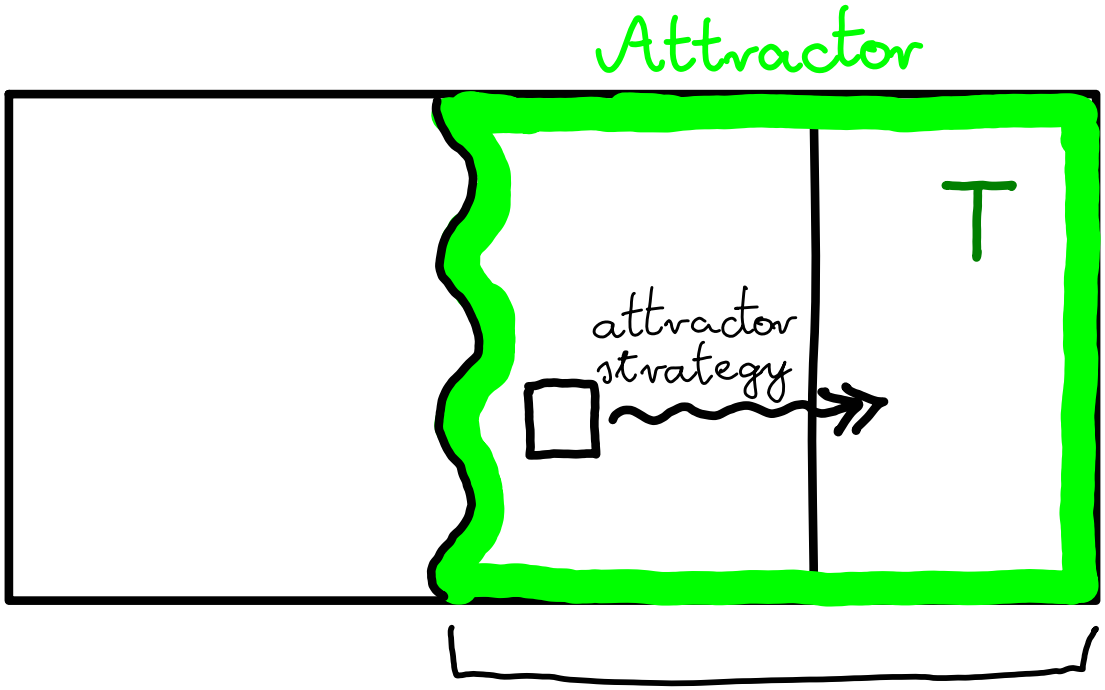
compute: $V = W_\square \uplus W_\triangle$ such that

- $\square$ has a winning strategy from $W_\square$, and
- $\triangle$ has a winning strategy from $W_\triangle$

PLAN

1. Motivating example
2. Games on graphs
3. **Reachability / safety games**
4. Büchi / co-Büchi games
5. Parity games
6. Better algorithms for parity games
7. Search complexity of computing optimal strategies

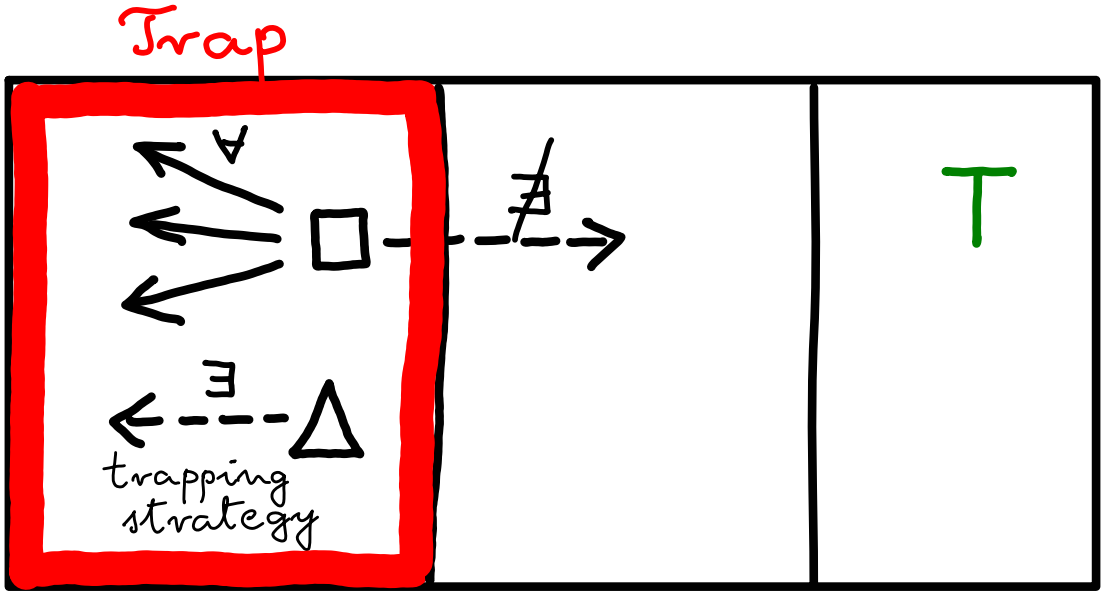
Reachability games



$W_\square = attr_\square(T)$

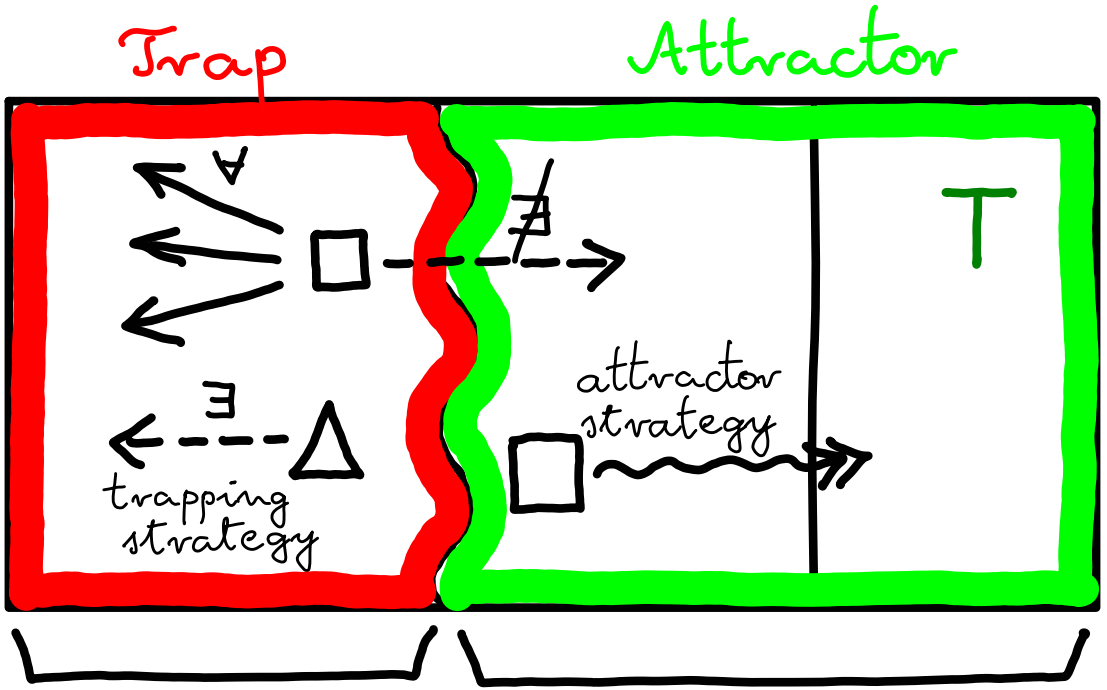
- **Reachability**: $W_\square = \{(v_0, v_1, v_2, \ldots) : v_i \in T \text{ for some } i \in \mathbb{N}\}$

Safety games



- **Reachability**: $W_{\square} = \{(v_0, v_1, v_2, \ldots) : v_i \in T \text{ for some } i \in \mathbb{N}\}$
- **Safety**: $W_{\Delta} = \{(v_0, v_1, v_2, \ldots) : v_i \in T \text{ for no } i \in \mathbb{N}\}$

Reachability/Safety games



$W_\Delta = V \setminus attr_\square(T)$ $W_\square = attr_\square(T)$

Fact

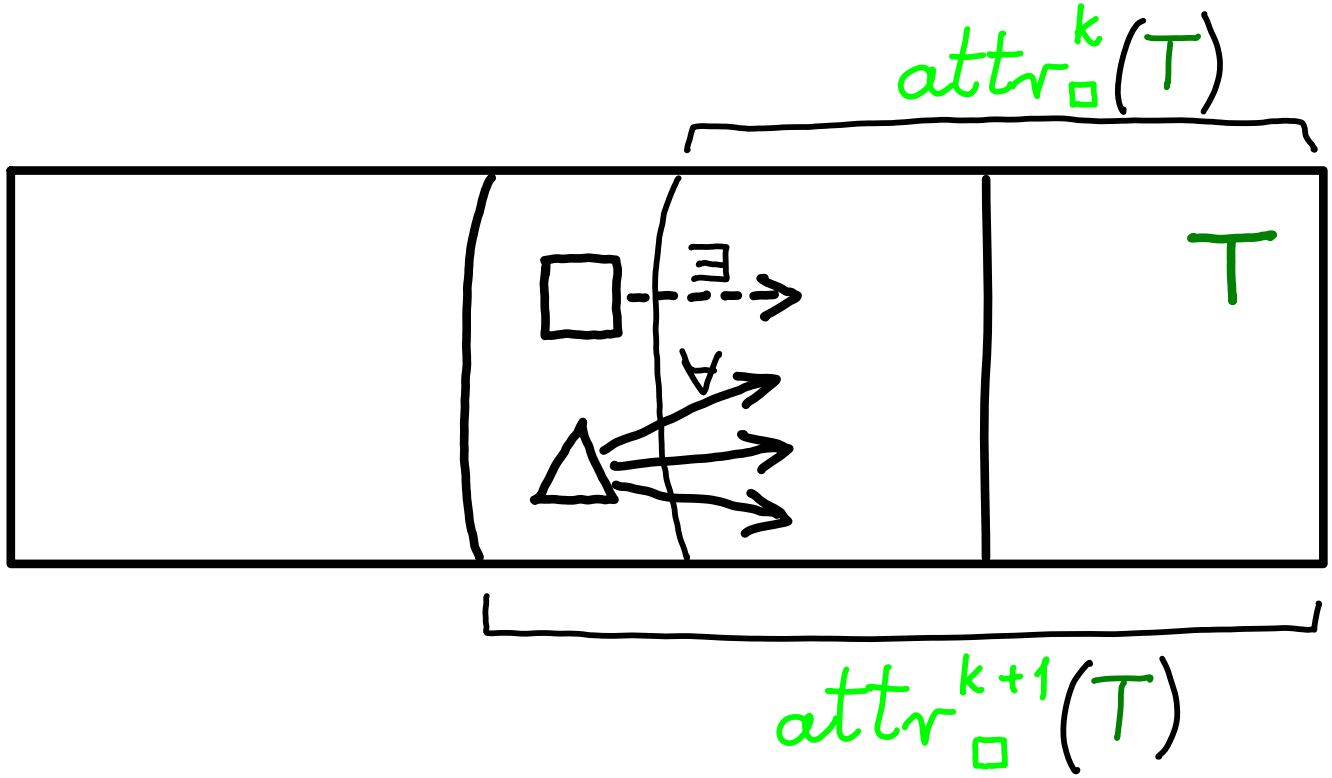
Reachability/safety games are **positionally determined**

Exercise 1

Explain in detail
why the complement of an attractor
is a trap for the other player,
and how to compute
positional attractor and trapping strategies.

Conclude that reachability/safety games
are positionally determined.

Solving Reachability/Safety games: Computing Attractors



Fact $attr_{\square}(T) = \bigcup_{k \geq 0} attr_{\square}^{k}(T)$

can be computed in $O(m)$ time

Exercise 2

Discuss implementation details that make it possible to compute attractors in time $O(m)$.

PLAN

1. Motivating example
2. Games on graphs
3. Reachability / safety games
4. **Büchi / co-Büchi games**
5. Parity games
6. Better algorithms for parity games
7. Search complexity of computing optimal strategies

GAMES ON GRAPHS: (co-)BÜCHI OBJECTIVES

- **Büchi** (repeated reachability)

  An infinite play is winning if

  vertices in $T \subseteq V$ occur **infinitely often**

- **co-Büchi** (eventual safety)

  An infinite play is winning if

  vertices in $T \subseteq V$ occur **finitely often**

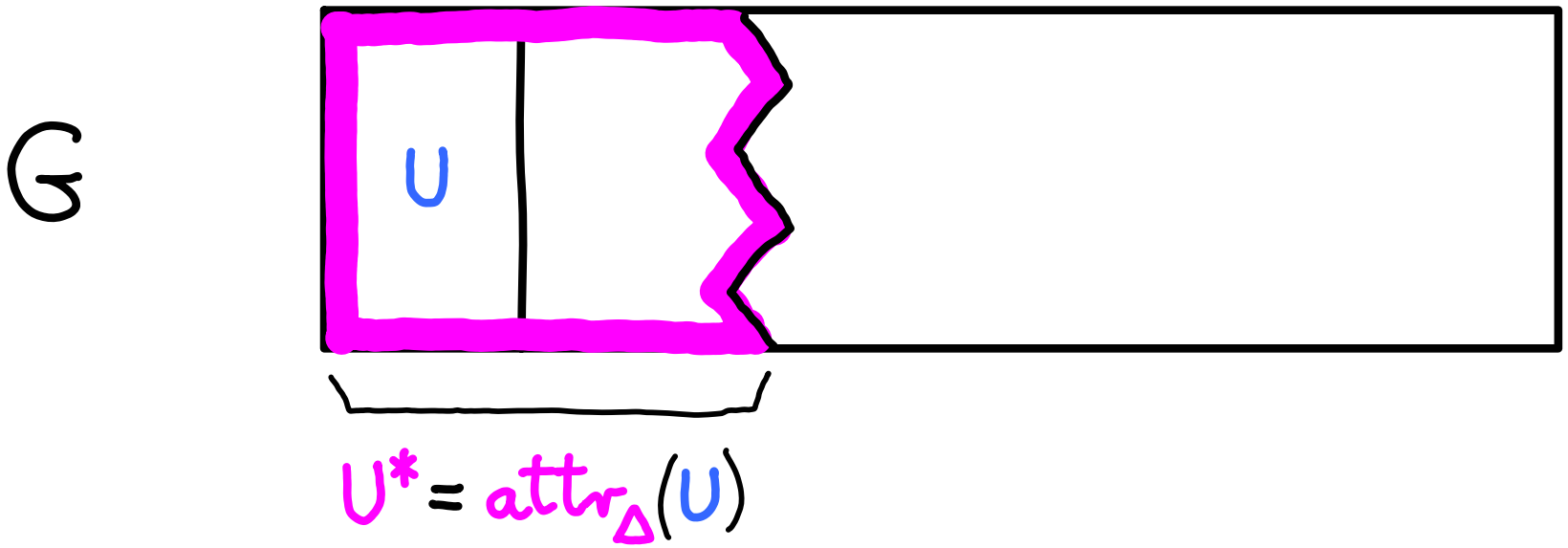
Oracle Lemma

Lemma If $U \subseteq win_{\Delta}(G)$

then $win_{\Delta}(G) = U^{*} \cup win_{\Delta}(G'')$

$win_{\square}(G) = win_{\square}(G'')$

Oracle Lemma

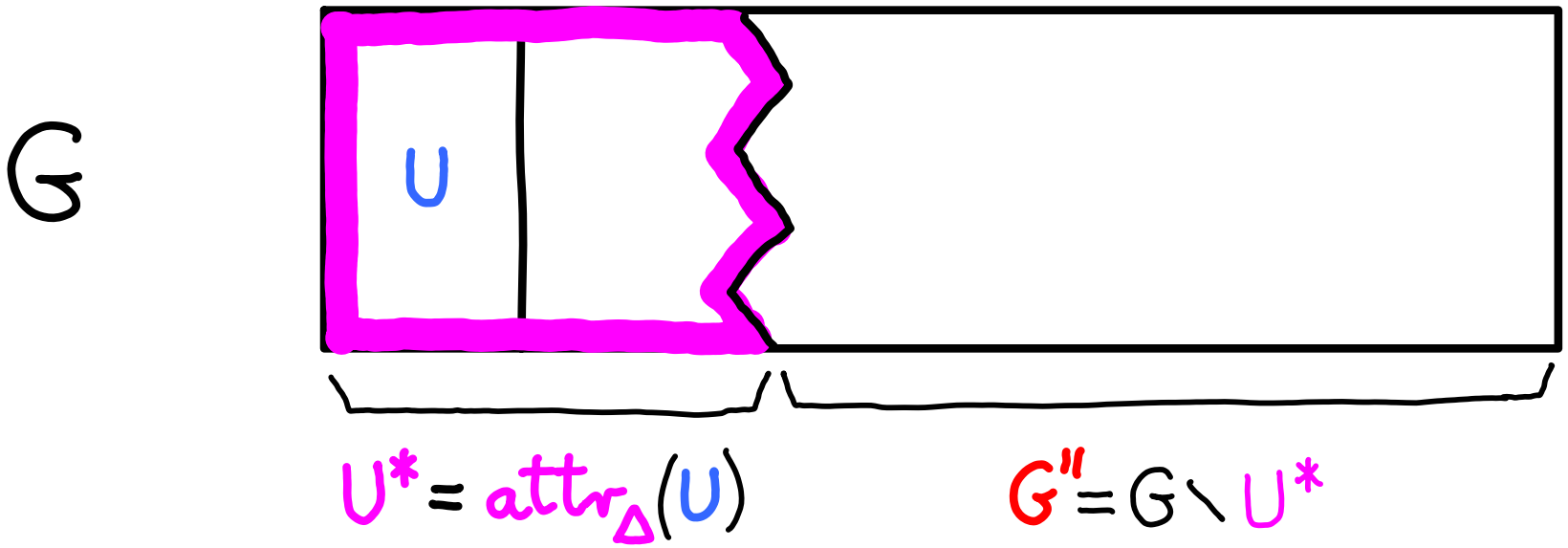


Lemma If $U \subseteq win_\Delta(G)$

then $win_\Delta(G) = U^* \cup win_\Delta(G'')$

$win_\square(G) = win_\square(G'')$

Oracle Lemma

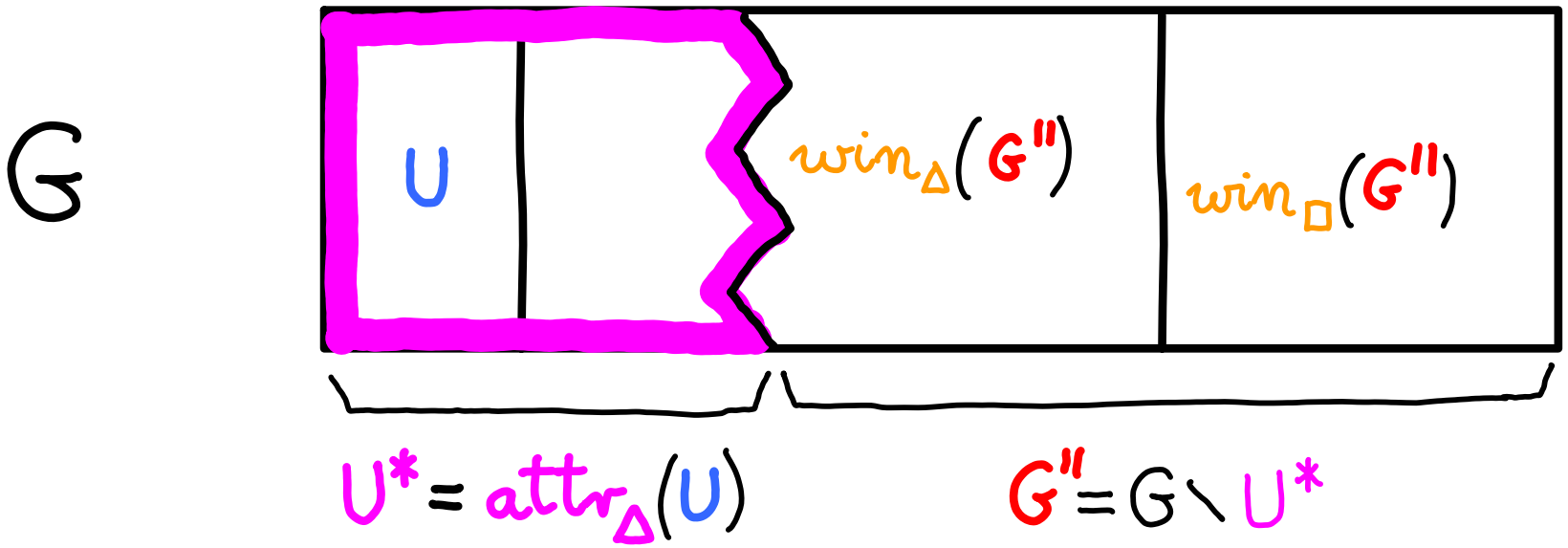


Lemma If $U \subseteq win_\Delta(G)$

then $win_\Delta(G) = U^* \cup win_\Delta(G'')$

$win_\square(G) = win_\square(G'')$

Oracle Lemma

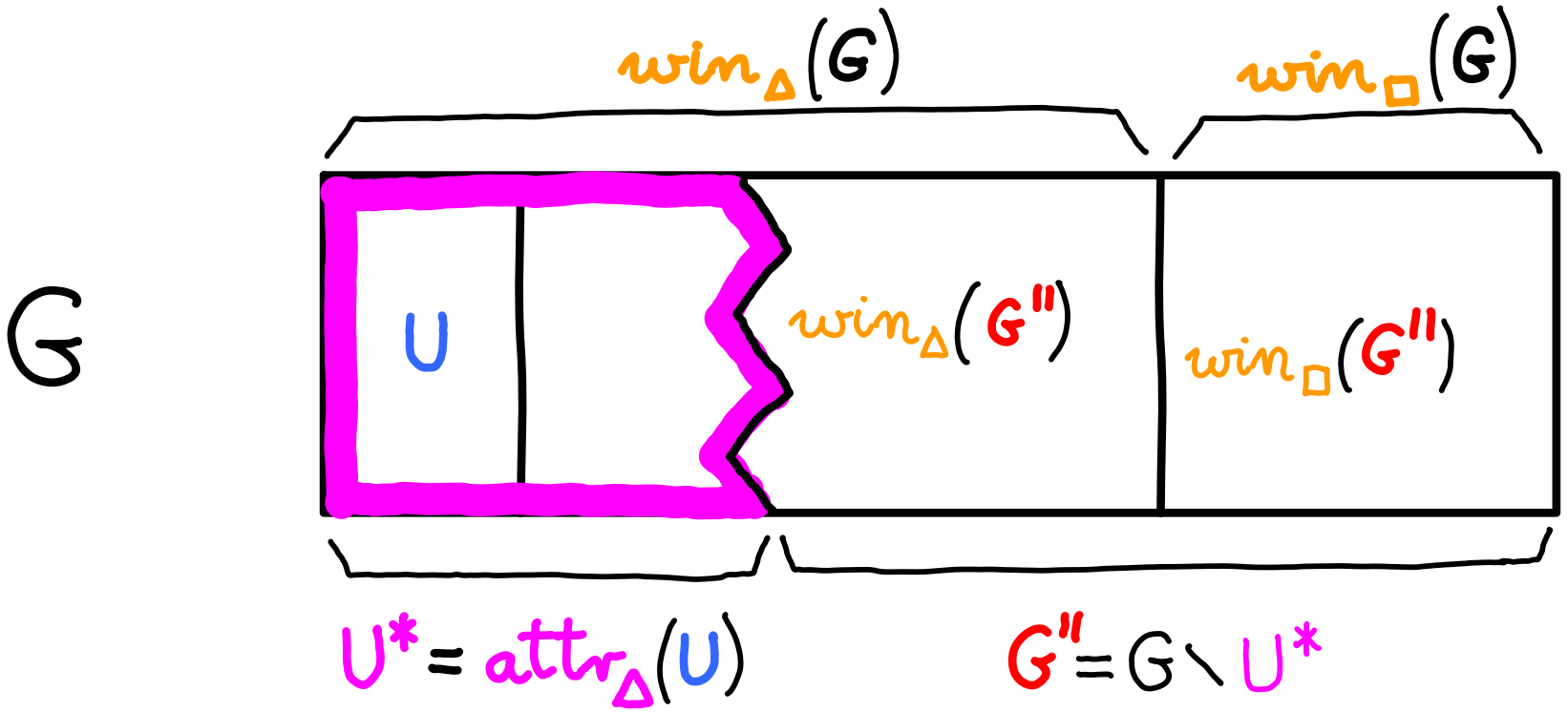


Lemma If $U \subseteq win_\Delta(G)$

then $win_\Delta(G) = U^* \cup win_\Delta(G'')$

$win_\square(G) = win_\square(G'')$

Oracle Lemma



Lemma If $U \subseteq win_\Delta(G)$

then $win_\Delta(G) = U^* \cup win_\Delta(G'')$

$win_\square(G) = win_\square(G'')$

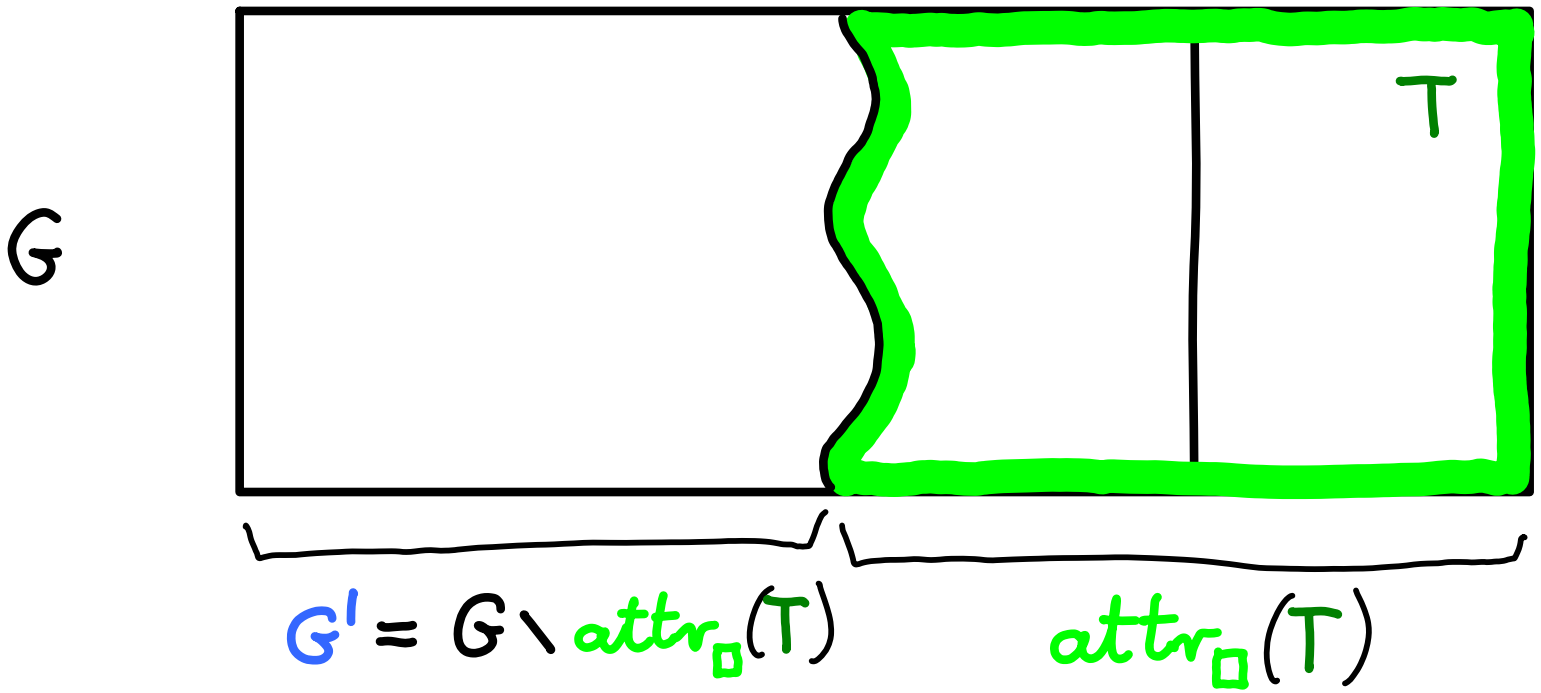
Exercise 3

Prove the oracle lemma for Büchi and co-Büchi games.

Are the winning strategies constructed positional?

If not, what assumptions do you need to make them so?

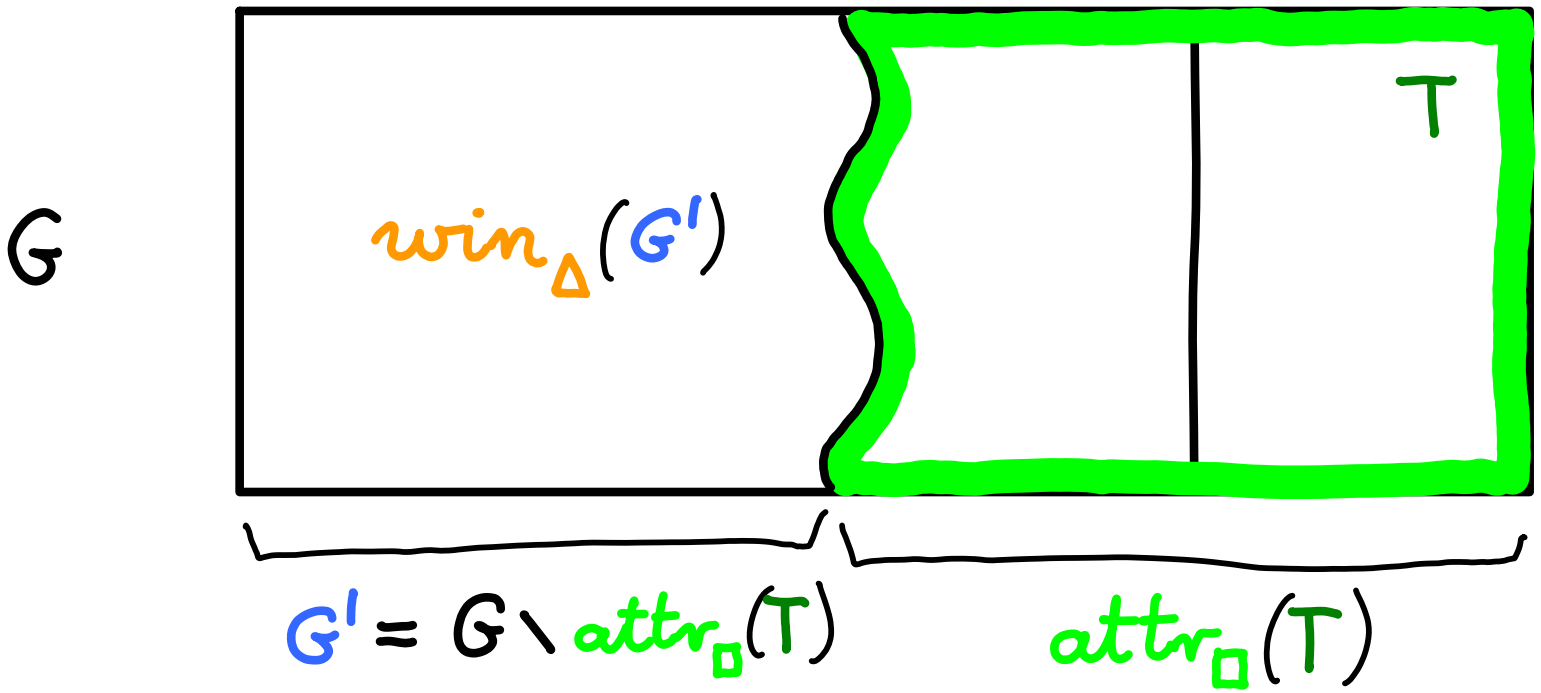
T-ATTRACTOR LEMMA



LEMMA

1. $win_{\triangle}(G') = V \setminus attr_{\square}(T) \subseteq win_{\triangle}(G)$
2. If $G' = \emptyset$ then $win_{\square}(G) = V$

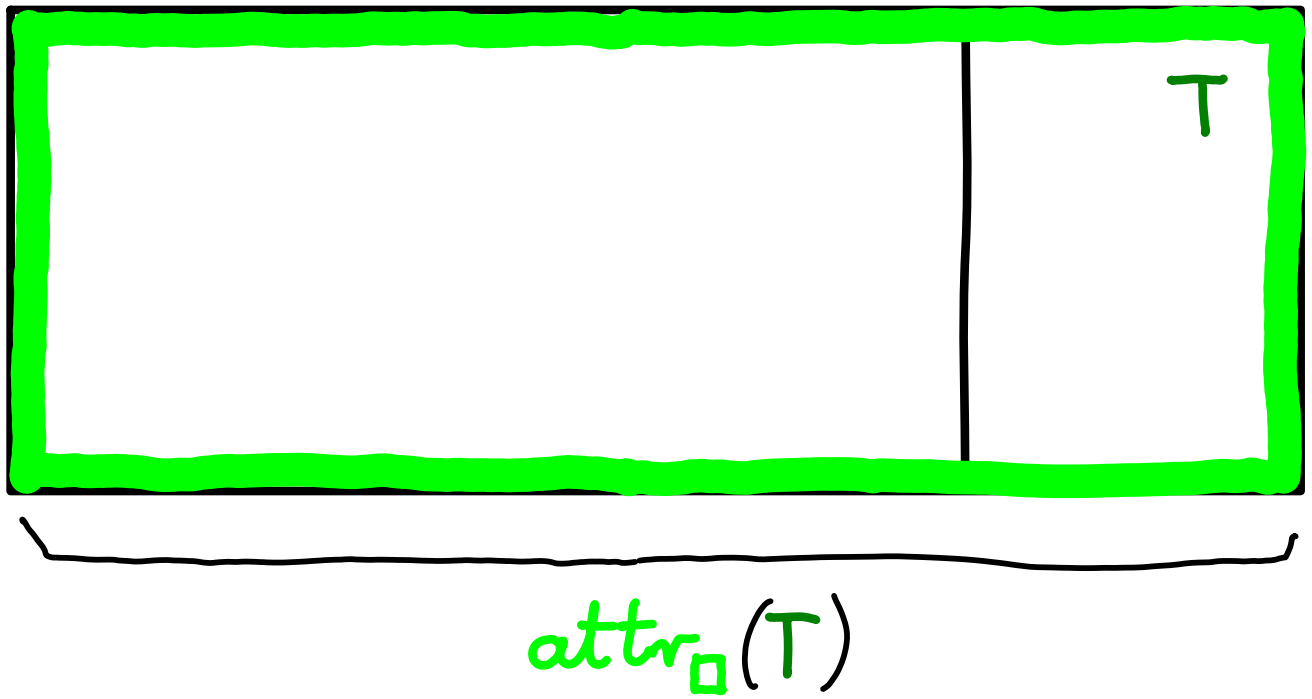
T-attractor Lemma



Lemma

1. $win_\triangle(G') = V \setminus attr_\square(T) \subseteq win_\triangle(G)$
2. If $G' = \emptyset$ then $win_\square(G) = V$

T-ATTRACTOR LEMMA

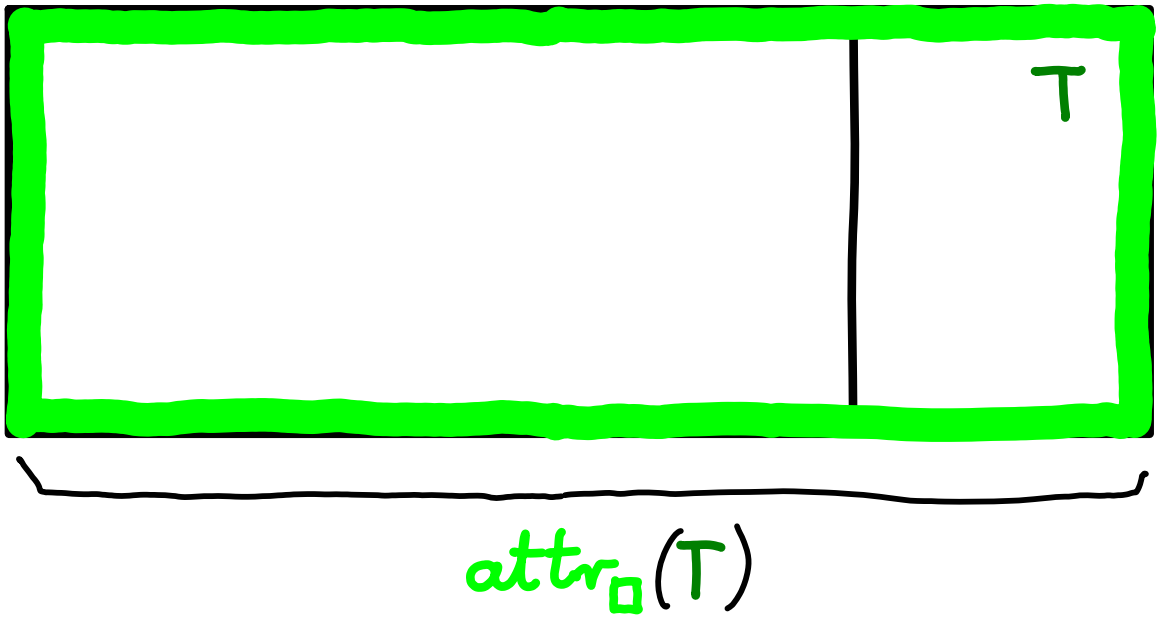


LEMMA

1. $win_{\triangle}(G') = V \setminus attr_{\square}(T) \subseteq win_{\triangle}(G)$

2. If $G' = \emptyset$ then $win_{\square}(G) = V$

A DIVIDE-AND-CONQUER ALGORITHM



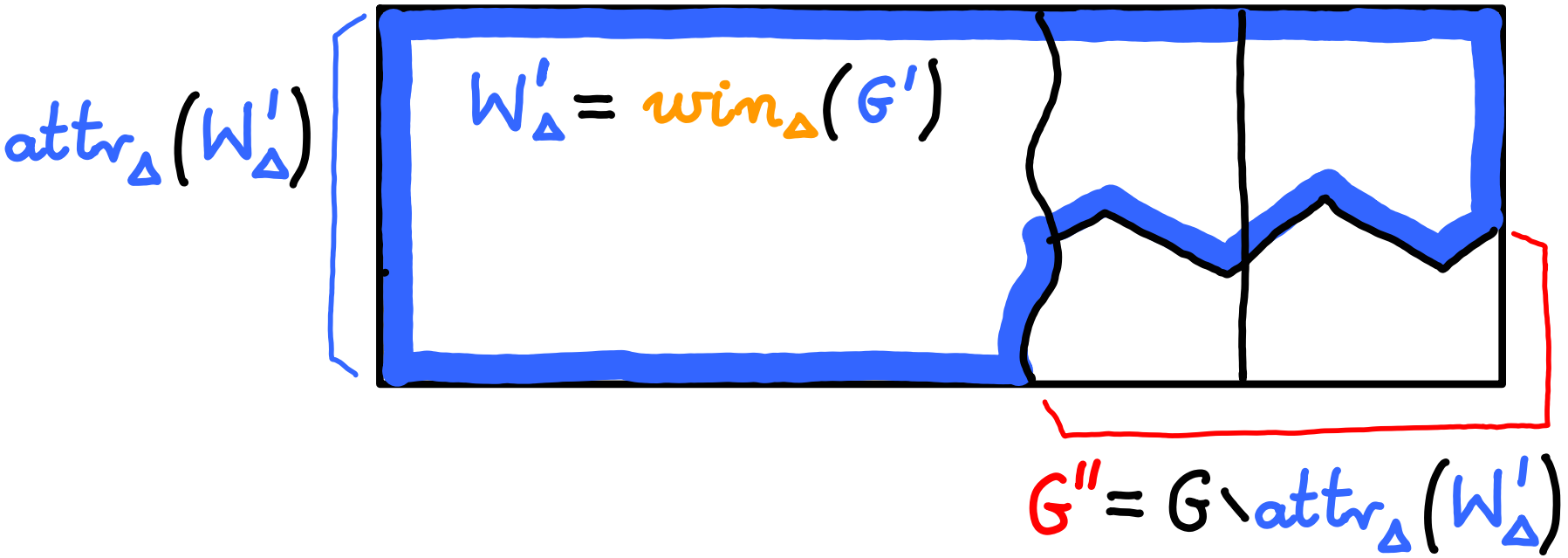
WIN(G): if $G' = \emptyset$ then $(W_{\square}, W_{\triangle}) := (V, \emptyset)$

else $(W''_{\square}, W''_{\triangle}) := \text{WIN}(G'')$

$(W_{\square}, W_{\triangle}) := \left(W''_{\square},\ attr_{\triangle}(W'_{\triangle}) \cup W''_{\triangle}\right)$

return $(W_{\square}, W_{\triangle})$

A DIVIDE-AND-CONQUER ALGORITHM



WIN(G):

if $G' = \emptyset$ then $(W_\square, W_\Delta) := (V, \emptyset)$

else $(W''_\square, W''_\Delta) := \text{WIN}(G'')$

$(W_\square, W_\Delta) := \left(W''_\square,\ attr_\Delta(W'_\Delta) \cup W''_\Delta\right)$

return $(W_\square, W_\Delta)$

POSITIONAL DETERMINACY OF (co-) BÜCHI GAMES

LEMMA

Büchi / co-Büchi games are **positionally determined**

(both on finite and **infinite graphs**)

Exercise 4

Prove the positional determinacy of Büchi and co-Büchi games.

Running time of the divide-and-conquer algorithm

$T(n)$

Win(G):

if $G' = \emptyset$ then $(W_\square, W_\triangle) := (V, \emptyset)$

else $(W''_\square, W''_\triangle) := \textbf{Win}(G'')$

$(W_\square, W_\triangle) := \left(W''_\square, attr_\triangle(W'_\triangle) \cup W''_\triangle\right)$

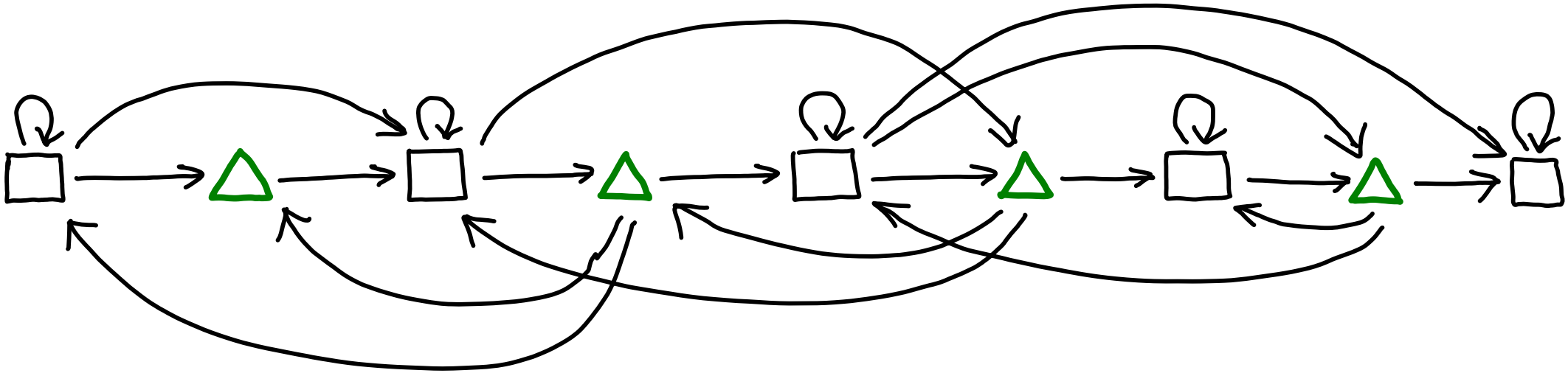
return $(W_\square, W_\triangle)$

$T(n-1)$

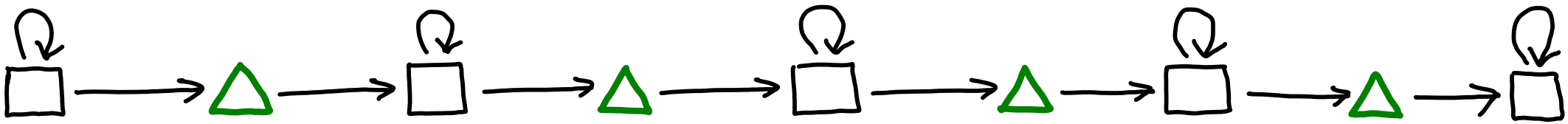
Recurrence: $T(n) \leq T(n-1) + O(m)$

Solution: $T(n) = O(n \cdot m)$

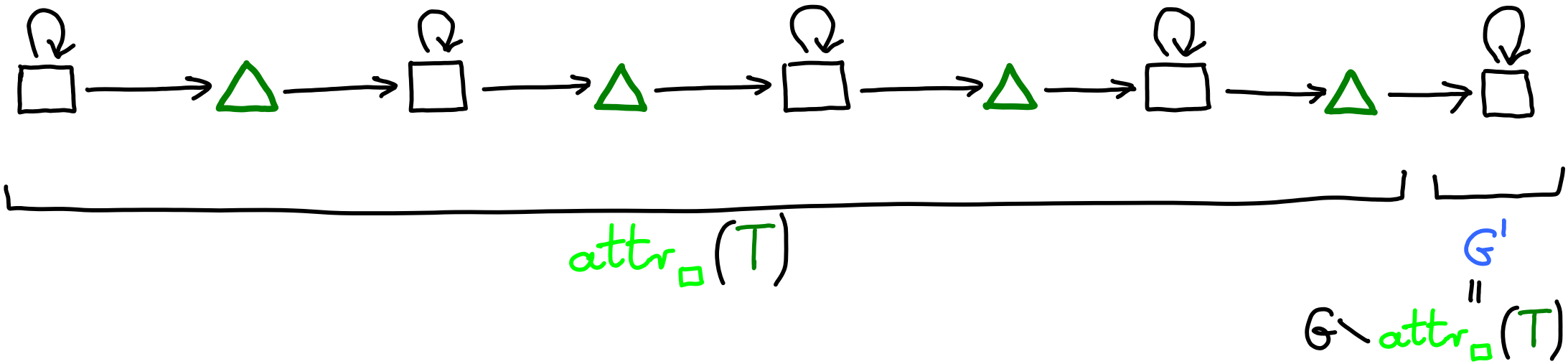
A Büchi game



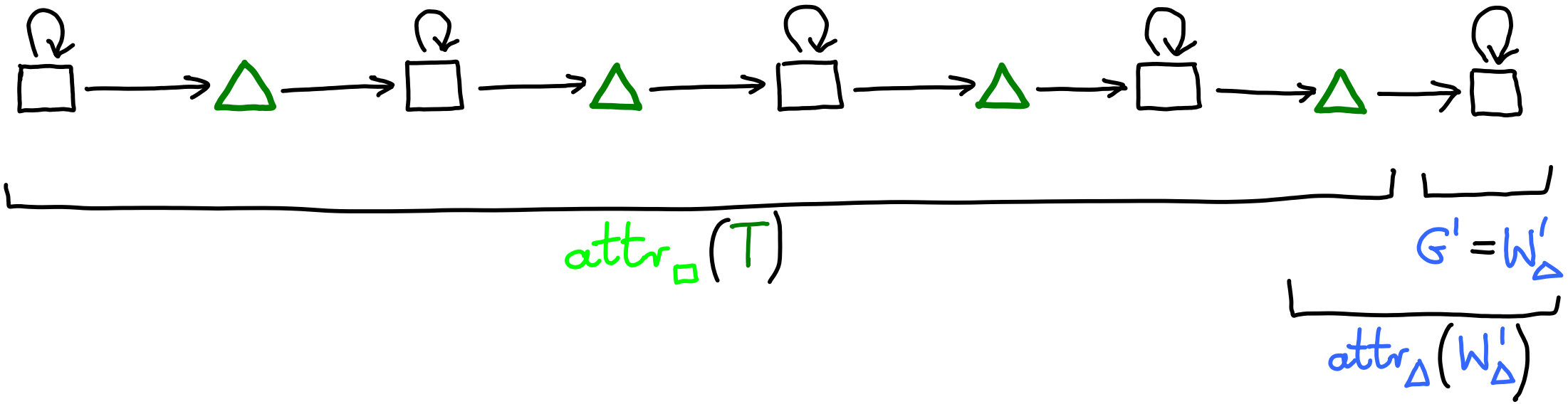
A Büchi game



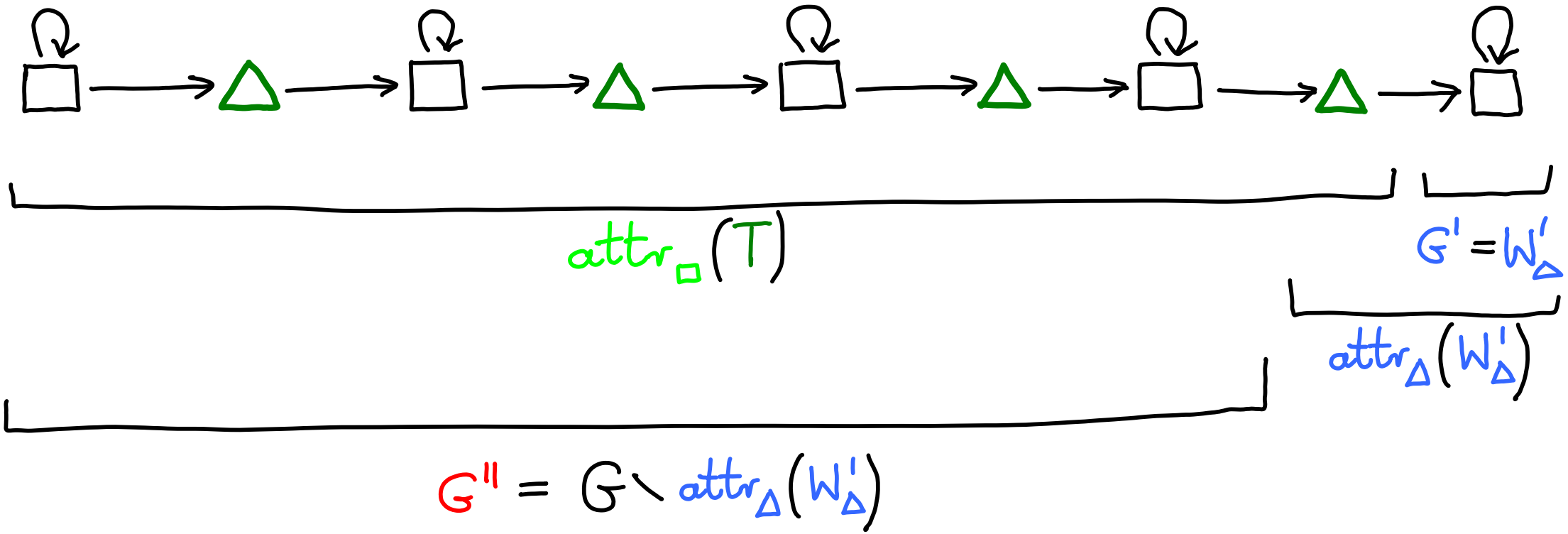
A Büchi game



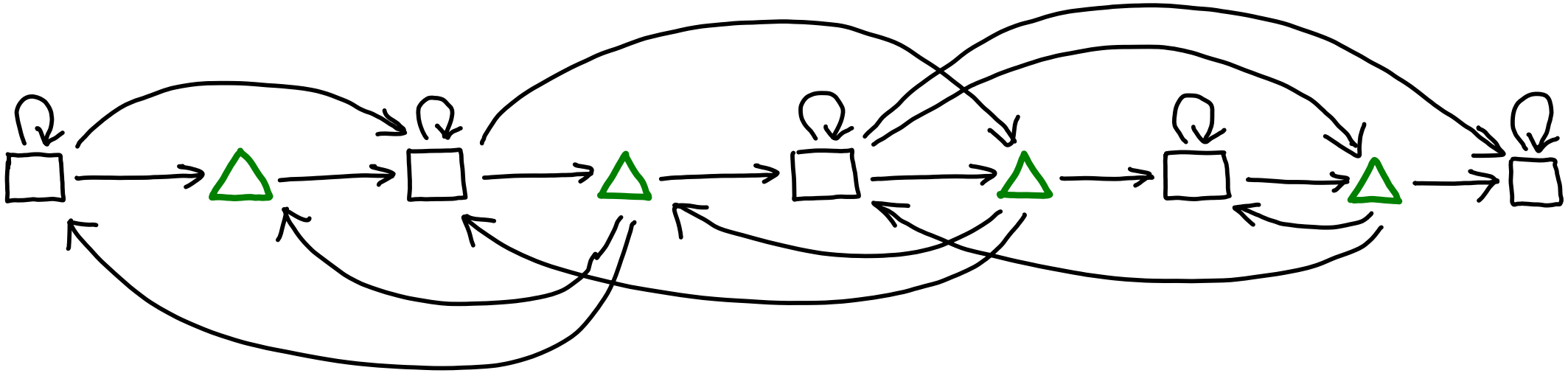
A Büchi game


$\mathrm{attr}_{\square}(T)$
$G' = W'_{\triangle}$
$\mathrm{attr}_{\triangle}(W'_{\triangle})$

A Büchi game


$\mathrm{attr}_{\square}(T)$
$G' = W'_{\triangle}$
$\mathrm{attr}_{\triangle}(W'_{\triangle})$
$G'' = G \setminus \mathrm{attr}_{\triangle}(W'_{\triangle})$

A Büchi game



$attr_{\square}(T)$

$G' = W'_{\triangle}$

$attr_{\triangle}(W'_{\triangle})$

$G'' = G \setminus attr_{\triangle}(W'_{\triangle})$

FACT The divide and conquer algorithm may take $\Theta(mn)$ time

AN $O(n^2)$ ALGORITHM FOR BÜCHI GAMES

[CHATTERJEE, HENZINGER 2012]

- Consider game graphs $G_1, G_2, G_3, \ldots, G_{\log n} = G$ each G_i with $\leq 2^i \cdot n$ edges, respectively
- Keep removing traps (complements of attractors)

  each time proceeding through graphs $G_1, G_2, G_3, \ldots, G_{\log n}$ until one of them gives a non-empty trap

---

KEY PROPERTY If a trap found in G_i first then its size is $\geq 2^i$

- Ammortized analysis charges $(2+2^2+\cdots+2^i)n \leq 2^{i+1} \cdot n$ work to the 2^i vertices removed

PLAN

1. Motivating example
2. Games on graphs
3. Reachability / safety games
4. Büchi / co-Büchi games
5. **Parity games**
6. Better algorithms for parity games
7. Search complexity of computing optimal strategies

GAMES ON GRAPHS: PARITY OBJECTIVES

$$\mathrm{Inf}(a_0, a_1, a_2, \ldots) = \{ a : a_i = a \text{ for } \textbf{infinitely many } i \in \mathbb{N} \}$$

Parity objective:

$$W_\square = \{ (v_0, v_1, v_2, \ldots) : \max(\mathrm{Inf}(p(v_0), p(v_1), p(v_2), \ldots) \text{ is } \textbf{even} \}$$

where $p : V \to \mathbb{N}$ is the priority function

Parity Games

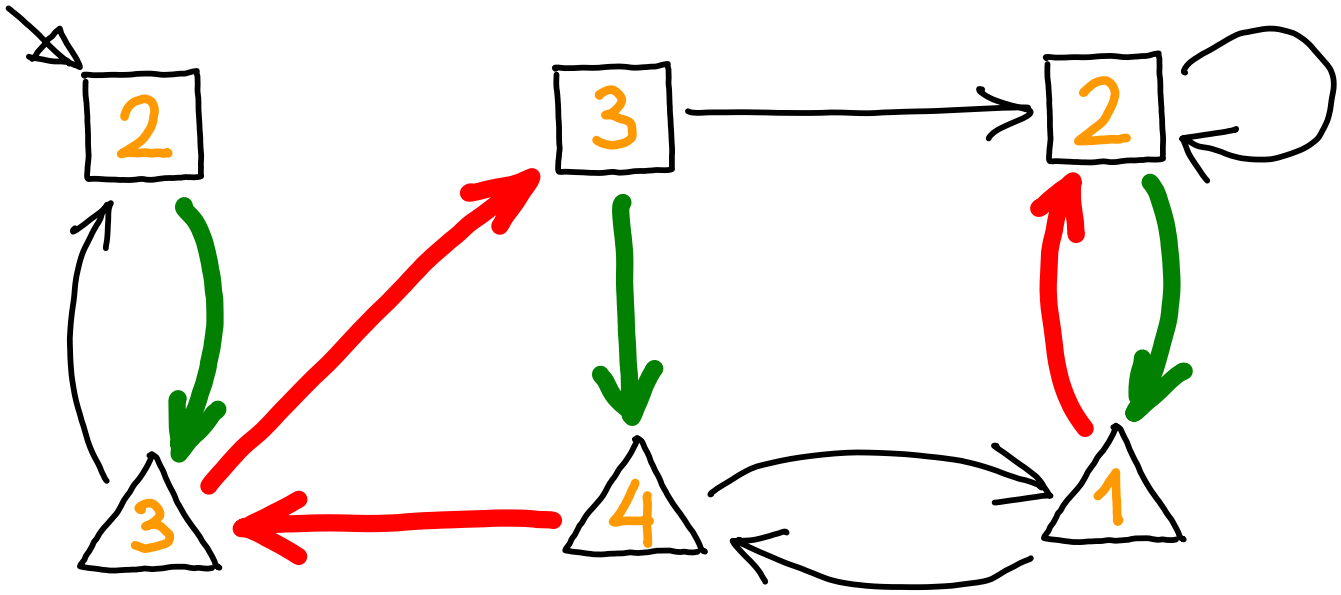
Players:

n vertices, m edges, d priorities

$n = 6$, $m = 11$, $d = 4$

Even = □ = 0

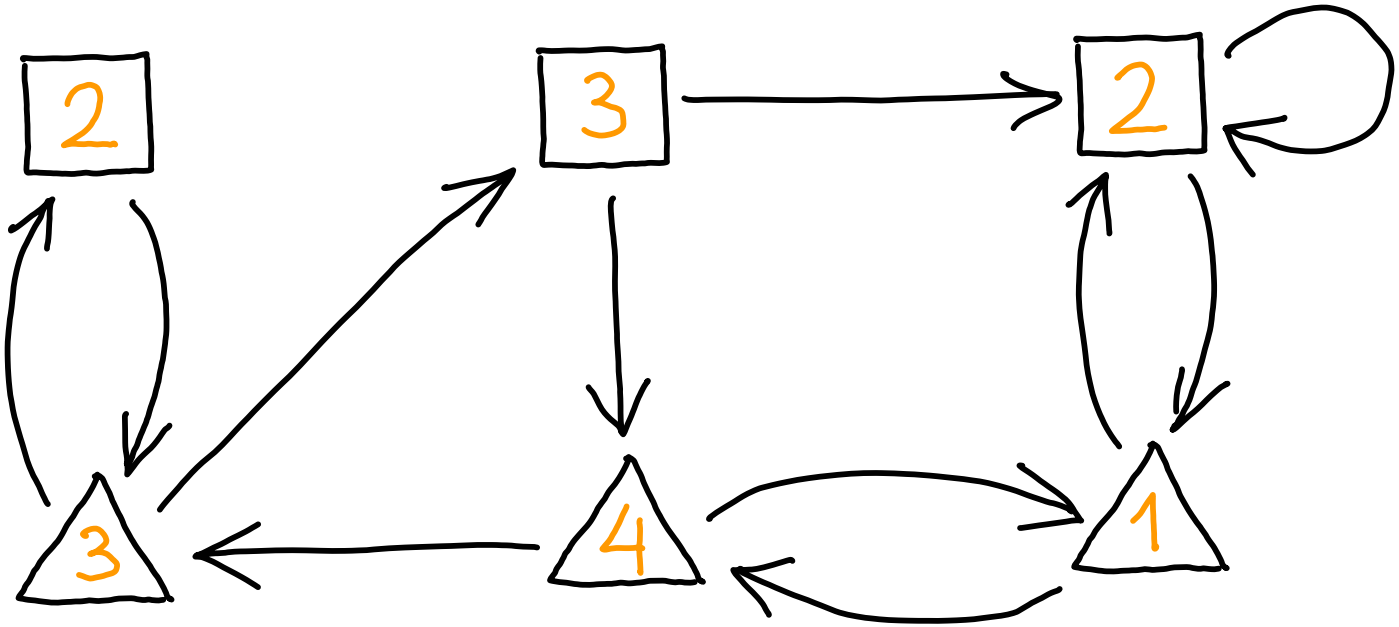
Odd = △ = 1



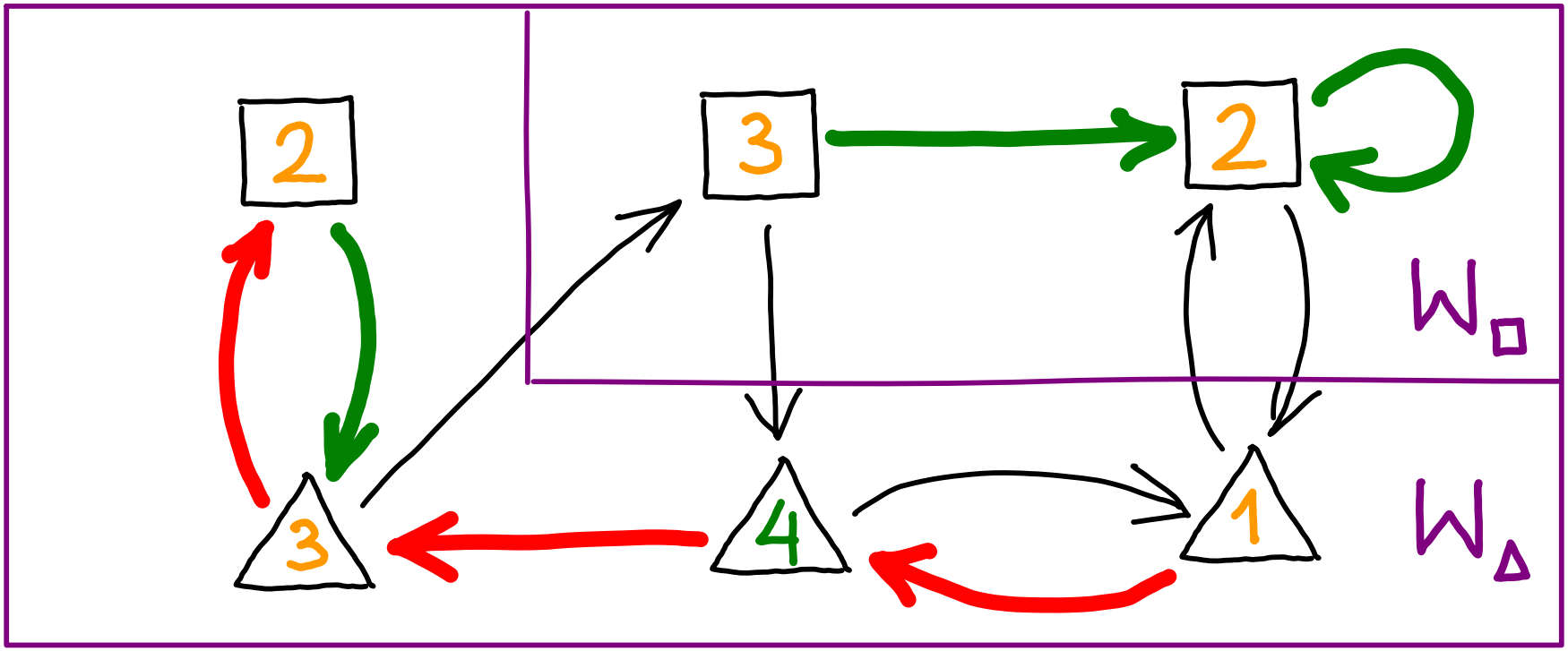
Winner of an infinite play:

parity of the highest priority occurring infinitely often

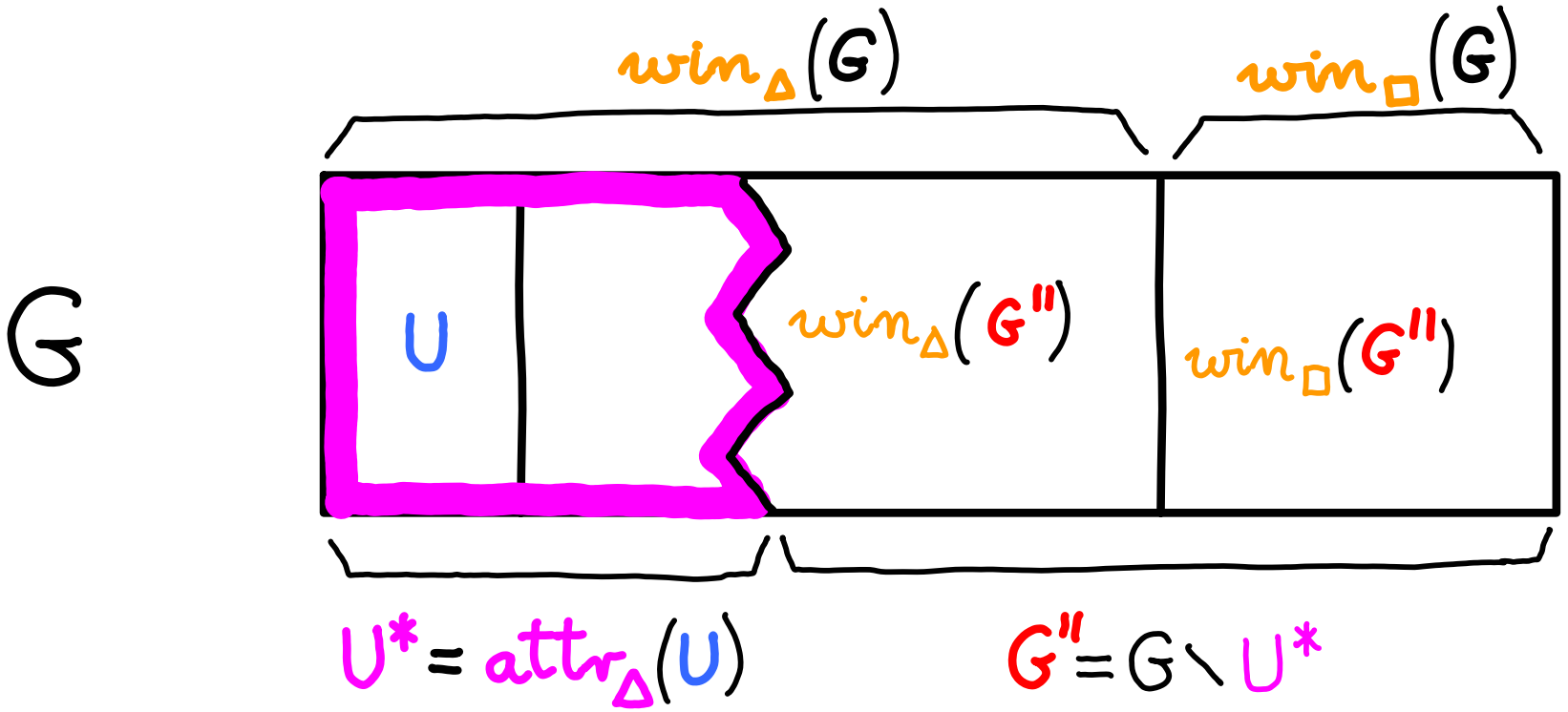
Solving parity games



Solving parity games



Oracle Lemma



Lemma If $U \subseteq win_\Delta(G)$

then $win_\Delta(G) = U^* \cup win_\Delta(G'')$

$win_\square(G) = win_\square(G'')$

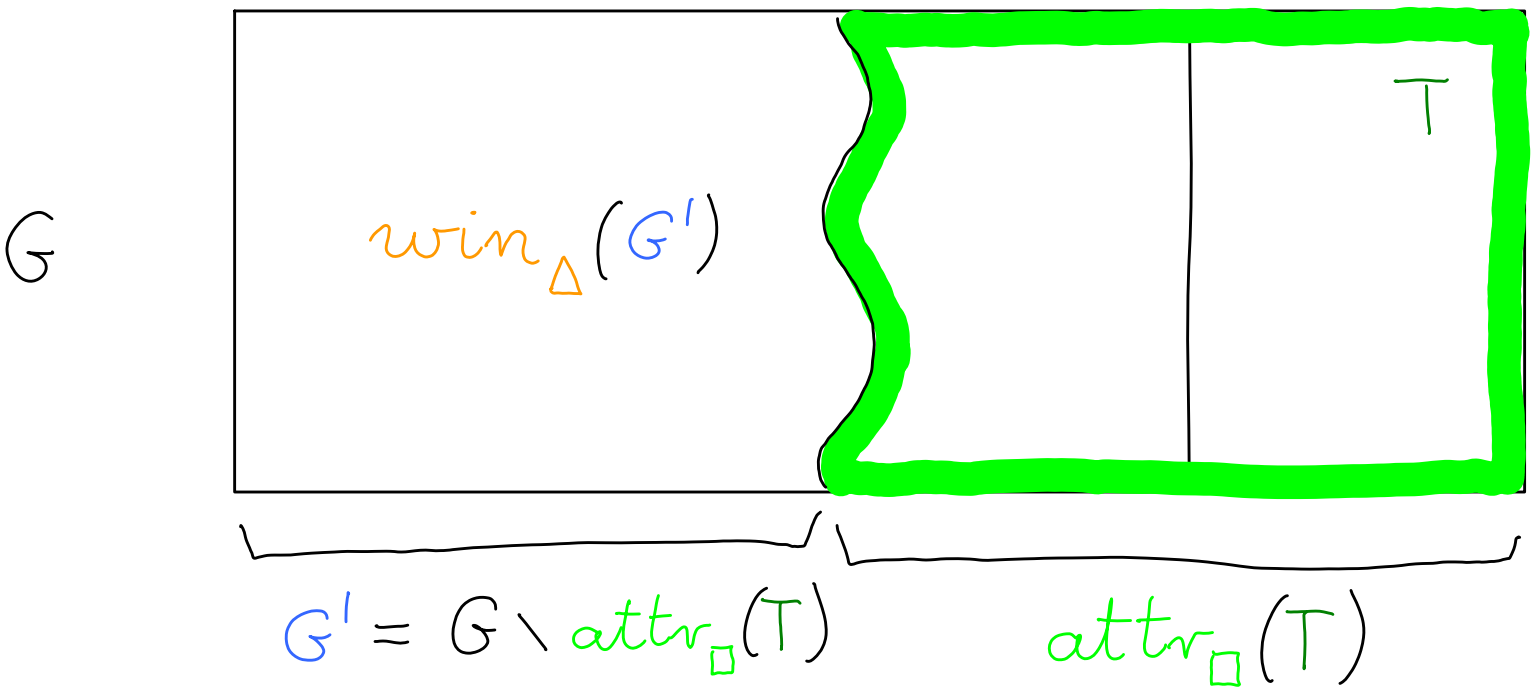
Exercise 5

- Does your proof of the oracle lemma for Büchi/co-Büchi games work also for parity games?

- Are the winning strategies constructed positional?

- If not, what assumptions do you need to make them so?

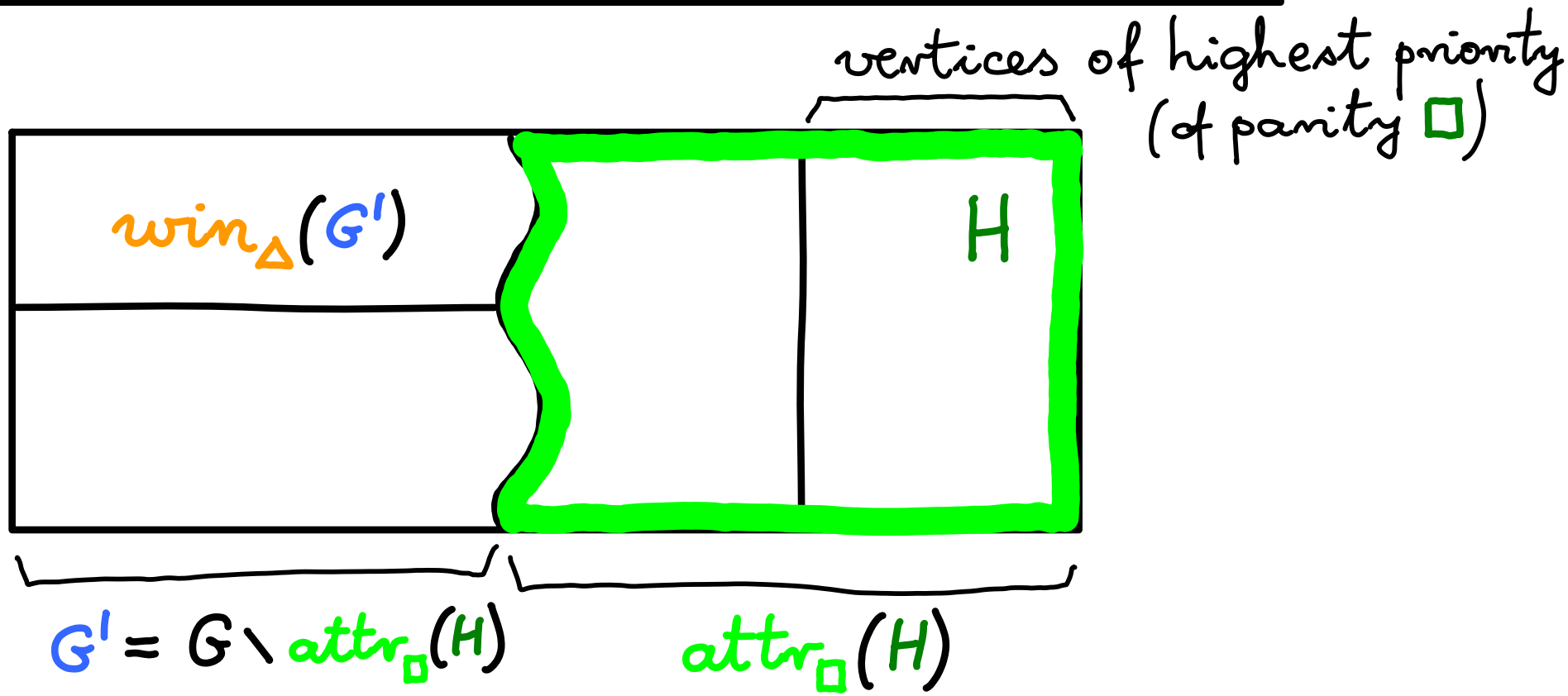
T-ATTRACTOR LEMMA



LEMMA

1. $win_{\triangle}(G') = V \setminus attr_{\square}(T) \subseteq win_{\triangle}(G)$

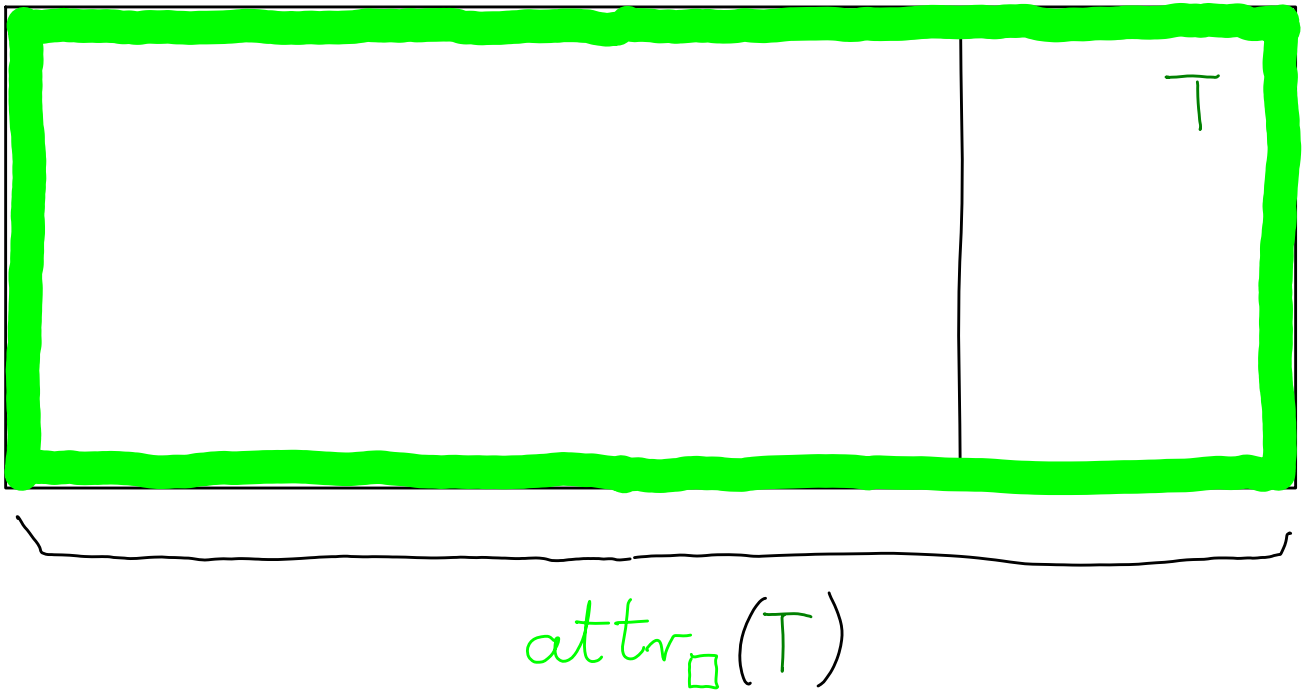
Highest-Priority attractor Lemma



Lemma

1. If $win_{\triangle}(G') \neq \emptyset$ then $win_{\triangle}(G') \subseteq win_{\triangle}(G)$

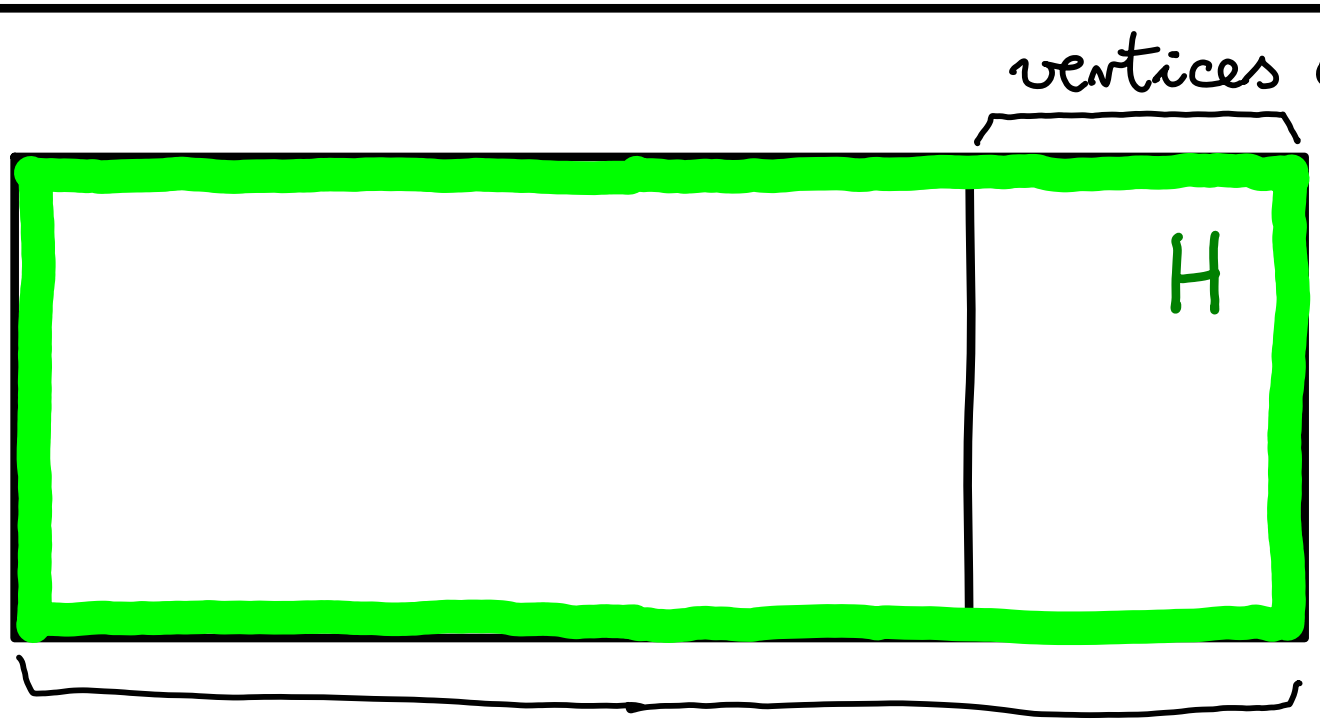
T-ATTRACTOR LEMMA



LEMMA

1. $win_{\triangle}(G') = V \setminus attr_{\square}(T) \subseteq win_{\triangle}(G)$
2. If $G' = \emptyset$ then $win_{\square}(G) = V$

Highest-Priority attractor Lemma

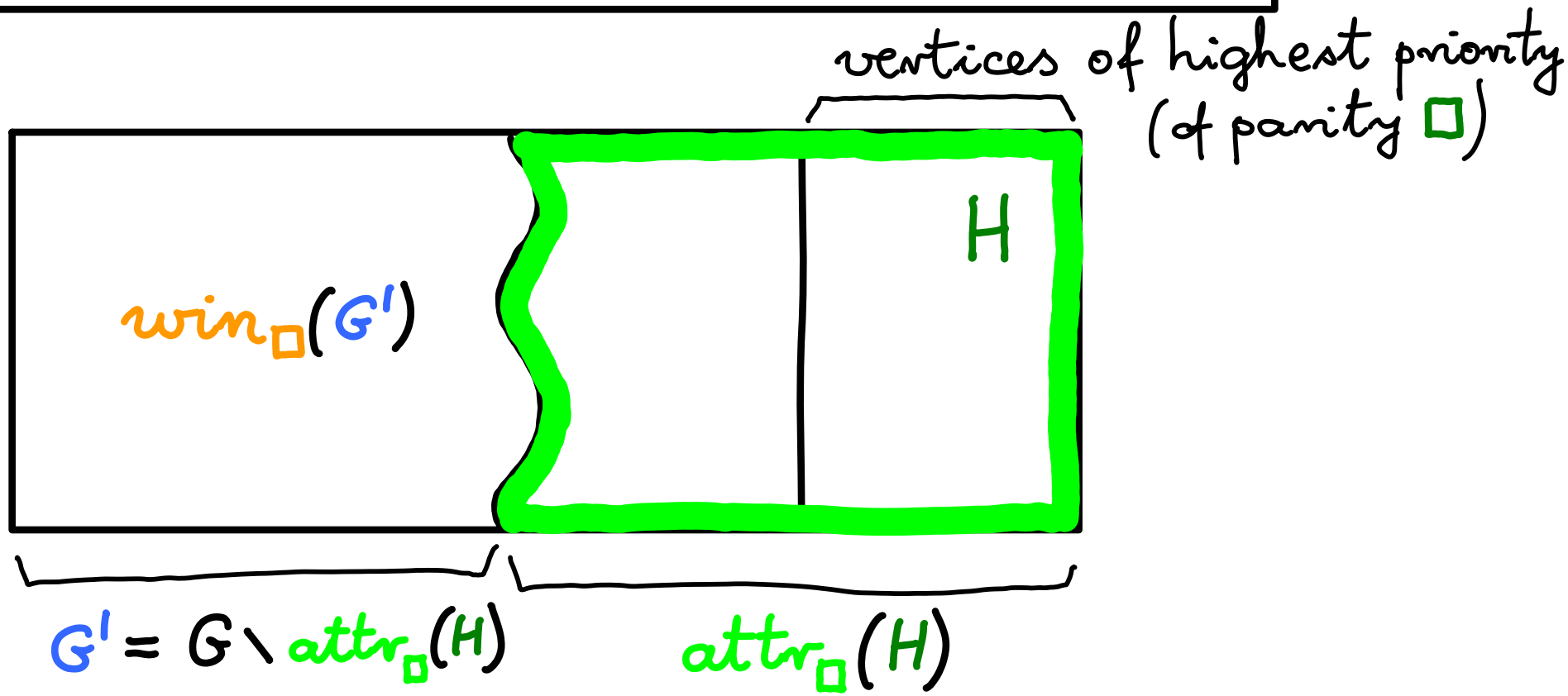


$G' = G \setminus attr_{\square}(H)$

Lemma

1. If $win_{\triangle}(G') \neq \emptyset$ then $win_{\triangle}(G') \subseteq win_{\triangle}(G)$
2. If $G' = \emptyset$ then $win_{\square}(G) = V$

Highest-Priority attractor Lemma



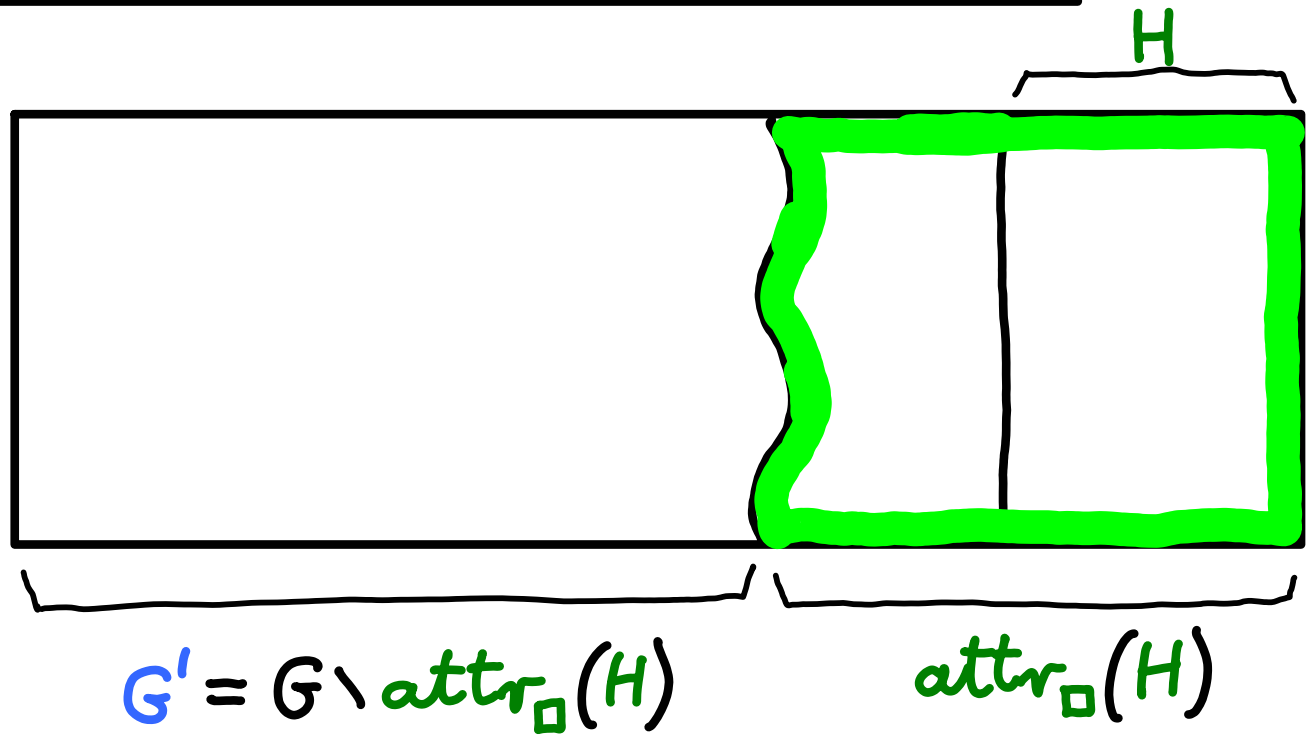
Lemma

1. If $win_{\triangle}(G') \neq \emptyset$ then $win_{\triangle}(G') \subseteq win_{\triangle}(G)$
2. If $win_{\triangle}(G') = \emptyset$ then $win_{\square}(G) = V$

Exercise 6

Prove the highest-priority attractor lemma.

A DIVIDE-AND-CONQUER ALGORITHM



Win(G): $(W'_{\square}, W'_{\triangle}) := \text{Win}(G')$

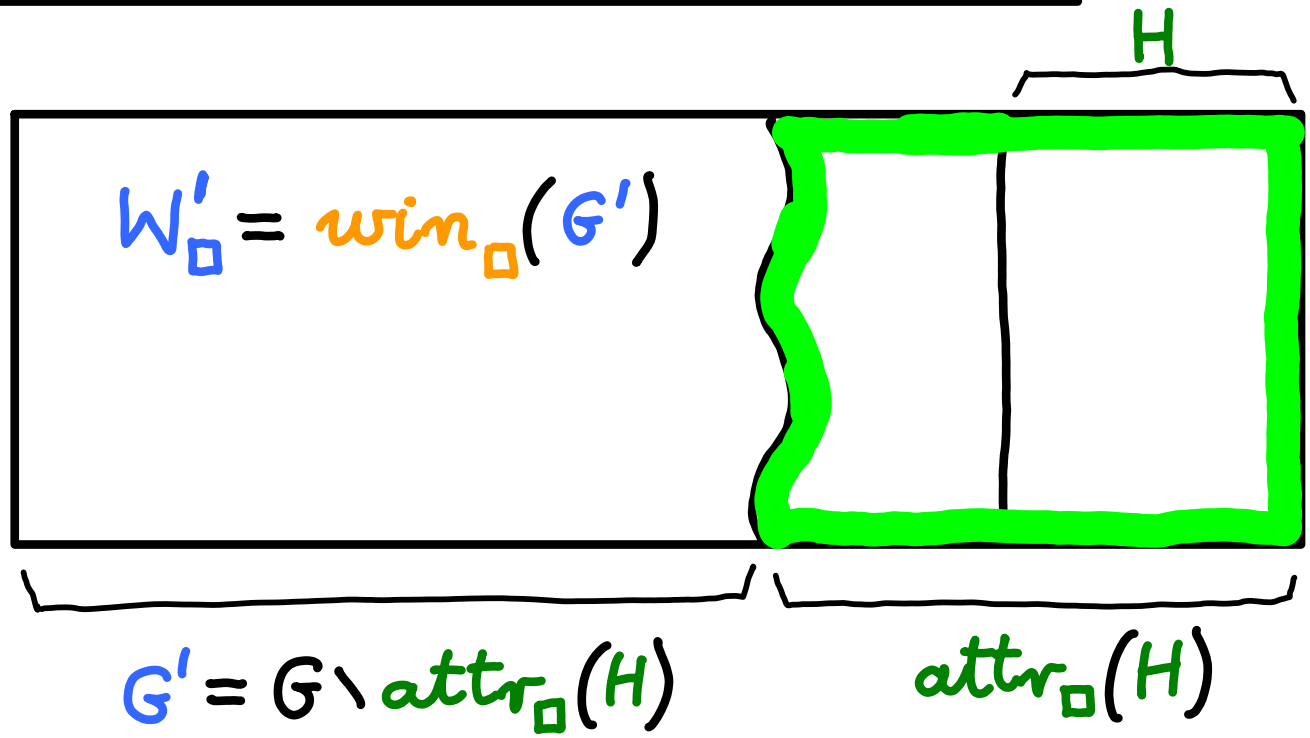
if $W'_{\triangle} = \emptyset$ then $(W_{\square}, W_{\triangle}) := (V, \emptyset)$

else $(W''_{\square}, W''_{\triangle}) := \text{Win}(G'')$

$(W_{\square}, W_{\triangle}) := \left(W''_{\square},\ attr_{\triangle}(W'_{\triangle}) \cup W''_{\triangle}\right)$

return $(W_{\square}, W_{\triangle})$

A DIVIDE-AND-CONQUER ALGORITHM



WIN(G):

$(W'_\square, W'_\triangle) := \text{WIN}(G')$

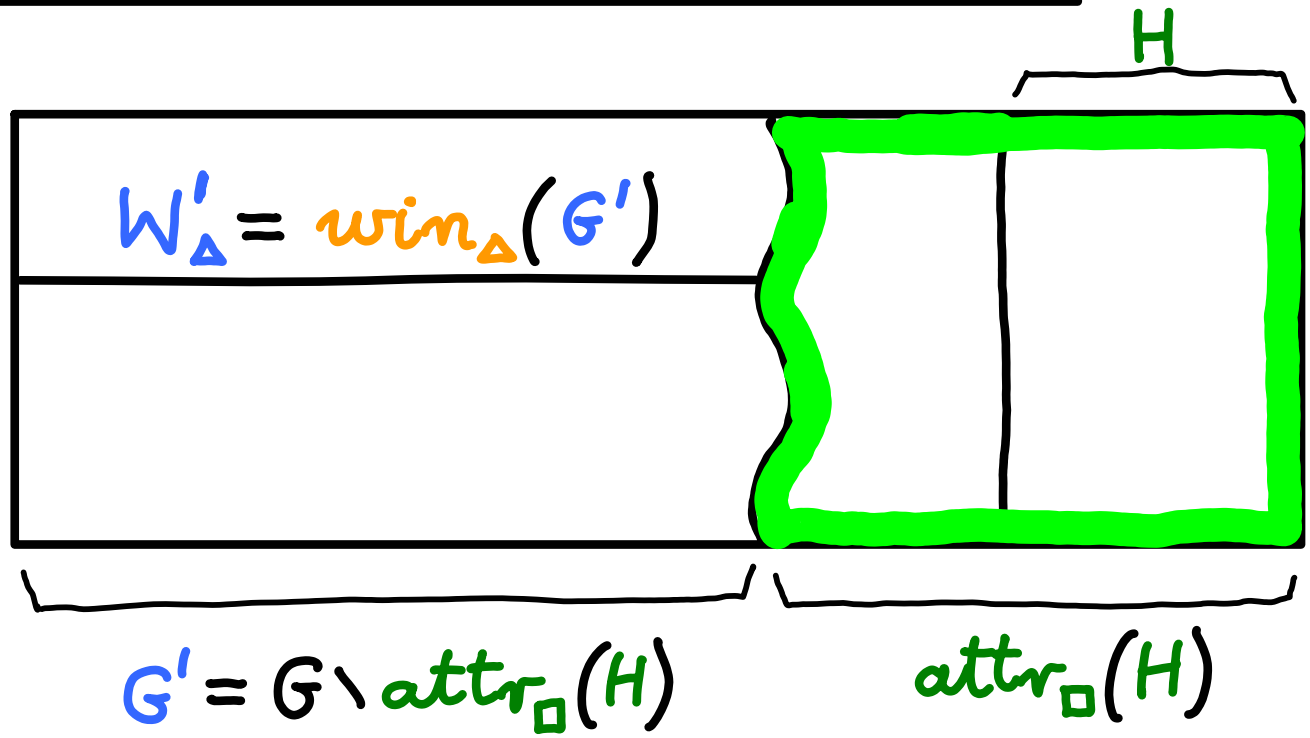
if $W'_\triangle = \emptyset$ then $(W_\square, W_\triangle) := (V, \emptyset)$

else $(W''_\square, W''_\triangle) := \text{WIN}(G'')$

$(W_\square, W_\triangle) := \left(W''_\square,\ attr_\triangle(W'_\triangle) \cup W''_\triangle\right)$

return $(W_\square, W_\triangle)$

A DIVIDE-AND-CONQUER ALGORITHM



WIN(G): $(W'_\square, W'_\triangle) := \text{WIN}(G')$

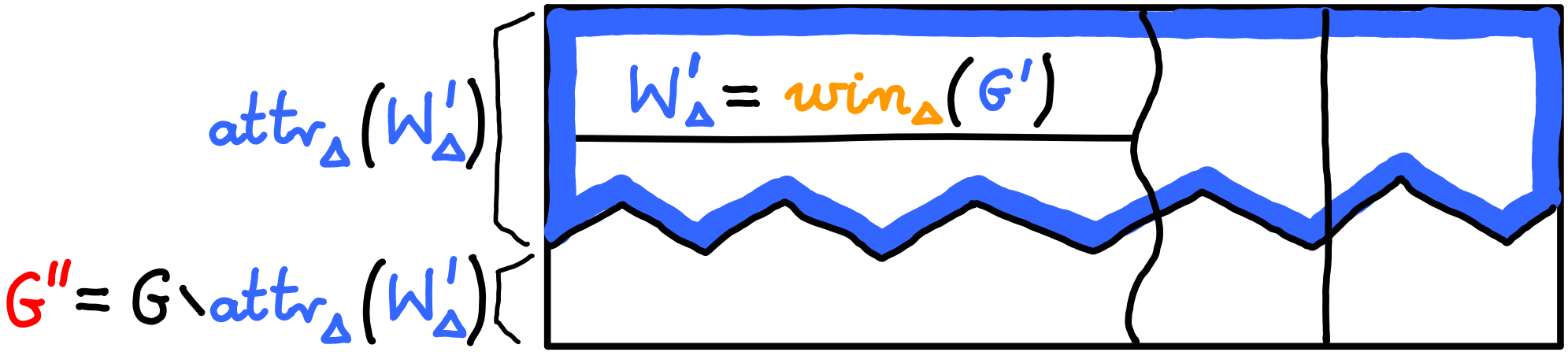
if $W'_\triangle = \emptyset$ then $(W_\square, W_\triangle) := (V, \emptyset)$

else $(W''_\square, W''_\triangle) := \text{WIN}(G'')$

$(W_\square, W_\triangle) := \left(W''_\square,\ attr_\triangle(W'_\triangle) \cup W''_\triangle\right)$

return $(W_\square, W_\triangle)$

A DIVIDE-AND-CONQUER ALGORITHM



WIN(G):

$(W'_\square, W'_\Delta) := \text{WIN}(G')$

if $W'_\Delta = \emptyset$ then $(W_\square, W_\Delta) := (V, \emptyset)$

else $(W''_\square, W''_\Delta) := \text{WIN}(G'')$

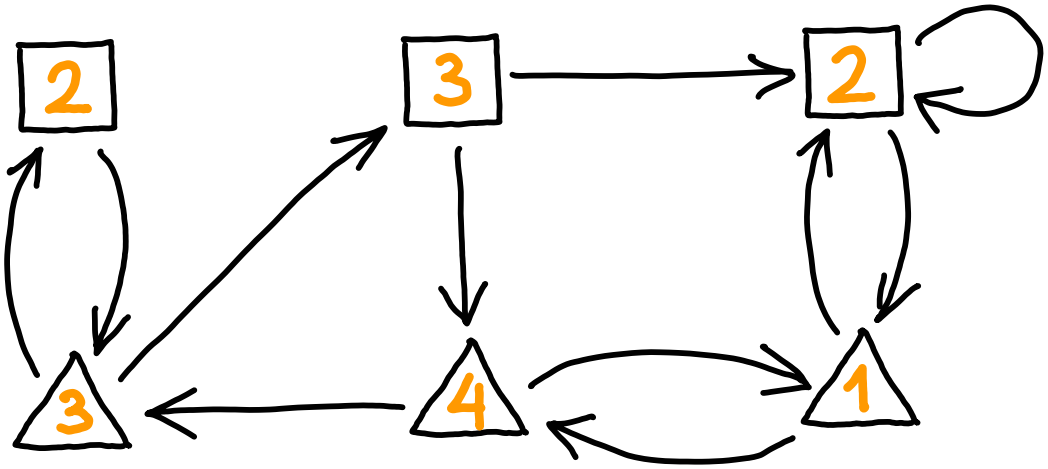
$(W_\square, W_\Delta) := \left(W''_\square,\ attr_\Delta(W'_\Delta) \cup W''_\Delta\right)$

return $(W_\square, W_\Delta)$

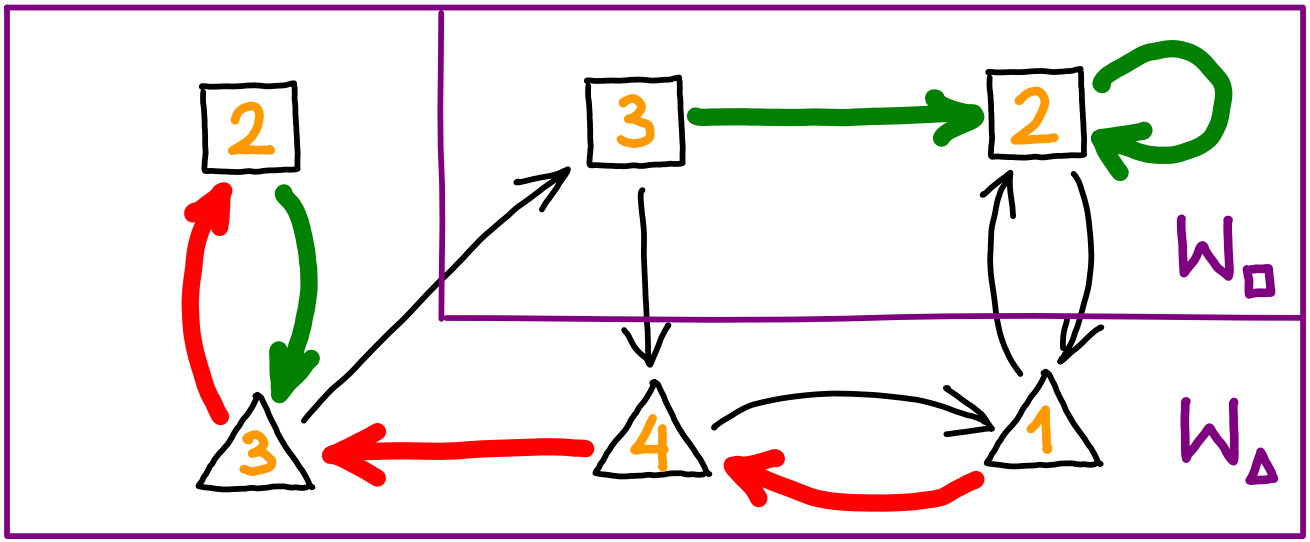
Exercise 7

Run the divide-and-conquer algorithm on the following parity game.

Indicate the winning sets for both players, and their positional winning strategies.



Positional Determinacy



Thm [Emerson, Jutla ; Mostowski 1991]

Parity games are **positionally determined**

Corollary

(Deciding the winner in) parity games is in $NP \cap co\text{-}NP$

Exercise 8

- Use the oracle and highest-priority attractor lemmas, and induction on the size of the game graph, to prove the positional determinacy for parity games.

- Design a polynomial-time algorithm for checking if a given positional strategy of a player in a parity game is a winning strategy for her.

Running time of the divide-and-conquer algorithm

$T(n)$

Win(G): $(W'_\square, W'_\triangle) :=$ **Win**(G') — $T(n-1)$

  if $W'_\triangle = \emptyset$ then $(W_\square, W_\triangle) := (V, \emptyset)$

  else $(W''_\square, W''_\triangle) :=$ **Win**(G'') — $T(n-1)$

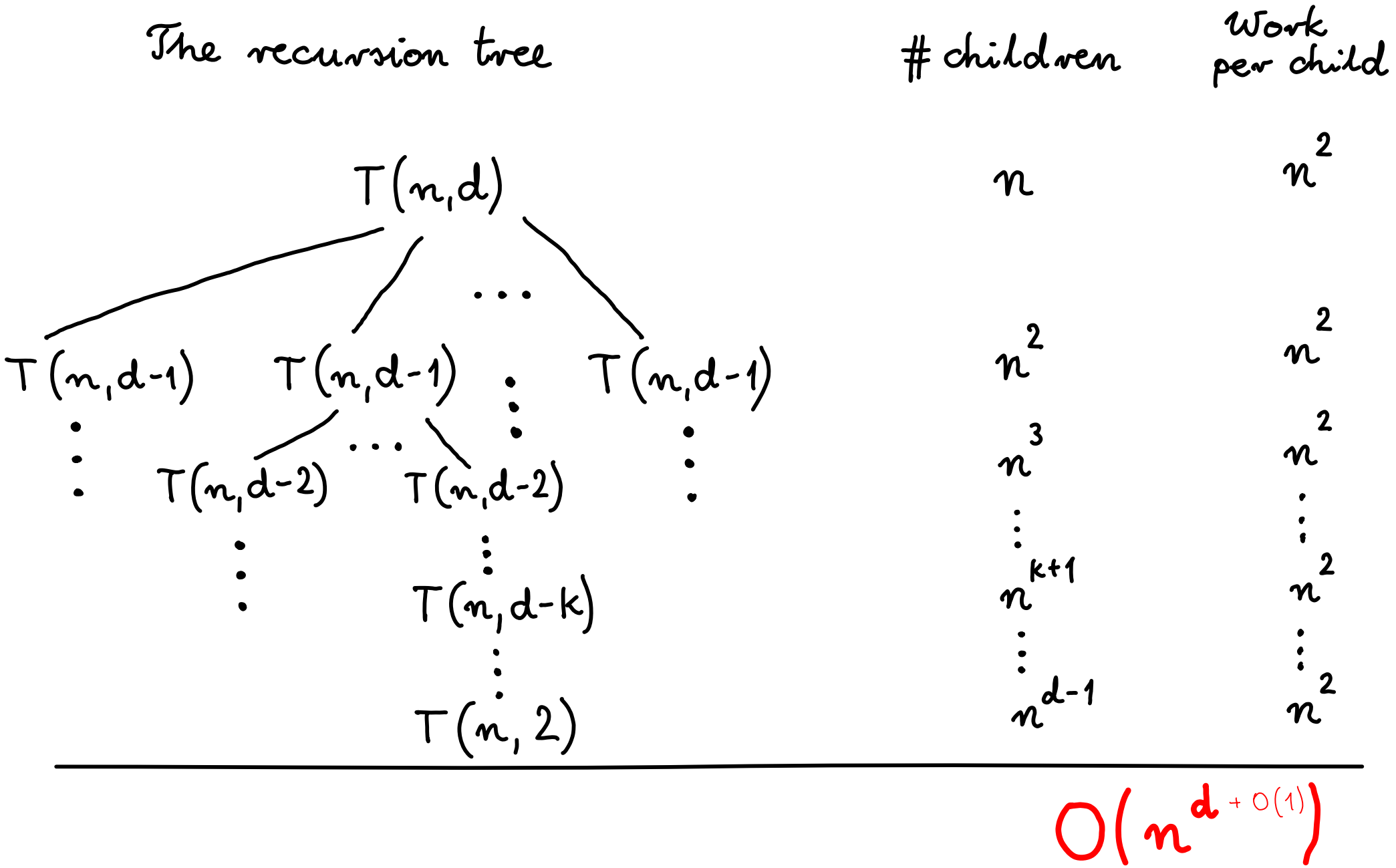
  $(W_\square, W_\triangle) := \left(W''_\square, \mathrm{attr}_\triangle(W'_\triangle) \cup W''_\triangle\right)$

  return $(W_\square, W_\triangle)$

Recurrence: $T(n) \leq 2 \cdot T(n-1) + O(n^2)$

Solution: $T(n) = O(2^n)$

Running time of the divide-and-conquer algorithm



Exercise 9

- Observe that the first recursive call **WIN**(G') is on the game G' with less than d distinct priorities.

- Argue that if the second recursive call **WIN**(G'') occurs then the game G'' has less vertices of priority d than G.

- Conclude that the following recurrence:

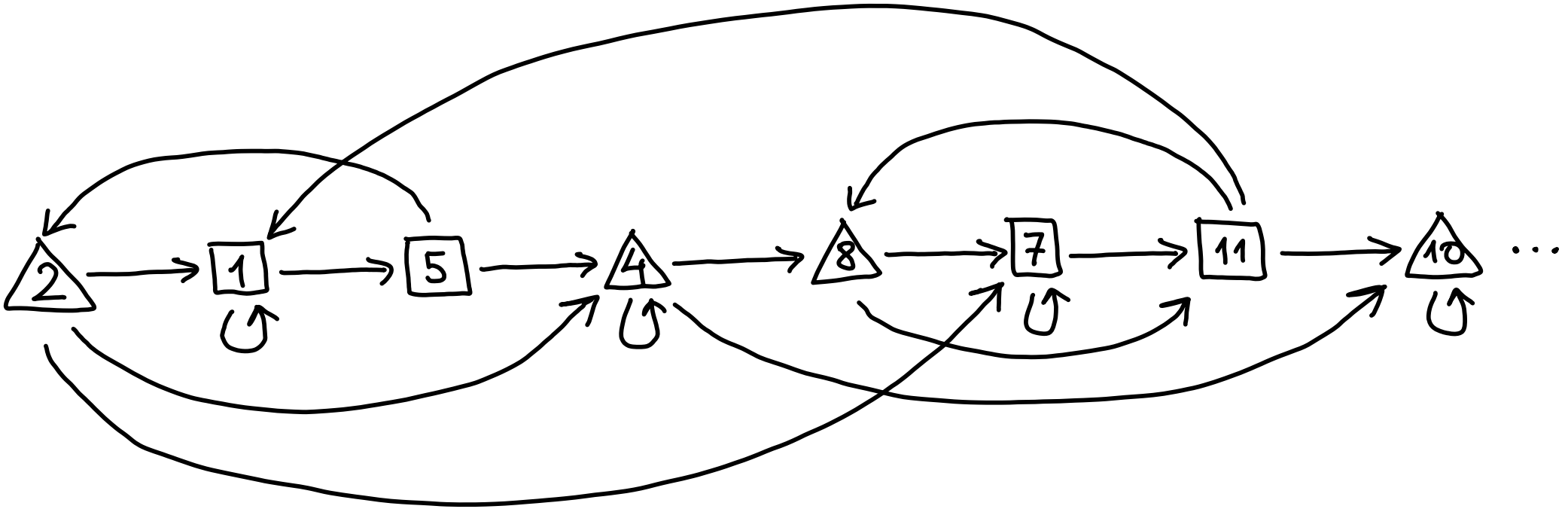
$$\begin{cases} T(n,1) = O(n) \\ T(n,d) \leq (n_d+1) \cdot T(n,d-1) + O(n_d \cdot m) \end{cases}$$

characterizes the running time of the algorithm.

- Prove that $T(n,d) = O\left(m \cdot \left(\frac{n+d}{d}\right)^d\right) = O\left(n^{d+O(1)}\right)$

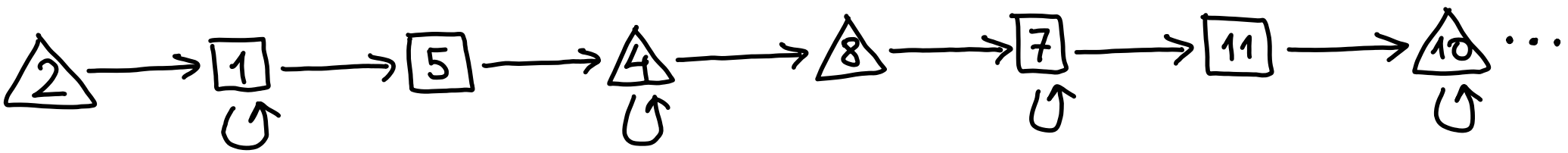
Exercise 10

What is the running time of the divide-and conquer algorithm on this parity game?



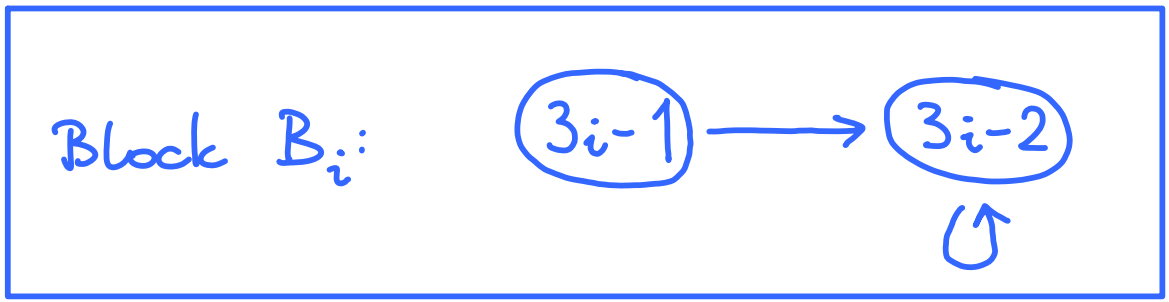
Exercise 10

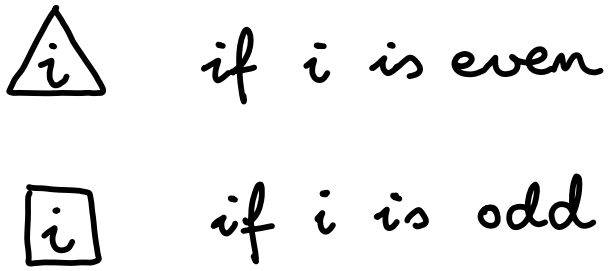
What is the running time of the divide-and conquer algorithm on this parity game?

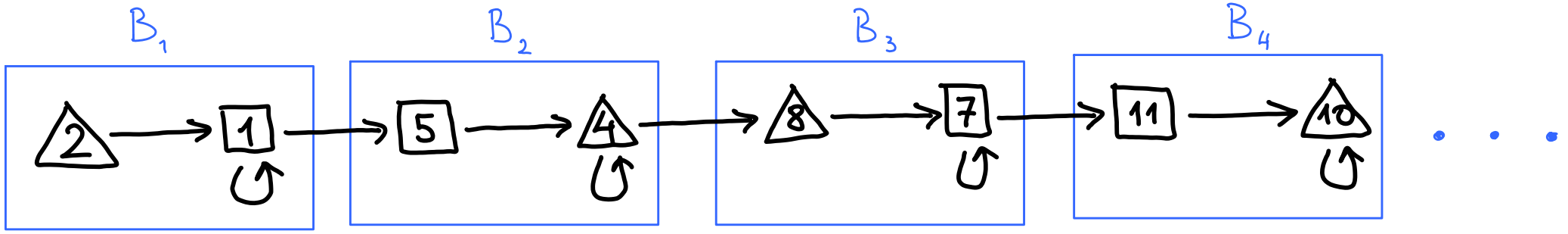


Exercise 10

What is the running time of the divide-and conquer algorithm on this parity game?







PLAN

1. Motivating example
2. Games on graphs
3. Reachability / safety games
4. Büchi / co-Büchi games
5. Parity games
6. **Better algorithms for parity games**
7. Search complexity of computing optimal strategies

Better algorithms

Many priorities

$d = \Omega(n^{\frac{1}{2}+\varepsilon})$

Randomized

Expected $n^{O(\sqrt{n})}$

Random Facet

Deterministic

$n^{O(\sqrt{n})}$

[J., Paterson, Zwick 2008]

Few priorities

$d = O(n^{1/2})$

$O(n^{\frac{d}{2}+O(1)})$

[Browne et al. 1997, Seidl 1996, J. 2000]

$O(n^{\frac{d}{3}+O(1)})$

[Schewe 2007]

Better algorithms

Many priorities

$d = \Omega(n^{\frac{1}{2}+\varepsilon})$

Randomized

Expected $n^{O(\sqrt{n})}$

Deterministic

$n^{O(\sqrt{n})}$

Few priorities

$d = O(n^{1/2})$

Randomized

$O(n^{\frac{d}{2}+O(1)})$

Deterministic

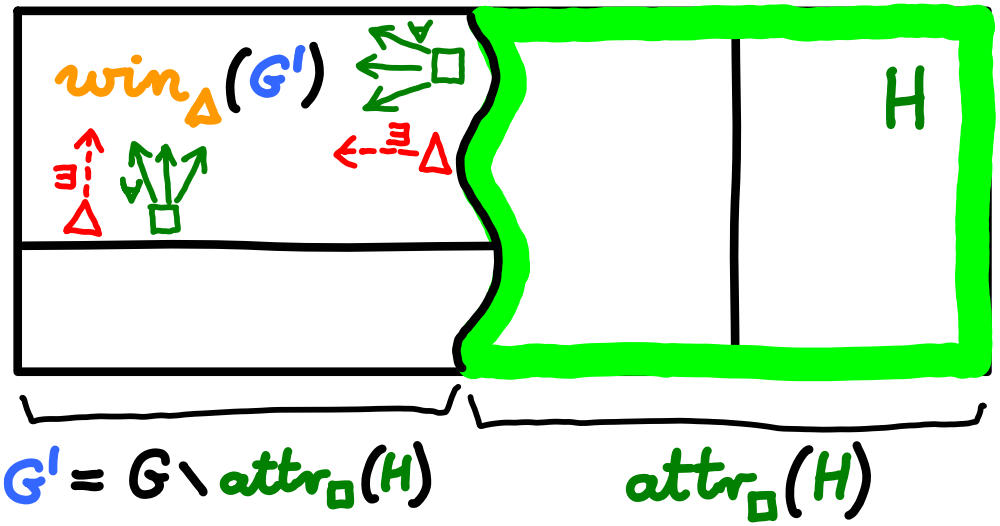
$O(n^{\frac{d}{3}+O(1)})$

Dominions

Def $D \subseteq V$ is an **dominion** for Δ if

Δ has a **trapping** strategy in D

that is **winning** for her



Fact $win_\Delta(G')$ is a dominion for Δ in G

Fact A dominion of size $\leq \ell$ can be found in $n^{O(\ell)}$ time

Exercise 11

Design a brute-force algorithm for detecting dominions of size $\leq \ell$ that runs in time $O(n^\ell)$.

A Subexponential Algorithm

$T(n)$

Win(G): If there is a dominion $D \neq \emptyset$ of size $\leq \ell$

then return $\textsc{Win}(G \setminus D) \oplus D$

else

$(W'_\square, W'_\triangle) := \textsc{Win}(G')$

if $W'_\triangle = \emptyset$ then $(W_\square, W_\triangle) := (V, \emptyset)$

else $(W''_\square, W''_\triangle) := \textsc{Win}(G'')$

$(W_\square, W_\triangle) := \left(W''_\square, \mathrm{attr}_\triangle(W'_\triangle) \cup W''_\triangle\right)$

return $(W_\square, W_\triangle)$

$n^{O(\ell)}$

$T(n-1)$

$T(n-1)$

$T(n-\ell)$

Recurrence: $T(n) \leq n^{O(\ell)} + T(n-1) + T(n-\ell)$

Solution: $\boxed{T(n) = n^{O(\sqrt{n})}}$ if $\ell = \sqrt{n}$

Exercise 12

- Define labelled tree τ_n
  - root labelled n
  - if $n > 3$ then the root has:
    - * left subtree τ_{n-1}
    - * right subtree $\tau_{n-\lceil\sqrt{2n}\rceil}$
  - if $n = 1, 2,$ or 3 then the root has no children
- Prove that on every path from root to leaf in τ_n there are at most $\lfloor\sqrt{2n}\rfloor$ right turns.

  Hint: Use induction and the fact that

  $$\text{if } \tfrac{1}{2}j^2 < n \leq \tfrac{1}{2}(j+1)^2 \text{ then } n - \lceil\sqrt{2n}\rceil \leq \tfrac{1}{2}j^2$$

- Conclude that τ_n has $n^{O(\sqrt{n})}$ leaves.

Better algorithms

Many priorities

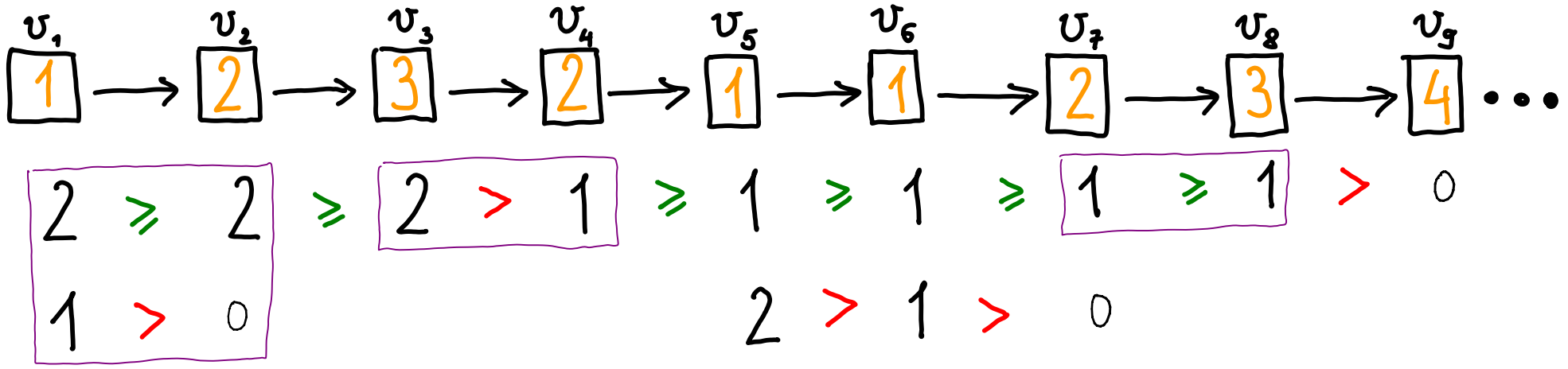
$d = \Omega(n^{\frac{1}{2}+\varepsilon})$

Few priorities

$d = O(n^{1/2})$

	Randomized	Deterministic
Expected	$n^{O(\sqrt{n})}$	$n^{O(\sqrt{n})}$
	$O(n^{\frac{d}{2}+O(1)})$	$O(n^{\frac{d}{3}+O(1)})$

A SUFFICIENT CONDITION FOR WINNING PLAYS



FACT

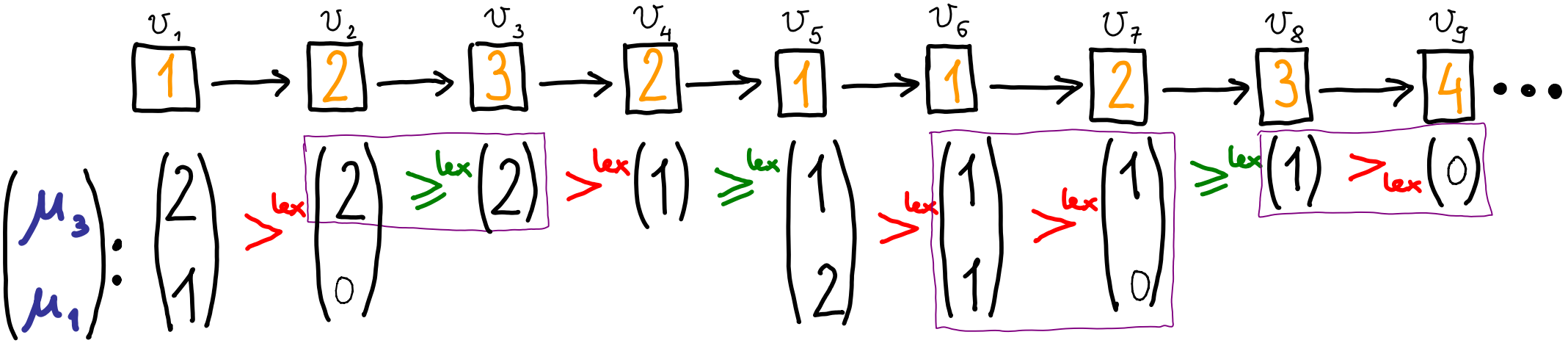
$v_1, v_2, v_3, \ldots$ is winning for $\square$

if

there are $\mu_1, \mu_3, \ldots, \mu_{d-1} : V \to \mathbb{N}$, s.t.

- $\mu_k(v_i) \geqslant \mu_k(v_{i+1})$ for $p(v_i) < k \leq d-1$
- $\mu_k(v_i) > \mu_k(v_{i+1})$ for $k = p(v_i)$

A SUFFICIENT CONDITION FOR WINNING PLAYS



FACT

$v_1, v_2, v_3, \ldots$ is winning for $\square$

if

there are $\mu_1, \mu_3, \ldots, \mu_{d-1} : V \to \mathbb{N}$, s.t.

$$\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v_i)} \end{pmatrix}(v_i) \geq^{lex} \begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v_i)} \end{pmatrix}(v_{i+1})$$

$>^{lex}$ if $p(v_i)$ is odd!

Exercise 13

- Prove the two facts.

- Argue that the following holds for the first characterization:

  if $\mu_k(v) = l$

  then, starting from v,

  at most l vertices of priority k will occur,

  before a vertex of priority $> k$ occurs.

- Does a similar property hold for the second characterization?

Characterizing existence of winning plays

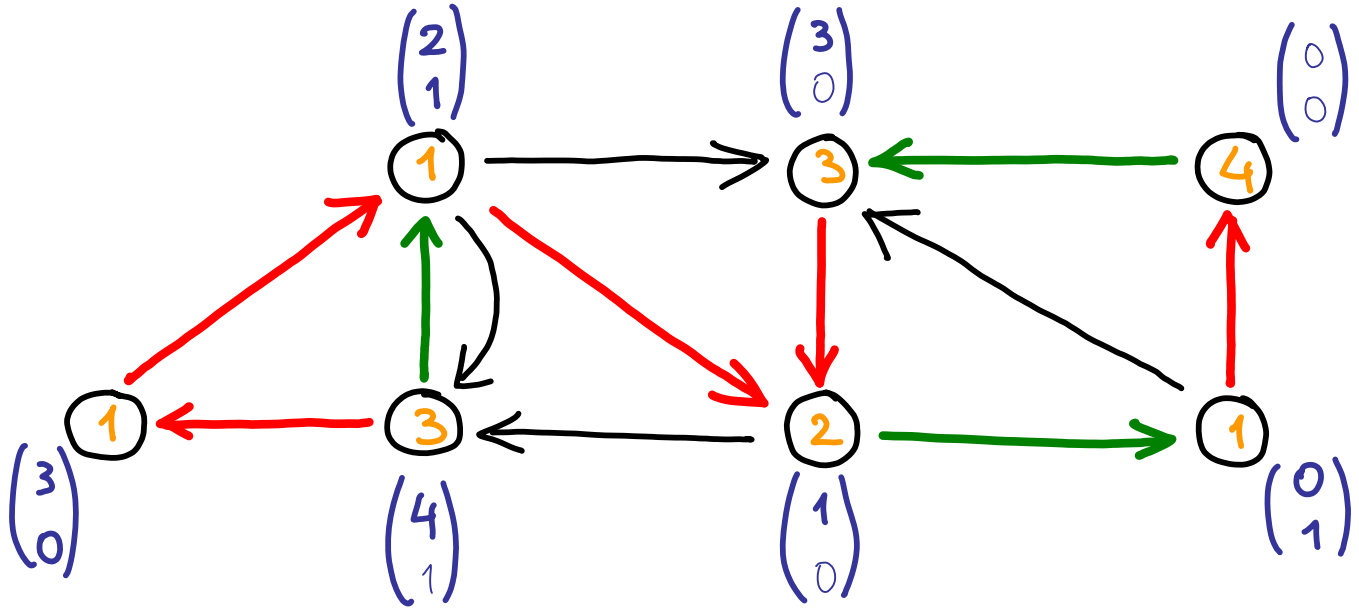
Lemma

There is a winning play (from every vertex) for $\square$ if there are $\mu_1, \mu_3, \ldots, \mu_{d-1} : V \to \mathbb{N}$, s.t.

$$\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(v) \geq^{\text{lex}} \min_{(v,w)\in E} \begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(w)$$

$>^{\text{lex}}$ if $p(v)$ is odd!

Characterizing existence of winning plays



$$\mu_3(v) >^{lex} \min_{(v,w)\in E} \mu_3(w) \quad \text{if } p(v) = 3$$

$$\mu_3(v) \geq^{lex} \min_{(v,w)\in E} \mu_3(w) \quad \text{if } p(v) = 2$$

$$\begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v) >^{lex} \min_{(v,w)\in E} \begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v) \quad \text{if } p(v) = 1$$

CHARACTERIZING UNIVERSALITY OF WINNING PLAYS

LEMMA

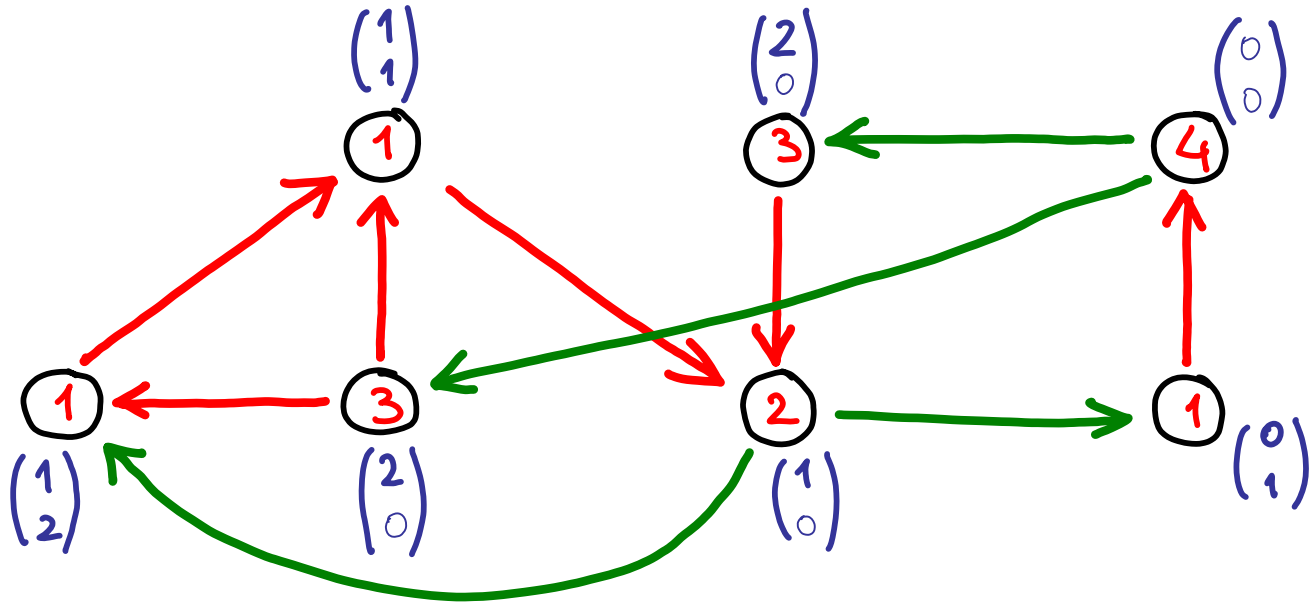
ALL plays (from every vertex) are winning for $\square$

if there are $\mu_1, \mu_3, \ldots, \mu_{d-1} : V \to \mathbb{N}$, s.t.

$$\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(v) \geq^{\text{lex}} \max_{(v,w)\in E} \begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(w)$$

$>^{\text{lex}}$ if $p(v)$ is odd!

Characterizing universality of winning plays



$$\mu_3(v) >^{lex} \max_{(v,w)\in E} \mu_3(w) \qquad \text{if } p(v) = 3$$

$$\mu_3(v) \geq^{lex} \max_{(v,w)\in E} \mu_3(w) \qquad \text{if } p(v) = 2$$

$$\begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v) >^{lex} \max_{(v,w)\in E} \begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v) \qquad \text{if } p(v) = 1$$

Characterizing existence of winning strategies

Theorem

There is a winning strategy for $\Box$ (from every vertex)

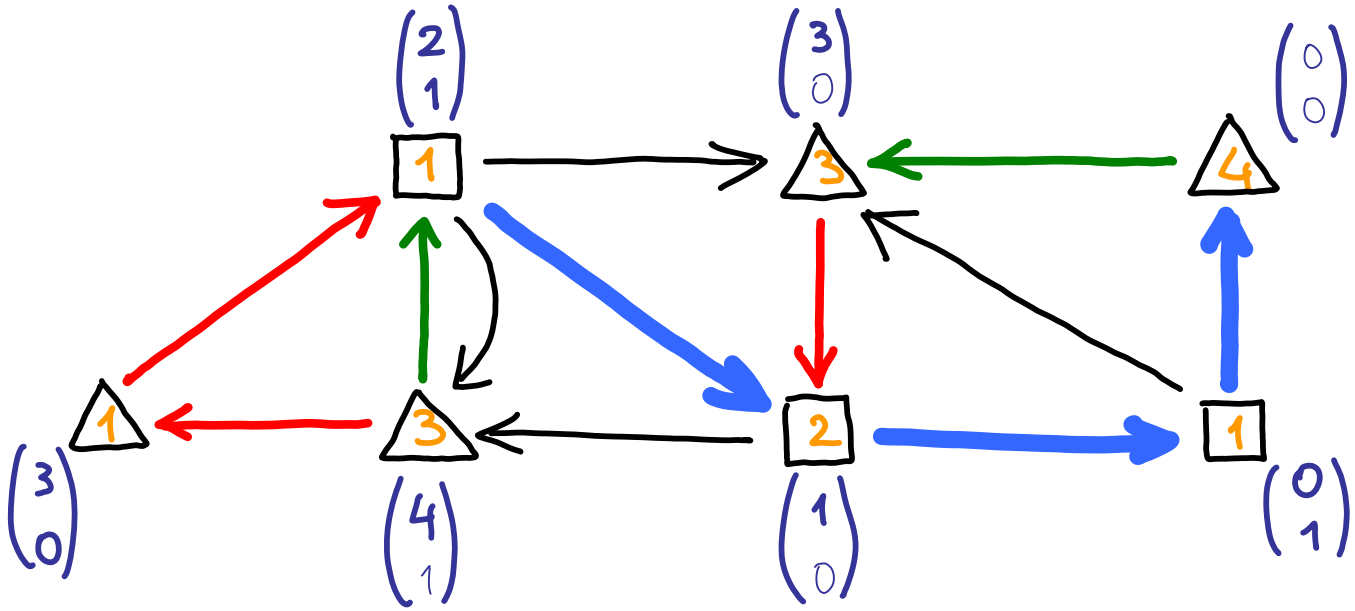
iff

there are $\mu_1, \mu_3, \ldots, \mu_{d-1} : V \to \mathbb{N}$, s.t.

- $$\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(v) \geq^{lex} \min_{(v,w)\in E} \begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(w)$$ $>^{lex}$ if $p(v)$ is odd! for $v \in V_\Box$

- $$\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(v) \geq^{lex} \max_{(v,w)\in E} \begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(w)$$ $>^{lex}$ if $p(v)$ is odd! for $v \in V_\Diamond$

Characterizing existence of winning strategies



$v \in V_\square$

$$\mu_3(v) >^{lex} \min_{(v,w)\in E} \mu_3(w)$$

$$\mu_3(v) \geqslant^{lex} \min_{(v,w)\in E} \mu_3(w)$$

$$\begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v) >^{lex} \min_{(v,w)\in E} \begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v)$$

$v \in V_\triangle$

$$\mu_3(v) >^{lex} \max_{(v,w)\in E} \mu_3(w) \quad \text{if } p(v) = 3$$

$$\mu_3(v) \geqslant^{lex} \max_{(v,w)\in E} \mu_3(w) \quad \text{if } p(v) = 2$$

$$\begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v) >^{lex} \max_{(v,w)\in E} \begin{pmatrix}\mu_3\\ \mu_1\end{pmatrix}(v) \quad \text{if } p(v) = 1$$

SMALL PROGRESS MEASURES

THEOREM

There is a winning strategy for $\Box$ (from every vertex)

iff

there are $\mu_1, \mu_3, \ldots, \mu_{d-1} : V \to \mathbb{N}$, s.t.

- $\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(v) \geq^{\text{lex}} \min_{(v,w)\in E} \begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(w)$ $\quad >^{\text{lex}}$ if $p(v)$ is odd! $\quad$ for $v \in V_\Box$

- $\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(v) \geq^{\text{lex}} \max_{(v,w)\in E} \begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_{p(v)} \end{pmatrix}(w)$ $\quad >^{\text{lex}}$ if $p(v)$ is odd! $\quad$ for $v \in V_\Diamond$

and $\mu_k : V \to \{0, 1, 2, \ldots, n_k\}$ where $n_k = \left| p^{-1}(k) \right|$

Exercise 14

Prove the small progress measure theorem.

Hint: Use positional determinacy of parity games and induction on the size of the winning set similar to that in the proof of correctness of the divide-and-conquer algorithm.

SMALL PROGRESS MEASURES

LEMMA

The number of tuples

$$\begin{pmatrix} \mu_{d-1} \\ \vdots \\ \mu_1 \end{pmatrix} \in \mathbb{N}^{d/2}$$

s.t. $\mu_k \in \{0, 1, 2, \ldots, n_k\}$ where $n_k = \left| p^{-1}(k) \right|$

is $\boxed{\leq \left(\frac{n}{d/2}\right)^{d/2}}$

$$= O\left(n^{\frac{d}{2}}\right)$$

Progress Measure Lifting Algorithm

1. Start with $\mu : V \ni v \mapsto (0, 0, \ldots, 0)$

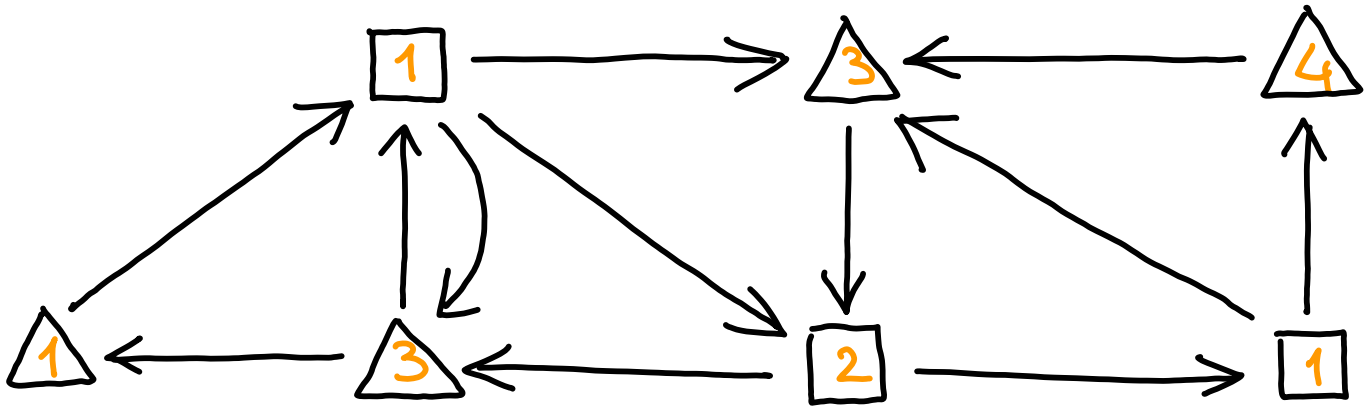
2. <u>While</u> μ **violates the progress inequality** at some $v \in V$, lift $\mu(v)$ (minimally) so that it doesn't

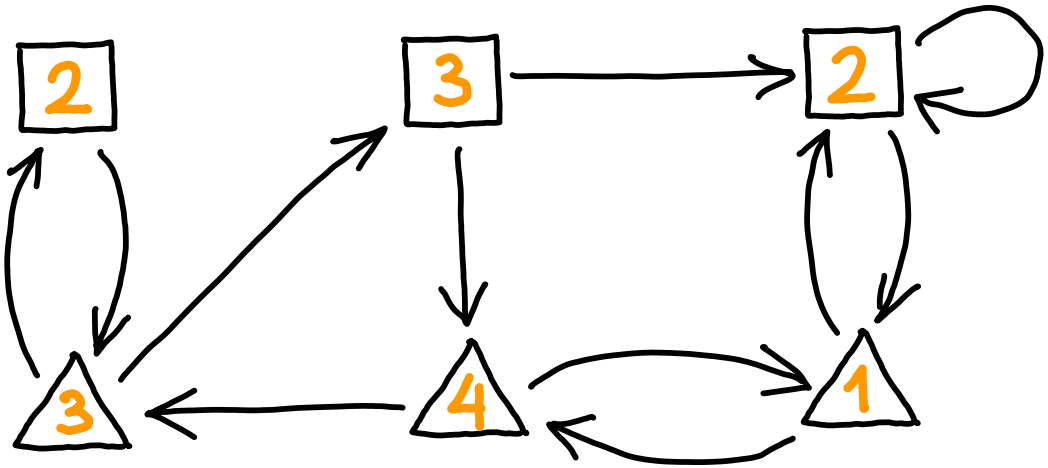
Running time: $\leqslant dm \cdot n^{\frac{d}{2}}$

$= O\left(n^{\frac{d}{2} + O(1)}\right)$

Exercise 15

Run the progress-measure-lifting algorithm on the following parity games:





Better algorithms

Many priorities

$$d = \Omega\left(n^{\frac{1}{2}+\varepsilon}\right)$$

Randomized	Deterministic
Expected $n^{O(\sqrt{n})}$	$n^{O(\sqrt{n})}$

Few priorities

$$d = O(n^{1/2})$$

Randomized	Deterministic
$O\left(n^{\frac{d}{2}+O(1)}\right)$	$O\left(n^{\frac{d}{3}+O(1)}\right)$

Smaller Progress Measures for Dominions

Thm If $D \subseteq V$ is a dominion of size $\leq \ell$, then there is a progress measure $\mu : V \to M_G^{\infty}(\ell)$, s.t. $\mu(D) \subseteq M_G(\ell)$,

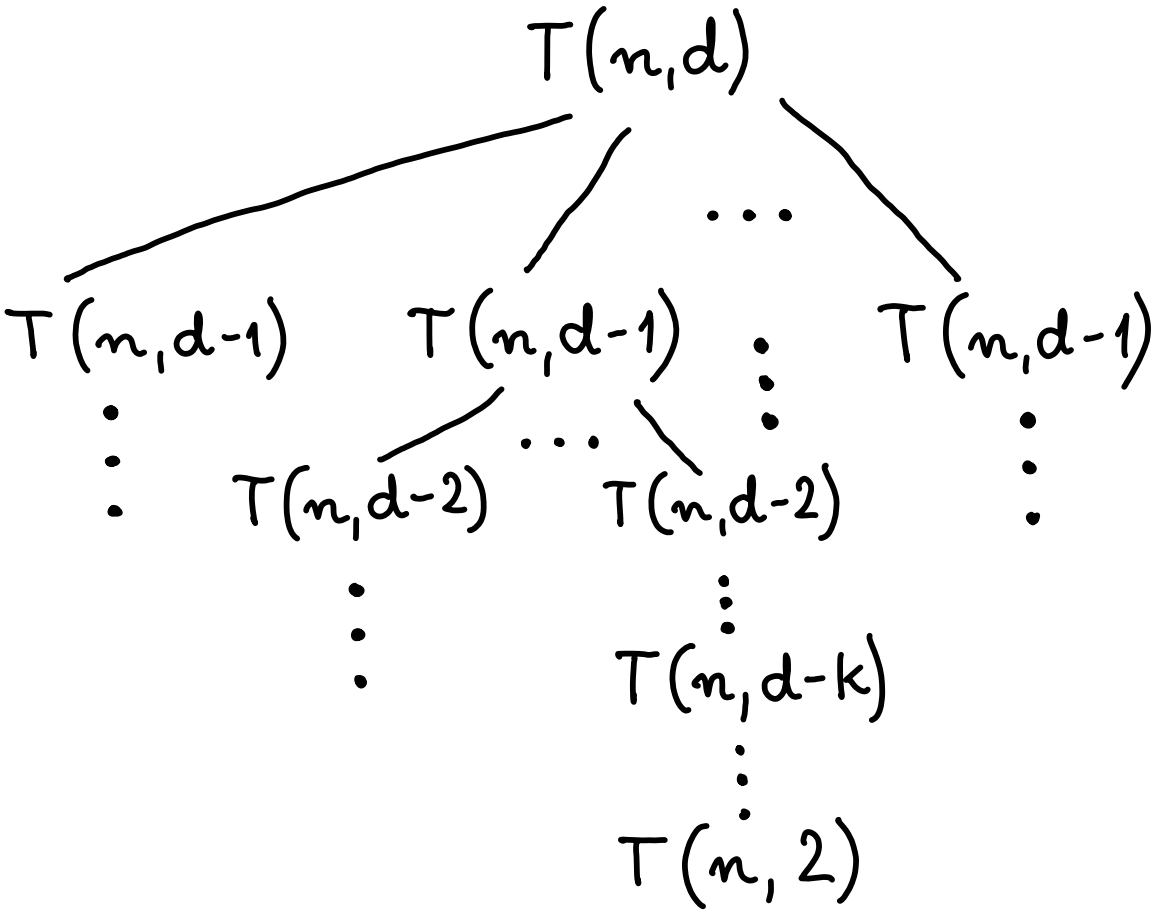
where $$M_G(\ell) = \left\{ \mu \in \mathbb{N}^{d/2} : \sum_{k=1}^{d/2} \mu_k \leq \ell \right\}$$

Cor

1. *All* dominions of *size* $\leq \ell/2$ can be found in *time* $O\binom{\ell + \frac{d}{2}}{\frac{d}{2}}$

2. *All* dominions of *size* $\leq n^{2/3}$ can be found in *time* $O(n^{d/3})$

The Recursion Tree for Schewe's Algorithm



# children	Dominion preprocessing
$\leq n^{1/3}$	$n^{d/3}$
$\leq n^{2/3}$	$n^{(d-1)/3}$
$\leq n^{3/3}$	$n^{(d-2)/3}$
$\leq n^{(k+1)/3}$	$n^{(d-k)/3}$
$\leq n^{(d-1)/3}$	

$$O\left(n^{\frac{d}{3}+O(1)}\right)$$

Exercise 16

- Prove that

$$\underbrace{\left|\left\{\mu \in \mathbb{N}^{d/2} : \sum_{k=1}^{d/2} \mu_k \leq \ell\right\}\right|}_{M_G(\ell)} = \binom{\ell + \frac{d}{2}}{\frac{d}{2}} = \prod_{i=1}^{d/2} \frac{\ell + i}{i}$$

- Argue that if $\ell = 2 \cdot \lceil n^{2/3} \rceil$ then $|M_G(\ell)| \leq n^{d/3}$ for all $d \geq 8$.

- Prove that if the dominion returned by the progress measure lifting algorithm, using $M_G(\ell)$, is of size $\leq \ell/2$ then there is no other dominion of size $\leq \ell/2$ in G.

EXERCISE 17

- Argue that the running time of Schewe's algorithm can be characterized by the recurrence:

$$\begin{cases} T(n,1) = O(n) \\ T(n,d) \leq T(n,d-1) + T\left(n - \lceil n^{2/3} \rceil, d\right) + O\left(n^{d/3 + O(1)}\right) \end{cases}$$

and hence also by:

$$\begin{cases} T(n,1) = O(n) \\ T(n,d) \leq n^{1/3} \cdot T(n,d-1) + O\left(n^{d/3 + O(1)}\right) \end{cases}$$

- Prove that $T(n,d) = O\left(n^{d/3 + O(1)}\right)$.

Exercise 18

- Give algorithms with the smallest asymptotic worst-case running time bound as you can find, for parity games with 2, 3, 4, 5, 6, 7, and 8 priorities.

PLAN

1. Motivating example
2. Games on graphs
3. Reachability / safety games
4. Büchi / co-Büchi games
5. Parity games
6. Better algorithms for parity games
7. Search complexity of computing optimal strategies

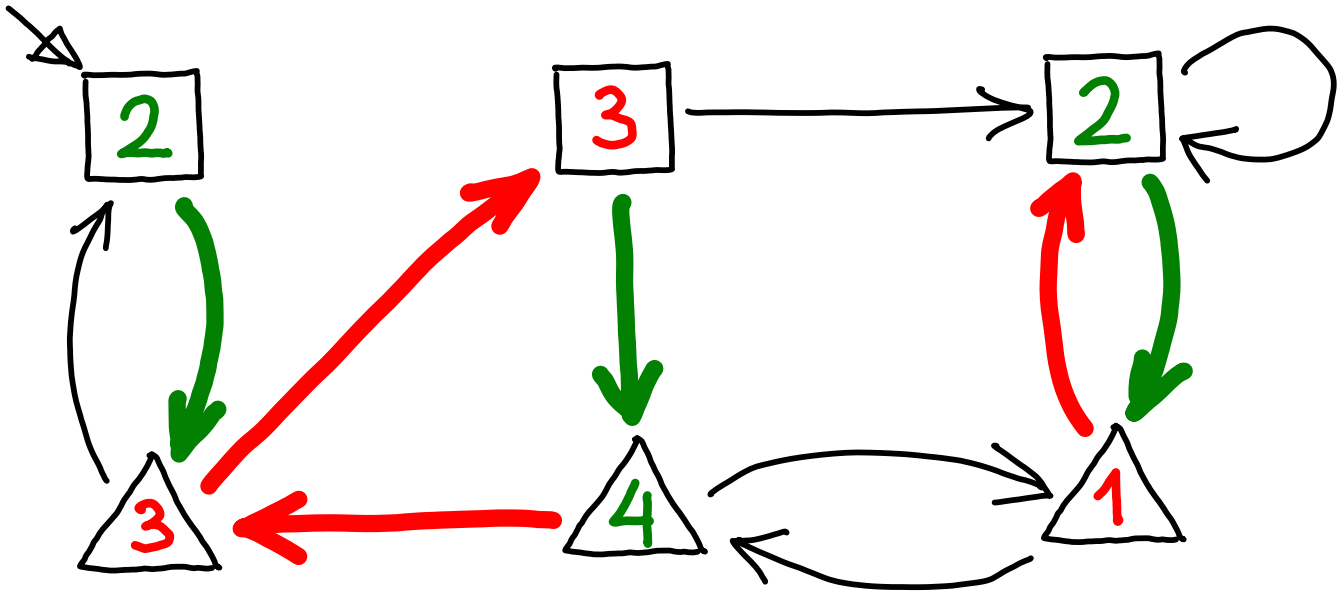
Parity Games

Players:

n vertices, m edges, d priorities

Even = □ = 0

Odd = △ = 1



Winner of an infinite play:

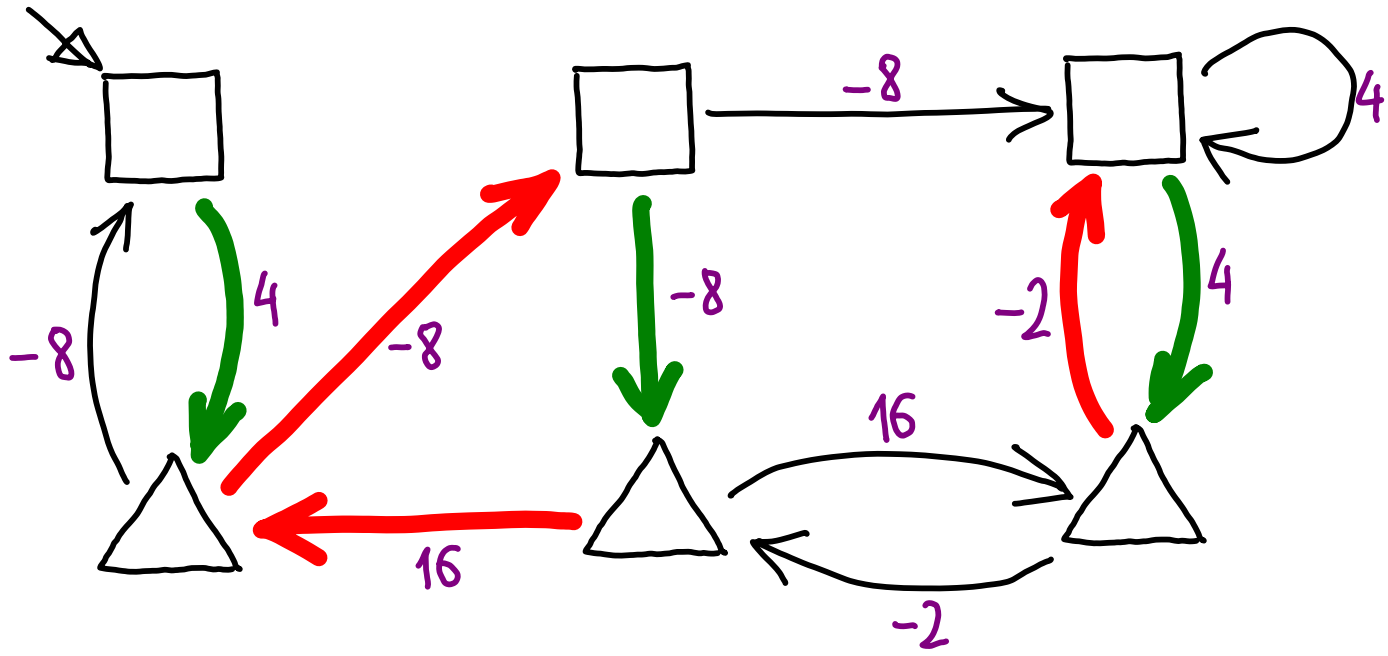
parity of the highest priority occurring infinitely often

Mean-Payoff Games

Players:

MAX = □

MIN = △



Winner of an infinite play $\pi = \langle v_0, v_1, v_2, \dots \rangle$

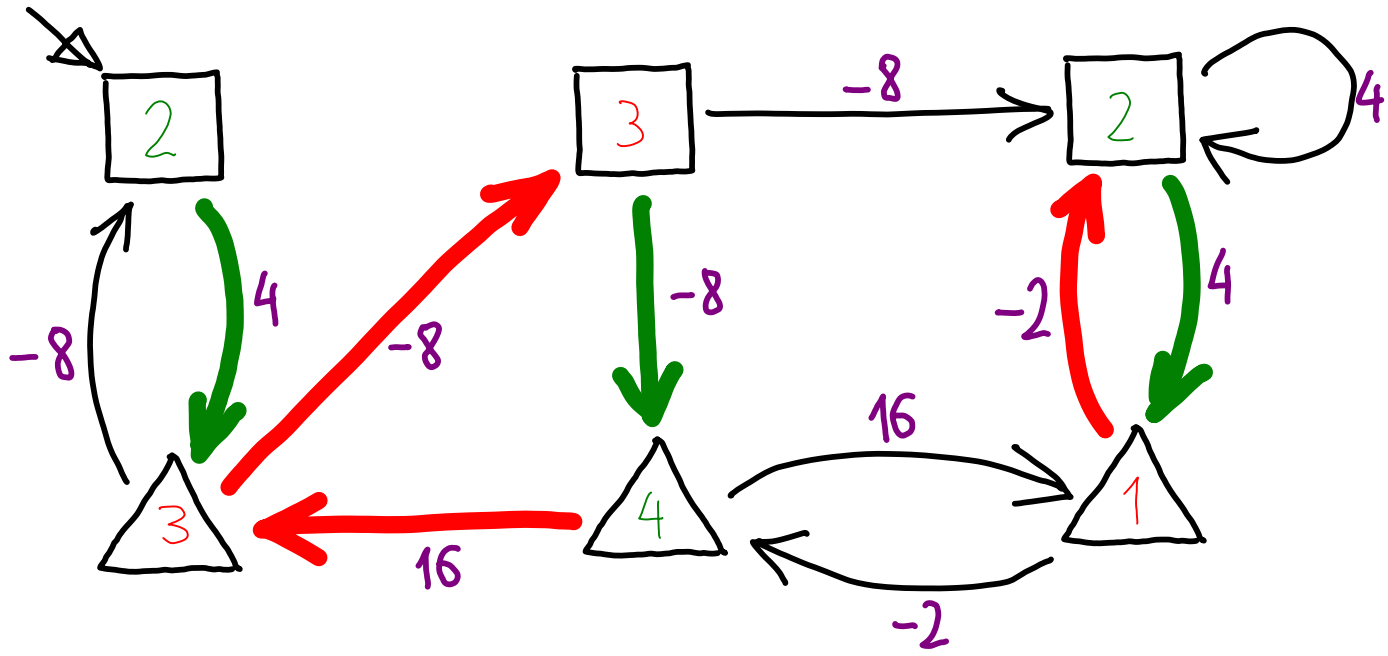
MAX if $\lim_{n \to \infty} \left(\frac{1}{n} \cdot \sum_{i=0}^{n-1} r(v_i, v_{i+1}) \right) \geqslant 0$

Mean-Payoff Games

Players:

Max = □

Min = △



$$(v, w) \longmapsto (-n)^{p(v)}$$

here 2 was good enough

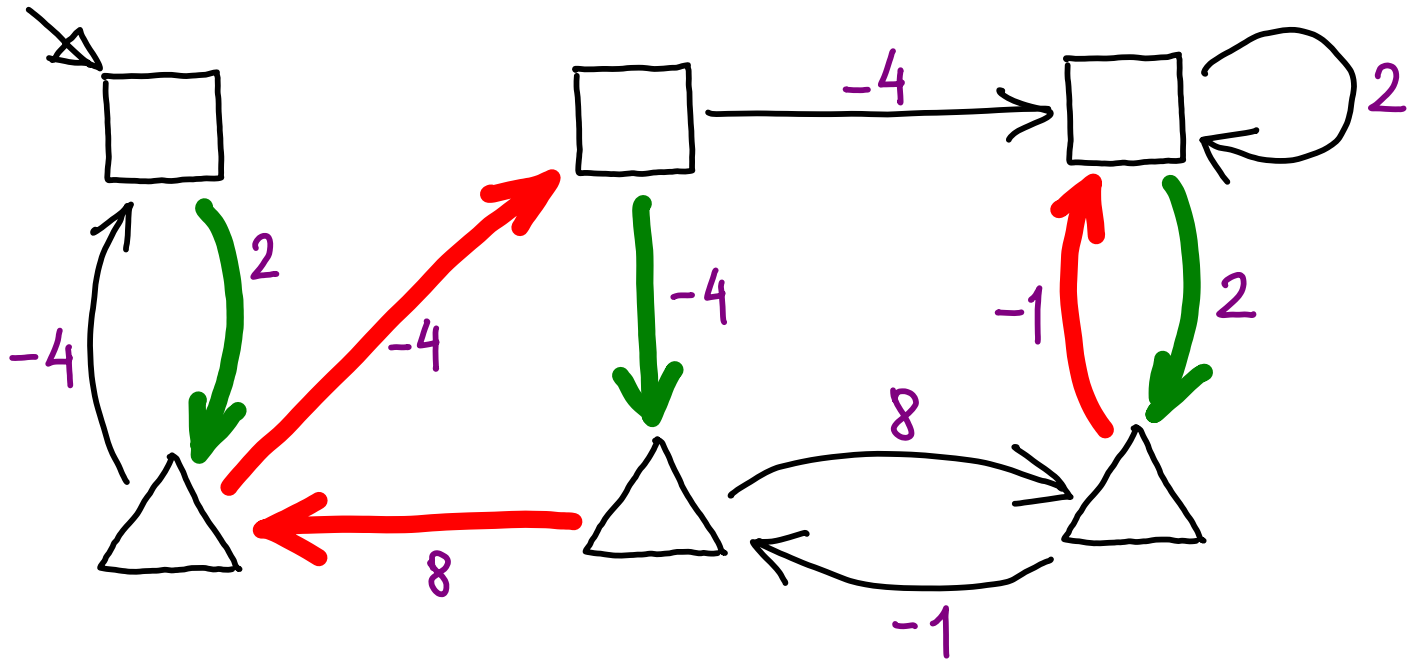
Discounted Games

Players:

MAX = □

MIN = △

$0 < \delta < 1$



Winner of an infinite play $\pi = \langle v_0, v_1, v_2, \dots \rangle$

MAX if $\sum_{i=0}^{\infty} \delta^i \cdot r(v_i, v_{i+1}) \geq 0$

Payoff, Value, Determinacy

$$\pi = \langle s_0, s_1, s_2, \ldots \rangle$$

Mean payoff : $$\mathcal{A}(\pi) = \lim_{n \to \infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} r(s_i, s_{i+1})$$

Discounted total: $$\mathcal{D}_\delta(\pi) = \sum_{i=0}^{\infty} \delta^i \cdot r(s_i, s_{i+1})$$

Lower value: $$Val_*(s) = \sup_{\chi \in \Sigma_{Max}} \inf_{\mu \in \Sigma_{Min}} \mathcal{P}(Play(s, \mu, \chi))$$

Upper value: $$Val^*(s) = \inf_{\mu \in \Sigma_{Min}} \sup_{\chi \in \Sigma_{Max}} \mathcal{P}(Play(s, \mu, \chi))$$

Determinacy: $$Val(s) = Val_*(s) = Val^*(s)$$

COMPARING COMPUTATIONAL COMPLEXITY OF INFINITE GAMES

THM [PURI 1995]

There is a polynomial-time reduction
from parity games to mean-payoff games

THM [1960's]

There is a polynomial-time reduction
from mean-payoff games to discounted games

THM [HARDY LITTLEWOOD 1930's]

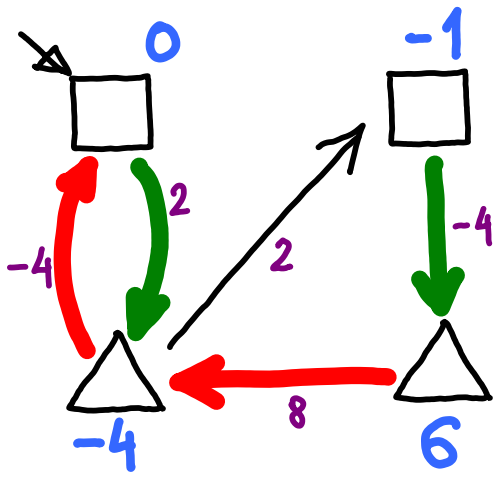
If $\lim_{n\to\infty}\left[\frac{1}{n}\cdot\sum_{i=0}^{n-1} a_i\right]$ exists

then $\lim_{n\to\infty}\left[\frac{1}{n}\cdot\sum_{i=0}^{n-1} a_i\right] = \lim_{\delta\nearrow 1}\left[(1-\delta)\cdot\sum_{i=0}^{\infty}\delta^i\cdot a_i\right]$

Optimality equations for discounted games

Opt(G):

$$v_s = \begin{cases} \max\limits_{(s,t)\in E} \left(r_{(s,t)} + \delta \cdot v_t \right) & \text{if } s \in S_{Max} \\ \min\limits_{(s,t)\in E} \left(r_{(s,t)} + \delta \cdot v_t \right) & \text{if } s \in S_{Min} \end{cases}$$



$\delta = \frac{1}{2}$

Lemma If $V \vDash Opt(G)$ then $V = Val^G$
and positional strategies choosing optimal successor are optimal

Optimality equations for discounted games

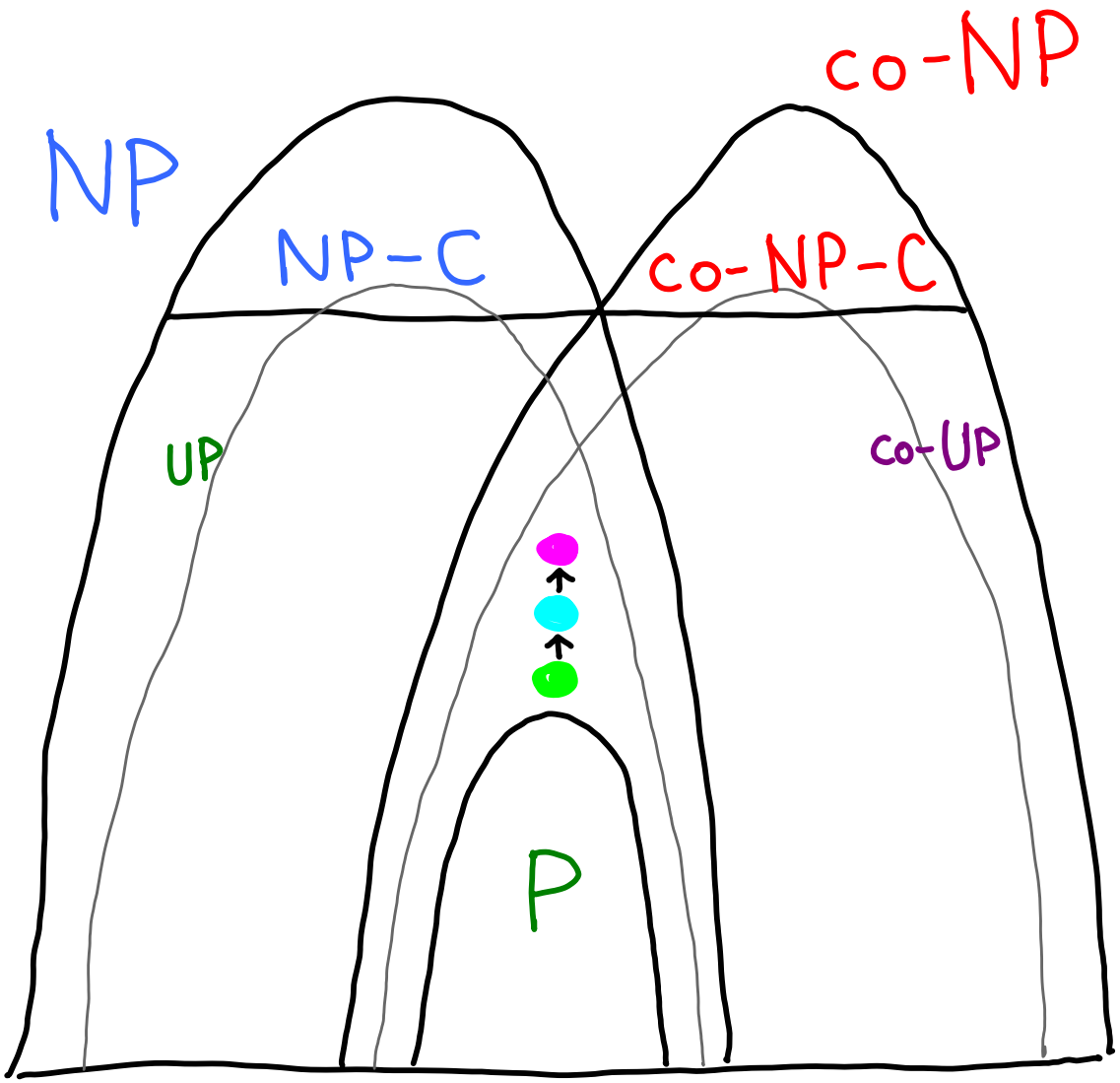
$$F : v_s \longmapsto \begin{cases} \max\limits_{(s,t)\in E} \left(r_{(s,t)} + \delta \cdot v_t \right) & \text{if } s \in S_{Max} \\ \min\limits_{(s,t)\in E} \left(r_{(s,t)} + \delta \cdot v_t \right) & \text{if } s \in S_{Min} \end{cases}$$

Fact $F : \mathbb{R}^n \to \mathbb{R}^n$ is a contraction

Corollary

- F has a **unique fixed point**, i.e., Opt(G) has a solution
- Discounted, mean-payoff, and parity games:
  - are positionally determined
  - are in UP ∩ co-UP

COMPUTATIONAL COMPLEXITY OF SOLVING INFINITE GAMES

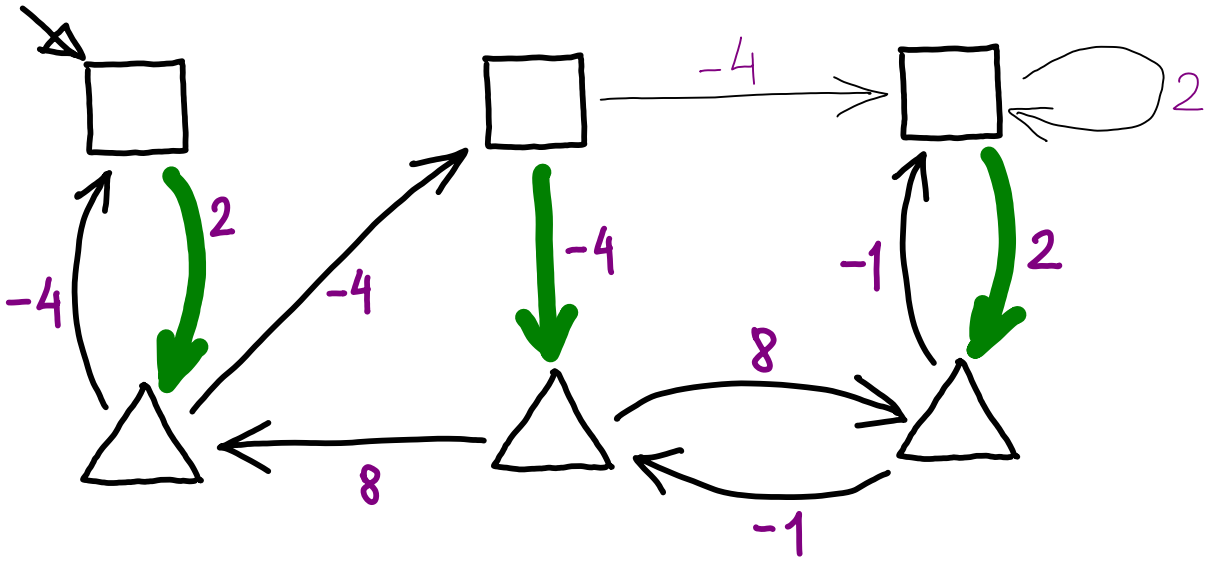


DISCOUNTED GAMES

AVERAGE-REWARD GAMES

PARITY GAMES

2-player Strategy Improvement



$\delta = \frac{1}{2}$

0. Pick $\chi \in \Pi_{max}$

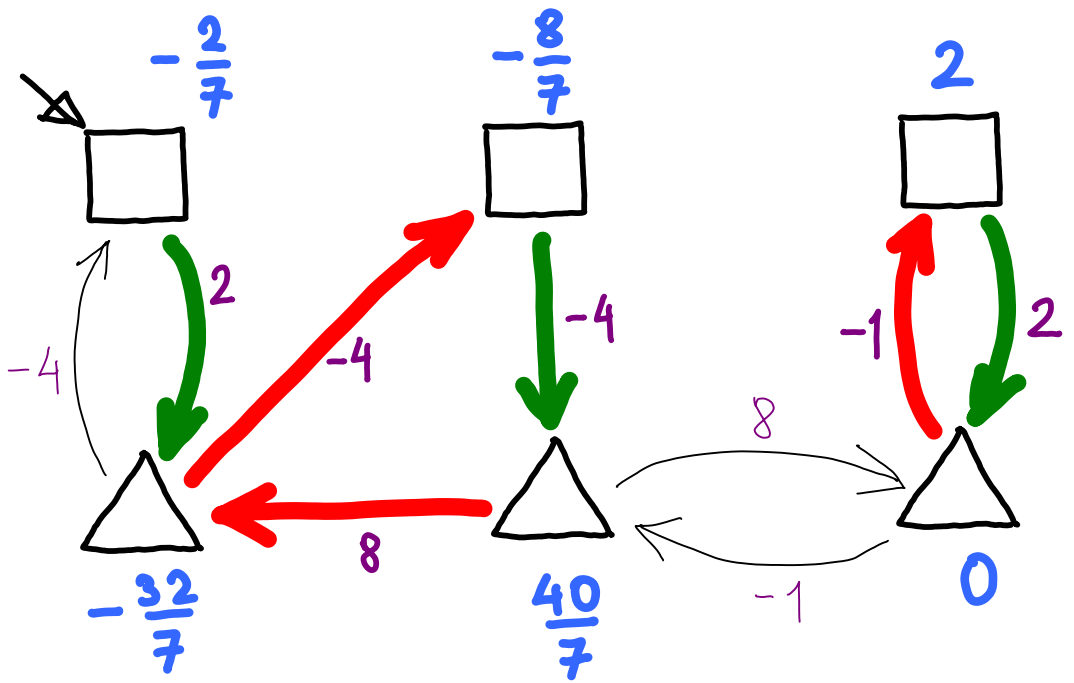
1. Find $V : S \to \mathbb{R}$, such that $V \models \mathrm{OPT}(\Gamma^{\chi})$

2. If $V \not\models \mathrm{OPT}(\Gamma)$
   then $\chi := \mathrm{Improve}(\chi, V)$; goto 1.

Compute the best response μ to χ for Min

Improve locally w.r.t. the best response

2-player Strategy Improvement



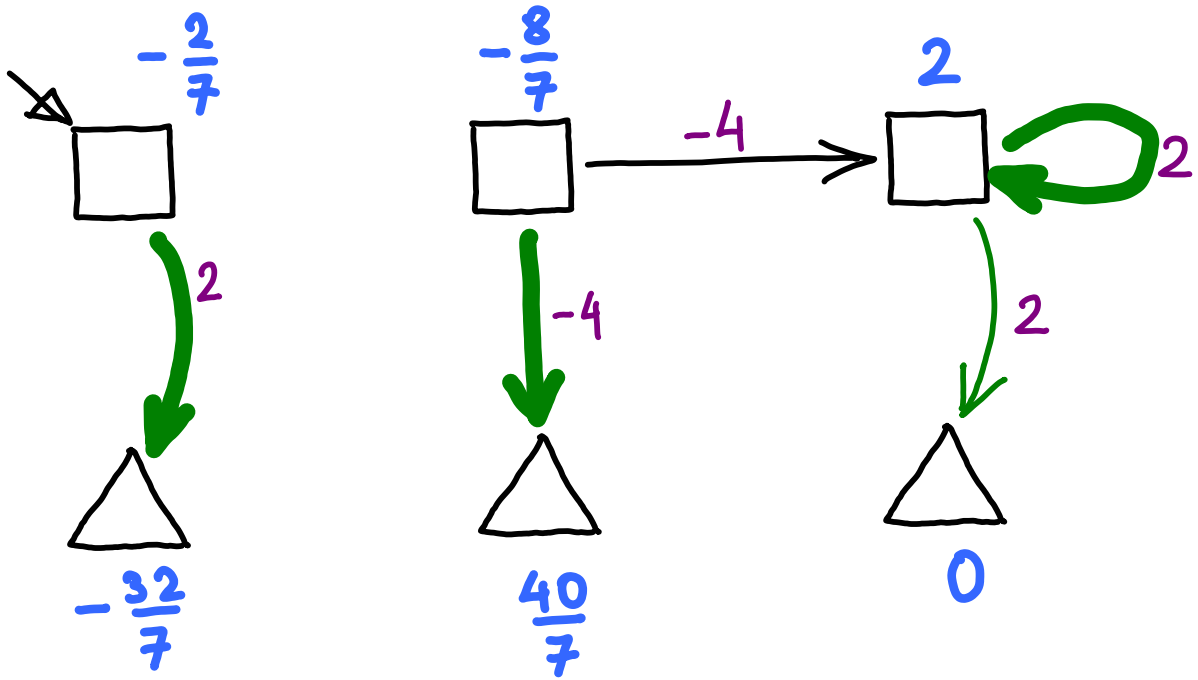
$\delta = \frac{1}{2}$

0. Pick $\chi \in \Pi_{Max}$
1. **Find $V: S \to \mathbb{R}$, such that $V = \mathrm{OPT}(G^{\chi})$**
2. If $V \neq \mathrm{OPT}(G)$
   then $\chi := \mathrm{Improve}(\chi, V)$; goto 1.

Compute the best response μ to χ for Min

Improve locally w.r.t. the best response

2-player Strategy Improvement



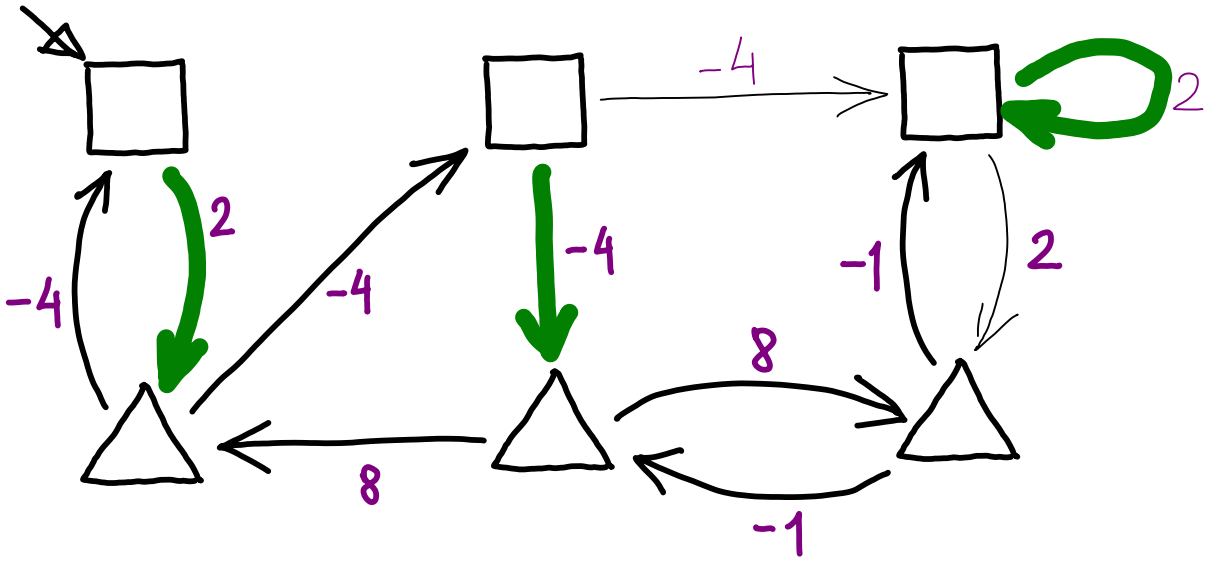
$\delta = \frac{1}{2}$

0. Pick $\chi \in \Pi_{Max}$
1. Find $V : S \to \mathbb{R}$, such that $V \models Opt(G^{\chi})$
2. **If** $V \not\models Opt(G)$
   then $\chi := Improve(\chi, V)$; goto 1.

Compute the best response μ to χ for Min

Improve locally w.r.t. the best response

2-player Strategy Improvement



$\delta = \frac{1}{2}$

0. Pick $\chi \in \Pi_{Max}$
1. **Find $V: S \to \mathbb{R}$, such that $V = \text{Opt}(G^{\chi})$**
2. If $V \neq \text{Opt}(G)$
   then $\chi := \text{Improve}(\chi, V)$; goto 1.

Compute the best response μ to χ for Min

Improve locally w.r.t. the best response

Correctness of 2-player Strategy Improvement

Improvement Lemma

If $V \vDash \mathrm{Opt}(G^{x})$,

$x' = \mathrm{Improve}(x, V)$, and

$V' \vDash \mathrm{Opt}(G^{x'})$

then $V' \geq V$, and

$V' > V$ if $x' \neq x$.

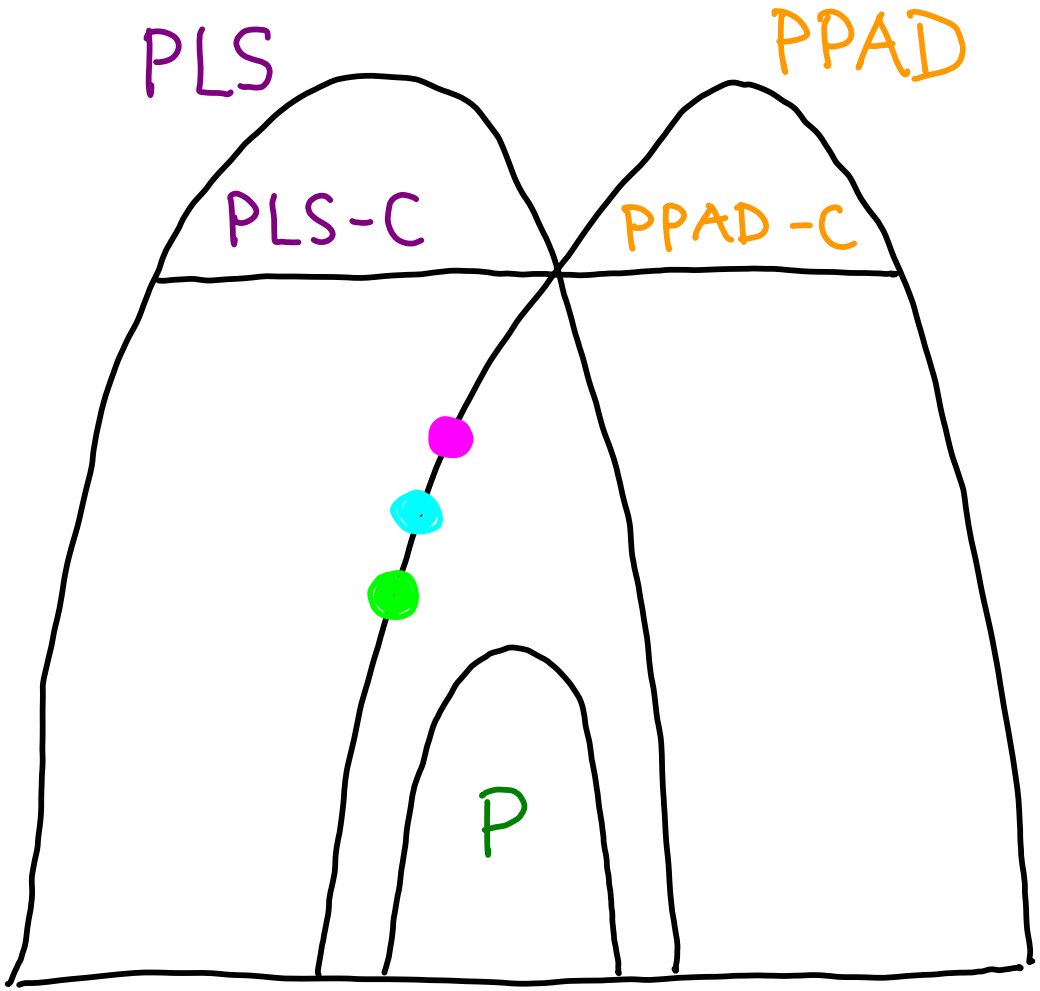
Termination Lemma

Strategy improvement terminates (in $\leq |\Pi_{\mathrm{Max}}|$ steps)

and returns $V \vDash \mathrm{Opt}(G)$.

Corollary Computing the value of discounted, average-reward, and parity games is in PLS

COMPUTATIONAL COMPLEXITY OF SOLVING INFINITE GAMES



DISCOUNTED GAMES

AVERAGE-REWARD GAMES

PARITY GAMES

Linear Complementarity Problem

Given: $M \in \mathbb{R}^{n \times n}$

$q \in \mathbb{R}^n$

Find: $z, w \in \mathbb{R}^n$

such that

linear
$$\begin{cases} z \geq 0 \\ w \geq 0 \\ w = Mz + q \end{cases}$$

complementarity $\{ \; z \perp w$ i.e., $z^T \cdot w = 0$

Fact

If $q \geq 0$

then $(\underset{z}{0}, \underset{w}{q}) \models LCP(M, q)$

Complementary cones of LCP

$$z \geq 0 \quad \perp \quad w = Mz + q \geq 0$$

iff

$$z \geq 0 \perp w \geq 0 \quad \text{and} \quad q = -Mz + Iw$$

iff

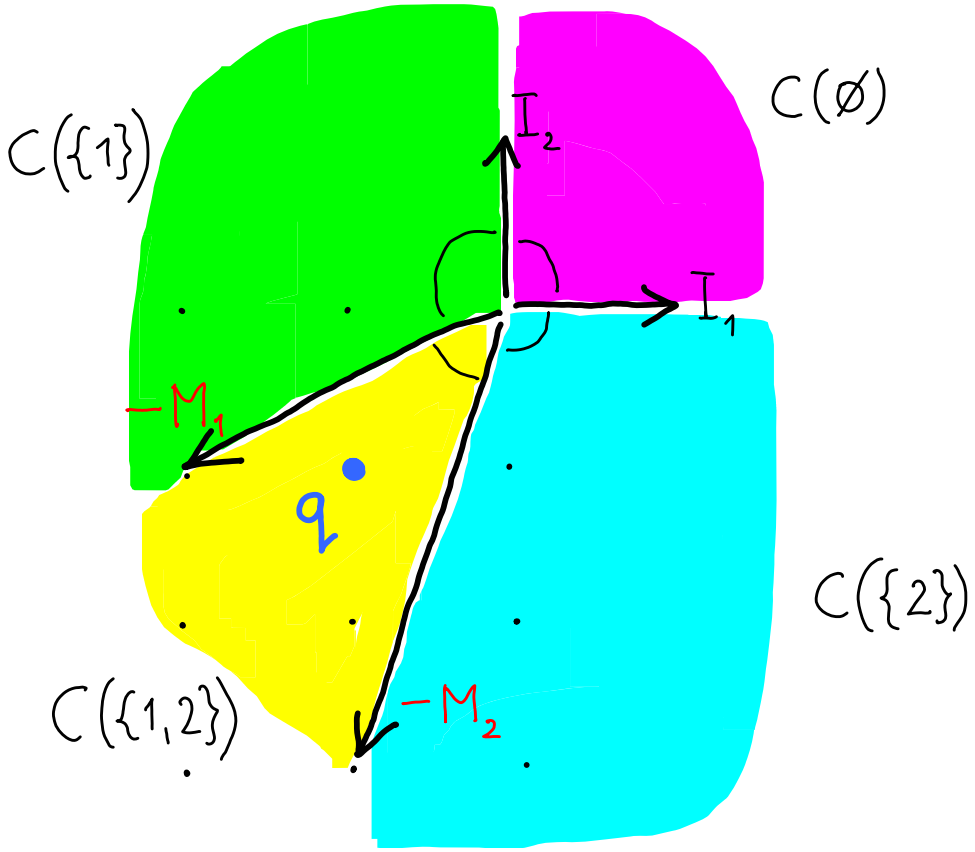
$$q \in C(\alpha) = \text{cone}\left(\{-M_k : k \in \alpha\} \cup \{I_k : k \notin \alpha\}\right)$$

for some $\alpha = \{k : w_k = 0\} \subseteq \{1, \ldots, n\}$

Complementary cones of a matrix

$$z \geqslant 0 \perp w \geqslant 0 \quad \text{and} \quad q = -Mz + Iw$$

$$M = \begin{pmatrix} \overset{M_1}{2} & \overset{M_2}{1} \\ 1 & 3 \end{pmatrix} \qquad q = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$$



$q \in C(\{1,2\})$

$\Downarrow$

$(z_1, z_2, 0, 0) \in LCP(M, q)$

where $q = -Mz$

P-MATRICES

DEF $M \in \mathbb{R}^{n \times n}$ is a **P-matrix**
if all its **principal minors** are **positive**

THM M is a P-matrix
iff $LCP(M,q)$ has a **unique solution** for every $q \in \mathbb{R}^n$.

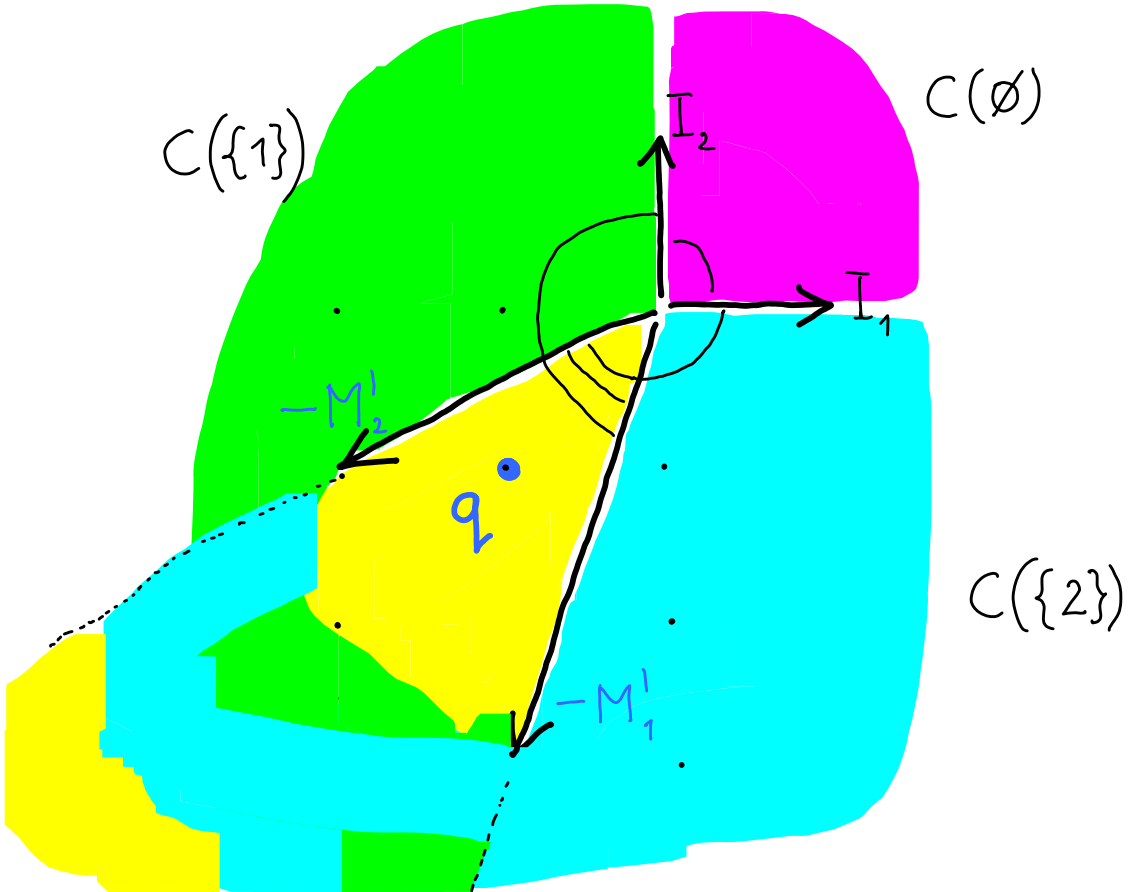
EXAMPLE $M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$ $M' = \begin{pmatrix} 1 & 2 \\ 3 & 1 \end{pmatrix}$

- M is a P-matrix: $\det(M_{11}) = 2$
  $\det(M_{22}) = 3$
  $\det(M) = 5$
- M' is not a P-matrix: $\det(M') = -5$

Complementary cones of a non P-matrix

$$w \geqslant 0 \perp z \geqslant 0 \quad \text{and} \quad q = -Mz + Iw$$

$$M' = \begin{pmatrix} \overset{M'_1}{1} & \overset{M'_2}{2} \\ 3 & 1 \end{pmatrix} \qquad q = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$$



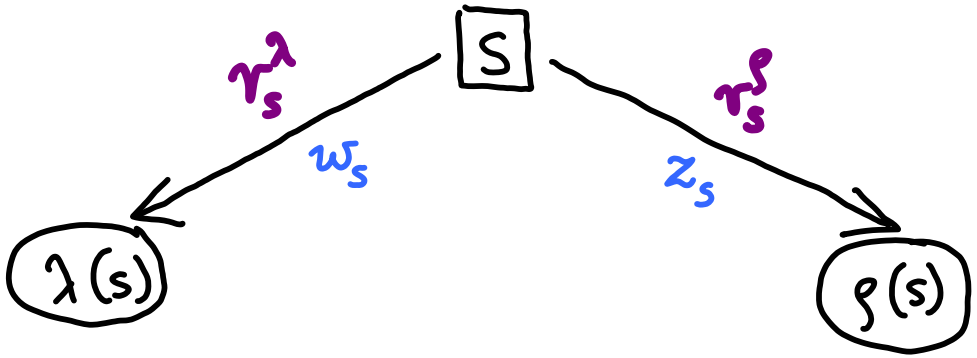
$q \in C(\{1,2\})$

$q \in C(\{1\})$

$q \in C(\{2\})$

Optimality equations and comlementarity

$$v_s = \max \left\{ r_s^\lambda + \delta \cdot v_{\lambda(s)} \, , \; r_s^\rho + \delta \cdot v_{\rho(s)} \right\}$$



slacks

$$v_s = w_s + r_s^\lambda + \delta \cdot v_{\lambda(s)}$$

$$v_s = z_s + r_s^\rho + \delta \cdot v_{\rho(s)}$$

$$z_s, w_s \geq 0 \qquad z_s \cdot w_s = 0$$

complementarity

OPTIMALITY EQUATIONS AND COMLEMENTARITY

$$s \in S_{Max}: \quad v_s = \max\left\{ r_s^\lambda + \delta \cdot v_{\lambda(s)} \,,\; r_s^\rho + \delta \cdot v_{\rho(s)} \right\}$$

$$s \in S_{Min}: \quad v_s = \min\left\{ r_s^\lambda + \delta \cdot v_{\lambda(s)} \,,\; r_s^\rho + \delta \cdot v_{\rho(s)} \right\}$$

Replace max/min with slacks and complementarity

$$s \in S_{Max}: \quad \begin{aligned} v_s &= w_s + r_s^\lambda + \delta \cdot v_{\lambda(s)} \\ v_s &= z_s + r_s^\rho + \delta \cdot v_{\rho(s)} \end{aligned}$$

slacks

$$s \in S_{Min}: \quad \begin{aligned} v_s &= -w_s + r_s^\lambda + \delta \cdot v_{\lambda(s)} \\ v_s &= -z_s + r_s^\rho + \delta \cdot v_{\rho(s)} \end{aligned}$$

$$s \in S: \quad w_s \geqslant 0 \;\perp\; z_s \geqslant 0$$

complementarity

Reduction to LCP: rewrite

$s \in S_{max}$:

$$v_s = w_s + r_s^\lambda + \delta \cdot v_{\lambda(s)}$$
$$v_s = z_s + r_s^\rho + \delta \cdot v_{\rho(s)}$$

slacks

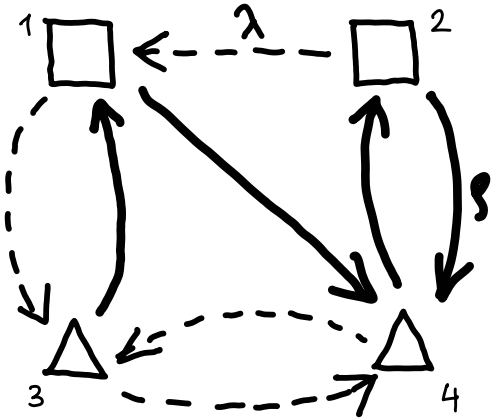
$s \in S_{min}$:

$$v_s = -w_s + r_s^\lambda + \delta \cdot v_{\lambda(s)}$$
$$v_s = -z_s + r_s^\rho + \delta \cdot v_{\rho(s)}$$

$$\rightsquigarrow$$

$$(\hat{I} - \delta \cdot \hat{T}^\lambda) v = w + \hat{I} r^\lambda$$
$$(\hat{I} - \delta \cdot \hat{T}^\rho) v = z + \hat{I} r^\rho$$

Example



$$\hat{I} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$\hat{T}^\rho = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix}$$

Reduction to LCP: eliminate v

$$\left(\hat{I} - \delta \cdot \widehat{T^{\lambda}}\right) v = w + \hat{I} r^{\lambda}$$

$$\left(\hat{I} - \delta \cdot \widehat{T^{s}}\right) v = z + \hat{I} r^{s}$$

Eliminate v:

$$w + \hat{I} r^{\lambda} = \underbrace{\left(\hat{I} - \delta \cdot \widehat{T^{\lambda}}\right) \cdot \left(\hat{I} - \delta \cdot \widehat{T^{s}}\right)^{-1}}_{M} \cdot \left(z + \hat{I} r^{s}\right)$$

$$w = Mz + q$$

$$w \geq 0 \quad \perp \quad z \geq 0$$

$$M = \left(\hat{I} - \delta \cdot \widehat{T^{\lambda}}\right) \cdot \left(\hat{I} - \delta \cdot \widehat{T^{s}}\right)^{-1}$$

$$q = M\left(\hat{I} r^{s}\right) - \left(\hat{I} r^{\lambda}\right)$$

Reduction to LCP: eliminate v

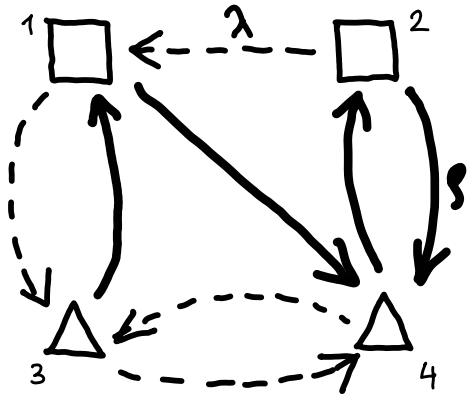
$$w = Mz + q$$

$$w \geq 0 \quad \perp \quad z \geq 0$$

$$M = (\hat{I} - \delta \cdot \widehat{T^{\lambda}}) \cdot (\hat{I} - \delta \cdot \widehat{T^{s}})^{-1}$$

$$q = M(\hat{I} r^{s}) - (\hat{I} r^{\lambda})$$

Example



$$\hat{I} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$\widehat{T^{s}} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix}$$

$$(\hat{I} - \delta \cdot \widehat{T^{s}}) = \begin{pmatrix} 1 & 0 & 0 & -\delta \\ 0 & 1 & 0 & -\delta \\ \delta & 0 & -1 & 0 \\ 0 & \delta & 0 & -1 \end{pmatrix}$$

STRICTLY DIAGONALLY DOMINANT MATRICES

FACT [Levy-Desplanques]

If $A \in \mathbb{R}^{n \times n}$ is strictly diagonally dominant, then A is non-singular.

FACT $(\hat{I} - \delta \cdot \hat{T}^{\lambda})$ and $(\hat{I} - \delta \cdot \hat{T}^{\beta})$ are s.d.d.

COROLLARY $M = (\hat{I} - \delta \hat{T}^{\lambda}) \cdot (\hat{I} - \delta \hat{T}^{\beta})^{-1}$ is well-defined

P-MATRICES OF THE FORM $B \cdot C^{-1}$

THM [Johnson-Tsatsomeros '95]

Let $M = B \cdot C^{-1}$, where $B, C \in \mathbb{R}^{n \times n}$

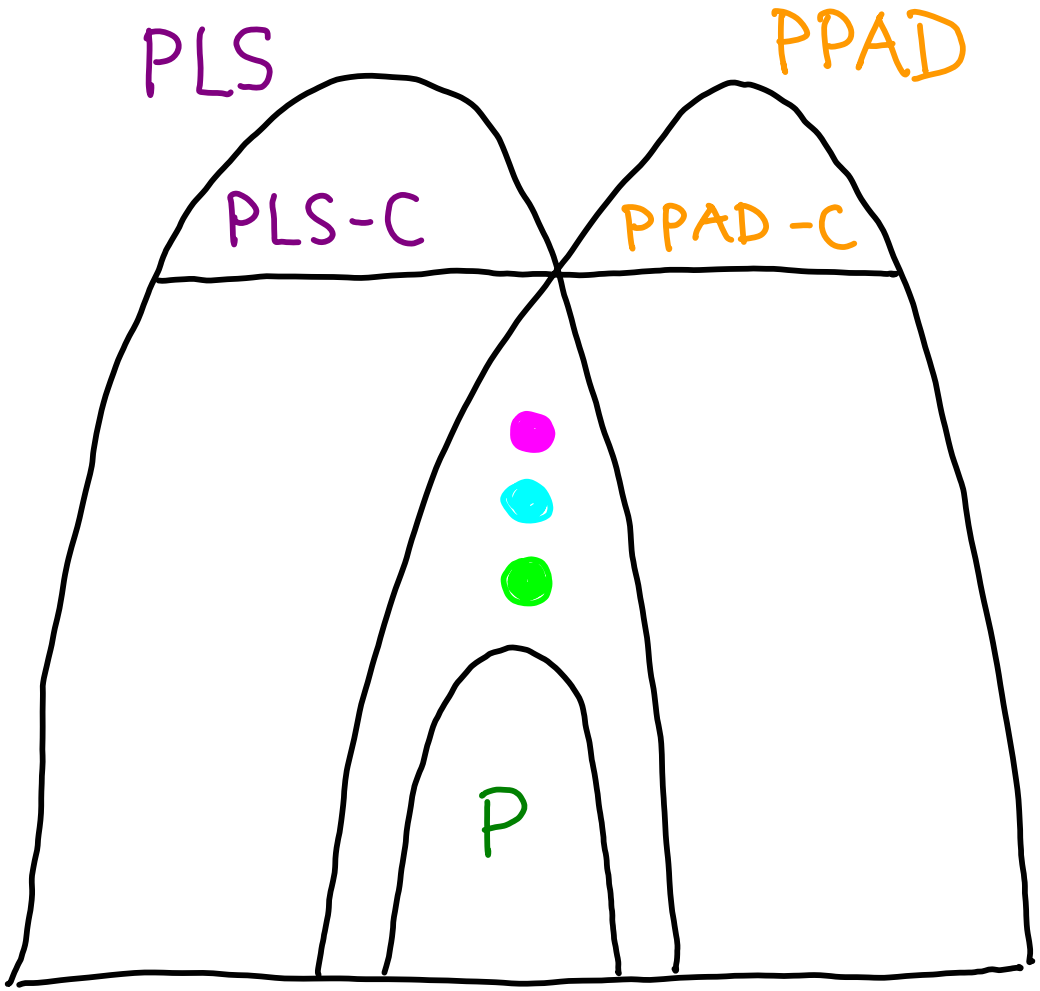
Then, M is a **P-matrix**

if $T \cdot B + (I - T) \cdot C$ is non-singular for all $T \in [0, I]$

COROLLARY

- $M = (\hat{I} - \delta \cdot \hat{T}^{\lambda}) \cdot (\hat{I} - \delta \cdot \hat{T}^{s})^{-1}$ is a **P-matrix.**
- Computing the value of discounted, average-reward, and parity games is in **PPAD**

COMPUTATIONAL COMPLEXITY OF SOLVING INFINITE GAMES



DISCOUNTED GAMES

AVERAGE-REWARD GAMES

PARITY GAMES

What is q?

Lemma

Let v^ρ be the vector of values of strategy pair ρ, i.e.,

$$v^\rho = r^\rho + \delta \cdot T^\rho \cdot v^\rho$$

Then:

$$q = \hat{I}\left(\left(r^\rho + \delta \cdot T^\rho \cdot v^\rho \right) - \left(r^\lambda + \delta \cdot T^\lambda \cdot v^\rho \right) \right)$$

take "right" edge and then follow ρ

take "left" edge and then follow ρ

Algorithm 1: Strategy improvement revisited

Def. $s \in S_{Max}$ is switchable (for strategy pair ρ) if

$$r_s^{\rho} + \delta \cdot v_{\rho(s)}^{\rho} < r_s^{\lambda} + \delta \cdot v_{\lambda(s)}^{\rho}$$

---

1. Start with arbitrary ρ, λ.
2. While there is a switchable vertex (for ρ) do
3. Find ρ' such that:
   a) $\rho' \restriction S_{Max} = \rho \restriction S_{Max}$
   b) no $s \in S_{Min}$ is switchable for ρ'

   } $\rho' \restriction S_{Min}$ is a best response to $\rho \restriction S_{Max}$
4. Switch all (or some) switchable vertices in S_{Max}

ALGORITHM 2: MURTY'S "LEAST-INDEX" METHOD

1. Fix a permutation ("indexing") of states.
2. While there is a switchable vertex do
3. Switch the switchable vertex with **least index**.

THM Murty's algorithm terminates (in $\leq 2^n$ steps).

"Nested" strategy improvement: 

New algorithm: 

Algorithm 3: Cottle-Danzig

1. For $s = 1, 2, \ldots, n$ do
2. If s is switchable then
3. [Drive r_s^ρ until s becomes **indifferent**
4. While driving r_s^ρ,
   switch a vertex in $\{1, 2, \ldots, s-1\}$
   when it becomes **indifferent**
5. Switch s and restore r_s^ρ

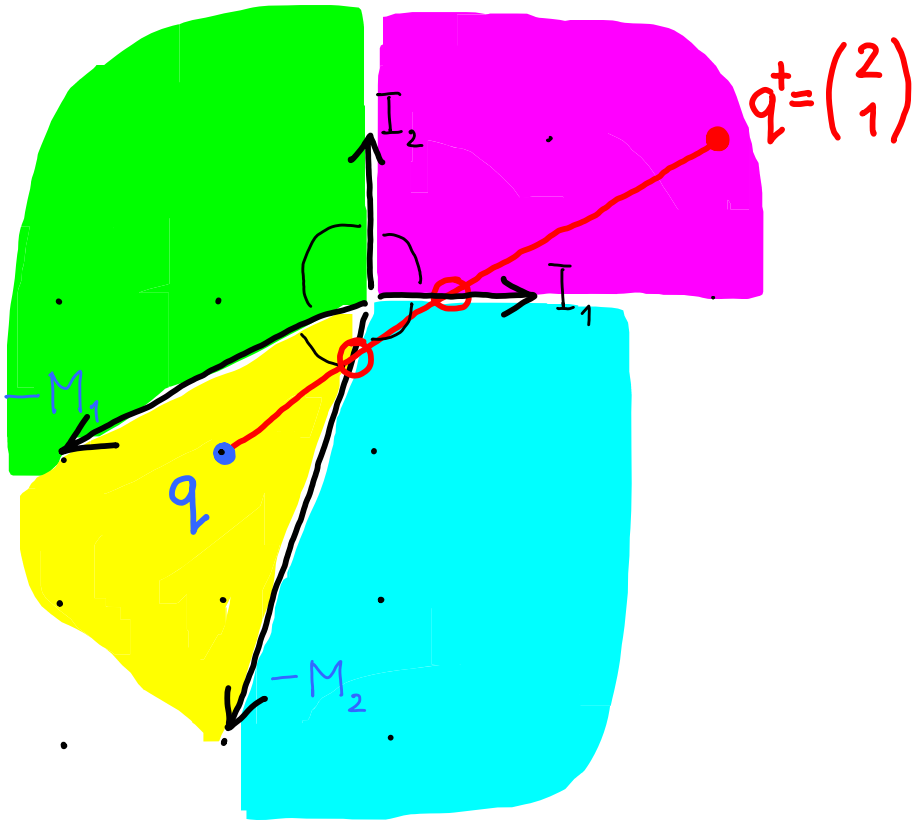
Fact Cottle-Danzig terminates (in $\leq 2^n$ steps)

Algorithm 4 : Lemke

1. Drive r^λ ("linearly"), until no vertex is switchable.
2. Drive r^λ ("linearly") back, until original r^λ is restored.

   While driving r^λ back,
   switch vertices when they become indifferent.

$$q = M(\hat{I} r^s) - (\hat{I} r^\lambda)$$



EXPONENTIAL TIME LOWER BOUNDS

THM [SAVANI, VON STENGEL 2004]

Lemke-Howson algorithm needs exponential time to find a Nash equilibrium

THM [FEARNLEY, J., SAVANI 2009]

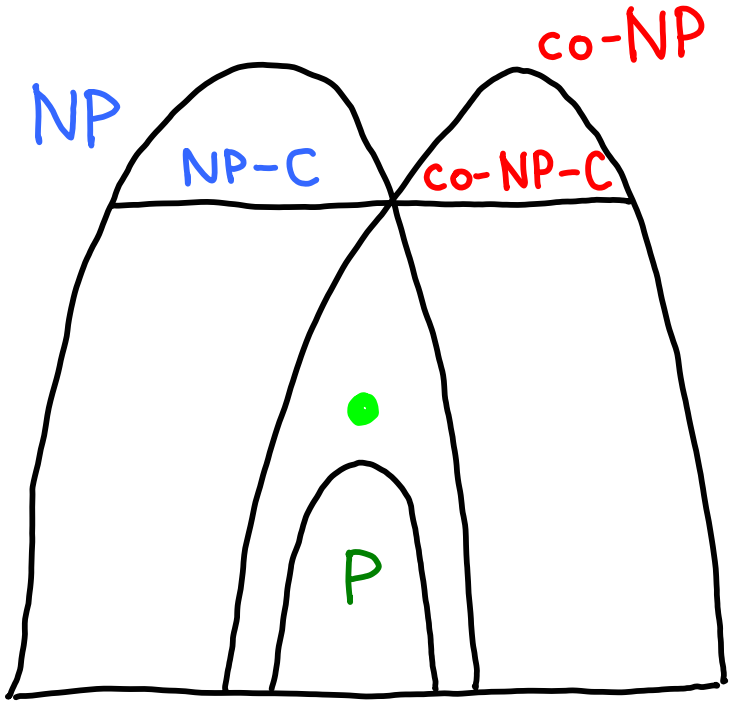
Murty's, Cottle-Danzig, and Lemke's algorithms need exponential time to solve parity games

LEMKE CAN BE FAST ?

THM [ADLER, MEGGIDO 1985]

There is an implementation of Lemke's algorithm that terminates in quadratic number of steps on random linear programs

How difficult is it to find a winning strategy?



P-MATRIX LCP
↑
COMPUTING OPTIMAL STRATEGIES