Heuristic and Exact Search in Mixed-Integer Programming

Gerald Gamrath and the SCIP team

Zuse Institute Berlin · gamrath@zib.de
SCIP Optimization Suite · http://scip.zib.de

International Workshop on Pattern Databases and Large-Scale Search
Zuse Institute Berlin · October 5, 2018

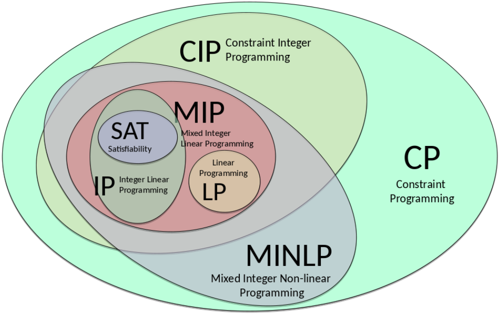
SPONSORED BY THE
Federal Ministry of Education and Research

Mixed-Integer Programming

General Form:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^I_{\geq 0} \times \mathbb{R}^C_{\geq 0} \end{aligned}$$

1. linear objective function
2. general linear constraints
3. general integer variables
4. continuous variables



How does this fit into this workshop?

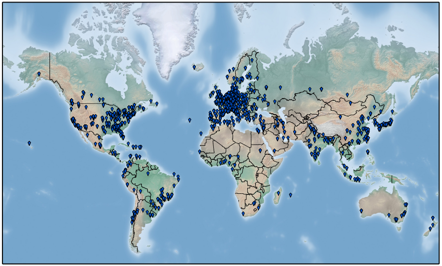
- no states?
- no actions?
- no clear goal state?
- no pattern databases!
- but still: a similar algorithm

SCIP: Solving Constraint Integer Programs

An open branch-cut-and-price framework with techniques from MIP, CP, SAT, and GO.

35+ active developers

- 10+ running Bachelor & Master projects
- 14+ running PhD projects
- 11 postdocs and professors



5 active development centers

- ZIB: SCIP, SoPlex, UG, ZIMPL
- TU Darmstadt: SCIP and SCIP-SDP
- FAU Erlangen-Nürnberg: SCIP
- RWTH Aachen & Uni. Lancaster: GCG

Many international contributors and users

- more than 14 000 downloads per year from 100+ countries

Careers

- 7 former developers are now building commercial optimization software at CPLEX, FICO Xpress, Gurobi, MOSEK, and GAMS
- 10 awards for Masters and PhD theses: MOS, EURO, GOR, DMV

Relaxations and bounds

A common approach for hard nonconvex optimization problems like MIP: compute bounds on the optimal value

$$\begin{aligned} z^* = \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^I_{\geq 0} \times \mathbb{R}^C_{\geq 0} \end{aligned}$$

Relaxations and bounds

A common approach for hard nonconvex optimization problems like MIP: compute bounds on the optimal value

$$\begin{aligned} z^* = \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^I_{\geq 0} \times \mathbb{R}^C_{\geq 0} \end{aligned}$$

1. Lower bound $L \leq z^*$: relaxation
   - in MIP: **LP relaxation**, $\mathbb{Z}^I \rightsquigarrow \mathbb{R}^I$
   - convex and “fast” to solve $\rightsquigarrow x^{\text{LP}}$

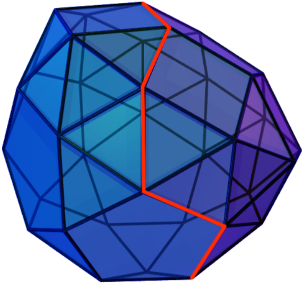
Relaxations and bounds

A common approach for hard nonconvex optimization problems like MIP: compute bounds on the optimal value

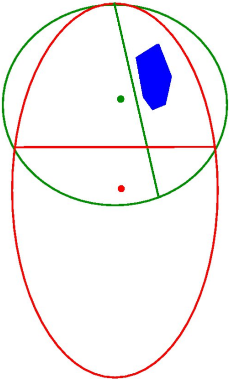
$$\begin{aligned} z^* = \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^I_{\geq 0} \times \mathbb{R}^C_{\geq 0} \end{aligned}$$

1. Lower bound $L \leq z^*$: relaxation
   - in MIP: **LP relaxation**, $\mathbb{Z}^I \rightsquigarrow \mathbb{R}^I$
   - convex and “fast” to solve $\rightsquigarrow x^{\text{LP}}$
2. Upper bound $U \geq z^*$: feasible solutions
   - if LP relaxation is “accidentally” feasible $\rightsquigarrow$ optimal solution
   - later: primal heuristics

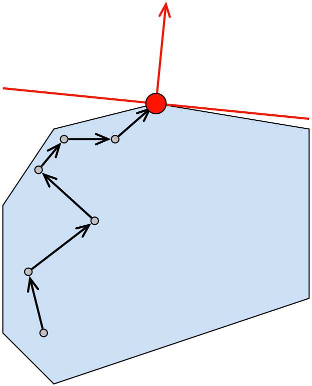
How to solve LPs?



Simplex algorithm



Ellipsoid method



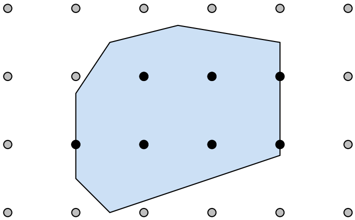
Interior point

LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree

Solution space

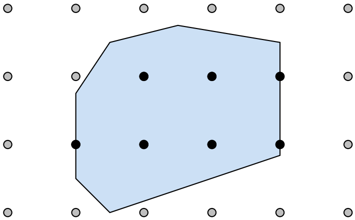


LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree

Solution space

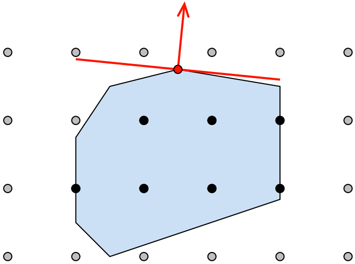


LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree

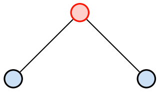
Solution space



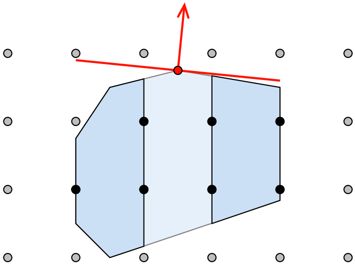
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



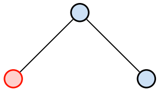
Solution space



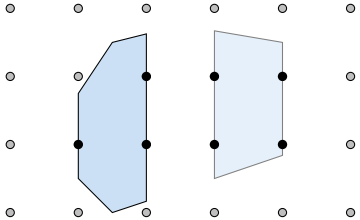
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



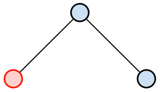
Solution space



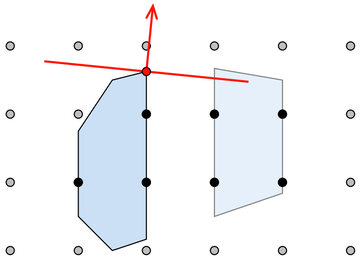
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



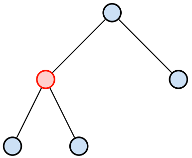
Solution space



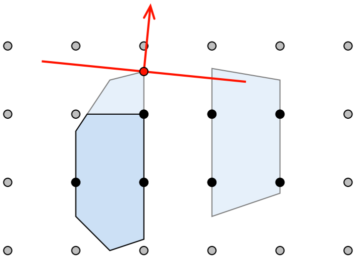
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



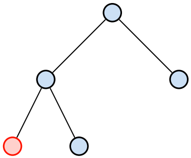
Solution space



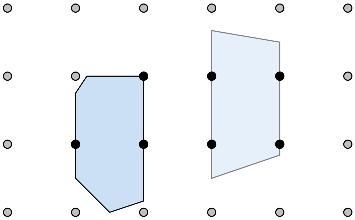
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



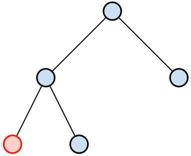
Solution space



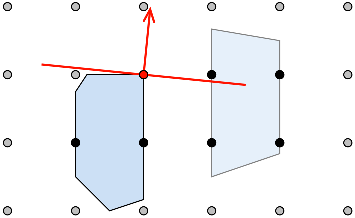
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



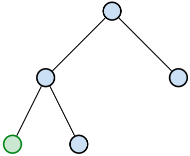
Solution space



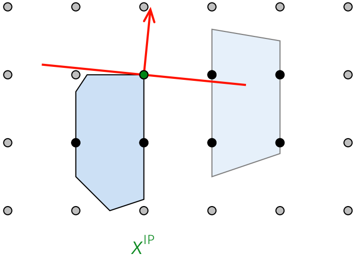
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



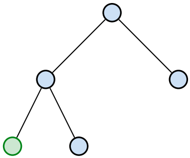
Solution space



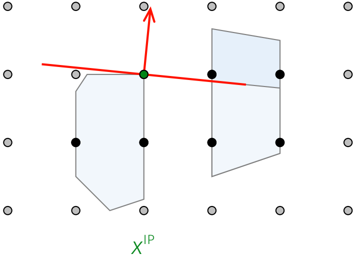
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



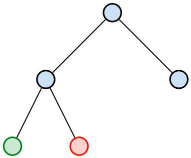
Solution space



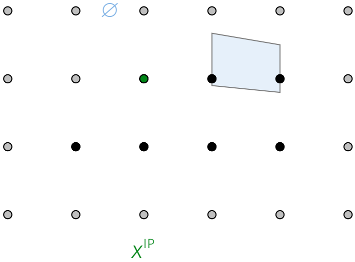
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



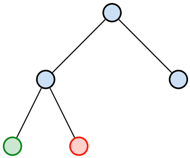
Solution space



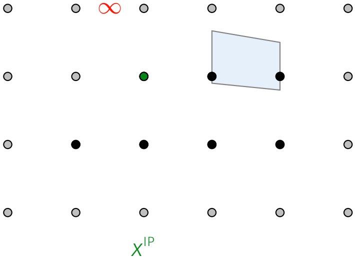
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



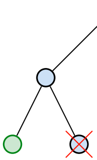
Solution space



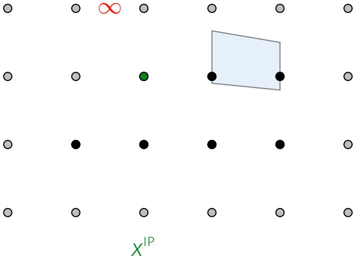
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



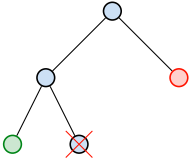
Solution space



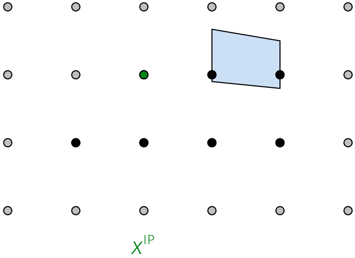
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



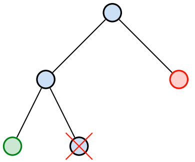
Solution space



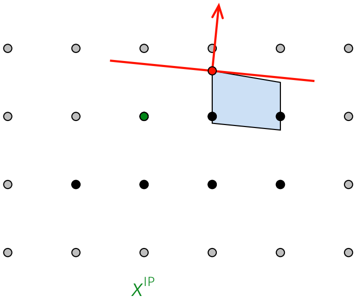
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



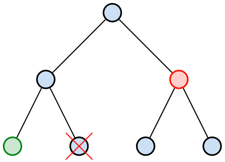
Solution space



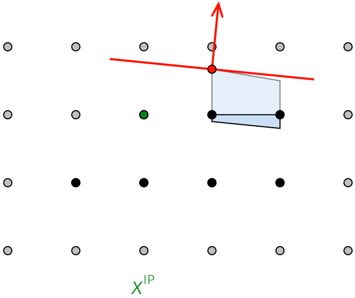
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



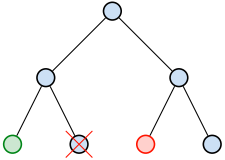
Solution space



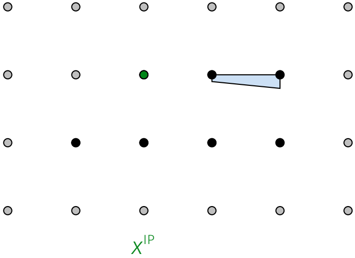
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



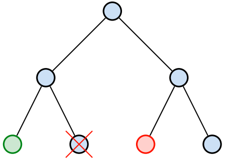
Solution space



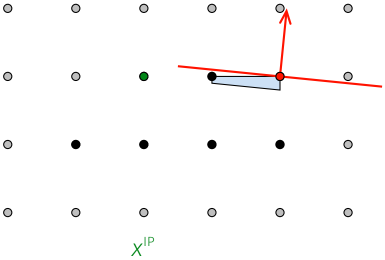
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



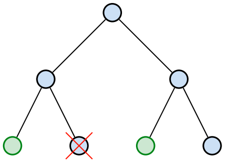
Solution space



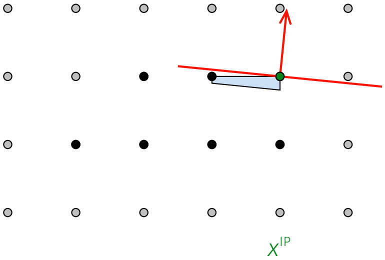
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



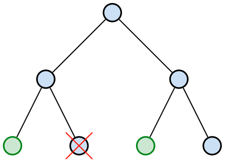
Solution space



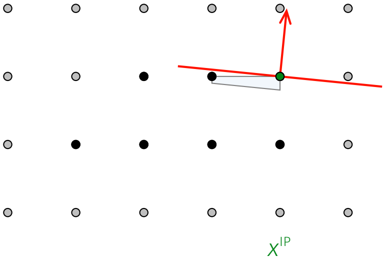
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



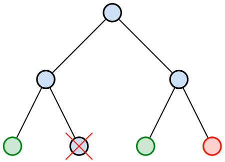
Solution space



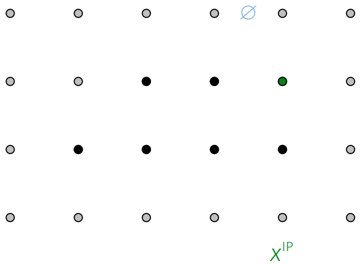
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



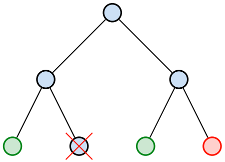
Solution space



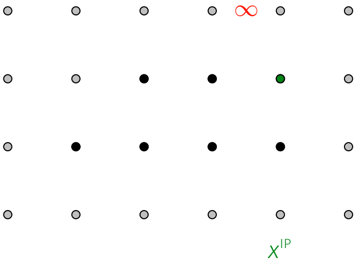
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



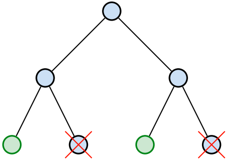
Solution space



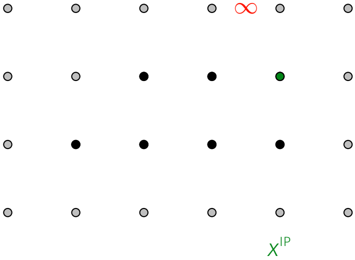
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



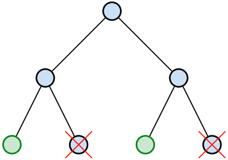
Solution space



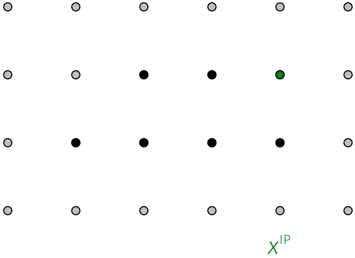
LP-based branch-and-bound

Systematic reduction of $U - L$ by divide-and-conquer (Land & Doig 1960, Dakin 1965)

Branch-and-bound tree



Solution space

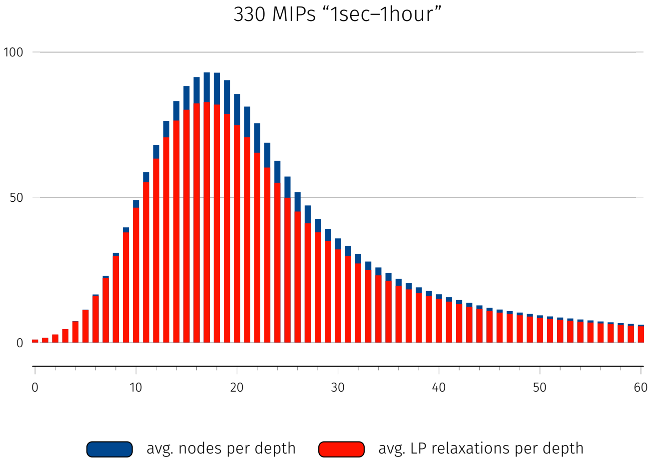


1. the union of leaf nodes contains all improving solutions
2. L = smallest LP bound over all leaf nodes: "best bound"
3. x^{LP} integer $\Rightarrow$ improve "incumbent" U
4. node LP infeasible or node LP bound $> U \Rightarrow$ prune
5. proven optimality gap $g = U - L \Rightarrow$ stop if this is ≤ 0

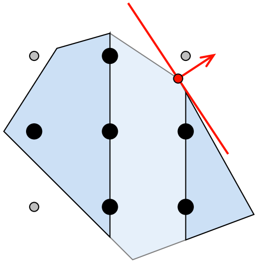
Dual simplex iterations during tree search

383.7/3.3 ≈ 116x speedup

depth	avg. iters
root	**383.7**
1	17.0
2	14.9
3	12.2
4	10.3
5	9.0
6	8.2
⋮	⋮
13	4.2
14	4.1
15	3.8
16	3.4
17	3.3
18	3.1
19	2.8
20	2.7
21	2.5
22	2.4
⋮	⋮



Branching rules



Task

- divide into (disjoint) subproblems
- improve local bounds
- dramatic performance impact.

Techniques

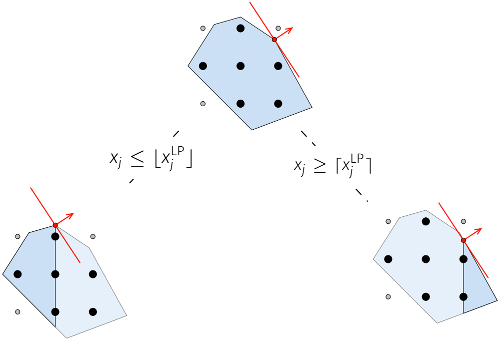
- branching on variables
  - most infeasible
  - least infeasible
  - random branching
  - strong branching
  - pseudocosts
  - reliability
  - VSIDS
  - hybrid reliability/inference
  - cloud branching
  - backdoor branching
  - ...
- branching on constraints
  - SOS1
  - SOS2
  - multiaggregated variables
  - general disjunctions

Dual gain

Branching children/descendants:

$$P_j^- := P \cap \{x_j \leq \lfloor x_j^{\mathrm{LP}} \rfloor\}, P_j^+ := P \cap \{x_j \geq \lceil x_j^{\mathrm{LP}} \rceil\}$$

Dual gain: LP objective between a descendant and its parent node P:



$$\Delta c_j^* := \min\{c^T x : x \in P_j^*\} - \min\{c^T x : x \in P\} \geq 0, \quad * \in \{-, +\}$$

Scoring function

Selecting fractional candidates based on scores for individual directions

$$s^{-} := \Delta c_j^{-}, s^{+} := \Delta c_j^{+} \forall j \in \mathcal{F}$$

requires scoring function:
$s(s^{-}, s^{+})$: $\mathbb{R}^2_{\geq 0} \to \mathbb{R}_{\geq 0}$

Possibilities:

- Weighted sum for $\lambda \in [0, 1]$:

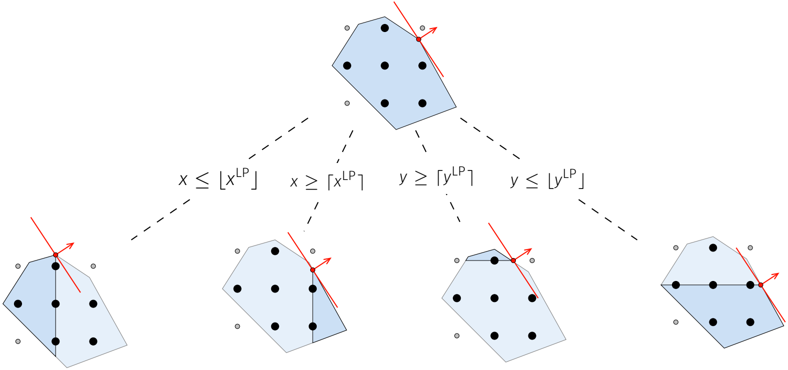
$$s(s^{-}, s^{+}) := \lambda \max\{s^{-}, s^{+}\} + (1 - \lambda) \min\{s^{-}, s^{+}\}$$

- Product for small $\epsilon > 0$:

$$s(s^{-}, s^{+}) := \max\{s^{-}, \epsilon\} \cdot \max\{s^{-}, \epsilon\}$$

Lookahead: strong branching (Gauthier & Ribiere 1977)

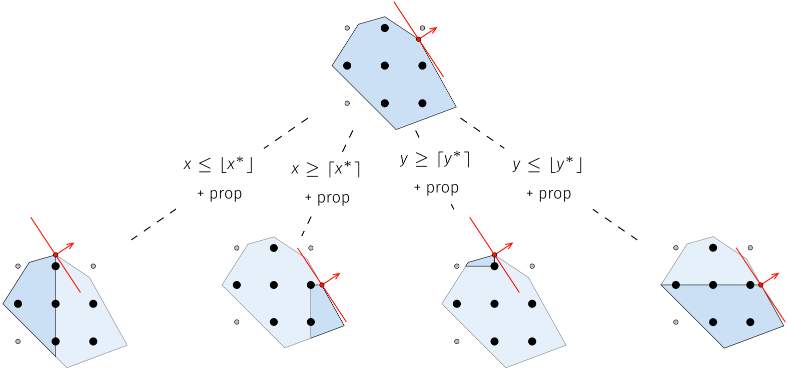
1. Perform an explicit look-ahead by solving all possible descendants of the current node.



2. Select a fractional variable $j \in \underset{j' \in \mathcal{F}}{\operatorname{argmax}} \{s\{\Delta c_{j'}^{-}, \Delta c_{j'}^{+}\}\}$.

Lookahead: strong branching with domain propagation (G. 2014)

1. Perform an explicit look-ahead by solving all possible descendants of the current node.



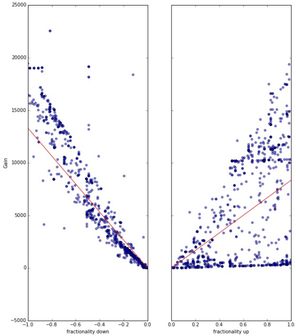
2. Select a fractional variable $j \in \underset{j' \in \mathcal{F}}{\operatorname{argmax}}\{s\{\Delta c_{j'}^-, \Delta c_{j'}^+\}\}$.

Lookback: pseudocosts (Benichou et al. 1971)

Estimate for objective gain based on past branching observations.

- unit gain:
  computed from fractionalities f_j^* and LP gains
- pseudocosts Ψ_j^*:
  average unit gain of branching history
- branching decision based on estimated gains:

$$s(f_j^- \Psi_j^-, f_j^+ \Psi_j^+)$$



Select a fractional variable $j \in \operatorname{argmax}_{j' \in \mathcal{F}} \{s(f_j^- \Psi_j^-, f_j^+ \Psi_j^+)\}$.

Combinations

Pseudocosts are uninitialized at the beginning of the search.

Reliability branching (Achterberg et al. 2004)

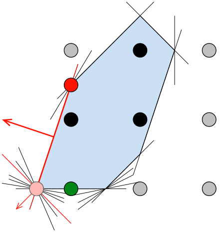
1. Determine the set of fractional variables $\mathcal{F} \neq \emptyset$.
2. Split $\mathcal{F}$ into reliable subset $\mathcal{F}^{\text{rel}}$ and unreliable subset $\mathcal{F}^{\text{url}}$.
3. Perform strong branching for all $j \in \mathcal{F}^{\text{url}}$
4. Record unit gains and update pseudocosts
5. Compare the best strong branching result with the best pseudocost prediction for the branching decision.

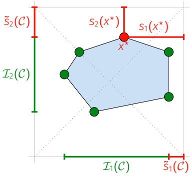
State-of-the-art: Hybrid branching (Achterberg & Berthold 2009)

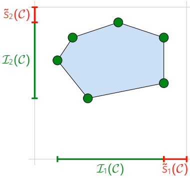
1. combine reliability branching with other branching scores:
    - cutoff information
    - inference information
    - conflict information
2. take degeneracy into account (G. et al. 2018)

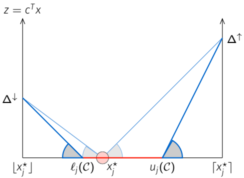
Cloud branching (G. et al. 2018)

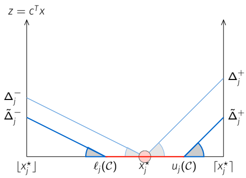
- LP solutions are typically (dual) degenerate
- multiple LP optima exist
- “the” optimal LP solution returned by the LP solver is more or less random
- compute a “cloud” $\mathcal{C}$ of alternative LP optima
- guide branching by this set of LP solutions rather than a single one
- new branching rule using only cloud information and modifications to most existing ones







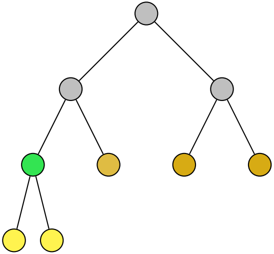




Node Selection

Basic rules

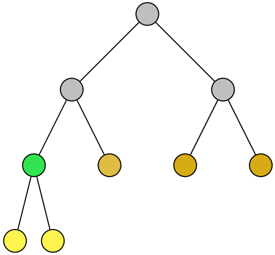
- depth first search (DFS)
  → exploit hot-start



Node Selection

Basic rules

- depth first search (DFS)
  $\rightarrow$ exploit hot-start
- best bound search (BBS)
  $\rightarrow$ improve dual bound



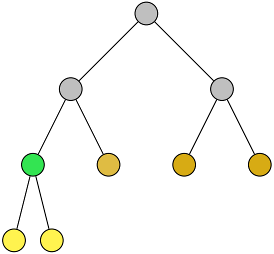
Best bound search

- select node with smallest lower bound
- empirically leads to fewest number of nodes
- lower bound is an admissible heuristic fucntion
- if LP bounds of all children are computed in advance, this is similar to $A^\star$

Node Selection

Basic rules

- depth first search (DFS)
  $\rightarrow$ exploit hot-start
- best bound search (BBS)
  $\rightarrow$ improve dual bound
- best estimate search (BES)
  $\rightarrow$ improve primal bound



Best estimate (Benichou et al. 1971)

Use learned pseudo costs to estimate objective value

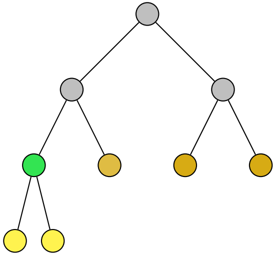
$$\hat{c} := c^T x_j^{\text{LP}} + \sum_{j \in \mathcal{F}} \min\{f_j^- \boldsymbol{\Psi}_j^-, f_j^+ \boldsymbol{\Psi}_j^+\}$$

of the best solution in the subtree rooted at a node with LP solution x^{LP}.

Node Selection

Basic rules

- depth first search (DFS)
  → exploit hot-start
- best bound search (BBS)
  → improve dual bound
- best estimate search (BES)
  → improve primal bound



Best estimate (Benichou et al. 1971)

Use learned pseudo costs to estimate objective value

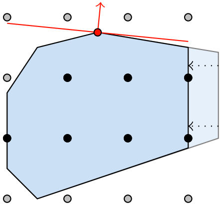
$$\hat{c} := c^T x_j^{\mathrm{LP}} + \sum_{j \in \mathcal{F}} \min\{f_j^- \Psi_j^-, f_j^+ \Psi_j^+\}$$

of the best solution in the subtree rooted at a node with LP solution x^{LP}.

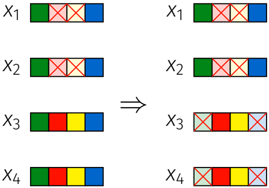
Usually best bound/estimate interleaved with DFS plunges to find solutions earlier.

Branch-and-bound is accelerated by many more techniques...

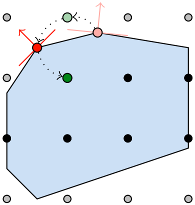
Presolving



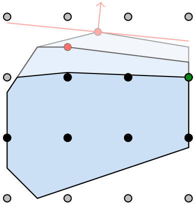
Domain Propagation



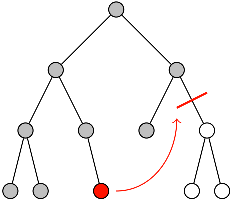
Primal Heuristics



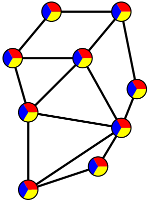
Cutting Planes



Conflict Analysis



Symmetry Handling

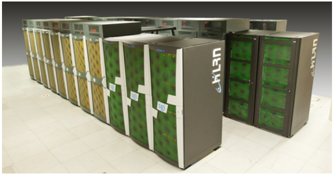


ug[SCIP] — the parallel version of SCIP

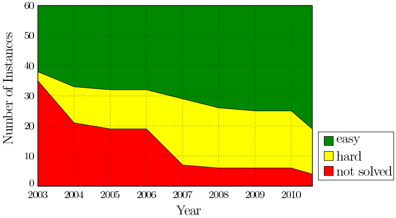
Some facts and results:

- shared (“FiberSCIP”) and distributed memory version (“ParaSCIP”)
- solves MIP and MINLP
- successful runs with up to 80.000 SCIP solvers
- solved 2 previously unsolved MIPLIB 2003 instances
  - ds: 4096 cores, about 76 hours, 3 billion nodes
  - stp3d: 7186 cores, about 33 hours, 10 million nodes (optimal solution given)
- and many MIPLIB 2010 instances

HLRN II:



MIPLIB 2003:



Conclusions

MIP solving:

- basic algorithm: branch-and-bound tree search
- good bounds are provided by LP relaxation
- accelerated by a bag of tricks

Discussion:

- what can we learn from each other?
- use MIP techniques for search?
- use search within MIP solver components?

Thank you for your attention!