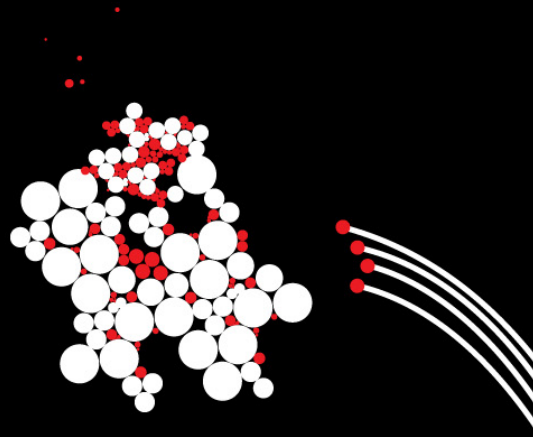
UNIVERSITY OF TWENTE.

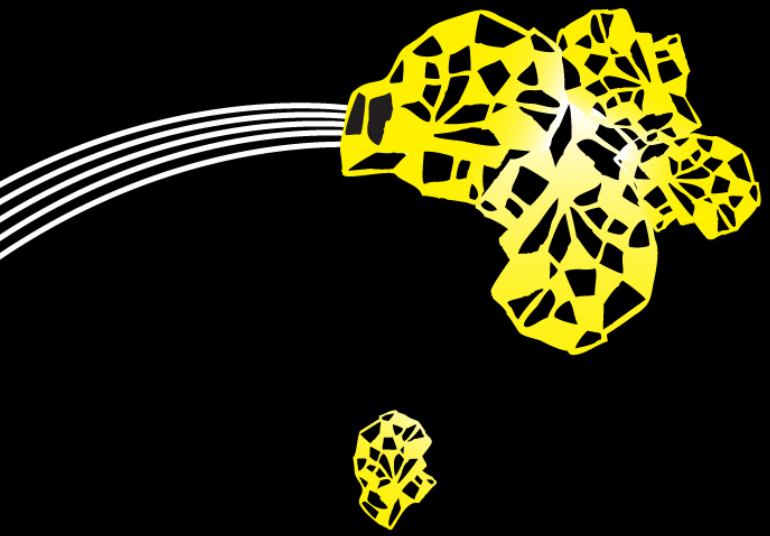


Lower bounds for ranking-based pivot rules

Yann Disser[1], Georg Loho[2,3], *Matthew Maat*[2] and Nils Mosis[1]

[1]TU Darmstadt
[2]University of Twente
[3]FU Berlin

March 2026, Grenoble



Linear programming and the simplex method

$$\begin{aligned} \min \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq 0 \end{aligned}$$

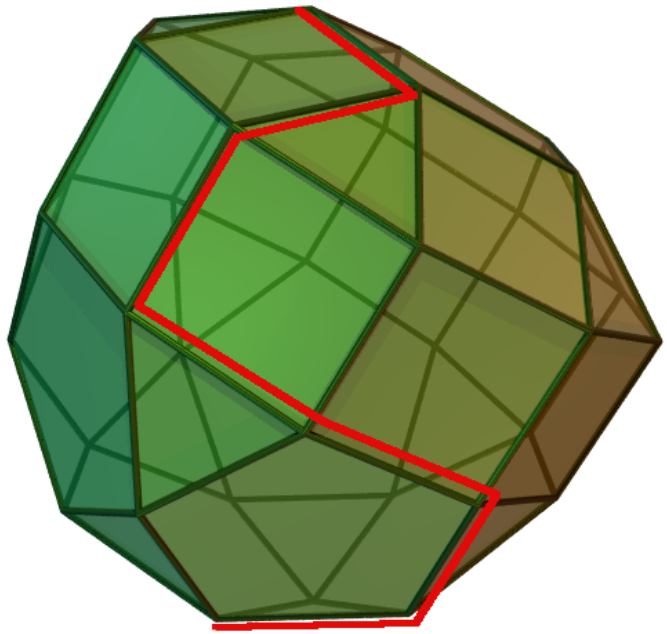


Figure: [1]

[1] https://en.wikipedia.org/wiki/Simplex_algorithm#/media/File:Simplex-method-3-dimensions.png

Linear programming and the simplex method

$$\begin{aligned} \min \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq 0 \end{aligned}$$

- Simplex: start with vertex of polyhedron
- Move along edge polyhedron

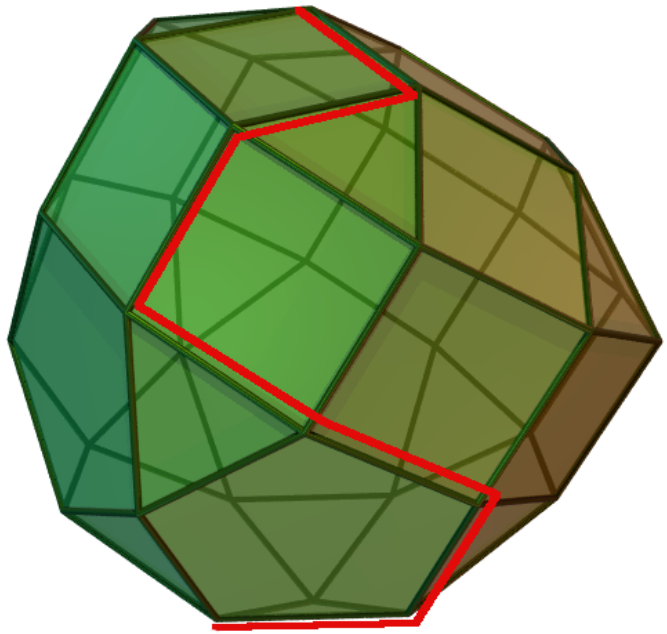


Figure: [1]

[1] https://en.wikipedia.org/wiki/Simplex_algorithm#/media/File:Simplex-method-3-dimensions.png

Linear programming and the simplex method

$$\begin{aligned} \min \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq 0 \end{aligned}$$

- Simplex: start with vertex of polyhedron
- Move along edge polyhedron
- Or: basis B (columns of A)
- Replace column of B/exchange one basic variable

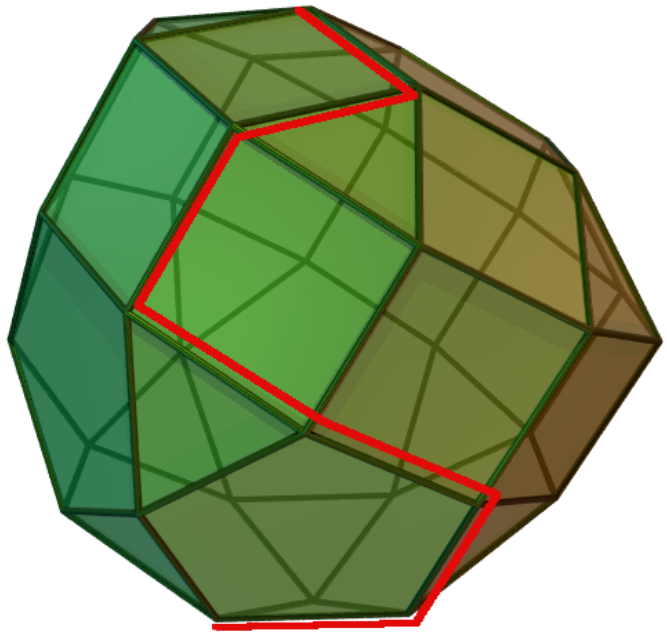
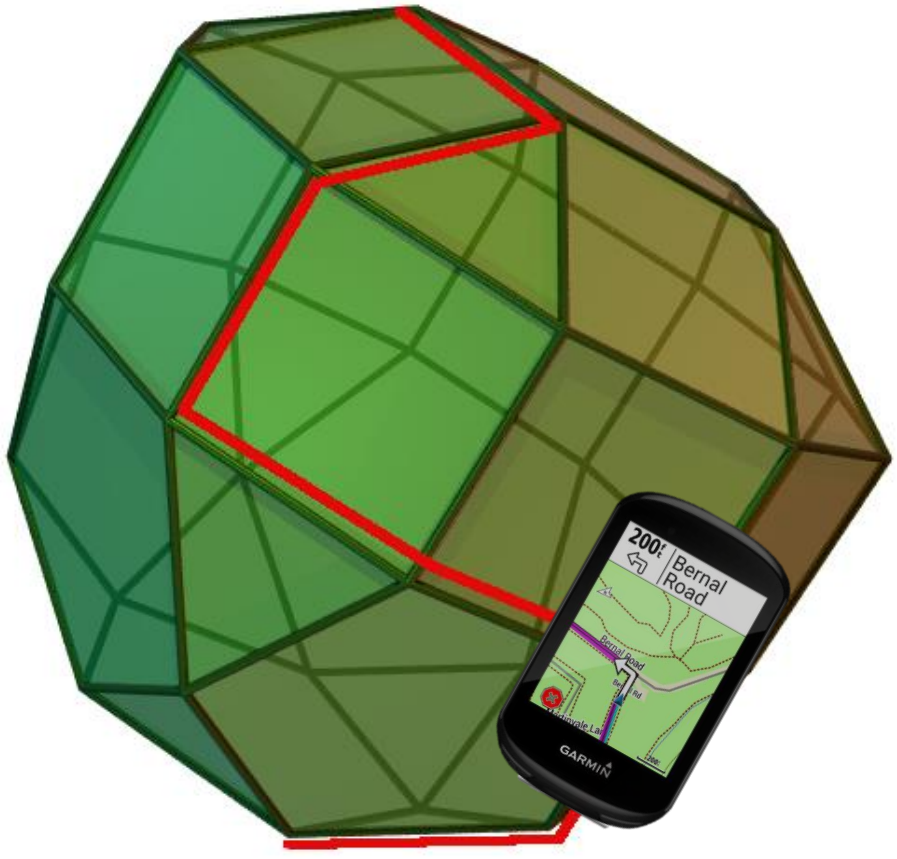


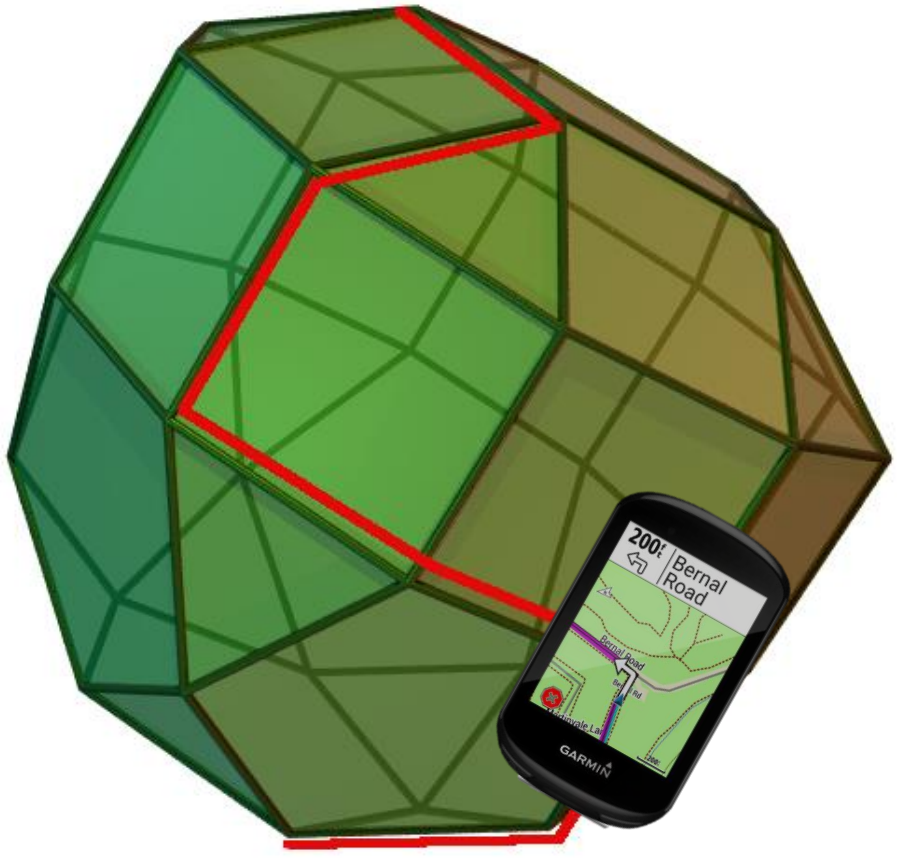
Figure: [1]

[1] https://en.wikipedia.org/wiki/Simplex_algorithm#/media/File:Simplex-method-3-dimensions.png

Pivot rules

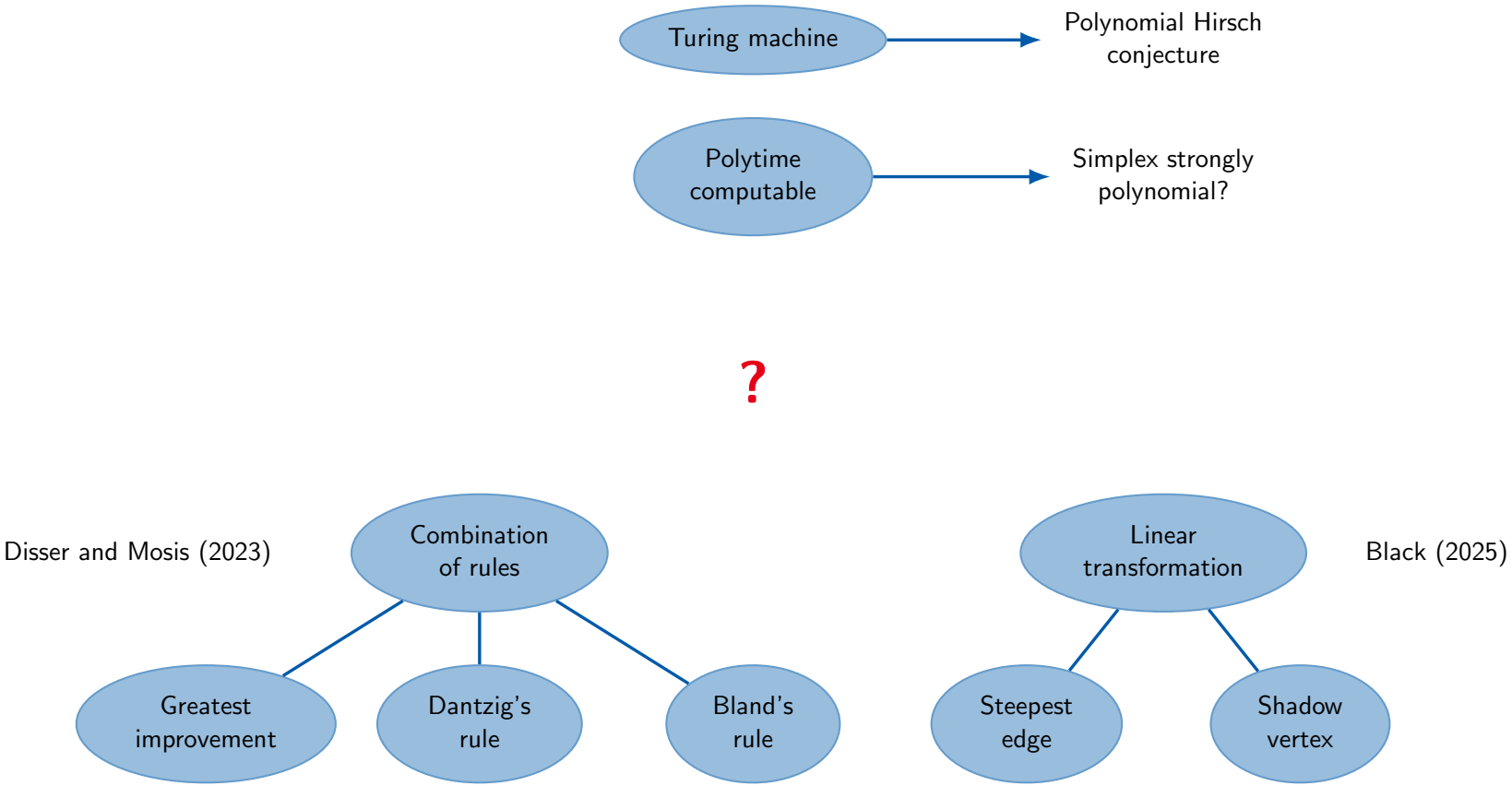


Pivot rules



- Is there a pivot rule that guarantees reaching the optimum in (strongly) polynomial time?

Pivot rule models



Pivot rule models

- $\min \boldsymbol{c}^T \boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0$ with $A \in \mathbb{R}^{m \times n}$.
  Focus on deterministic rules

Pivot rule models

- $\min \boldsymbol{c}^T \boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0$ with $A \in \mathbb{R}^{m \times n}$.
  Focus on deterministic rules
- A pivot rule is a function

$$\Pi_{m,n} \colon \mathcal{L}_{m,n} \times \mathcal{P}([n]) \times [H_{m,n}] \to \mathcal{P}([n]) \times [H_{m,n}]$$
$$((A, \boldsymbol{b}, \boldsymbol{c}), B, h) \mapsto (S, h')$$

Pivot rule models

- $\min \boldsymbol{c}^T \boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0$ with $A \in \mathbb{R}^{m \times n}$.
  Focus on deterministic rules
- A pivot rule is a function

$$\Pi_{m,n} \colon \mathcal{L}_{m,n} \times \mathcal{P}([n]) \times [H_{m,n}] \to \mathcal{P}([n]) \times [H_{m,n}]$$
$$((A, \boldsymbol{b}, \boldsymbol{c}), B, h) \mapsto (S, h')$$

- Split pivot rule up

$$\Pi_{m,n}((A, \boldsymbol{b}, \boldsymbol{c}), B, h) = D^{\Pi}_{m,n}(I^{\Pi}_{m,n}((A, \boldsymbol{b}, \boldsymbol{c}), B), h)$$

With $I^{\Pi}_{m,n}$ **information function** and $D^{\Pi}_{m,n}$ **decision function**.

Pivot rule models

- $\min \boldsymbol{c}^T \boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0$ with $A \in \mathbb{R}^{m \times n}$.
  Focus on deterministic rules
- A pivot rule is a function

$$\Pi_{m,n} \colon \mathcal{L}_{m,n} \times \mathcal{P}([n]) \times [H_{m,n}] \to \mathcal{P}([n]) \times [H_{m,n}]$$
$$((A, \boldsymbol{b}, \boldsymbol{c}), B, h) \mapsto (S, h')$$

- Split pivot rule up

$$\Pi_{m,n}((A, \boldsymbol{b}, \boldsymbol{c}), B, h) = D^{\Pi}_{m,n}(I^{\Pi}_{m,n}((A, \boldsymbol{b}, \boldsymbol{c}), B), h)$$

  With $I^{\Pi}_{m,n}$ **information function** and $D^{\Pi}_{m,n}$ **decision function**.
- Classify pivot rules based on a fixed information function $I^{\Pi}_{m,n}$.

Information functions

- Examples:
  - Steepness of edges
  - Indices of improving variables
  - A ranking by some metric

Information functions

- Examples:
  - Steepness of edges
  - Indices of improving variables
  - A ranking by some metric
- Ranking-based pivot rules: $I^{\Pi}_{m,n}$ encodes information from one or more rankings[2]

[2] And some symmetry requirements

Information functions

- Examples:
  - Steepness of edges
  - Indices of improving variables
  - A ranking by some metric
- Ranking-based pivot rules: $I^{\Pi}_{m,n}$ encodes information from one or more rankings[2]
- We show lower bounds for:
  - $I^{\Pi}_{m,n}$ is ranking of indices (*index-based* rule)
  - Memoryless rules where $I^{\Pi}_{m,n}$ has rankings of indices, reduced cost, objective improvement **at the same time**.

[2] And some symmetry requirements

Theorems

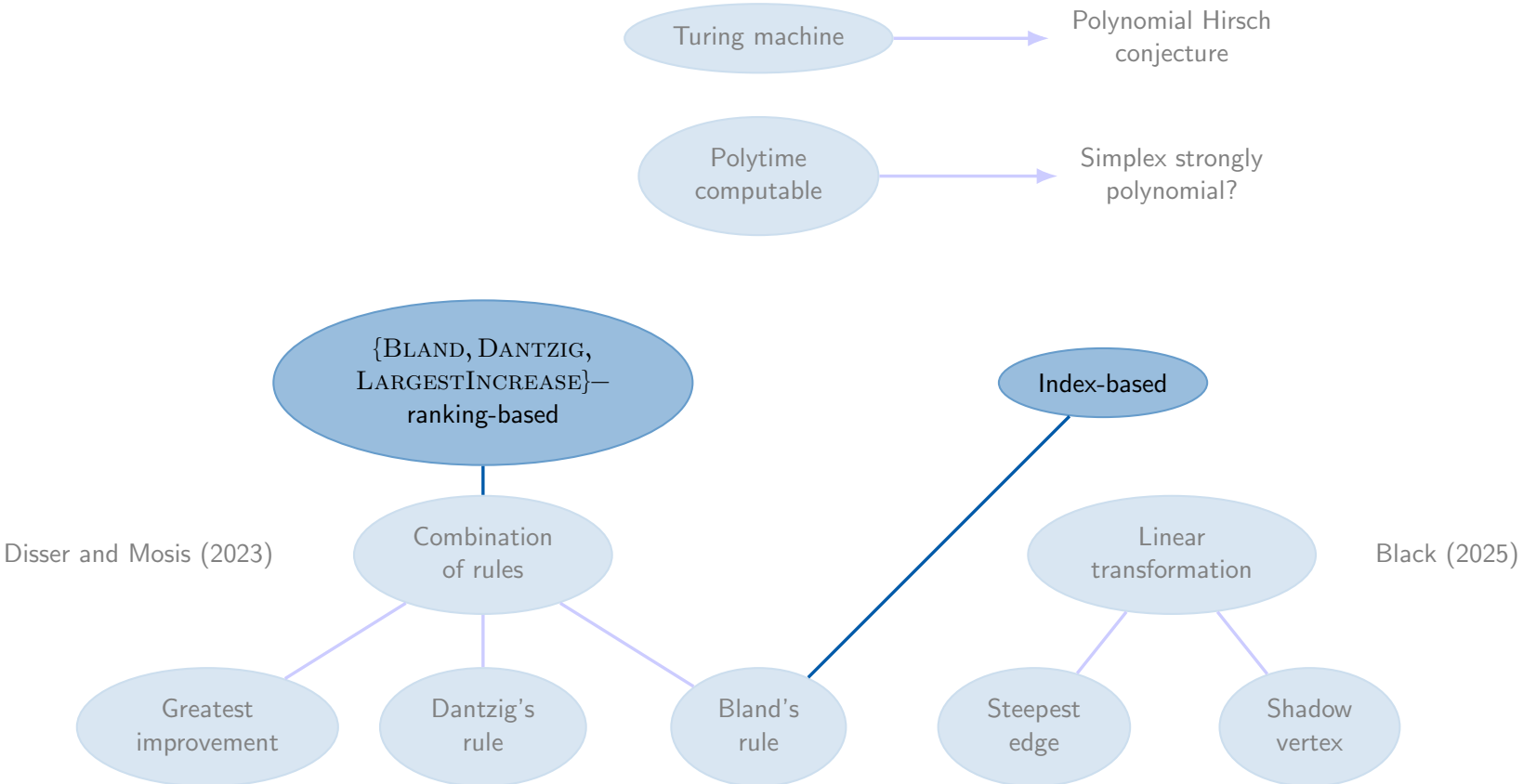
Theorem (Disser, Loho, M., Mosis)

Using an index-based pivot rule that uses $o(N/\log(N))$ memory states, the simplex algorithm needs a number of steps that is superpolynomial in N in the worst case, with N the input size.

Theorem (Disser, Loho, M., Mosis)

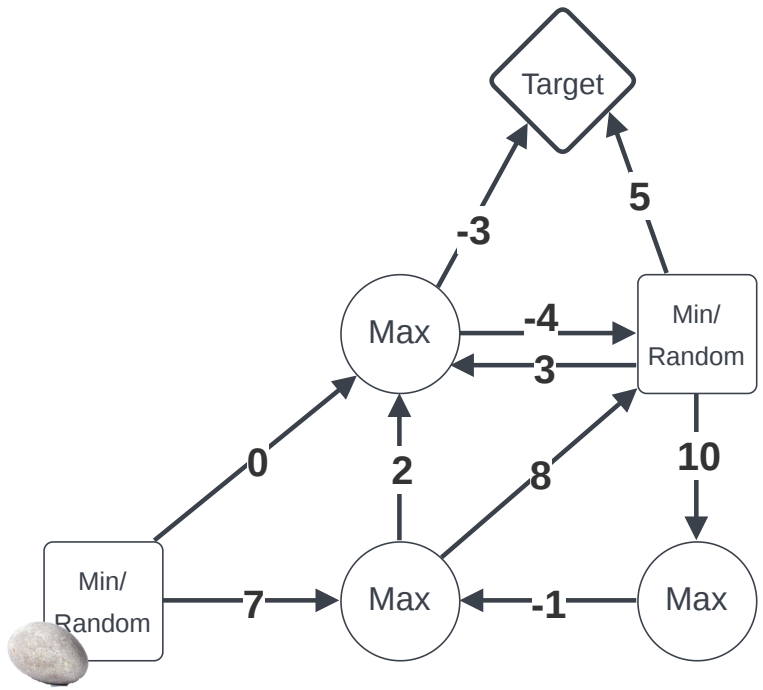
Using a memoryless $\{\textsc{Bland}, \textsc{Dantzig}, \textsc{LargestIncrease}\}$-ranking-based pivot rule, the simplex algorithm takes $\Omega(2^{\sqrt{N}})$ iterations in the worst case, with N the input size.

Pivot rule models



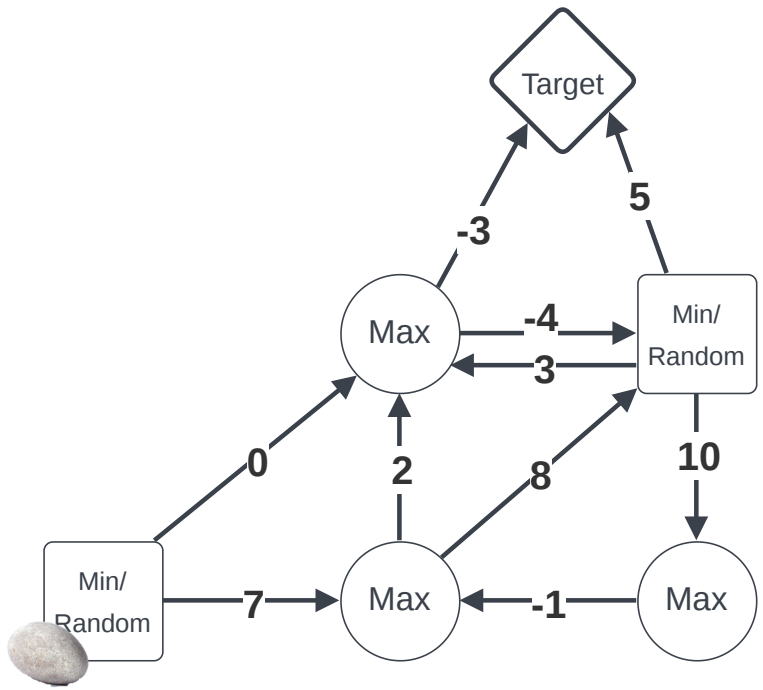
LSP and weak unichain MDP

- Directed graph with edge costs, Max/Min or Max/Rand nodes
- Move pebble to target vertex



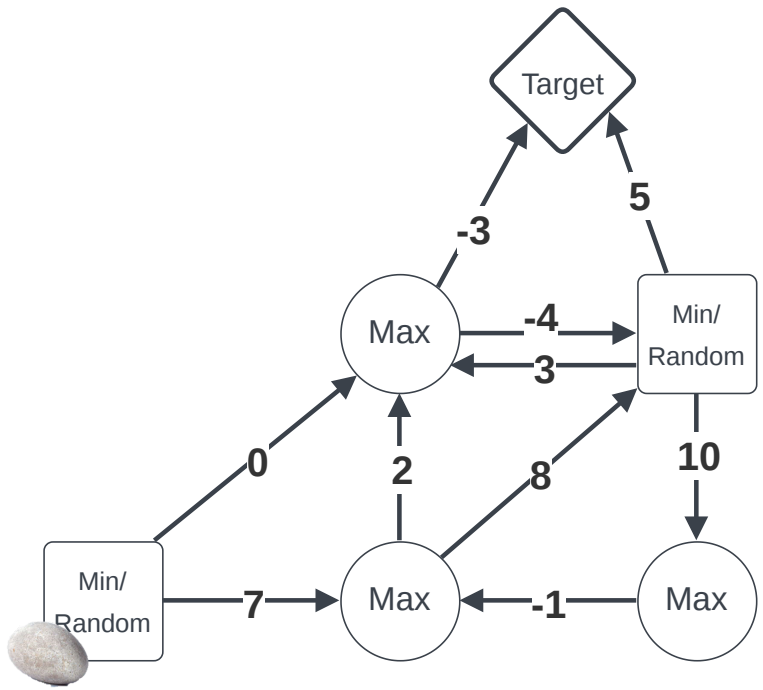
LSP and weak unichain MDP

- Directed graph with edge costs, Max/Min or Max/Rand nodes
- Move pebble to target vertex
- Player goals: maximize/minimize weight of path to target



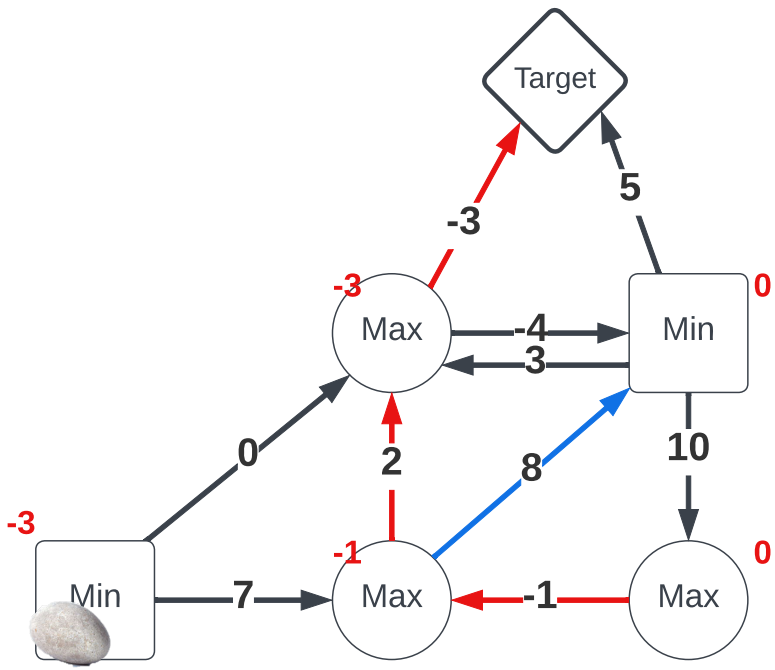
LSP and weak unichain MDP

- Directed graph with edge costs, Max/Min or Max/Rand nodes
- Move pebble to target vertex
- Player goals: maximize/minimize weight of path to target
- Find optimal strategies and value



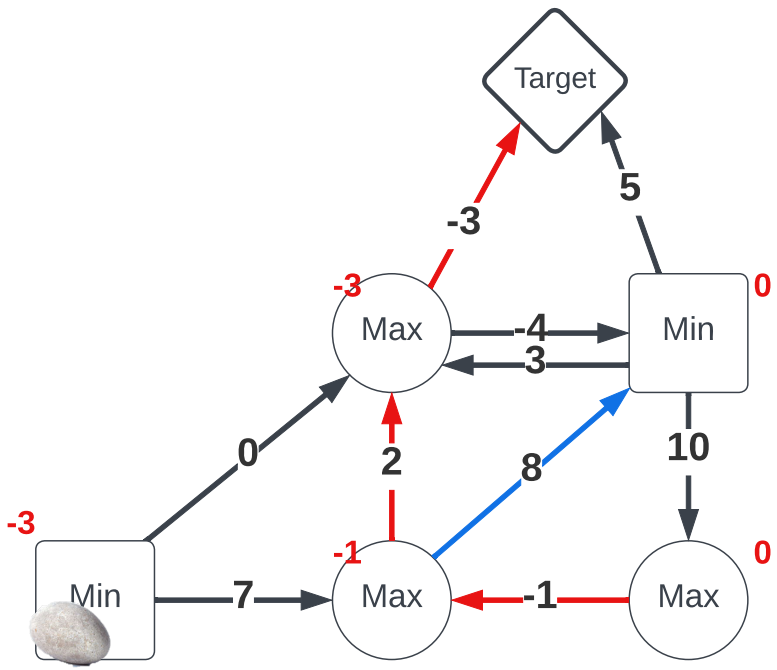
Strategy improvement

- Fix an admissible strategy for Maximizer



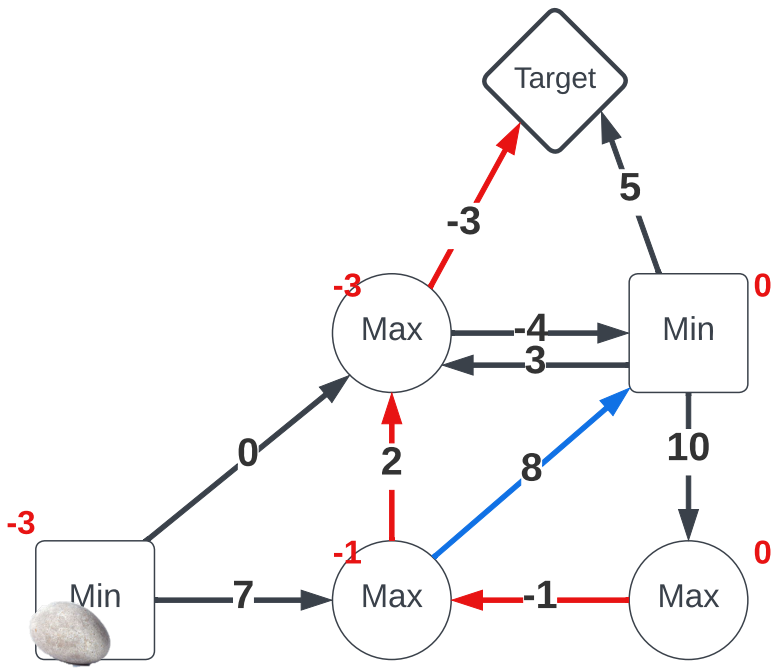
Strategy improvement

- Fix an admissible strategy for Maximizer
- Make improving switches until optimal



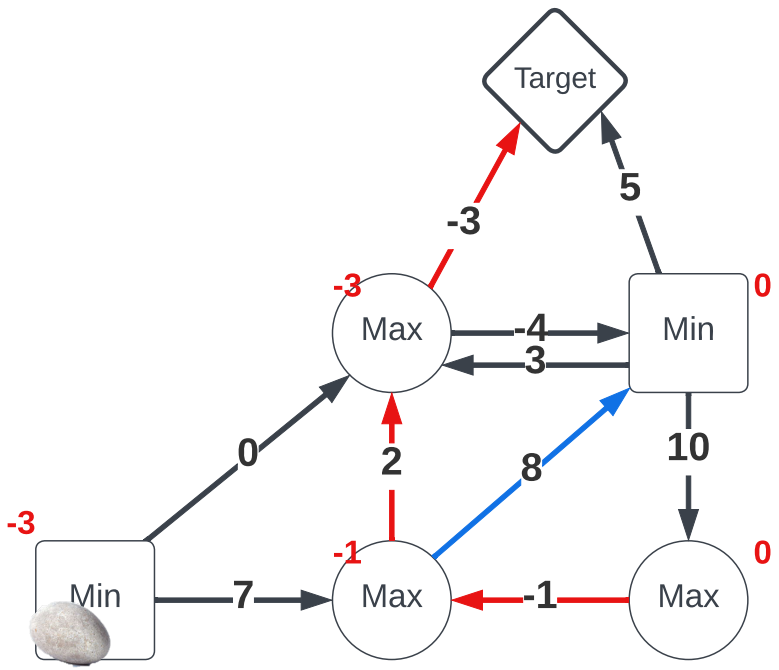
Strategy improvement

- Fix an admissible strategy for Maximizer
- Make improving switches until optimal
- Improvement rule

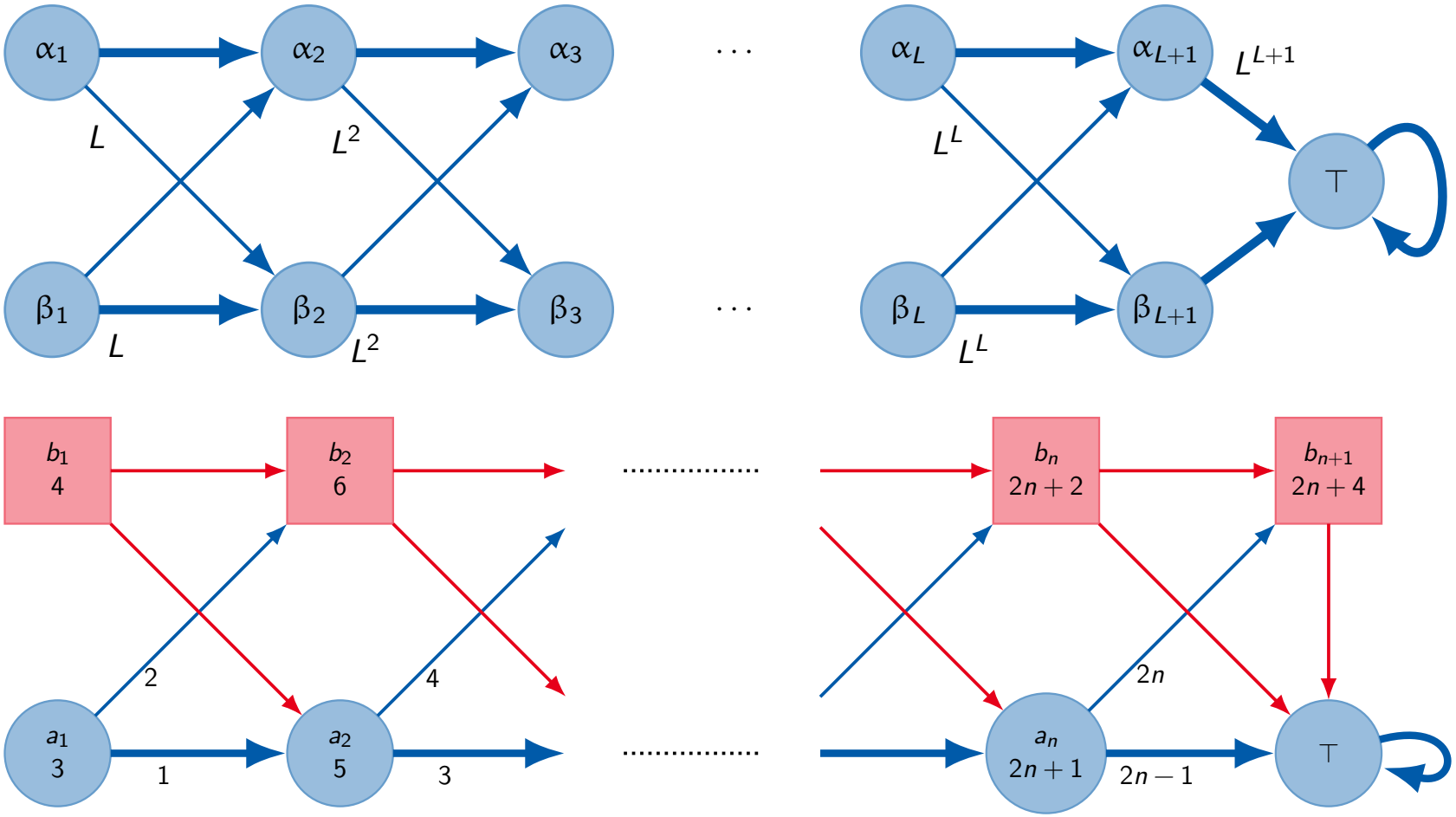


Strategy improvement

- Fix an admissible strategy for Maximizer
- Make improving switches until optimal
- Improvement rule
- Equivalent to simplex method in related LP
  (M. (2026+), Schewe (2009), Friedmann (2011))



Proof technique: binary counter games



Proof technique

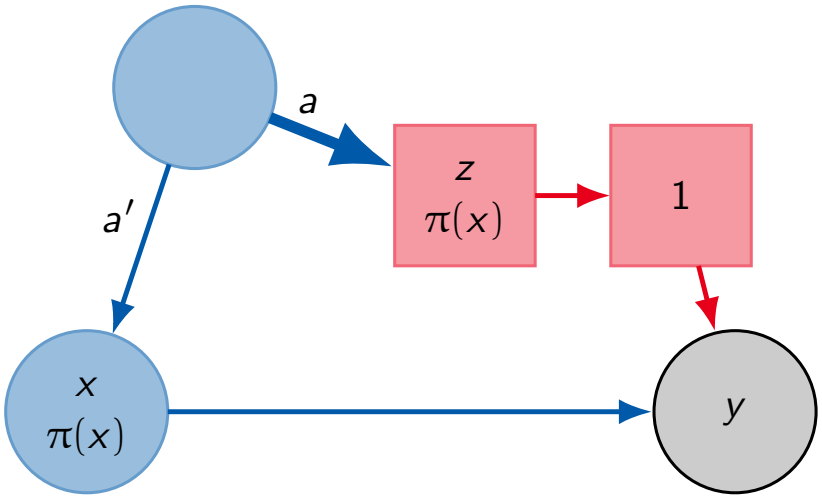
- Both structures exponential with Bland's rule
- Goal: add gadgets to imitate Bland's rule

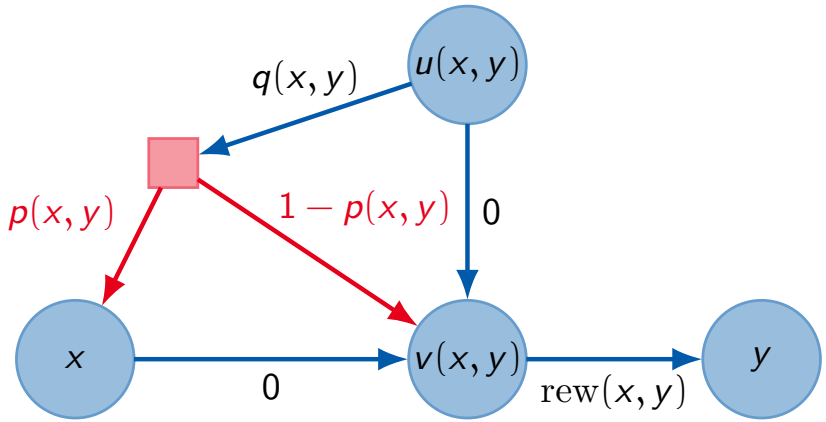
Proof technique

- Both structures exponential with Bland's rule
- Goal: add gadgets to imitate Bland's rule
- LSP for index-based, MDP for $\{\textsc{Bland}, \textsc{Dantzig}, \textsc{LargestIncrease}\}$-ranking-based
- Turn LSP/MDP into equivalent LP

Proof technique

- Both structures exponential with Bland's rule
- Goal: add gadgets to imitate Bland's rule
- LSP for index-based, MDP for $\{\textsc{Bland}, \textsc{Dantzig}, \textsc{LargestIncrease}\}$-ranking-based
- Turn LSP/MDP into equivalent LP
- Constant number of improving moves





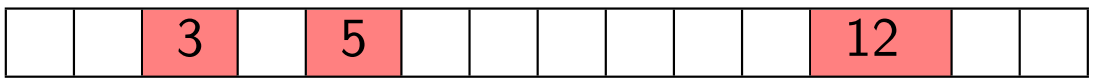
Proof technique: index-based rules

- Rule uses order of indices and memory

Proof technique: index-based rules

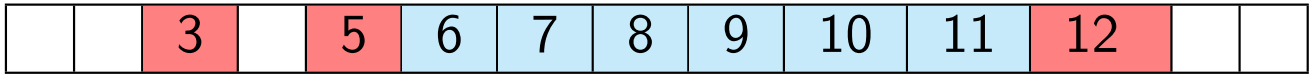
- Rule uses order of indices and memory
- Keep number of improving moves constant
  - Choice of j-th variable will repeat after at most $o(\frac{n}{\log(n)})$ steps.

h	1	2	3	4
j	3	5	12	3
h'	2	3	4	1



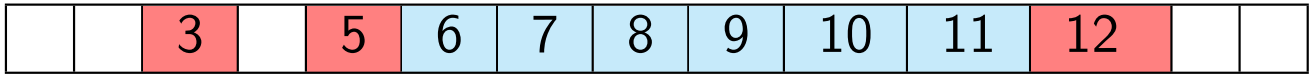
Proof technique: index-based rules

- Two options:
  - There is a gap of size $\omega(log(n))$



Proof technique: index-based rules

- Two options:
  - There is a gap of size $\omega(log(n))$

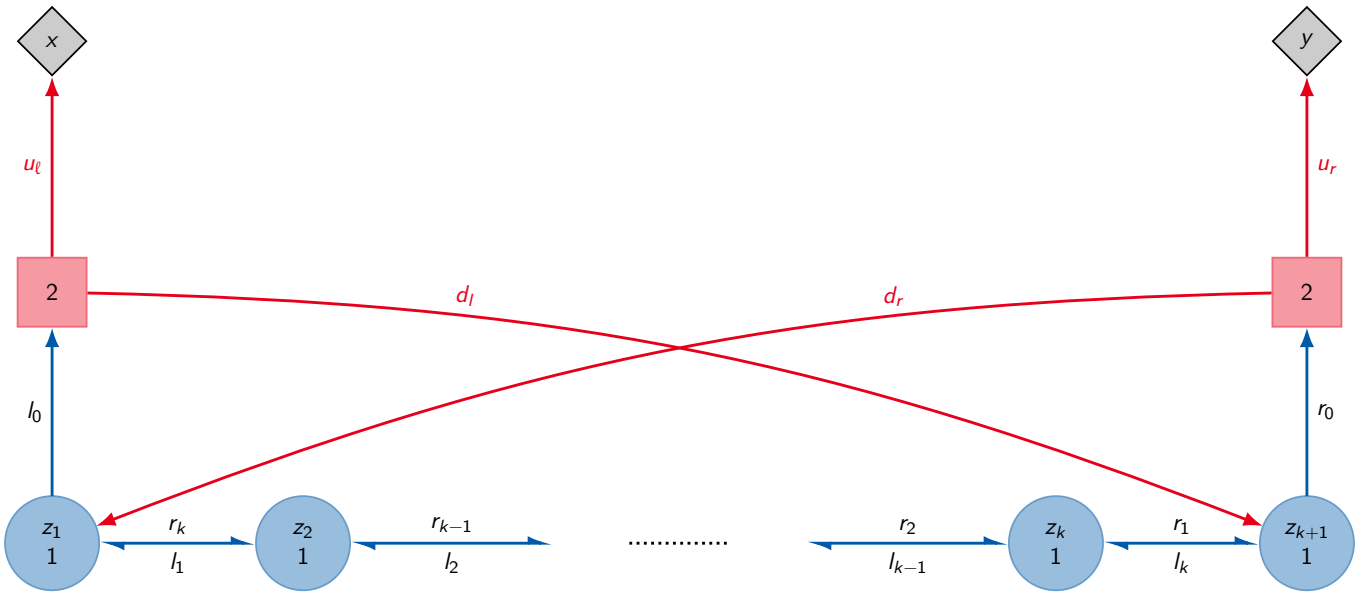


  - Make binary counter with its edges at $5, 6, \ldots, 11$ in Bland order: $\omega(\log(n))$ bits

Proof technique: index-based rules

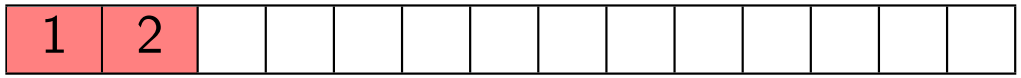
- Two options:
  - There is a gap of size $\omega(log(n))$

  - Make binary counter with its edges at $5, 6, \ldots, 11$ in Bland order: $\omega(\log(n))$ bits
  - Decoy switches that get reactivated every cycle



Proof technique: index-based rules

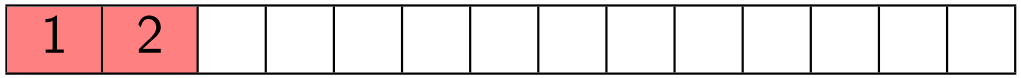
- Two options:
  - Output of pivot rule is clustered



  - Make one big binary counter with $\omega(\log(n))$ bits, copy every edge $o(\frac{n}{\log(n)})$ times

Proof technique: index-based rules

- Two options:
  - Output of pivot rule is clustered



  - Make one big binary counter with $\omega(\log(n))$ bits, copy every edge $o(\frac{n}{\log(n)})$ times
- Binary counter takes $2^{\omega(\log(n))} = n^{\omega(1)}$ iterations

Proof technique: memoryless ranking-based rules

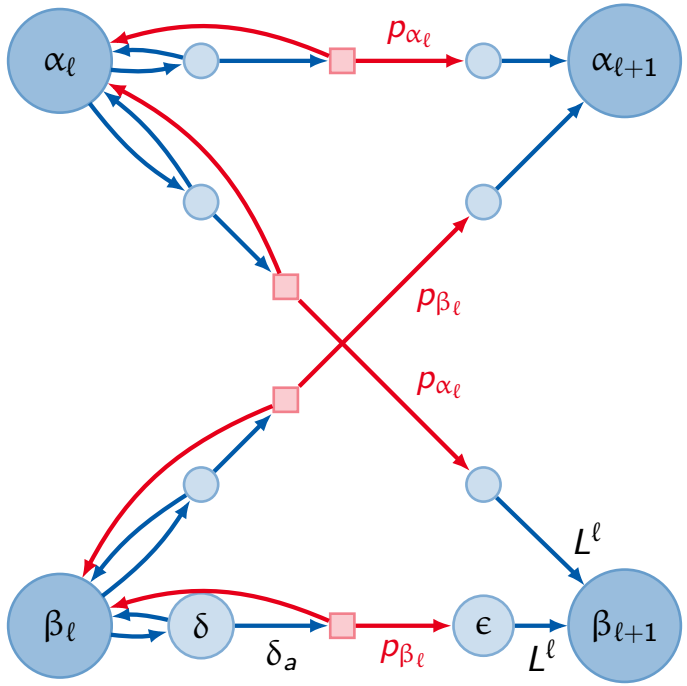
- Ranking by index, reduced cost and objective improvement

Proof technique: memoryless ranking-based rules

- Ranking by index, reduced cost and objective improvement
- Bland's rule, constant number of improving moves

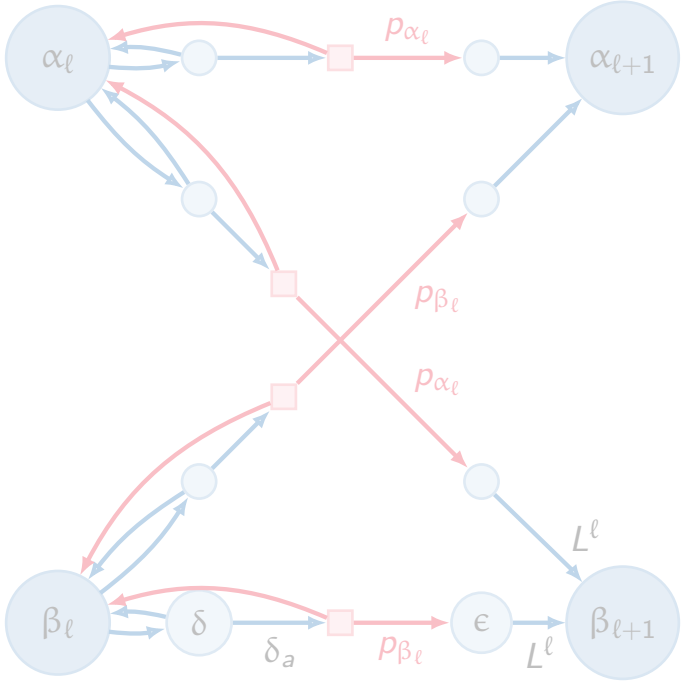
Proof technique: memoryless ranking-based rules

- Ranking by index, reduced cost and objective improvement
- Bland's rule, constant number of improving moves
- Make sure all rankings coincide:



Proof technique: memoryless ranking-based rules

- Ranking by index, reduced cost and objective improvement
- Bland's rule, constant number of improving moves
- Make sure all rankings coincide:
- Pivot rule effectively only sees one ranking



Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).

Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).
  - Then we copy every edge $\sqrt{n}$ times

Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).
  - Then we copy every edge $\sqrt{n}$ times
  - Then rule behaves like Bland's rule

Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).
  - Then we copy every edge $\sqrt{n}$ times
  - Then rule behaves like Bland's rule
- Case 2:
  - Pivot rule can output indices $\Omega(\sqrt{n})$.

Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).
  - Then we copy every edge $\sqrt{n}$ times
  - Then rule behaves like Bland's rule
- Case 2:
  - Pivot rule can output indices $\Omega(\sqrt{n})$.
  - We construct MDPs for n where this happens

Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).
  - Then we copy every edge $\sqrt{n}$ times
  - Then rule behaves like Bland's rule
- Case 2:
  - Pivot rule can output indices $\Omega(\sqrt{n})$.
  - We construct MDPs for n where this happens
  - Build a binary counter with $\sqrt{n}$ levels, with indices reversed

Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).
  - Then we copy every edge $\sqrt{n}$ times
  - Then rule behaves like Bland's rule
- Case 2:
  - Pivot rule can output indices $\Omega(\sqrt{n})$.
  - We construct MDPs for n where this happens
  - Build a binary counter with $\sqrt{n}$ levels, with indices reversed
  - This behaves like Bland's rule

Proof technique: memoryless ranking-based rules

- Pivot rule always picks the k-th lowest index!
- Case 1:
  - Pivot rule always outputs small indices ($o(\sqrt{n})$).
  - Then we copy every edge $\sqrt{n}$ times
  - Then rule behaves like Bland's rule
- Case 2:
  - Pivot rule can output indices $\Omega(\sqrt{n})$.
  - We construct MDPs for n where this happens
  - Build a binary counter with $\sqrt{n}$ levels, with indices reversed
  - This behaves like Bland's rule
- In all cases, takes $\Omega(2^{\sqrt{N}})$ iterations

Contact/references I

Contact: m.t.maat@utwente.nl

Black, A. E. (2025). Exponential lower bounds for many pivot rules for the simplex method. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 86–99. Springer.

Disser, Y. and Mosis, N. (2023). A Unified Worst Case for Classical Simplex and Policy Iteration Pivot Rules. In *34th International Symposium on Algorithms and Computation (ISAAC 2023)*, volume 283, pages 27:1–27:17, Dagstuhl, Germany.

Friedmann, O. (2011). *Exponential lower bounds for solving infinitary payoff games and linear programs*. PhD Thesis, lmu.

Contact/references II

Friedmann, O., Hansen, T. D., and Zwick, U. (2011). Subexponential lower bounds for randomized pivoting rules for the simplex algorithm. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, STOC '11, page 283–292, New York, NY, USA. Association for Computing Machinery.

Maat, M. (2026+). Connecting strategy improvement, the simplex algorithm and lopsidedness.

Schewe, S. (2009). From parity and payoff games to linear programming. In *International Symposium on Mathematical Foundations of Computer Science*, pages 675–686. Springer.