Separation for the Max-Cut Problem using Target Cuts and Graph Contraction

Thorsten Bonato

Research Group Discrete and Combinatorial Optimization
University of Heidelberg

Joint work with:
Gerhard Reinelt (University of Heidelberg)
Giovanni Rinaldi (IASI, Rome)

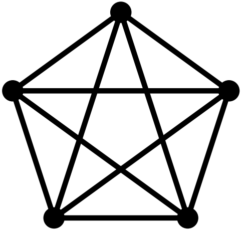
1st Alpen-Adria Workshop on Optimization
Klagenfurt, June 4, 2010

Outline

Outline

Definition

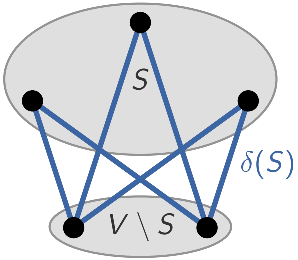
Let $G = (V, E, c)$ be an undirected weighted graph.



Definition

Let $G = (V, E, c)$ be an undirected weighted graph.

Any $S \subseteq V$ induces a set $\delta(S)$ of edges with exactly one end in S. The set $\delta(S)$ is called a cut of G with shores S and $V \setminus S$.

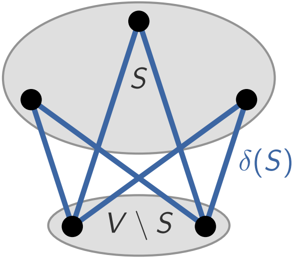


Definition

Let $G = (V, E, c)$ be an undirected weighted graph.

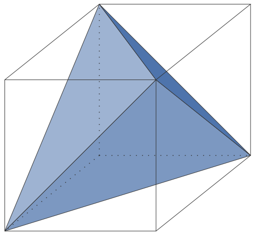
Any $S \subseteq V$ induces a set $\delta(S)$ of edges with exactly one end in S. The set $\delta(S)$ is called a cut of G with shores S and $V \setminus S$.

Finding a cut with maximum aggregate edge weight is known as max-cut problem.



Cut polytope CUT(G)

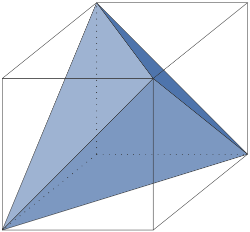
Convex hull of all incidence vectors of cuts of G.



$\mathrm{CUT}(K_3)$

Cut polytope CUT(G)

Convex hull of all incidence vectors of cuts of G.



$\mathrm{CUT}(K_3)$

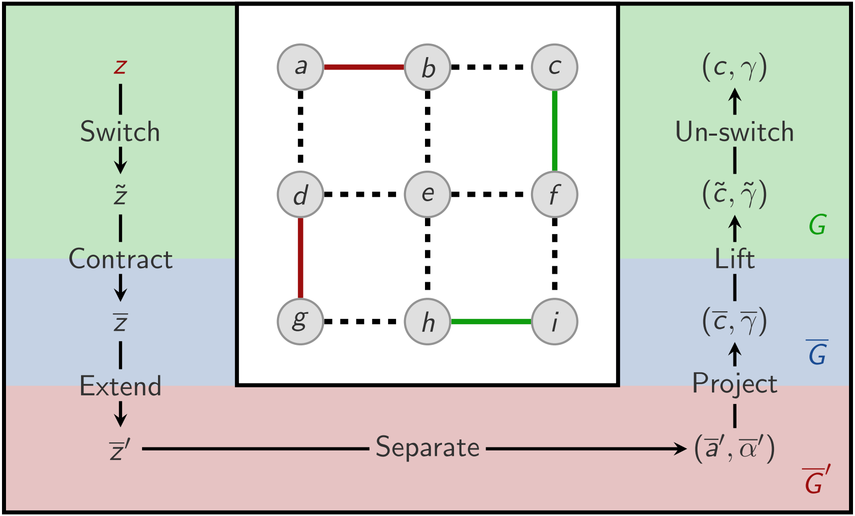
Semimetric polytope MET(G)

Relaxation of the max-cut IP formulation described by two inequality classes:

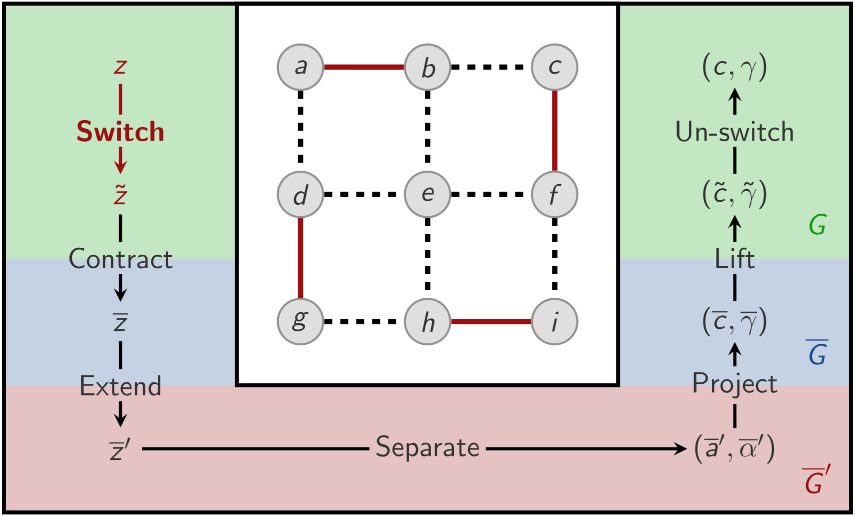
Odd-cycle:	$x(F) - x(C \setminus F) \leq \lvert F\rvert - 1,$	for each cycle C of G, for all $F \subseteq C, \lvert F\rvert$ odd.
Trivial:	$0 \leq x_e \leq 1,$	for all $e \in E$.

Outline

Input: LP solution $z \in \mathrm{MET}(G) \setminus \mathrm{CUT}(G)$.

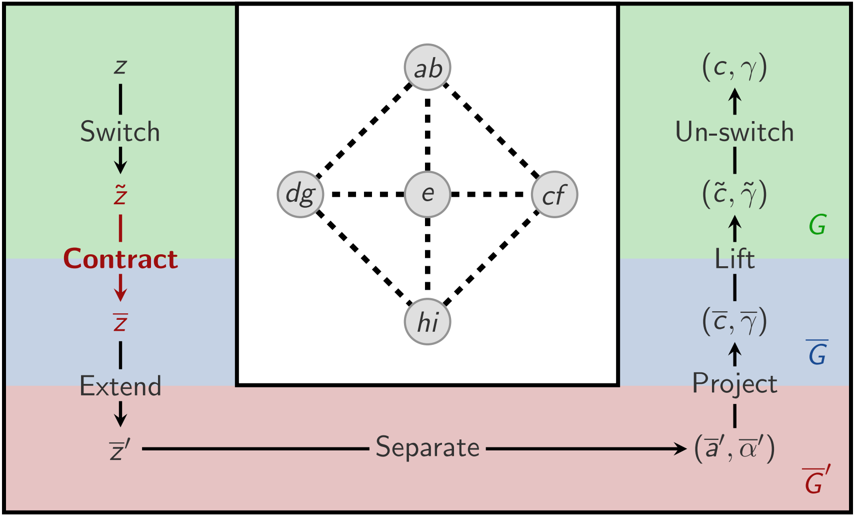


Transform 1-edges into 0-edges.

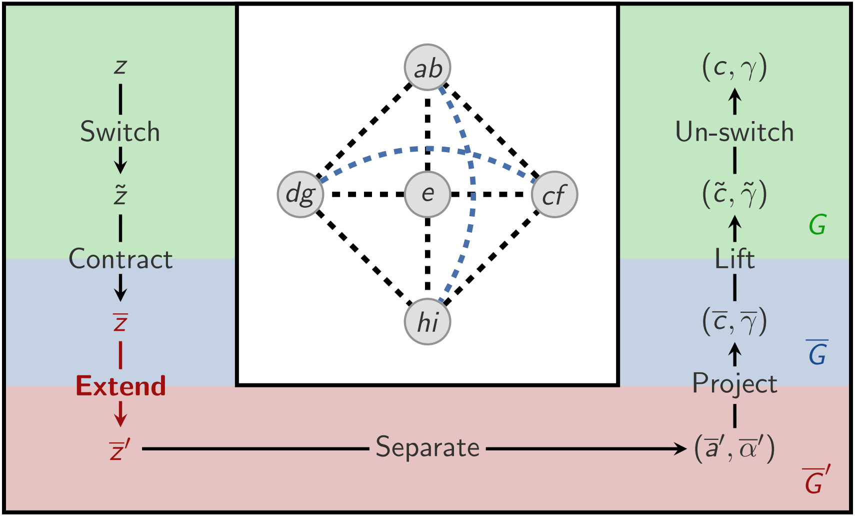


Outline of the Separation using Graph Contraction

Contract 0-edges. Allows heuristic odd-cycle separation.

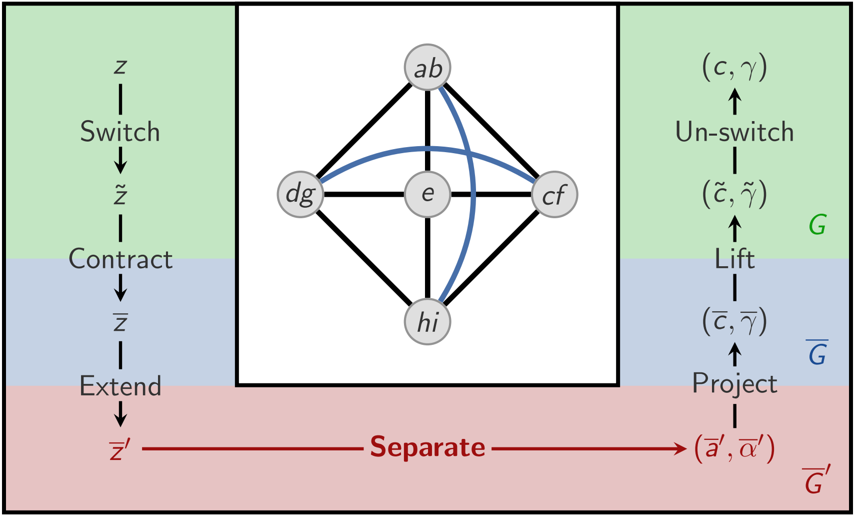


Introduce artificial LP values for non-edges.

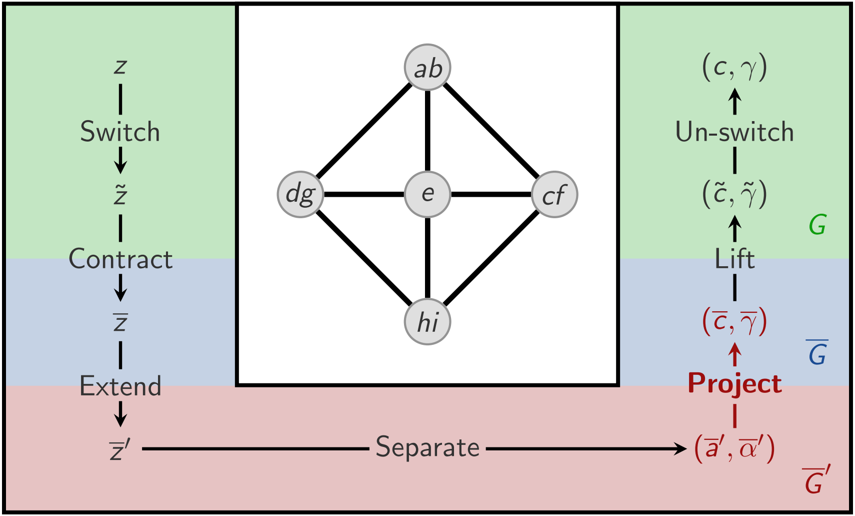


Outline of the Separation using Graph Contraction

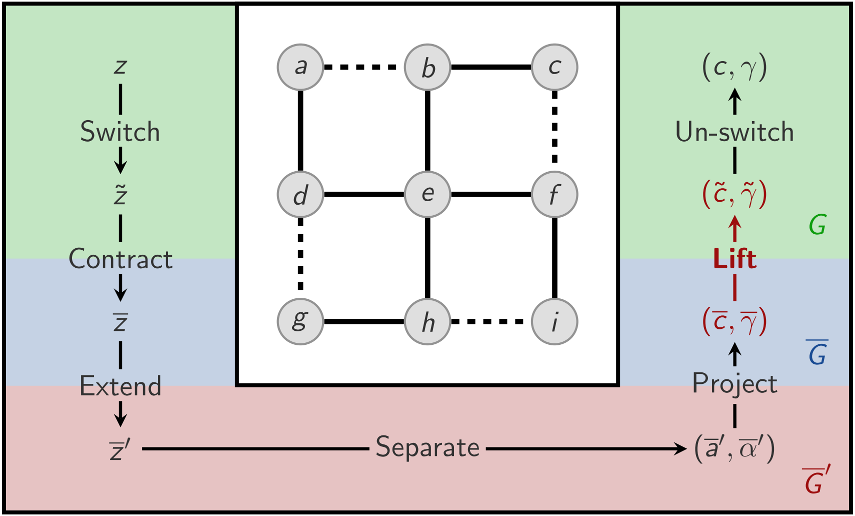
Separate extended LP solution.



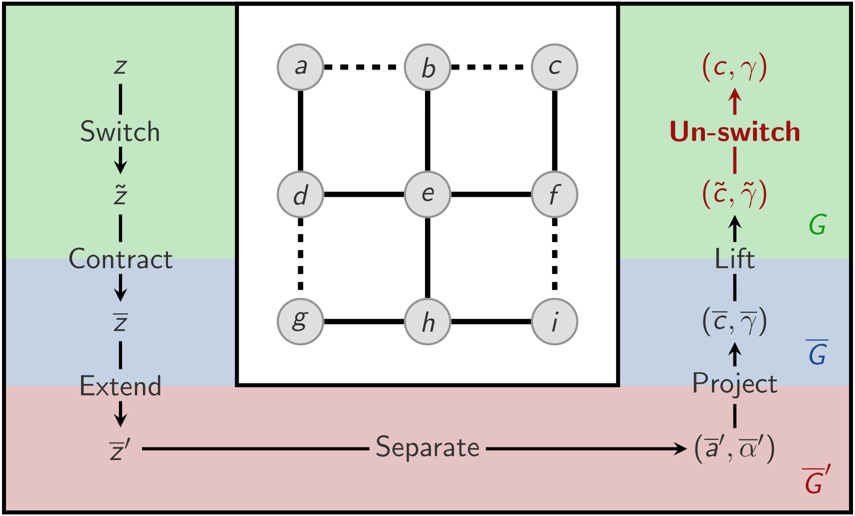
Project out nonzero coefficients related to non-edges.



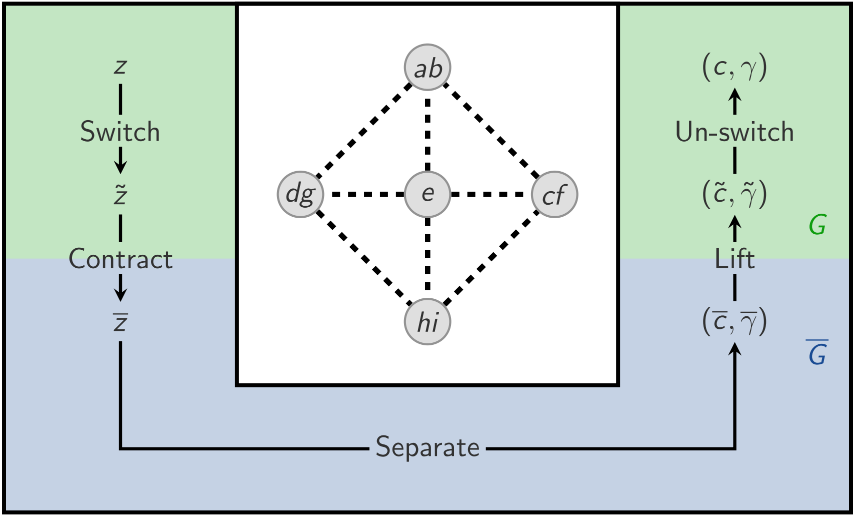
Lift inequality.



Switch lifted inequality.



In the scope of this talk, we omit the extension.



Outline

Target cuts were introduced by Buchheim, Liers, and Oswald.

Given:

- Polytope $P := \text{conv}\{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$,
- Point $x^* \notin P$.

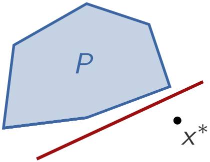
Target cuts were introduced by Buchheim, Liers, and Oswald.

Given:

- Polytope $P := \mathrm{conv}\{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$,
- Point $x^* \notin P$.

Standard separation

Find a valid inequality $a^T x \leq \alpha$
that separates x^* from P.



Target Cuts vs. Standard Separation

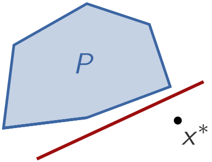
Target cuts were introduced by Buchheim, Liers, and Oswald.

Given:

- Polytope $P := \text{conv}\{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$,
- Point $x^* \notin P$.

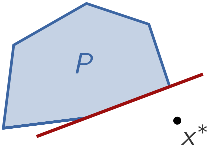
Standard separation

Find a valid inequality $a^T x \leq \alpha$ that separates x^* from P.



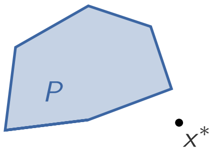
Target cut separation

Find a facet inducing inequality $a^T x \leq \alpha$ that separates x^* from P.



Given:

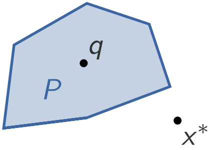
- Polytope $P := \text{conv}\{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$,
- Point $x^* \notin P$,



Target Cuts in a Nutshell

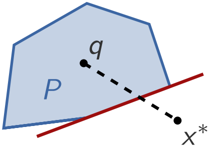
Given:

- Polytope $P := \operatorname{conv}\{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$,
- Point $x^* \notin P$,
- Point $q \in \operatorname{relint}(P)$.



Given:

- Polytope $P := \operatorname{conv}\{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$,
- Point $x^* \notin P$,
- Point $q \in \operatorname{relint}(P)$.



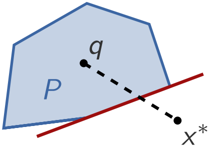
Goal

Find the inequality $a^T x \leq \alpha$ which induces the facet of P that is intersected by the line $\overline{qx^*}$.

Target Cuts in a Nutshell

Given:

- Polytope $P := \mathrm{conv}\{x_1, \dots, x_n\} \subseteq \mathbb{R}^d$,
- Point $x^* \notin P$,
- Point $q \in \mathrm{relint}(P)$.



Goal

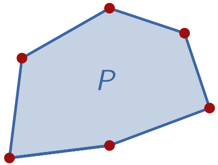
Find the inequality $a^T x \leq \alpha$ which induces the facet of P that is intersected by the line $\overline{qx^*}$.

Target Cut LP

$$\begin{aligned} \max \quad & a^T(x^* - q) \\ \text{s.t.} \quad & a^T(x_i - q) \leq 1, \text{ for } i = 1, \dots, n \\ & a \in \mathbb{R}^d. \end{aligned} \tag{TC}$$

Main problem

(TC) has one row for each vertex of P.

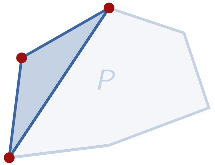


Main problem

(TC) has one row for each vertex of P.

Idea

Start with a subset of vertices and extend it iteratively.

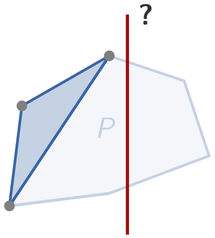


Main problem

(TC) has one row for each vertex of P.

Idea

Start with a subset of vertices and extend it iteratively.



Requires an oracle that

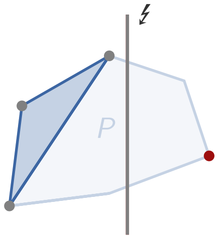
1. checks whether a given inequality $b^T x \leq \beta$ is valid for P,

Main problem

(TC) has one row for each vertex of P.

Idea

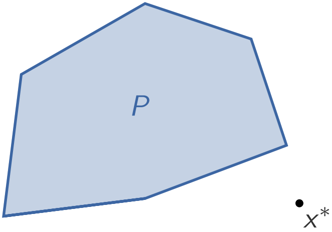
Start with a subset of vertices and extend it iteratively.



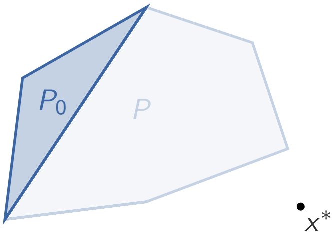
Requires an oracle that

1. checks whether a given inequality $b^T x \leq \beta$ is valid for P,
2. if not, provides at least one violating vertex of P.

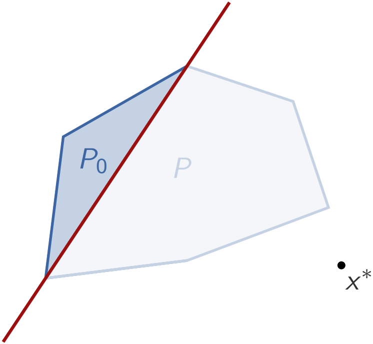
Delayed Row Generation: An Example



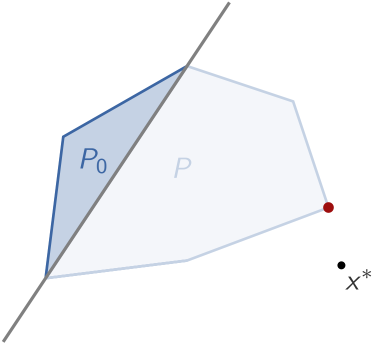
Delayed Row Generation: An Example



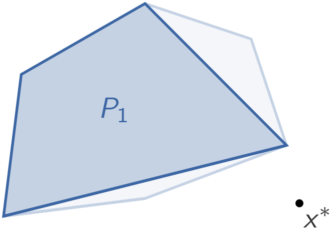
Delayed Row Generation: An Example



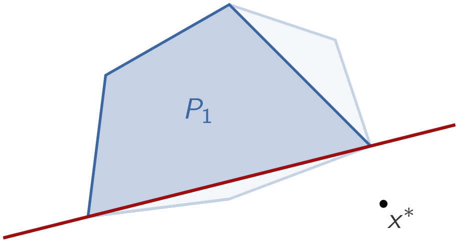
Delayed Row Generation: An Example



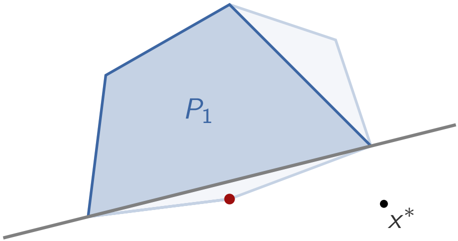
Delayed Row Generation: An Example



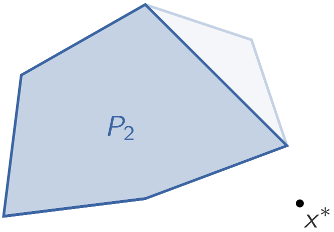
Delayed Row Generation: An Example



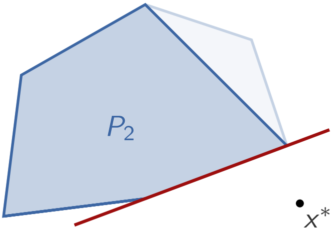
Delayed Row Generation: An Example



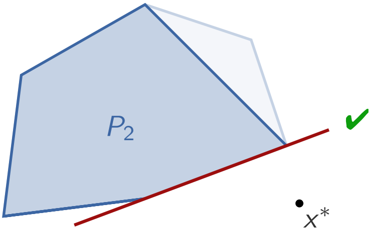
Delayed Row Generation: An Example



Delayed Row Generation: An Example



Delayed Row Generation: An Example



Exact approach

- Solve $\max\{b^T x \mid x \in P\}$ for given b.
- Use of exact algorithm in each oracle call can slow down the overall target cut separation.

Exact approach

- Solve $\max\{b^T x \mid x \in P\}$ for given b.
- Use of exact algorithm in each oracle call can slow down the overall target cut separation.

Heuristic approach

1. Call a fast heuristic.
2. If no violating vertex was found, call exact algorithm.

Exact approach

- Solve $\max\{b^T x \mid x \in P\}$ for given b.
- Use of exact algorithm in each oracle call can slow down the overall target cut separation.

Heuristic approach

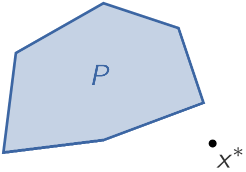
1. Call a fast heuristic.
2. If no violating vertex was found, call exact algorithm.

Oracle varieties

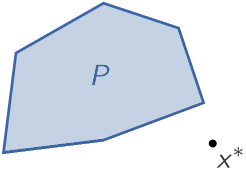
Try to find multiple violating vertices per oracle call.

Oracle: Heuristic vs. Exact Approach

Heuristic approach

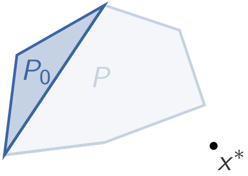


Exact approach

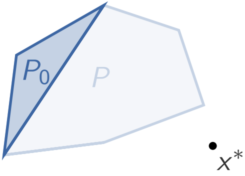


Oracle: Heuristic vs. Exact Approach

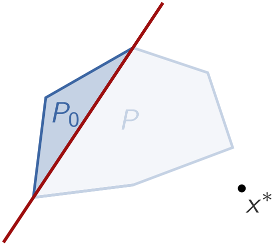
Heuristic approach



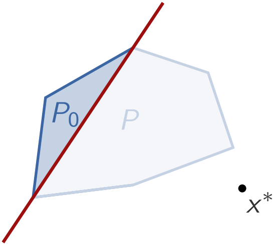
Exact approach



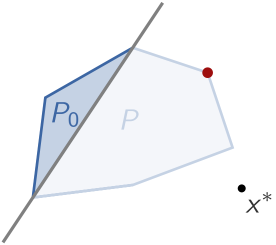
Heuristic approach



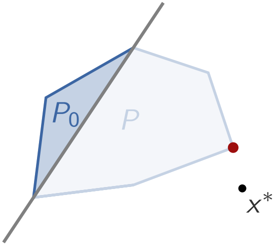
Exact approach



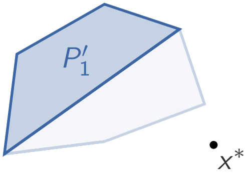
Heuristic approach



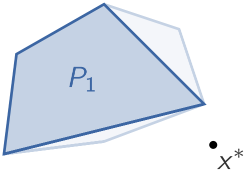
Exact approach



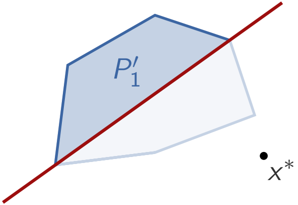
Heuristic approach



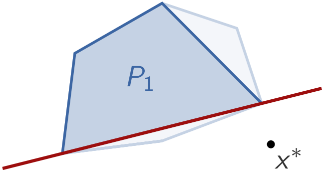
Exact approach



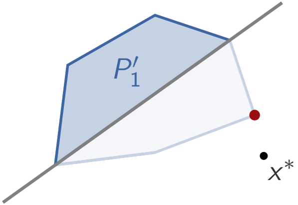
Heuristic approach



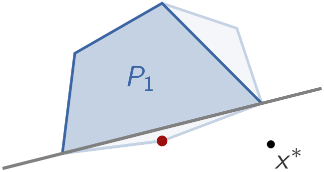
Exact approach



Heuristic approach

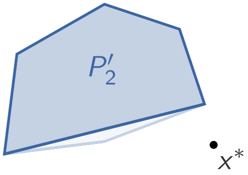


Exact approach

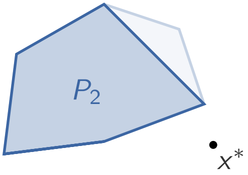


Oracle: Heuristic vs. Exact Approach

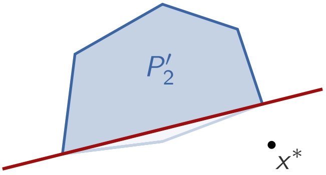
Heuristic approach



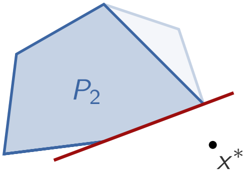
Exact approach



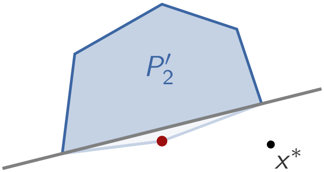
Heuristic approach



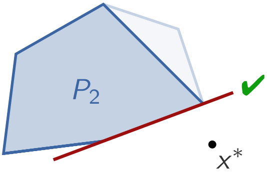
Exact approach



Heuristic approach

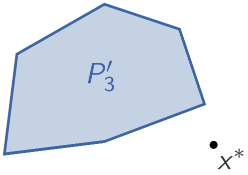


Exact approach

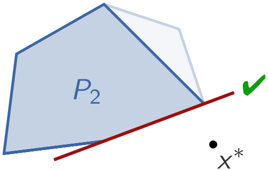


Oracle: Heuristic vs. Exact Approach

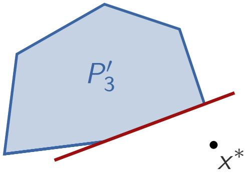
Heuristic approach



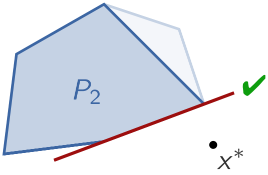
Exact approach



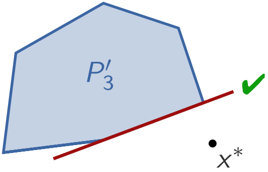
Heuristic approach



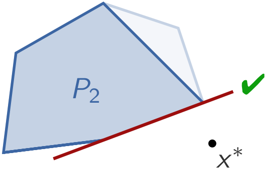
Exact approach



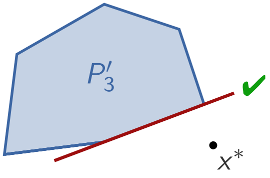
Heuristic approach



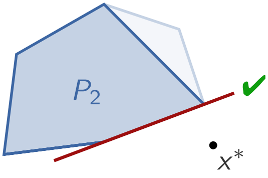
Exact approach



Heuristic approach



Exact approach



More but faster calls of the heuristic

vs.

Fewer but slower calls of the exact algorithm

Outline

Graph contraction / selection

- Use graph contraction to reduce the size of the initial graph.
- If contracted graph is too large, select a connected subgraph:
  - at random,
  - prefer edges with an LP value close to 0.5,
  - ...

Graph contraction / selection

- Use graph contraction to reduce the size of the initial graph.
- If contracted graph is too large, select a connected subgraph:
  - at random,
  - prefer edges with an LP value close to 0.5,
  - ...

Oracles for delayed row generation

- Exact algorithms:
  - **Branch & Cut**,
  - Branch & Bound using SDP relaxations.
- Heuristics:
  - **Kernighan-Lin** (multiple solution version),
  - Goemans-Williamson.

Test set

- Carried out a single B&C optimization of the problem `bqp250-1` with 250 nodes and 3339 edges [cf. Biq Mac Library].
- Extracted 42 intermediate LP solutions that were passed to the target cut separator.

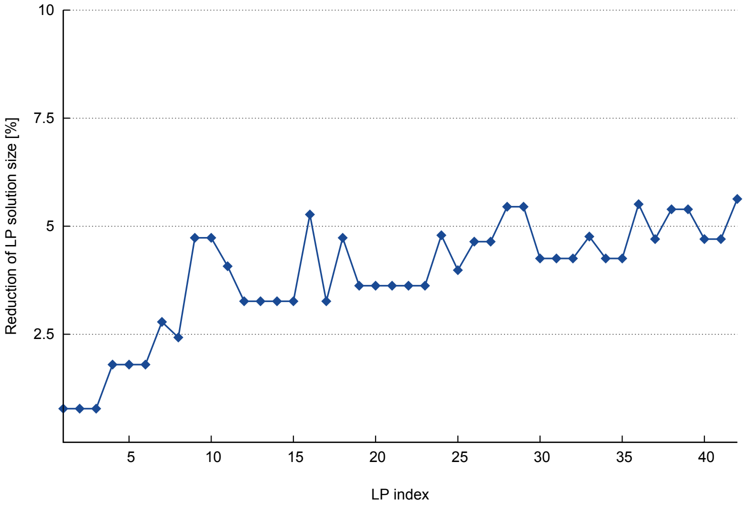
Test set

- Carried out a single B&C optimization of the problem `bqp250-1` with 250 nodes and 3339 edges [cf. Biq Mac Library].
- Extracted 42 intermediate LP solutions that were passed to the target cut separator.

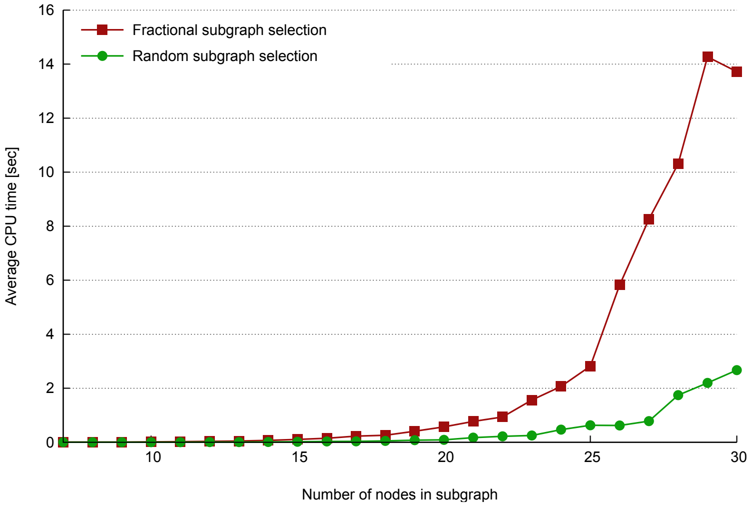
Measured quantities

- Average CPU time of the target cut separation over the 42 LP solutions.
- Rate of success, i. e., the percentage of the target cut separation attempts that found at least one cutting plane.

Reduction of LP Solution Size by Graph Contraction

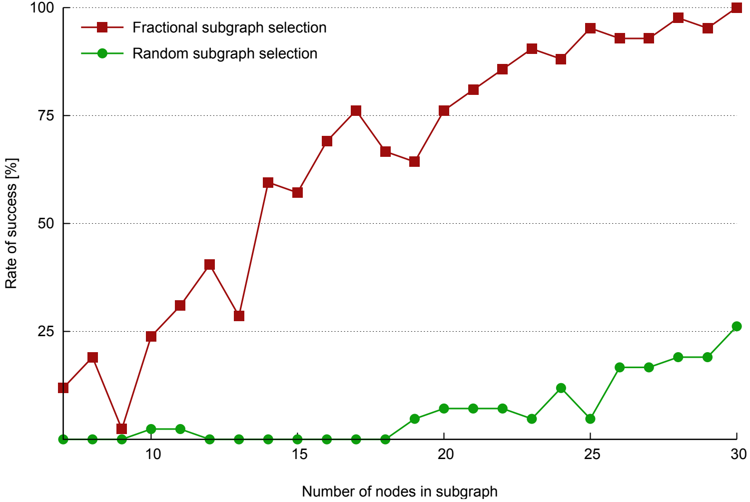


Average CPU Time with Delayed Row Generation

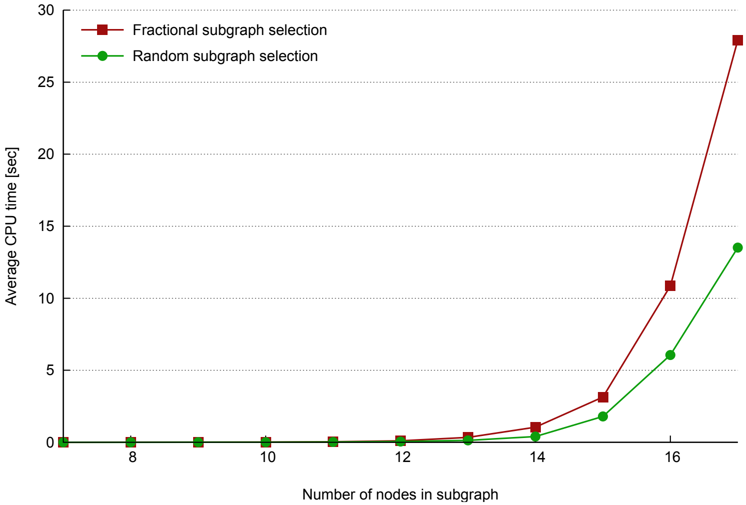


[Intel Xeon 2.8 GHz, 8GB shared RAM.]

Rate of Success with Delayed Row Generation

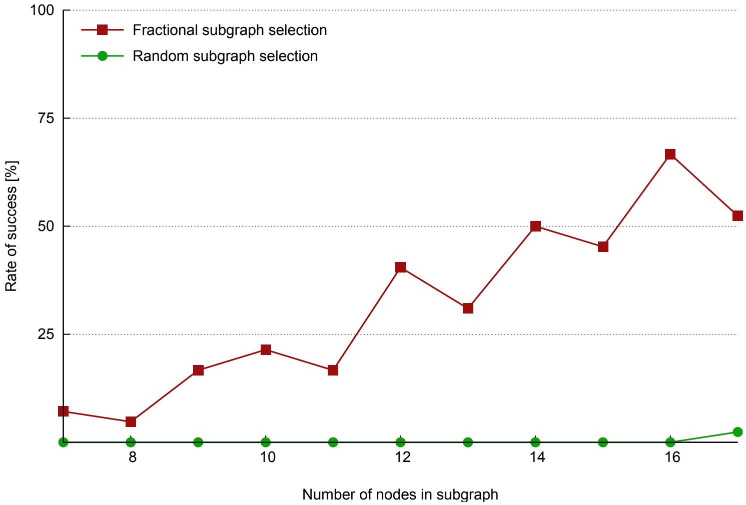


Average CPU Time without Delayed Row Generation



[Intel Xeon 2.8 GHz, 8GB shared RAM.]

Rate of Success without Delayed Row Generation



Conclusion

- Size of the subgraphs has to be small for fast separation.
  - ≤ 20 nodes with delayed row generation,
  - ≤ 14 nodes without delayed row generation.

Conclusion

- Size of the subgraphs has to be small for fast separation.
  - ≤ 20 nodes with delayed row generation,
  - ≤ 14 nodes without delayed row generation.
- Subgraph selection is a crucial factor.
  - Random subgraph selection: fast but unreliable.
  - Fractional subgraph selection: slow but more effective.

Conclusion

- Size of the subgraphs has to be small for fast separation.
  - ≤ 20 nodes with delayed row generation,
  - ≤ 14 nodes without delayed row generation.
- Subgraph selection is a crucial factor.
  - Random subgraph selection: fast but unreliable.
  - Fractional subgraph selection: slow but more effective.
- Small subgraph size can cause small impact of the generated cutting planes.

Conclusion

- Size of the subgraphs has to be small for fast separation.
  - ≤ 20 nodes with delayed row generation,
  - ≤ 14 nodes without delayed row generation.
- Subgraph selection is a crucial factor.
  - Random subgraph selection: fast but unreliable.
  - Fractional subgraph selection: slow but more effective.
- Small subgraph size can cause small impact of the generated cutting planes.
- **Open question**: How to detect the most suitable subgraphs?

Conclusion

- Size of the subgraphs has to be small for fast separation.
  - ≤ 20 nodes with delayed row generation,
  - ≤ 14 nodes without delayed row generation.
- Subgraph selection is a crucial factor.
  - Random subgraph selection: fast but unreliable.
  - Fractional subgraph selection: slow but more effective.
- Small subgraph size can cause small impact of the generated cutting planes.
- **Open question**: How to detect the most suitable subgraphs?

Thank you for your attention!