# General disjunction branching based on objective function improvement

Gregor Hendel, Stephen J. Maher, Yuji Shinano

Zuse Institute Berlin

Berlin, Germany

18th May 2016

# Partitioning in MIP

- Consider the general mixed integer programming problem

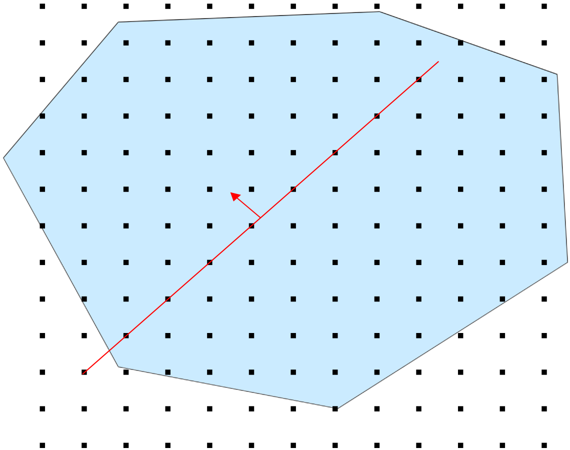
$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \\ & x_j \in \mathbb{Z}^+ \quad \forall j \in J \end{aligned}$$

  where $J \subseteq N = \{1, \ldots, n\}$.
- State-of-the-art mixed integer programming solvers employ a branch-and-cut algorithms.
- Many key features of MIP solvers rely on the partitioning of the feasible region, most importantly
  - Branching
  - Primal heuristics

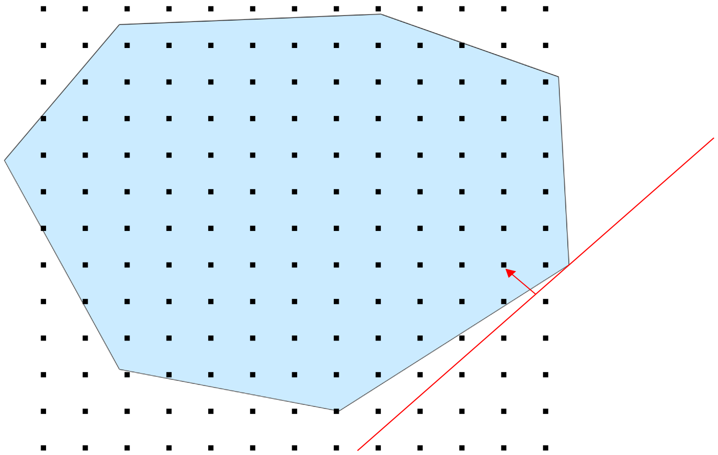
# Partitioning by branching

- Branching is one of the most important ingredients to a MIP solver.
- Typically, branching is used to partition the feasible region.
- Goal is to eliminate integer infeasible solutions.
- MIP solvers consist of many different branching rules to select the best partition.

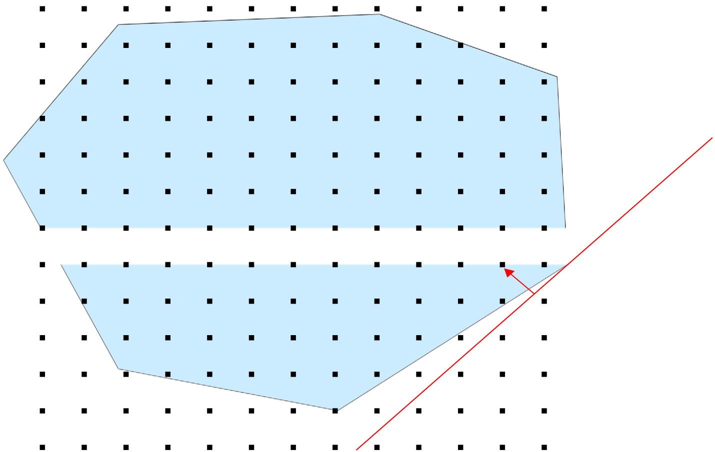
# Partitioning by branching



# Partitioning by branching



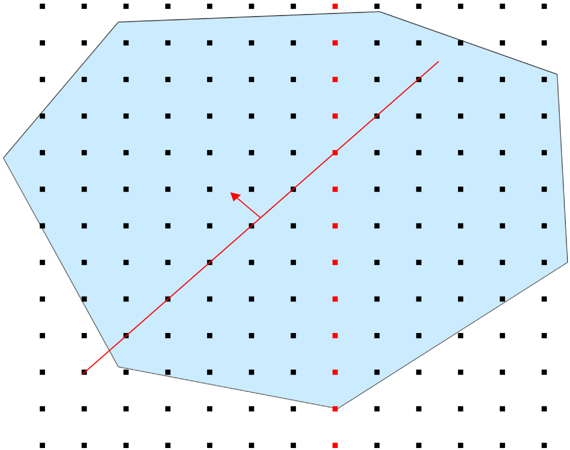
# Partitioning by branching



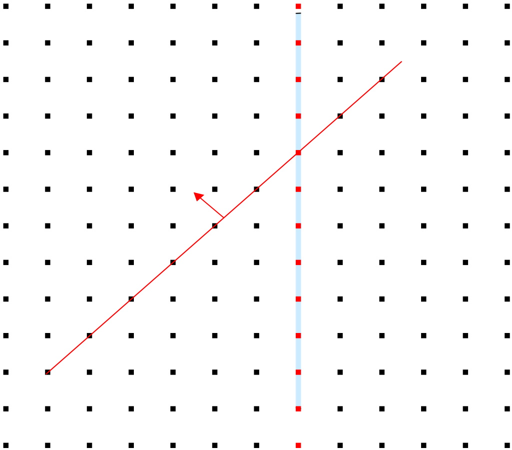
# Partitioning by primal heuristics

- Primal heuristics are valuable in quickly identifying primal feasible solutions.
- Large neighbourhood search heuristics work on the belief that a restricted problem is easier to solve than the original.
  - Local branching
  - Proximity search
  - RENS
  - RINS
  - ...
- Purpose of the restriction is not to provide an exact partition, but form a promising feasible region.

# Partition by primal heuristics



# Partition by primal heuristics



# Combining partitioning ideas

## Research question

Is it possible to combine the concepts from large-neighbourhood search heuristics to develop an alternative partitioning approach for branching?

# Combining partitioning ideas

## Research question

Is it possible to combine the concepts from large-neighbourhood search heuristics to develop an alternative partitioning approach for branching?

- Local branching
- Proximity search
- Branching on general disjunctions

# Combining ideas

## Local branching

- Given an incumbent solution $x^*$, define two subproblems by adding the invalid constraints

$$\sum_{j \in J : x_j^* = 0} x_j + \sum_{j \in J : x_j^* = 1} (1 - x_j) \leq k \quad \text{and} \quad \sum_{j \in J : x_j^* = 0} x_j + \sum_{j \in J : x_j^* = 1} (1 - x_j) \geq k + 1.$$

## Proximity search

- Forcing objective function improvement through the addition of the constraint

$$c^T x \leq c^T x^* - \gamma.$$

## Branching on general disjunctions

- A valid disjunction is given by

$$\pi^T x \leq \pi_0 \quad \text{and} \quad \pi^T x \geq \pi_0 + 1.$$

  when $(\pi, \pi_0)$ are all integer.

# Combining ideas

- Both local branching and proximity search add constraints that can be valid general disjunctions for the original problem.
- The objective function constraint from proximity search is valid if the objective function is always integer.

**However,**

- Constraints parallel to the objective function can greatly impact the performance of MIP solvers.

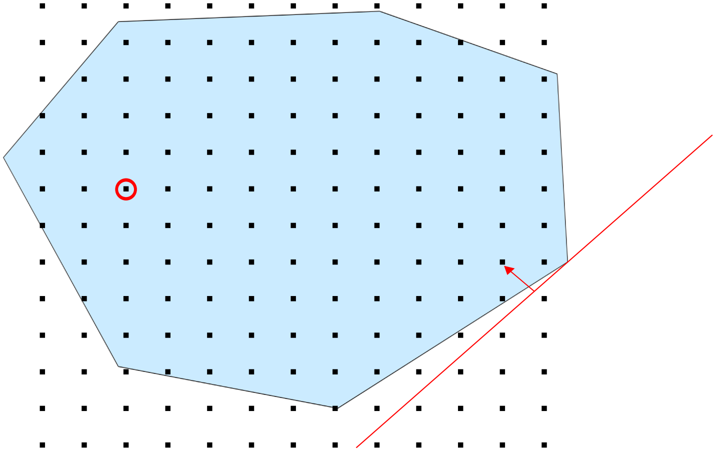
# Proposed branching scheme

- Forcing objective function improvement through objective function disjunctions
- Given an incumbent solution $x^*$, partition the feasible region with the addition of the constraints

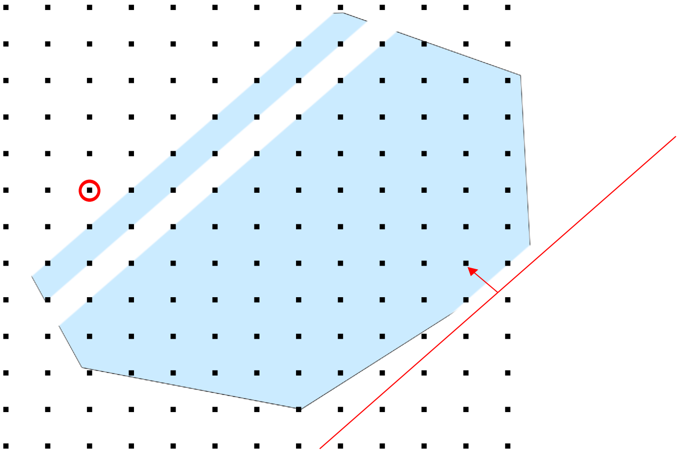
$$\lceil c^T x^{LP} \rceil \leq c^T x \leq c^T x^* - 3 \quad \text{and} \quad c^T x^* - 2 \leq c^T x \leq c^T x^* - 1$$

- Only possible for problems with an integer objective function (after presolving).

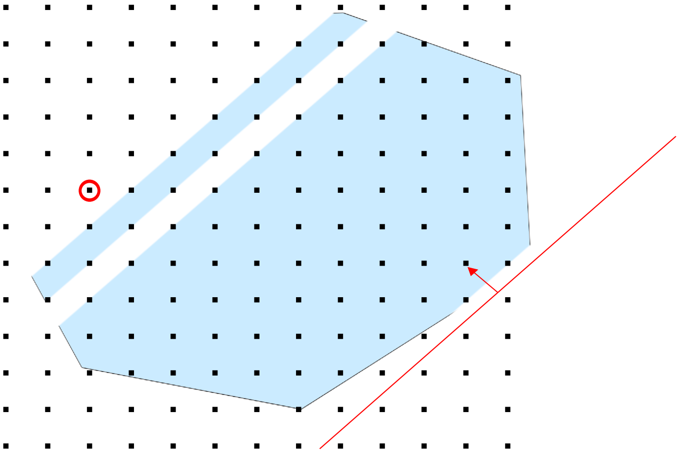
# Proposed branching scheme



# Proposed branching scheme



# Proposed branching scheme

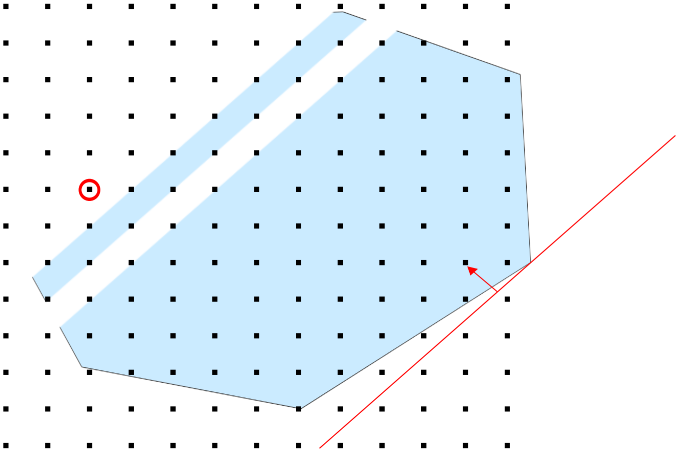


- *Concern:* Constraint parallel to objective function can increase dual degeneracy.

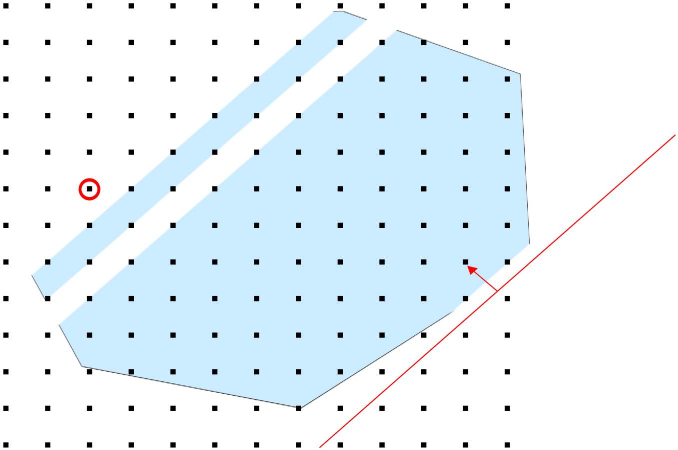
# Implementation details

- Implemented as a branching rule within the solver SCIP.
- Executed after the LP relaxation at a node is solved to optimality.
  - Requires an incumbent solution.
  - Delay based on nodes is imposed.
- Default node selector is used.
- Interleaves with the tactical branching rules within SCIP.

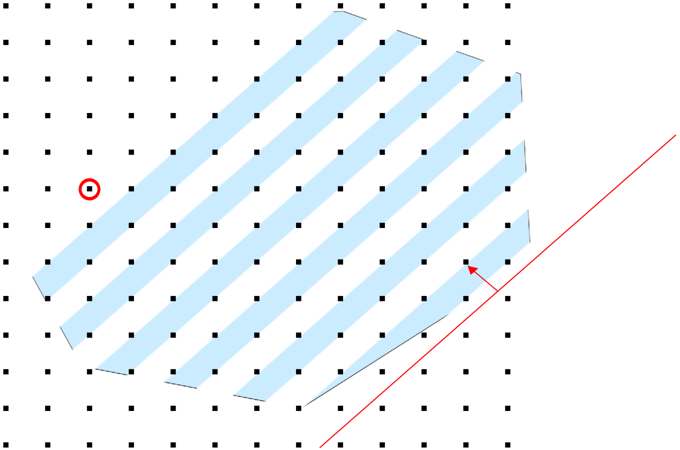
## Single partition



# Double partition



## Multiple partitions



# Test set

- Instances collected from the benchmark test sets of MIPLIB 3, MIPLIB 2003 and MIPLIB 2010.
- Only included instances with an integer objective function after the presolving.
- Excluded instances that were not solved by SCIP using default settings or the proposed branching scheme within a time limit of 7200 seconds.

**Comparison - shifted geometric mean**

- Time
- Nodes
- Primal integral

## Nodes

| | Single | Single - delay | Double | Double - delay |
|---|---|---|---|---|
| All optimal | 1.2328 | 1.0907 | 1.1933 | 1.0439 |
| Nodes: 0-100 | 1.2875 | 1 | 1.026 | 1 |
| Nodes: 101-1000 | 1.3389 | 1.3586 | 1.3083 | 1.2223 |
| Nodes: 1001-10000 | 2.0662 | 1.281 | 1.8838 | 1.0958 |
| At least 10001 nodes | 1.0941 | **0.9996** | 1.0657 | **0.9992** |

## Solving time

| | Single | Single - delay | Double | Double - delay |
|---|---|---|---|---|
| All optimal | 1.2165 | 1.0634 | 1.2115 | 1.0499 |
| Nodes: 0-100 | 1.0252 | 1.0153 | **0.9853** | 1 |
| Nodes: 101-1000 | 1.2719 | 1.1231 | 1.2056 | 1.071 |
| Nodes: 1001-10000 | 1.7771 | 1.1713 | 1.9112 | 1.1097 |
| At least 10001 nodes | 1.12 | 1.0291 | 1.0996 | 1.0502 |

## Primal Integral

| | Single | Single - delay | Double | Double - delay |
|---|---|---|---|---|
| All optimal | 1.1113 | 1.0052 | 1.101 | **0.9793** |
| Nodes: 0-100 | 1.009 | 1.0064 | **0.984** | **0.9884** |
| Nodes: 101-1000 | 1.1804 | 1.0095 | 1.2007 | 1.002 |
| Nodes: 1001-10000 | 1.2733 | 1.0097 | 1.2951 | 1.0071 |
| At least 10001 nodes | 1.045 | 1.0006 | 1.0056 | **0.9403** |

# Parallelisation

## Motivation

Assess whether additional presolving can improve the performance branching scheme.

- Additional presolve can decrease the performance impact of the objective constraint.
- Greater number of independent branch-and-bound trees are created.

# Results

| | Max Time | Mean Nodes | Mean Primal Integral |
|---|---|---|---|
| aflow40b | 1.2909 | 1.607 | 1.0995 |
| bnatt350 | 1.0008 | **0.9806** | **0.9975** |
| cov1075 | 1.1568 | **0.9473** | 1.0174 |
| csched010 | 1.047 | **0.9971** | 1.071 |
| dfn-gwin-UUM | 1.0414 | 1.0739 | 1.0481 |
| iis-100-0-cov | **0.671** | 1.0146 | 1.0574 |
| iis-bupa-cov | **0.9728** | 1.0179 | 1.0919 |
| iis-pima-cov | 1.2048 | **0.8346** | **0.9503** |
| m100n500k4r1 | 1.3179 | 1.0376 | 1.0823 |
| mcsched | 1.5268 | 2.1608 | 1.0573 |
| mik-250-1-100-1 | 1.7258 | 1.3224 | 1 |
| mzzv11 | 1.1707 | **0.5807** | 1.0513 |
| neos-1601936 | 1.3234 | 1.7193 | 1.6239 |
| neos-849702 | 1.0019 | 1.0811 | **0.9978** |
| net12 | **0.9412** | 1.1927 | 1.3687 |
| noswot | 1.1644 | **0.8475** | 1 |
| ns1208400 | 1.528 | 1.81 | **0.8866** |
| ns1766074 | 1.0146 | **0.9946** | 1.0119 |
| ns1830653 | 1.0155 | 1.1641 | **0.9591** |
| nsrand-ipx | 1.3219 | 1.1014 | 1.0868 |
| pw-myciel4 | 1.2649 | **0.906** | **0.9943** |
| rococoC10-001000 | **0.9339** | 1.1053 | **0.9807** |
| roll3000 | 1.4606 | 1.2444 | **0.9695** |
| rout | 1.0773 | 1.0304 | 1.0684 |
| stein45 | 1.1975 | 1.0804 | 1 |
| zib54-UUE | 1.0625 | 1.0135 | 1.0184 |

# Conclusion

- Discussed the concept of partitioning by branching and primal heuristics.
- Presented an exact solution approach based upon LNS heuristics and branching on general disjunctions.
- Demonstrated an improvement in the primal integral in the sequential setting.
- Described and evaluated the potential use of the approach in a parallel setting.