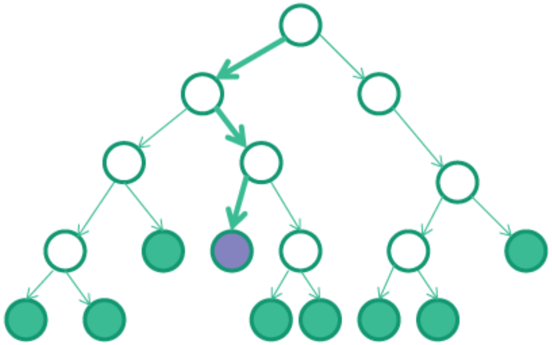
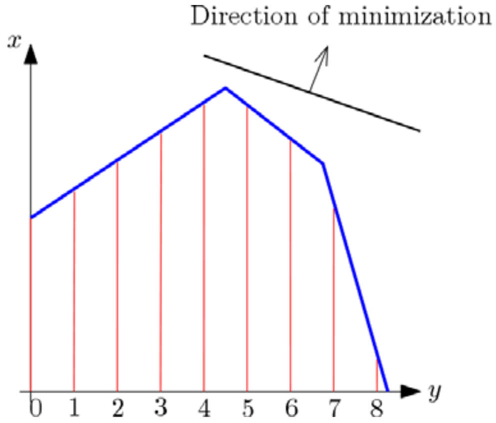
NTNU – Trondheim
Norwegian University of
Science and Technology

MILP algorithms: branch-and-bound and branch-and-cut

Content

- The Branch-and-Bound (BB) method.
  - the framework for almost all commercial software for solving mixed integer linear programs
- Cutting-plane (CP) algorithms.
- Branch-and-Cut (BC)
  - The most efficient general-purpose algorithms for solving MILPs





Basic idea of Branch-and-bound

BB is a divide and conquer approach: break problem into subproblems (sequence of LPs) that are easier to solve

Consider MILP:

$$\begin{aligned} J^* = \min_{(x,y)} \quad & c^T x + d^T y \\ \text{s.t.} \quad & (x,y) \in X \end{aligned}$$

where X is the set of feasible solutions,

$$X = \left\{(x,y) \in \mathbb{R}^n_+ \times \mathbb{Z}^p_+ : Ax + By \geq b\right\}$$

Let $X = X_1 \cup X_2 \cup \ldots \cup X_K$ be a decomposition of the feasible solution set X into smallers sets X_k, and let $J^k = \min\{c^T x + d^T y : (x,y) \in X_k\}$ for $k = 1, \ldots, K$. Then $J^* = \max_k J^k$.

(Wolsey (1998), Prop. 7.1).

Decomposing the initial formulation *P*

Let $(x^{\mathrm{R}}, y^{\mathrm{R}}) \in P$ be the solution of the initial LP relaxation,

$$\begin{aligned} J_{\mathrm{R}} = \min_{(x,y)} \quad & c^T x + d^T y \\ \text{s.t.} \quad & (x,y) \in P \\ & P = \left\{(x,y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ : Ax + By \geq b\right\} \end{aligned} \tag{1}$$

If $y \notin \mathbb{Z}$, i.e. some y_j is fractional, we try to eliminate this solution by decomposing the formulation in terms of adding bounds on integer variables.

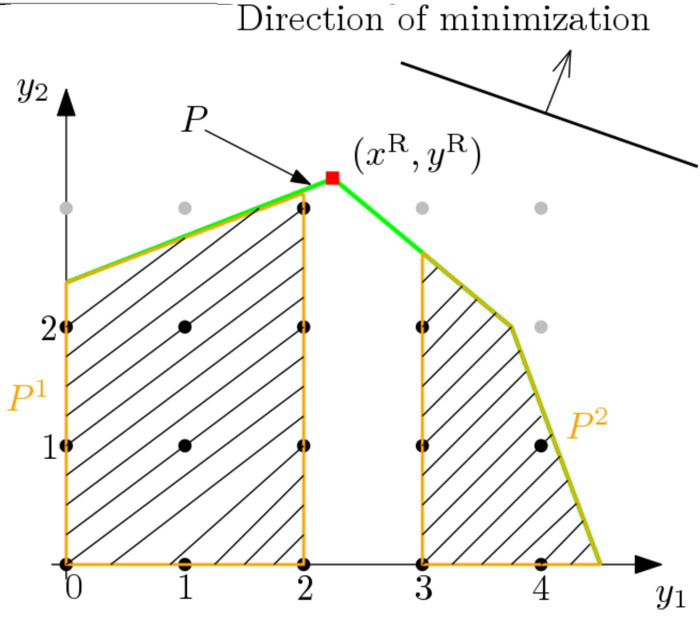
Let $y_j^{\mathrm{R}} \notin \mathbb{Z}$ be a chosen variable that is fractional in LP solution. A feasible solution $(x,y) \in X$ must then satisfy

$$y_j \leq \lfloor y_j^{\mathrm{R}} \rfloor \qquad \text{or} \qquad y_j \geq \lceil y_j^{\mathrm{R}} \rceil$$

We can then search for the optimal solution in the two disjoint sets

$$\begin{aligned} P^1 &:= P \cap \{y : y_j \leq \lfloor y_j^R \rfloor\}, \\ P^2 &:= P \cap \{y : y_j \geq \lceil y_j^R \rceil\}, \end{aligned}$$

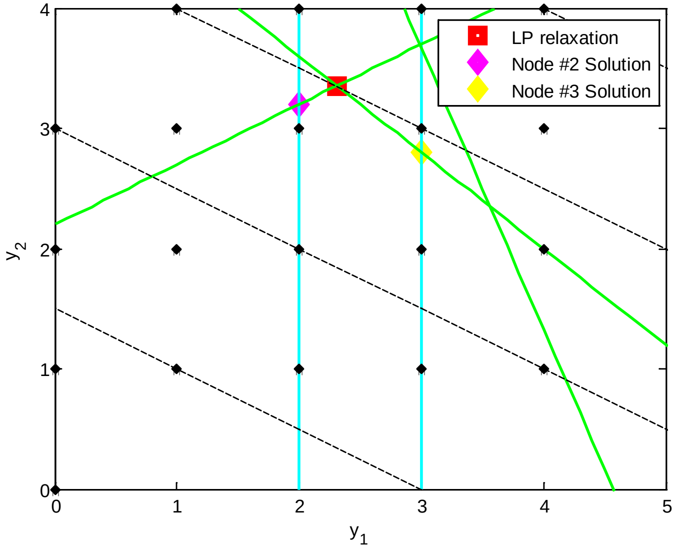
by solving *two* new LP relaxations.

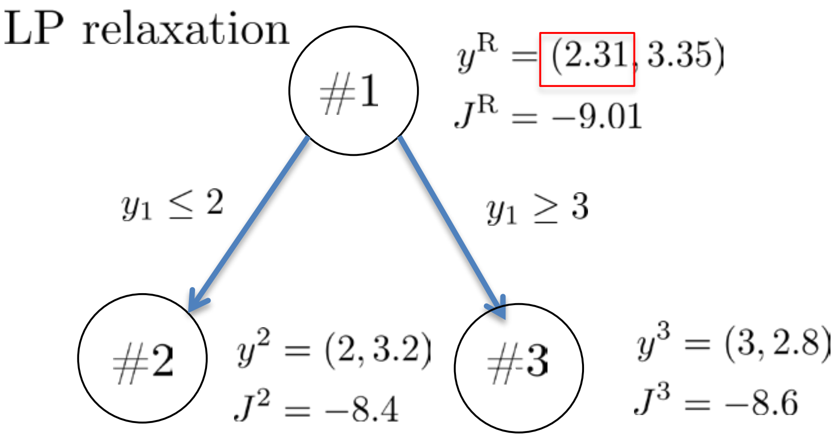


Enumeration tree

IP example 1:

$$
\begin{aligned}
J^* = \min_{y} \quad & -1y_1 - 2y_2 \\
\text{s.t.} \quad & -\frac{1}{2}y_1 + y_2 \leq \frac{11}{5} \\
& 4y_1 + 5y_2 \leq 26 \\
& 7y_1 + 3y_2 \leq 32 \\
& y \in \mathbb{Z}_+^2
\end{aligned}
$$





LP relaxation

$\#1$

$y^{\text{R}} = (\boxed{2.31}, 3.35)$

$J^{\text{R}} = -9.01$

$y_1 \leq 2$

$y_1 \geq 3$

$\#2$

$y^2 = (2, 3.2)$

$J^2 = -8.4$

$\#3$

$y^3 = (3, 2.8)$

$J^3 = -8.6$

How to proceed without complete enumeration?

Implicit enumeration: Utilize solution bounds

Let (x^*, y^*) optimal solution with objective value J^*.

$$\begin{aligned} J^* = \min_{(x,y)} \quad & c^T x + d^T y \\ \text{s.t.} \quad & Ax + By \geq b \\ & (x,y) \in \mathbb{R}^n_+ \times \mathbb{Z}^p_+ \end{aligned}$$

- LP relaxation is a convex problem: A lower bound on J^* is provided by the LP relaxation with objective value J_{R}.
- Any *integer* feasible solution, $(\bar{x}, \bar{y})$ with objective value $\bar{J}$, provides an upper bound on J.

Consequently, we have a lower and an upper bound on J^*:

$$\boxed{J_{\mathrm{R}} \leq J^* \leq \bar{J}}$$

Defines the **duality gap**:

$$\mathrm{DG} := [100\%] \cdot \frac{|\bar{J} - J_{\mathrm{LB}}|}{|\bar{J}|}$$

where J_{LB} is the best lower bound on J^*.

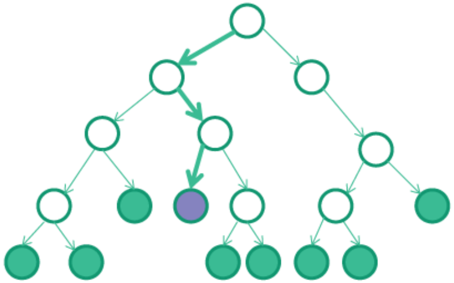
Pruning (= beskjæring av tre)

Utilize convexity of the LP relaxations to prune the enumeration tree.

1. Pruning by optimality : A solution is integer feasible; the solution cannot be improved by further decomposing the formulation and adding bounds.
2. Pruning by bound: A solution J^i in a node *i* is worse than the best known upper bound, i.e. $J^i \geq \bar{J}$
3. Pruning by infeasibility : A solution is (LP) infeasible.

(Remember lower and upper bounds on J):

$$\boxed{J_{\mathrm{R}} \leq J^* \leq \bar{J}}$$

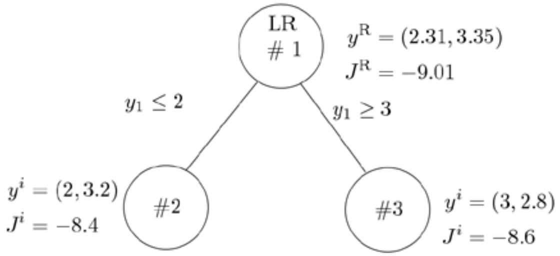


Branching: choosing a fractional variable

If $J^i < \bar{J}$ and $y^i \notin \mathbb{Z}_+^p$ after obtaining the LP solution in a node, the branch cannot be pruned.

$$\Downarrow$$

We need to divide (or branch) the formulation further



Which variable to choose?

Branching rules

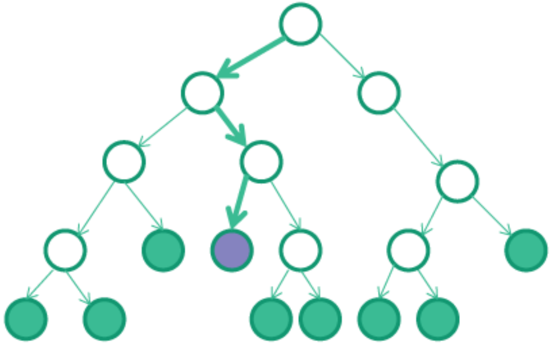
- Most fractional variable: branch on variable with fractional part closest to 0.5.
- Strong branching: tentative branch on each fractional variable (by a few iterations of the dual simplex) to check progress before actual branching is performed.
- Pseudocost branching: keep track of success variables already branched on.
- Branching priorities.

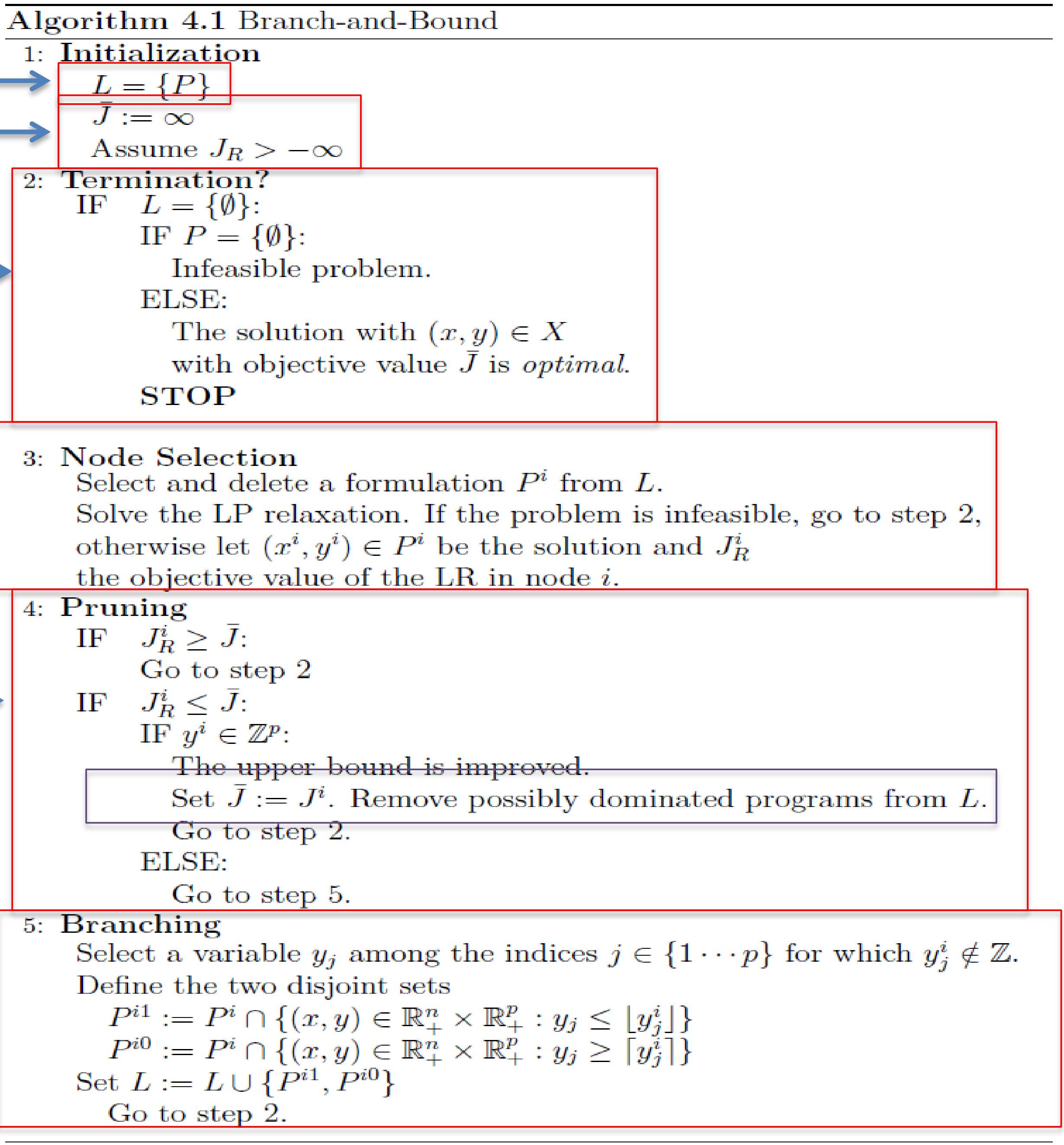
Node selection

Each time a branch cannot be pruned, two new children-nodes are created.

Node selection rules concerns which node (and hence which LP) to solve next:

- Depth-first search.
- Breath-first search.
- Best-bound search.
- Combinations.





```
Algorithm 4.1 Branch-and-Bound
1: Initialization
     $L = \{P\}$
     $\bar{J} := \infty$
     Assume $J_R > -\infty$
2: Termination?
     IF   $L = \{\emptyset\}$:
          IF $P = \{\emptyset\}$:
             Infeasible problem.
          ELSE:
             The solution with $(x, y) \in X$
             with objective value $\bar{J}$ is optimal.
          STOP
3: Node Selection
     Select and delete a formulation $P^i$ from $L$.
     Solve the LP relaxation. If the problem is infeasible, go to step 2,
     otherwise let $(x^i, y^i) \in P^i$ be the solution and $J_R^i$
     the objective value of the LR in node $i$.
4: Pruning
     IF   $J_R^i \geq \bar{J}$:
          Go to step 2
     IF   $J_R^i \leq \bar{J}$:
          IF $y^i \in \mathbb{Z}^p$:
             The upper bound is improved.
             Set $\bar{J} := J^i$. Remove possibly dominated programs from $L$.
             Go to step 2.
          ELSE:
             Go to step 5.
5: Branching
     Select a variable $y_j$ among the indices $j \in \{1 \cdots p\}$ for which $y_j^i \notin \mathbb{Z}$.
     Define the two disjoint sets
        $P^{i1} := P^i \cap \{(x, y) \in \mathbb{R}_+^n \times \mathbb{R}_+^p : y_j \leq \lfloor y_j^i \rfloor\}$
        $P^{i0} := P^i \cap \{(x, y) \in \mathbb{R}_+^n \times \mathbb{R}_+^p : y_j \geq \lceil y_j^i \rceil\}$
     Set $L := L \cup \{P^{i1}, P^{i0}\}$
        Go to step 2.
```

L is a list of with the nodes

Initialize upper bound.
Assume LP is bounded

Solve and check LP relaxation
in root node

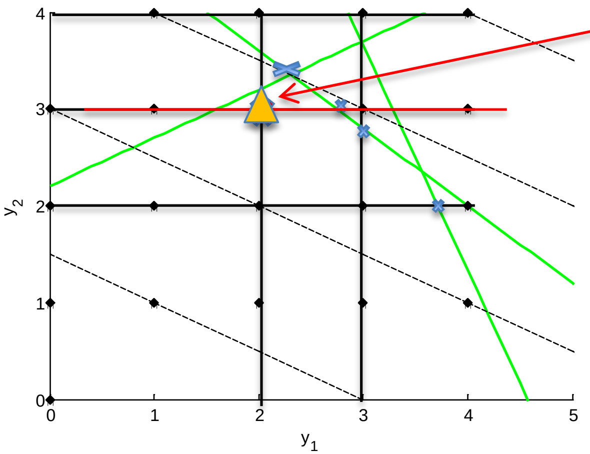
Select node a solve new LP
with added branching constraint

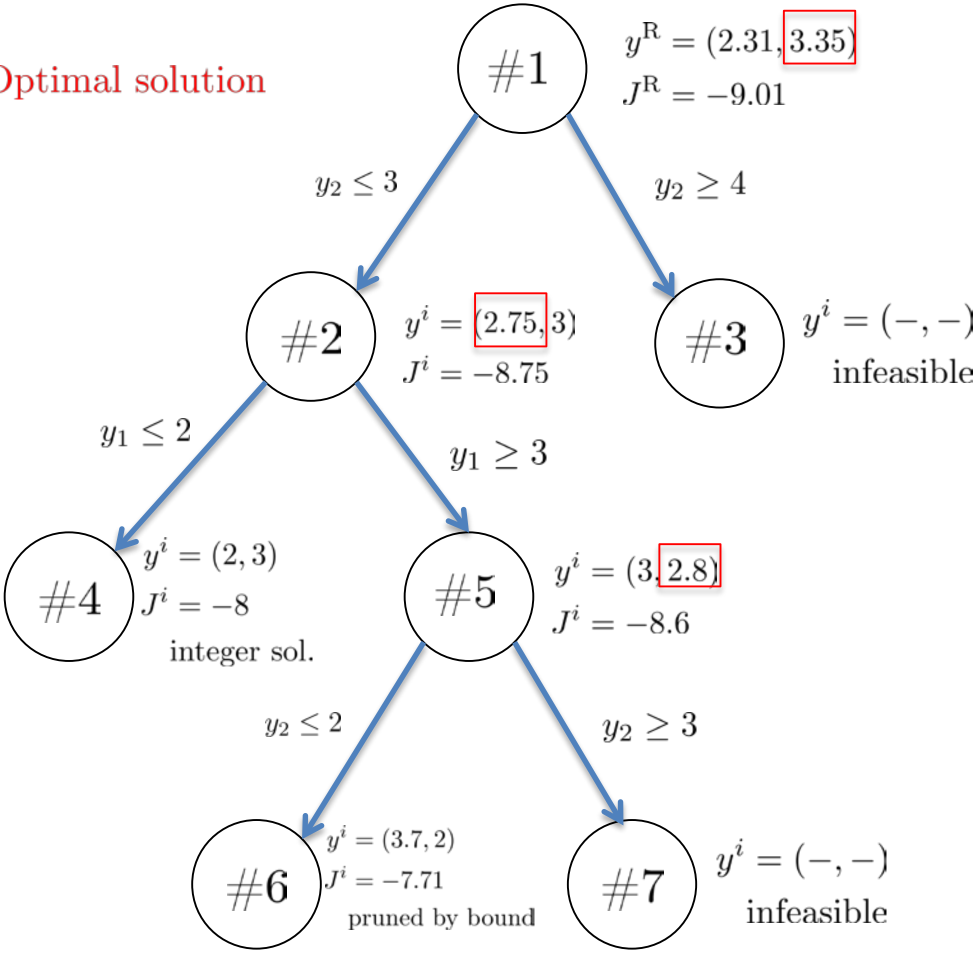
Check if solution can be pruned.
Removed nodes from L where the
solution is dominated by the best
lower bound

Choose branching variable,
add nodes to the list L of
unsolved nodes

Earlier IP example

- Branching rule: most fractional
- Node selection rule: best-bound





IP:

$$
\begin{aligned}
J^* = \min_{y} \quad & -1y_1 - 2y_2 \\
\text{s.t.} \quad & -\frac{1}{2}y_1 + y_2 \leq \frac{11}{5} \\
& 4y_1 + 5y_2 \leq 26 \\
& 7y_1 + 3y_2 \leq 32 \\
& y \in \mathbb{Z}^2_+
\end{aligned}
$$

No more nodes: search is finished

Software

Optimization modeling languages:

- Matlab through YALMIP
- GAMS: Generalized Algebraic Modeling System
- AMPL: A mathematical programming language

MILP software:

- CPLEX
- Gurobi
- Xpress-MP

BB example in GAMS:
The generalized assignment problem (GAP):

Given n assignments/tasks and m agents/servers/vehicles to carry out the tasks:

$i = 1 \dots n$: index of tasks
$j = 1 \dots m$: index of available agents
d_{ij} : cost of assigning task i to agent j
b_j : resource available from agent j
a_{ij} : resource required by agent j to do taks i
y_{ij} : a binary varible equal to 1 if agent j is assigned to do task i

$$\min_y \sum_{i=1}^{n} \sum_{j=1}^{m} d_{ij} y_{ij}$$

s.t.

$$\sum_{j=1}^{m} y_{ij} = 1, \qquad i = 1 \dots n \qquad : \text{Each task is assigned to exactly one agent}$$

$$\sum_{i=1}^{n} a_{ij} y_{ij} \le b_j, \quad j = 1 \dots m \qquad : \text{Total assignment for agent } j \text{ cannot exceed its capacity}$$

$$y_{ij} \in \{0, 1\}$$

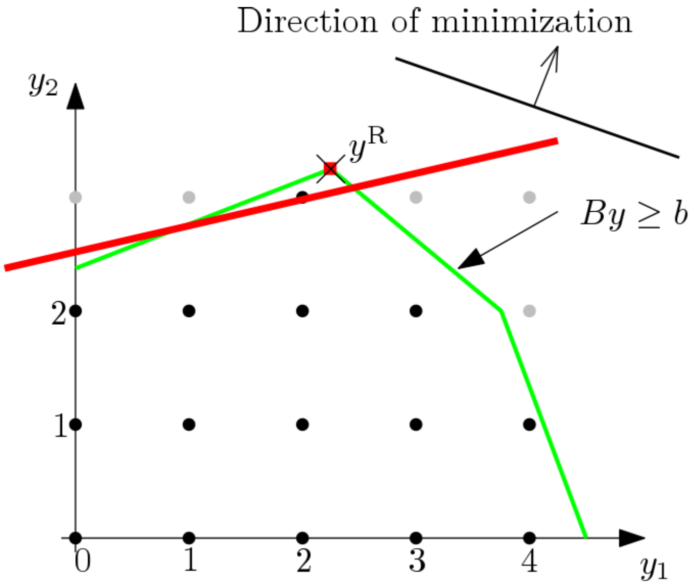
Recall the LP relaxation:

Given IP

$$\begin{aligned} \min_{y} \quad & J = d^T y \\ \text{s.t.} \quad & By \geq b \\ & y \in \mathbb{Z}_+^p \end{aligned}$$

with fractional solution y^{R} of the LP relaxation. The two basic approaches for eliminating this solution are

- Decompose the solution space (BB).
- Add valid inequalities that is valid for all integer feasible points $y \in X$, but *violated* at y^{R}. Such valid inequalities are called cuts.



Adding such valid inequalities means that we *cut* off the integer infeasible point.

The above procedure is called the separation problem:

> Given a fractional solution $\hat{y} \in P$, find a valid inequality $\pi y \leq \pi_0$, from a family of valid inequalities, or prove that no such inequality exists.

The Cutting-plane algorithm

IP:

$$\begin{aligned} \min_{y} \quad & J = d^T y \\ \text{s.t.} \quad & By \geq b \\ & y \in \mathbb{Z}_+^p \end{aligned} \qquad (1)$$

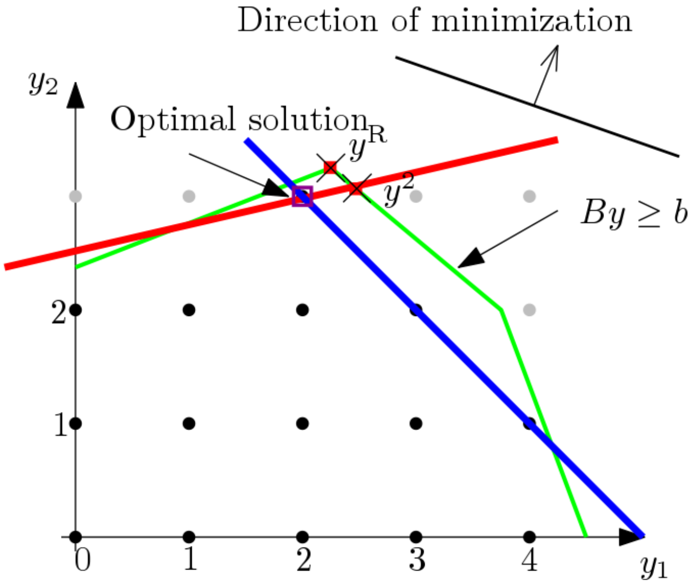
where $X = \{y \in \mathbb{Z}^p : By \geq b\}$.

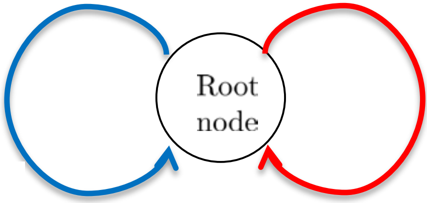
Repeat recursively:

Solve the LP relaxation over a given formulation P^i. If the relaxation is unbounded or infeasible, or the optimal solution of the LR y^i belongs to X, then STOP.

Otherwise, find a cutting plan $\pi^i y \leq \pi_0^i$ separating y^i from X.

Set $P^{i+1} = P^i \cap \{y : \pi^i y \leq \pi_0^i\}$ and repeat for formulation P^{i+1}.





$y^R = (2.31, 3.35)$

$J^R = -9.01$

Frist cut added
(weak)

$y^1 = (2.48, 3.20)$

$J^1 = -8.88$

Second cut added
(strong/facet)

$y^* = (2, 3)$

$J^* = -8$

integer sol.

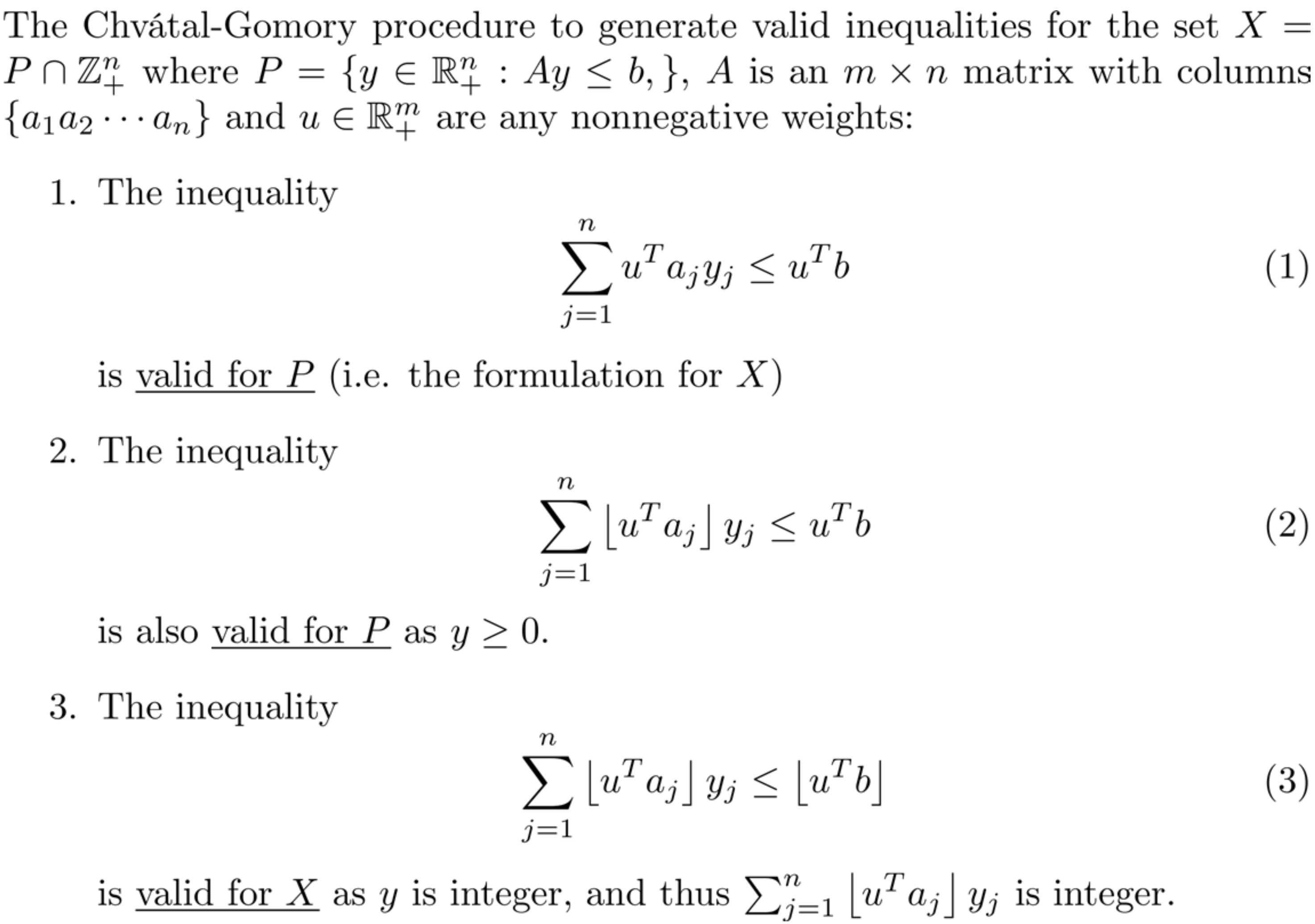
The same algorithm can be applied to MILPs, only with different cutting-planes

Generating valid inequalities

- Whenever an LP solution $(x^i, y^i) \notin X$, then there exist infinitely many cutting planes separating (x^i, y^i) from X.
- Many *families* of valid inequalities for linear integer and mixed integer sets have be developed. Some of these are
  1. Chvatal-Gomory cuts (IPs)
  2. Gomory mixed integer cuts
  3. Mixed integer rounding inequalities
  4. Lift-and-project
  5. Cover inequalities
  6. Split and intersection cuts
- The quality of the cuts generated by a separation algorithm is often closely related to the time spent on the cut generation.

Several of the above families of valid inequalities are closely related. See Cornuejols G. 2007, Valid inequalities for mixed integer linear programs, *Mathematical Programming*, 112(1), 3-44.

Example on cut generation:
Chvátal-Gomory valid inequalities



The Chvátal-Gomory procedure to generate valid inequalities for the set $X = P \cap \mathbb{Z}_+^n$ where $P = \{y \in \mathbb{R}_+^n : Ay \leq b,\}$, A is an $m \times n$ matrix with columns $\{a_1 a_2 \cdots a_n\}$ and $u \in \mathbb{R}_+^m$ are any nonnegative weights:

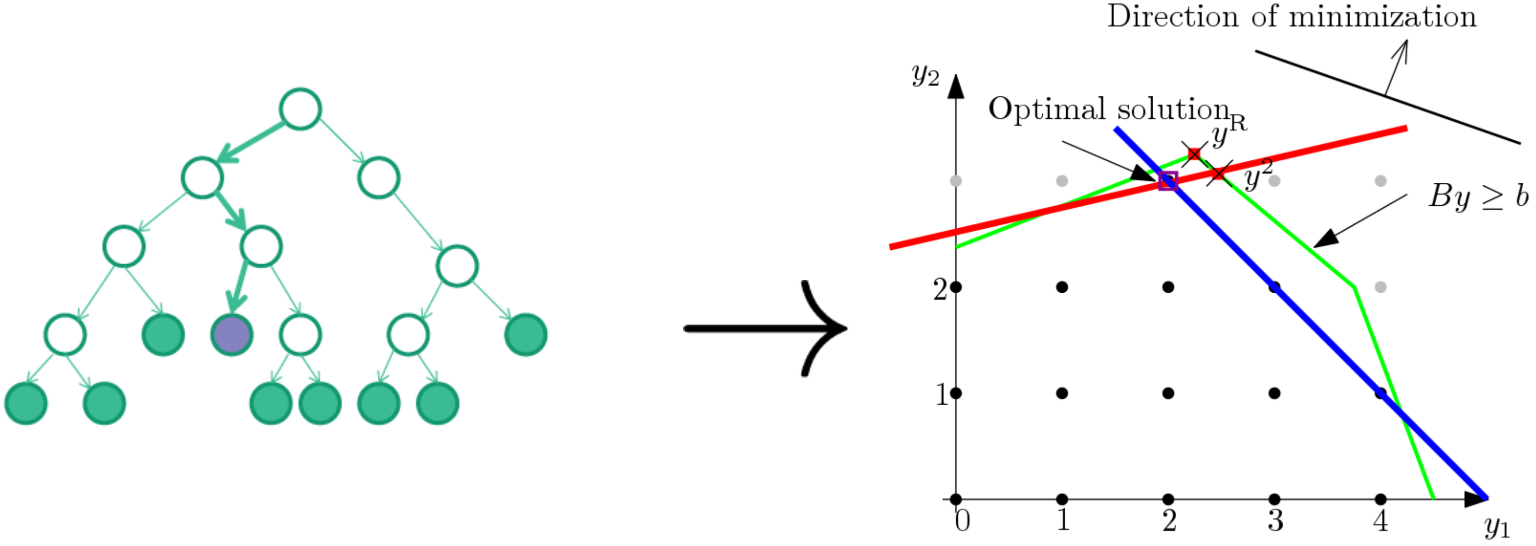
1. The inequality
$$\sum_{j=1}^{n} u^T a_j y_j \leq u^T b \qquad (1)$$
is valid for P (i.e. the formulation for X)

2. The inequality
$$\sum_{j=1}^{n} \left\lfloor u^T a_j \right\rfloor y_j \leq u^T b \qquad (2)$$
is also valid for P as $y \geq 0$.

3. The inequality
$$\sum_{j=1}^{n} \left\lfloor u^T a_j \right\rfloor y_j \leq \left\lfloor u^T b \right\rfloor \qquad (3)$$
is valid for X as y is integer, and thus $\sum_{j=1}^{n} \left\lfloor u^T a_j \right\rfloor y_j$ is integer.

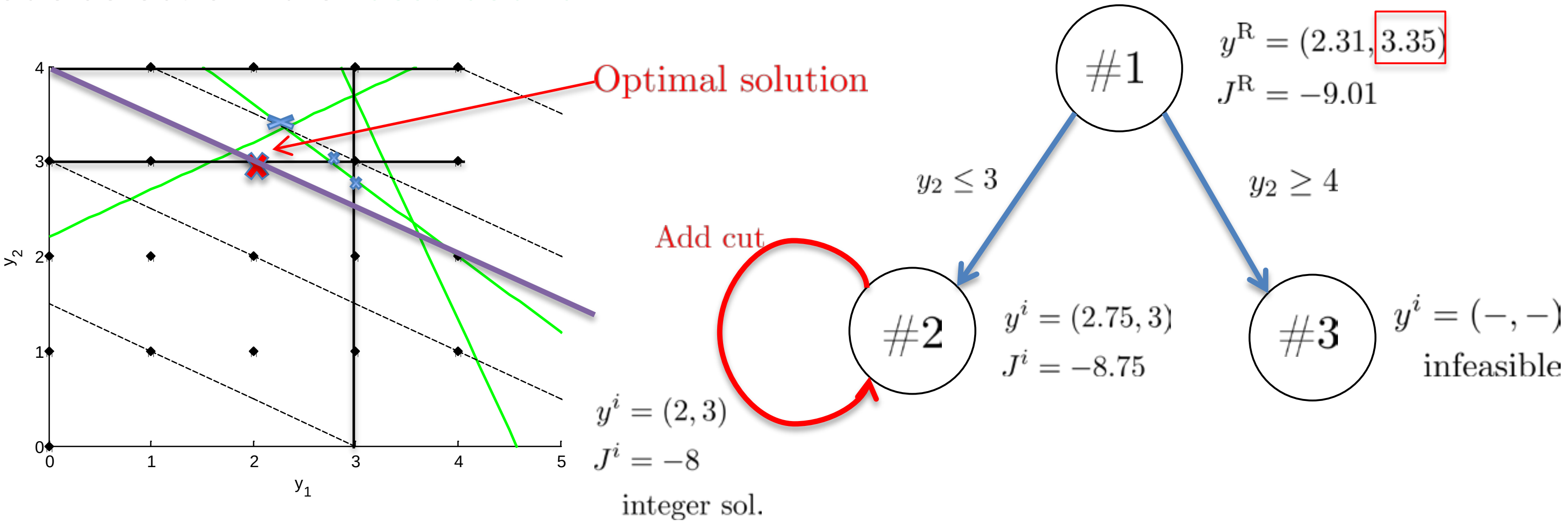
Branch-and-cut

- Pure cutting plane algorithms applied to MILPs often show slow convergence.
- By recursively adding cuts, the resulting LP may become very large, causing numerical difficulties for an LP solver.
- The cutting plane approach for solving MILPs is hence normally integrated within the branch-and-bound algorithm as variations of the branch-and-cut scheme.



Earlier IP example: Branch-and-Cut

- Branching rule: most fractional
- Node selection rule: best-bound



Optimal solution

$y^{\mathrm{R}} = (2.31, \boxed{3.35})$

$J^{\mathrm{R}} = -9.01$

$y_2 \leq 3$

$y_2 \geq 4$

Add cut

$y^i = (2.75, 3)$

$J^i = -8.75$

$y^i = (-, -)$

infeasible

$y^i = (2, 3)$

$J^i = -8$

integer sol.

Integer solution: no need to branch further.

List of remaining nodes to check is empty, $L = \{\emptyset\}$

Optimal solution

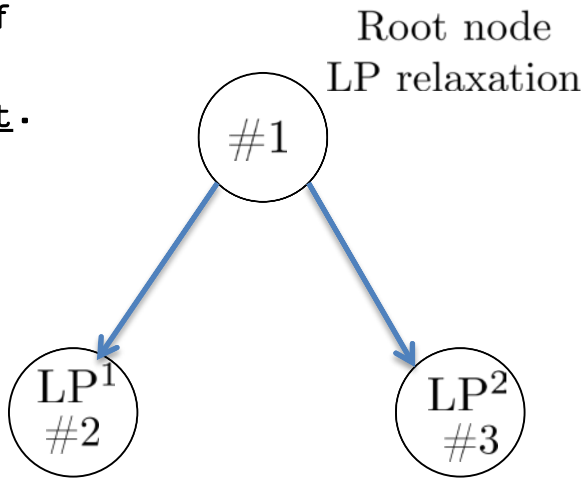
IP:

$$
\begin{aligned}
J^* = \min_{y} \quad & -1y_1 - 2y_2 \\
\text{s.t.} \quad & -\frac{1}{2}y_1 + y_2 \leq \frac{11}{5} \\
& 4y_1 + 5y_2 \leq 26 \\
& 7y_1 + 3y_2 \leq 32 \\
& y \in \mathbb{Z}^2_+
\end{aligned}
$$

The GAP problem in GAMS with Branch and Cut

Choice of LP algorithm in Branch and Bound

- Very important for the numerical efficiency of branch-and-bound methods.
- Re-use optimal basis from one node to the next.



$$\text{LP: } J_{\text{LP}} = \min_x \{c^T x : Ax = b, x \in \mathbb{R}^n_+\}$$

$$\textit{Dual}\text{ LP: } J_{\text{DLP}} = \max_\lambda \{\lambda^T b : \lambda^T A \le c, \lambda \in \mathbb{R}^m\}$$

Primal simplex requires a primal feasible starting point (phase 1 problem)
$\Rightarrow$ Adding a bound $y \geq \lfloor y_j^i \rfloor$ to eliminate fractional LP solution makes solution from parent node primal infeasible as starting point

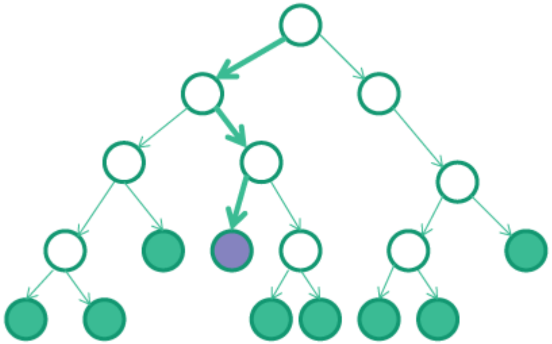
Solution from parent node is still dual feasible

Use dual simplex $\Rightarrow$ requiring few iterations to regain primal feasibility

Solution of large-scale MILPs

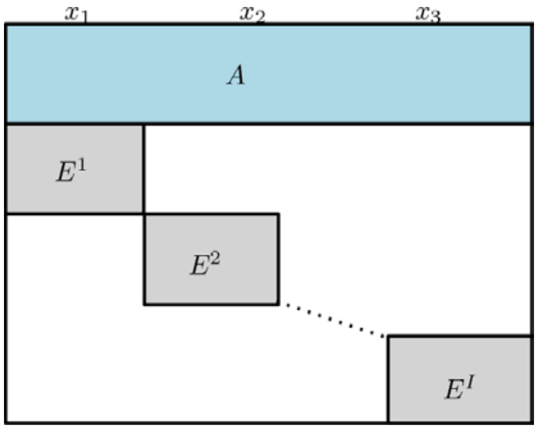
Important aspects of the branch-and-cut algorithm:

- Presolve routines
- Parallelization of branch-and-bound
- Efficiency of LP algorithm



Utilize structures in problem:

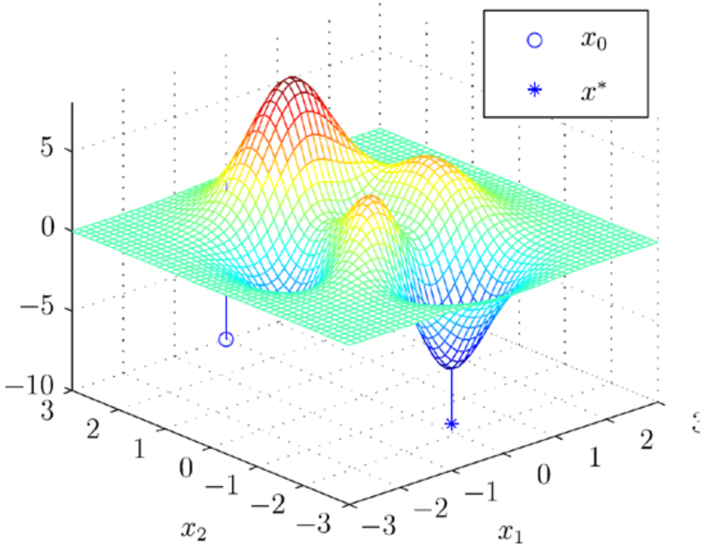
- Decomposition algorithms



- Apply heuristics to generate a feasible solution

MINLP: challenges

$$\begin{aligned} J^* = \min_{x,y} \quad & f(x,y) \\ \text{s.t.} \quad & g(x,y) \leq 0, \\ & x \in \mathbb{R}^n_+, \\ & y \in \{0,1\}^q \, , \end{aligned}$$



- Major difference in approach if $g(x,y) \leq 0$ is convex.
- Nonlinear model, e.g. $x_{k+1} = f(x_k, y_k)$: automatically nonconvex MINLP

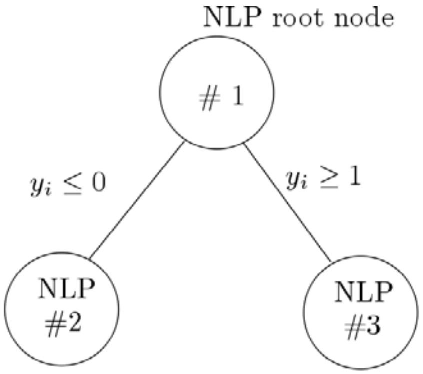
Nonconvex ⇒ Lower bound obatined by NLP relaxation is no longer valid lower bound on J^*

$$\Downarrow$$

No duality gap

MINLP: solution approaches

$$
\begin{aligned}
J^* = \min_{x,y} \quad & f(x,y) \\
\text{s.t.} \quad & g(x,y) \leq 0, \\
& x \in \mathbb{R}^n_+, \\
& y \in \{0,1\}^q \, ,
\end{aligned}
$$

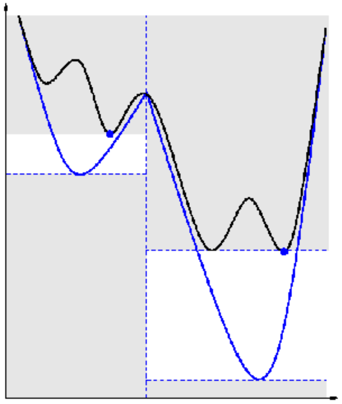


If the NLP relaxation, $y \in [0,1]^p$, renders a convex NLP:

- Nonlinear branch-and-bound: Solve NLP in each node, solution returned is globally optimal.
- Several other approaches, e.g. Outer approximation, Extended cutting-plane.
- Software: Bonmin, SBB, DICOPT, Knitro, etc.

If NLP relaxation is a nonconvex NLP:

- Piecewise linearization of nonlinearities: MINLP ⇒ approximated MILP.
- Ignore the fact that the NLP solution is no valid lower bound.
- Apply rigorous global optimization algorithms, e.g. spatial branch-and-bound (BARON)



Source: ZIB Berlin

Conclusions

- Branch-and-bound defines the basis for all modern MILP codes.
- Pure cutting-plane approaches are ineffective for large MILPs.
- BB is very efficient when integrated with advanced cut-generation, leading to branch-and-cut methods.
- Solving large-scale MINLPs are significantly more difficult than MILPs.