LP/MIP based techniques for solving MINLPs

Stefan Vigerske
Humboldt-Universität zu Berlin

DFG Research Center MATHEON
Mathematics for key technologies

17th December 2010, Corporate Technology, Siemens AG
Overview

1. Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2. Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques

LP/MIP based techniques for solving MINLPs
Mixed Integer Nonlinear Program (MINLP)

minimize $f(x)$

such that $g_j(x) \leq 0$, $j \in J$,

$x \in X = \{x \in \mathbb{R}^n : Dx \leq d, x_l \leq x \leq x_u\}$,

$x_I \in \mathbb{Z}^{|I|}$,

where

- $f : X \rightarrow \mathbb{R}$ and $g_j : X \rightarrow \mathbb{R}$, $j \in J$, are differentiable functions
- $I \subseteq \{1, \ldots, n\}$ the set of integer variables
Mixed Integer Nonlinear Program (MINLP)

minimize \ f(x) \\
such that \ g_j(x) \leq 0, \quad j \in J, \\
x \in X = \{x \in \mathbb{R}^n : Dx \leq d, \underline{x} \leq x \leq \bar{x}\}, \\
x_I \in \mathbb{Z}^{|I|},

where

- \( f : X \to \mathbb{R} \) and \( g_j : X \to \mathbb{R}, \ j \in J, \) are differentiable functions
- \( I \subseteq \{1, \ldots, n\} \) the set of integer variables

Important Special Cases:

- Convex MINLP: \( f \) and \( g_j \) are convex functions on \( X \)
- MIQQP (Mixed Integer Quadratically Constraint Quadratic Program): \( f \) and \( g_j \) are quadratic functions
Overview: Convex MINLP

1. Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2. Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques

Survey papers:


▷ P. Bonami, M. Kilinç, J. Linderoth, Algorithms and Software for Convex Mixed Integer Nonlinear Programs, 2010
Outline

1 Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2 Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques
NLP subproblems

Some NLP subproblems that will be useful:

**NLP relaxation**

\[
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g_j(x) \leq 0, \ j \in J, \\
& \quad x \in X, \\
& \quad x_I \in [\underline{x}_I, \overline{x}_I]
\end{align*}
\]

- \(x_I\) and \(\overline{x}_I\) possible restrictions on integer variables (as in B&B)
- solution yields lower bound for MINLP restricted to \([\underline{x}_I^k, \overline{x}_I^k]\)
Some NLP subproblems that will be useful:

### NLP relaxation

\[
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g_j(x) \leq 0, \ j \in J, \\
& \quad x \in X, \\
& \quad x_I \in [\underline{x}_I, \overline{x}_I]
\end{align*}
\]

- \(x_I\) and \(\overline{x}_I\) possible restrictions on integer variables (as in B&B)
- solution yields lower bound for MINLP restricted to \([\underline{x}_I^k, \overline{x}_I^k]\)

### NLP subproblem

\[
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g_j(x) \leq 0, \ j \in J, \\
& \quad x \in X, \\
& \quad x_I = \hat{x}_I
\end{align*}
\]

- \(\hat{x}_I \in X_I \cap \mathbb{Z}^{\|I\|}\) a candidate for integer part of a solution
- feasible solution yields upper bound for MINLP

Feasibility subproblem

\[
\begin{align*}
\min & \quad u \\
\text{s.t.} & \quad g_j(x) \leq u, \ j \in J, \\
& \quad x \in X, \\
& \quad u \in \mathbb{R}, \\
& \quad x_I = \hat{x}_I
\end{align*}
\]
Some NLP subproblems that will be useful:

<table>
<thead>
<tr>
<th>NLP relaxation</th>
<th>NLP subproblem</th>
<th>Feasibility subproblem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\min f(x)$</td>
<td>$\min f(x)$</td>
<td>$\min u$</td>
</tr>
<tr>
<td>s.t. $g_j(x) \leq 0, \ j \in J$, $x \in X$, $x_I \in [\bar{x}_I, \bar{x}_I]$</td>
<td>s.t. $g_j(x) \leq 0, \ j \in J$, $x \in X$, $x_I = \hat{x}_I$</td>
<td>s.t. $g_j(x) \leq u, \ j \in J$, $x \in X$, $u \in \mathbb{R}$, $x_I = \hat{x}_I$</td>
</tr>
<tr>
<td>$\triangleright$ $x_I$ and $\bar{x}_I$ possible restrictions on integer variables (as in B&amp;B)</td>
<td>$\triangleright$ $\hat{x}_I \in X_I \cap \mathbb{Z}^{</td>
<td>I</td>
</tr>
<tr>
<td>$\triangleright$ solution yields lower bound for MINLP restricted to $[\underline{x}_I^k, \bar{x}_I^k]$</td>
<td>$\triangleright$ feasible solution yields upper bound for MINLP</td>
<td>$\triangleright$ solution yields minimal infeasible solution w.r.t. $\hat{x}_I$</td>
</tr>
</tbody>
</table>

LP/MIP based techniques for solving MINLPs
replace nonlinearities in MINLP by supporting hyperplanes (use convexity)

given $K$ points $x^k$, $k = 1, \ldots, K$

MIP outer-approximation of MINLP

$$
\begin{align*}
\text{min } & \alpha \\
\text{s.t. } & \alpha \geq f(x^k) + \nabla f(x^k)^T (x - x^k), \\
& g_j(x^k) + \nabla g_j(x^k)^T (x - x^k) \leq 0, \\
& x \in \mathcal{X}, \quad x_I \in \mathbb{Z}^{\|I\|}
\end{align*}
$$

with $J^k \subseteq J$ set of constraints for which to include linearization in $x^k$
When to solve which NLP subproblem and MIP relaxation?

- **NLP-based Branch-and-Bound**
  - Formulation: \[ \text{solve NLP relaxation to compute lower bound in each node} \]
  - Branch on fractional integer variables
  - Performs well if NLP relaxation is tight
  - Not in the scope of this talk

- **Outer-Approximation**
  - Formulation: \[ \text{similar to Outer-Approximation} \]
  - Good performance only in some special cases (e.g., many discrete variables)
  - Not discussed in this talk

- **Generalized Benders Decomposition**
  - Formulation: \[ \text{Generalized Benders Decomposition} \]
  - Formulation: \[ \text{similar to Outer-Approximation} \]
  - Good performance only in some special cases (e.g., many discrete variables)
  - Not discussed in this talk

- **Extended cutting plane**
  - Formulation: \[ \text{Extended cutting plane} \]

- **LP/NLP-based Branch-and-Bound**
  - Formulation: \[ \text{LP/NLP-based Branch-and-Bound} \]
  - Formulation: \[ \text{LP/MIP based techniques for solving MINLPs} \]
When to solve which NLP subproblem and MIP relaxation?

Strategies:

- **NLP-based Branch-and-Bound** [Leyffer, 1993, 2001]
  - solve NLP relaxation to compute lower bound in each node
  - branch on fractional integer variables
  - performs well if NLP relaxation is tight
  - not in the scope of this talk
When to solve which NLP subproblem and MIP relaxation?

Strategies:

- **NLP-based Branch-and-Bound** [Leyffer, 1993, 2001]
  - solve NLP relaxation to compute lower bound in each node
  - branch on fractional integer variables
  - performs well if NLP relaxation is tight
  - not in the scope of this talk

- **Outer-Approximation** [Duran and Grossmann, 1986, Yuan et al., 1988, Fletcher and Leyffer, 1994]

- **Generalized Benders Decomposition** [Geoffrion, 1972]
  - similar to Outer-Approximation
  - good performance only in some special cases (e.g., many discrete variables)
  - not discussed in this talk
When to solve which NLP subproblem and MIP relaxation?

Strategies:

- **NLP-based Branch-and-Bound** [Leyffer, 1993, 2001]
  - solve NLP relaxation to compute lower bound in each node
  - branch on fractional integer variables
  - performs well if NLP relaxation is tight
  - not in the scope of this talk

- **Outer-Approximation** [Duran and Grossmann, 1986, Yuan et al., 1988, Fletcher and Leyffer, 1994]

- **Generalized Benders Decomposition** [Geoffrion, 1972]
  - similar to Outer-Approximation
  - good performance only in some special cases (e.g., many discrete variables)
  - not discussed in this talk

- **Extended cutting plane** [Westerlund and Petterson, 1995]
When to solve which NLP subproblem and MIP relaxation?

Strategies:

- **NLP-based Branch-and-Bound** [Leyffer, 1993, 2001]
  - solve NLP relaxation to compute lower bound in each node
  - branch on fractional integer variables
  - performs well if NLP relaxation is tight
  - not in the scope of this talk

- **Outer-Approximation** [Duran and Grossmann, 1986, Yuan et al., 1988, Fletcher and Leyffer, 1994]

- **Generalized Benders Decomposition** [Geoffrion, 1972]
  - similar to Outer-Approximation
  - good performance only in some special cases (e.g., many discrete variables)
  - not discussed in this talk

- **Extended cutting plane** [Westerlund and Pettersson, 1995]

- **LP/NLP-based Branch-and-Bound** [Quesada and Grossmann, 1992, Bonami et al., 2008, Abhishek et al., 2010]
Outer-Approximation for Convex MINLP

[Duran and Grossmann, 1986, Yuan et al., 1988, Fletcher and Leyffer, 1994]

Basic Theorem [Duran and Grossmann, 1986]

MINLP and the following problem have the same optimal solution

\[
\begin{align*}
\min & \quad \alpha \\
\text{s.t.} & \quad \alpha \geq f(x^k) + \nabla f(x^k)^T(x - x^k), \\
& \quad g_j(x^k) + \nabla g_j(x^k)^T(x - x^k) \leq 0, \quad j \in J, \\
& \quad x \in \mathbb{X}, \quad x_I \in \mathbb{Z}^{|I|}, \\
\end{align*}
\]

where \(x^k\) is the optimal solution for the NLP subproblem in \(x_I^k\) or the NLP feasibility problem in \(x_I^k\).
### Basic Theorem [Duran and Grossmann, 1986]

MINLP and the following problem have the **same optimal solution**

\[
\begin{align*}
\text{min } \alpha \\
\text{s.t. } \alpha & \geq f(x^k) + \nabla f(x^k)^T (x - x^k), \\
g_j(x^k) + \nabla g_j(x^k)^T (x - x^k) & \leq 0, \\
x & \in X, \\
x^k & \in \mathbb{Z}^{|I|}, \\
\end{align*}
\]

where \(x^k\) is the optimal solution for the NLP subproblem in \(x^k_i\) or the NLP feasibility problem in \(x^k_i\).

**Idea:** relax equivalent MIP to MIP approximation by solving a sequence of NLP sub- and feasibility problems
Outer-Approximation Algorithm

1. solve NLP relaxation
   \[ x^1 := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X \} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)
Outer-Approximation Algorithm [Duran and Grossmann, 1986]

1. solve NLP relaxation
   \[ x^1 := \text{argmin}\{f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X \} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)

3. iterate until lower bound = upper bound or MIP is infeasible:
   \[ x^k \in X \cap \mathbb{Z}^{||I||} \] by solving MIP relaxation

3.1 compute lower bound and \( x^k \in X \cap \mathbb{Z}^{||I||} \) by solving MIP relaxation

LP/MIP based techniques for solving MINLPs
Outer-Approximation for Convex MINLP

Outer-Approximation Algorithm

[Duran and Grossmann, 1986]

1. solve NLP relaxation
   \[ x^1 := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X \} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)

3. iterate until lower bound = upper bound or MIP is infeasible:
   3.1 compute lower bound and \( x^k_I \in X \cap \mathbb{Z}^{|I|} \) by solving MIP relaxation
   3.2 solve NLP subproblem
      \[ x^k := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X, \ x_I = x^k_I \} \]
   3.3 if feasible, update upper bound
   3.4 if infeasible, solve NLP feasibility problem
      \[ x^k := \arg\min \{ u : g_j(x) \leq u, \ j \in J, \ x \in X, \ u \in \mathbb{R}, \ x_I = x^k_I \} \]
   3.5 add linearizations of \( g_j(\cdot) \) in \( x^k \) to MIP relaxation
Outer-Approximation Algorithm

1. solve NLP relaxation
   \[ x^1 := \text{argmin}\{f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X\} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)

3. iterate until lower bound = upper bound or MIP is infeasible:
   3.1 compute lower bound and \( x^k_I \in X \cap \mathbb{Z}^{|I|} \) by solving MIP relaxation (not to optimality in early iterations)
   3.2 solve NLP subproblem
      \[ x^k := \text{argmin}\{f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X, \ x_I = x^k_I\} \]
   3.3 if feasible, update upper bound
   3.4 if infeasible, solve NLP feasibility problem
      \[ x^k := \text{argmin}\{u : g_j(x) \leq u, \ j \in J, \ x \in X, \ u \in \mathbb{R}, \ x_I = x^k_I\} \]
   3.5 add linearizations of \( g_j(\cdot) \) in \( x^k \) to MIP relaxation
Outer-Approximation Algorithm [Duran and Grossmann, 1986]

1. solve NLP relaxation
   \[ x^1 := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X \} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)

3. iterate until lower bound \( \geq \) upper bound or MIP is infeasible:
   3.1 compute lower bound and \( x^k_i \in X \cap \mathbb{Z}^{\|I\|} \) by solving MIP relaxation (not to optimality in early iterations)
   3.2 solve NLP subproblem
      \[ x^k := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X, \ x_I = x^k_I \} \]
   3.3 if feasible, update upper bound
   3.4 if infeasible, solve NLP feasibility problem
      \[ x^k := \arg\min \{ u : g_j(x) \leq u, \ j \in J, \ x \in X, \ u \in \mathbb{R}, \ x_I = x^k_I \} \]
   3.5 add linearizations of \( g_j(\cdot) \) in \( x^k \) to MIP relaxation
   3.6 optional: if \( X_I \subseteq [0, 1]^{\|I\|} \), add cut to forbid \( x^k_I \) in MIP

LP/MIP based techniques for solving MINLPs
Extended Cutting Plane Algorithm

- extension of Kelley’s cutting plane method [Kelley, 1961]
- does not rely on solving NLP subproblems
Extended Cutting Plane Algorithm

- extension of Kelley’s cutting plane method [Kelley, 1961]
- does not rely on solving NLP subproblems

Extended Cutting Plane Algorithm
[Westerlund and Petterson, 1995]

1. initialize MIP outer-approximation by omitting all nonlinear constraints from MINLP
Extended Cutting Plane Algorithm

- extension of Kelley’s cutting plane method [Kelley, 1961]
- does not rely on solving NLP subproblems

Extended Cutting Plane Algorithm

[Westerlund and Petterson, 1995]

1. initialize MIP outer-approximation by omitting all nonlinear constraints from MINLP
2. iterate:
   2.1 solve MIP relaxation to obtain $x^k \in X$ with $x^k_I \in \mathbb{Z}^{|I|}$
   2.2 if solution is feasible for MINLP, finish
   2.3 add linearization of $g_j(\cdot)$ in $x^k$ to MIP relaxation for most violated nonlinear constraint
      $(j = \arg\max_{j \in J} \{g_j(x^k)\})$
Extended Cutting Plane Algorithm

- extension of Kelley’s cutting plane method [Kelley, 1961]
- does not rely on solving NLP subproblems

Extended Cutting Plane Algorithm  
[Westerlund and Petterson, 1995]

1. initialize MIP outer-approximation by omitting all nonlinear constraints from MINLP
2. iterate:
   2.1 solve MIP relaxation to obtain $x^k \in X$ with $x^k_I \in \mathbb{Z}^{|I|}$
   2.2 if solution is feasible for MINLP, finish
   2.3 add linearization of $g_j(\cdot)$ in $x^k$ to MIP relaxation for most violated nonlinear constraint  
      $(j = \arg\max_{j \in J}\{g_j(x^k)\})$
      alternatively: add linearization for all or all violated constraints

LP/MIP based techniques for solving MINLPs
Extended Cutting Plane Algorithm

extension of Kelley’s cutting plane method [Kelley, 1961]
does not rely on solving NLP subproblems

Extended Cutting Plane Algorithm
[Westerlund and Petterson, 1995]

1. initialize MIP outer-approximation by omitting all nonlinear constraints from MINLP
2. iterate:
   2.1 solve MIP relaxation to obtain $x^k \in X$ with $x^k_I \in \mathbb{Z}^{|I|}$
   2.2 if solution is feasible for MINLP, finish
   2.3 add linearization of $g_j(\cdot)$ in $x^k$ to MIP relaxation for most violated nonlinear constraint
      \( j = \arg\max_{j \in J} \{ g_j(x^k) \} \)
      alternatively: add linearization for all or all violated constraints

works well on almost linear problems
LP/NLP based Branch-and-Bound

integrate Outer-Approximation into MIP alike branch-and-bound

LP/NLP based Branch-and-Bound Alg.  [Quesada and Grossmann, 1992]

1. solve NLP relaxation

\[ x^1 := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X \} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)
integrate Outer-Approximation into MIP alike branch-and-bound

LP/NLP based Branch-and-Bound Alg. [Quesada and Grossmann, 1992]

1. solve NLP relaxation

\[ x^1 := \text{argmin}\{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X \} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)

3. perform branch-and-bound:
   3.1 compute lower bound and \( \tilde{x}^k \) by solving LP relaxation of current node
   3.2 if \( \tilde{x}^k \) is feasible for MINLP, update upper bound, go to 3.1
   3.3 if \( \tilde{x}^k_i \) is fractional, branch on some variable \( x_i, \ i \in I \), go to 3.1
integrate Outer-Approximation into MIP alike branch-and-bound

**LP/NLP based Branch-and-Bound Alg.** [Quesada and Grossmann, 1992]

1. solve NLP relaxation

   \[ x^1 := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X \} \]

2. initialize MIP outer-approximation with linearizations of \( g_j(\cdot) \) in \( x^1 \)

3. perform branch-and-bound:
   3.1 compute lower bound and \( \tilde{x}^k \) by solving LP relaxation of current node
   3.2 if \( \tilde{x}^k \) is feasible for MINLP, update upper bound, go to 3.1
   3.3 if \( \tilde{x}_i^k \) is fractional, branch on some variable \( x_i, \ i \in I \), go to 3.1
   3.4 solve NLP subproblem or NLP feasibility problem

   \[ x^k := \arg\min \{ f(x) : g_j(x) \leq 0, \ j \in J, \ x \in X, \ x_I = \tilde{x}_I^k \} \]
   
   or

   \[ x^k := \arg\min \{ u : g_j(x) \leq u, \ j \in J, \ x \in X, \ u \in \mathbb{R}, \ x_I = \tilde{x}_I^k \} \]

3.5 add linearizations of \( g_j(\cdot) \) in \( x^k \) to LP relaxation
1 Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2 Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques

LP/MIP based techniques for solving MINLPs
handle equalities

- assume presence of nonlinear equalities

\[ h_j(x) = 0, \quad j \in J_- \]

- generally introduce nonconvexities

&Socceki and Grossmann, 1987:

\[ \nabla h_j(x) \nabla (x - x_k)^T \leq 0, \quad \text{if } \lambda_{kj} > 0 \]

\[ \nabla h_j(x) \nabla (x - x_k)^T \geq 0, \quad \text{if } \lambda_{kj} < 0 \]

where \( \lambda_{kj} \) is the dual multiplier associated with \( h_j(x, y) = 0 \) in the NLP, \( j \in J_- \) remains a rigorous method only in special cases

LP/MIP based techniques for solving MINLPs
assume presence of nonlinear equalities

\[ h_j(x) = 0, \quad j \in J_- \]

generally introduce nonconvexities

proposed relaxation [Kocis and Grossmann, 1987]:

\[ \nabla h_j(x^k)^T (x - x^k) \leq 0, \quad \text{if } \lambda_j^k > 0, \]
\[ \nabla h_j(x^k)^T (x - x^k) \geq 0, \quad \text{if } \lambda_j^k < 0, \]

where \( \lambda_j^k \) is the dual multiplier associated with \( h_j(x, y) = 0 \) in the NLP, \( j \in J_- \)

remains a rigorous method only in special cases
Assume $f(\cdot)$ or $g_j(\cdot)$ nonconvex or nonlinear equalities $h(x, y) = 0$ present.

- Outer-Approximation runs in two difficulties:
  1. NLP subproblems may not be solved to global optimality
  2. gradient-based linearizations may cut into feasible region
- apply heuristic strategies to reduce effect of nonconvexity
Assume $f(\cdot)$ or $g_j(\cdot)$ nonconvex or nonlinear equalities $h(x, y) = 0$ present.

⇒ Outer-Approximation runs in two difficulties:

1. NLP subproblems may not be solved to global optimality
2. gradient-based linearizations may cut into feasible region

⇒ apply heuristic strategies to reduce effect of nonconvexity

Augmented Penalty / Equality Relaxation MIP master problem

$$\begin{align*}
\min_{\alpha} & \quad \alpha + \sum_{k=1}^{K} w_p^k p^k + w_q^k q^k \\
\text{s.t.} & \quad \alpha \geq f(x^k) + \nabla f(x^k)^T (x - x^k), \\
& \quad g_j(x^k) + \nabla g_j(x^k)^T (x - x^k) \leq p^k, \\
& \quad \text{sign}(\lambda_k) \nabla h_j(x^k)^T (x - x^k) \leq q^k, \\
& \quad x \in X, \quad x_I \in \mathbb{Z}^{|I|}, \quad \alpha \in \mathbb{R}, \quad p^k, q^k \geq 0
\end{align*}$$

[Viswanathan and Grossmann, 1990]
Consider LP/NLP based Branch-and-Bound.

Similar Strategies as in MIP:
- pseudo-cost branching
- strong branching
- reliability branching
- depth first search
- best first search
- diving methods
- two-phase methods
Consider LP/NLP based Branch-and-Bound.

Cutting Planes for convex MINLP:

- **MIP cutting planes** (Gomory, MIR, flowcover, ...)

\[
\alpha \geq \hat{\alpha} - i + (\hat{\alpha} + i - \hat{\alpha} - i) x_i,
\]

where \(\hat{\alpha} - i\) is the lower bound for \(x_i = 0\); \(\hat{\alpha} + i\) is the lower bound for \(x_i = 1\).

20% improvement in mean solve time for FilMINT [Bonami et al., 2010]
Consider LP/NLP based Branch-and-Bound.

Cutting Planes for convex MINLP:
- MIP cutting planes (Gomory, MIR, flowcover, ...)
- Gomory Cuts for conic constraints [Cezik and Iyengar, 2005]
- Mixed Integer Rounding for Second-Order-Cone Constraints: [Atamtürk and Narayanan, 2010]

\[
\alpha \geq \hat{\alpha} - i + (\hat{\alpha} + i - \hat{\alpha} - i) x_i,
\]
where \( \hat{\alpha} - i = \) lower bound for \( x_i = 0 \);
\( \hat{\alpha} + i = \) lower bound for \( x_i = 1 \)

20% improvement in mean solve time for FilMINT [Bonami et al., 2010]
Consider LP/NLP based Branch-and-Bound.

Cutting Planes for convex MINLP:

- **MIP cutting planes** (Gomory, MIR, flowcover, ...)
- **Gomory Cuts for conic constraints** [Cezik and Iyengar, 2005]
- **Mixed Integer Rounding for Second-Order-Cone Constraints:** [Atamtürk and Narayanan, 2010]
- **disjunctive inequalities / lift-and-project:** [Stubbs and Mehrotra, 1999, Drewes, 2009, Kilinç et al., 2010a]
- **simple lift-and-project inequality from (strong) branching:** [Kilinç et al., 2010b]

\[
\alpha \geq \hat{\alpha}_i^- + (\hat{\alpha}_i^+ - \hat{\alpha}_i^-) x_i,
\]

where \( \hat{\alpha}_i^- \) = lower bound for \( x_i = 0 \); \( \hat{\alpha}_i^+ \) = lower bound for \( x_i = 1 \)

20% improvement in mean solve time for FilMINT [Bonami et al., 2010]
Heuristics for MINLP:

- **diving** heuristics for NLP-based branch-and-bound [Bonami and Gonçalves, 2008]
  - as in MIP, but try to fix several variables at each iteration
  - dive such that all nonlinear integer variables get integral values, then fix all nonlinear variables (to current value in NLP relaxation), solve remaining MIP
  - 21% improvement in mean B&B time
Heuristics for MINLP:

» **dive**
  - heuristics for NLP-based branch-and-bound [Bonami and Gonçalves, 2008]
  - as in MIP, but try to fix several variables at each iteration
  - dive such that all nonlinear integer variables get integral values, then fix all nonlinear variables (to current value in NLP relaxation), solve remaining MIP
  - 21% improvement in mean B&B time

» **large neighborhood search** (RENS, RINS, DINS, local branching, . . .) [Berthold et al., 2009a, Nannicini et al., 2009, Liberti et al., 2009]
Heuristics for MINLP:

- **diving** heuristics for NLP-based branch-and-bound [Bonami and Gonçalves, 2008]
  - as in MIP, but try to fix several variables at each iteration
  - dive such that all nonlinear integer variables get integral values, then fix all nonlinear variables (to current value in NLP relaxation), solve remaining MIP
  - 21% improvement in mean B&B time

- **large neighborhood search** (RENS, RINS, DINS, local branching, ...) [Berthold et al., 2009a, Nannicini et al., 2009, Liberti et al., 2009]

- sub-MIP heuristic **“Undercover”** (later...) [Berthold and Gleixner, 2009]
Heuristics for MINLP:

- **diving** heuristics for NLP-based branch-and-bound [Bonami and Gonçalves, 2008]
  - as in MIP, but try to fix several variables at each iteration
  - dive such that all nonlinear integer variables get integral values, then fix all nonlinear variables (to current value in NLP relaxation), solve remaining MIP
  - 21% improvement in mean B&B time

- large neighborhood search (RENS, RINS, DINS, local branching, ...) [Berthold et al., 2009a, Nannicini et al., 2009, Liberti et al., 2009]

- sub-MIP heuristic “Undercover” (later...) [Berthold and Gleixner, 2009]

- Feasibility Pump [Bonami et al., 2009a, Bonami and Gonçalves, 2008]
Basic Idea: Create a sequence of integer feasible points $\hat{x}^k$ and nonlinear feasible points $x^k$ by alternating solution of

$$\hat{x}^{K+1} := \text{argmin} \|x - x^K\|_1$$

s.t. $g_j(x^k) + \nabla g_j(x^k)^T(x - \hat{x}^k) \leq 0,$

$$j \in J, \quad k \leq K$$

$x \in X, \quad x_I \in \mathbb{Z}^{|I|}$

$$x^{K+1} := \text{argmin} \|x - \hat{x}^K\|_2$$

s.t. $g_j(x) \leq 0, \quad j \in J$

$x \in X$
**Basic Idea:** Create a sequence of integer feasible points $\hat{x}^k$ and nonlinear feasible points $x^k$ by alternating solution of

<table>
<thead>
<tr>
<th>MIP relaxation</th>
<th>NLP relaxation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{x}^{k+1} := \text{argmin }</td>
<td></td>
</tr>
<tr>
<td>s.t. $g_j(x^k) + \nabla g_j(x^k)^T(x - \hat{x}^k) \leq 0,$</td>
<td>s.t. $g_j(x) \leq 0, \quad j \in J$</td>
</tr>
<tr>
<td>$\quad j \in J, \quad k \leq K$</td>
<td>$\quad x \in X$</td>
</tr>
<tr>
<td>$x \in X, \quad x_I \in \mathbb{Z}^{</td>
<td>I</td>
</tr>
</tbody>
</table>

- **Convergence for convex problems:** finds feasible point $x^K$ or proofs infeasibility
- **Heuristic** for nonconvex problems
The Feasibility Pump for MINLP

[Bonami et al., 2009a, Bonami and Gonçalves, 2008]

**Basic Idea:** Create a sequence of integer feasible points $\hat{x}^k$ and nonlinear feasible points $x^k$ by alternating solution of

**Alternative: Rounding**

$$
\hat{x}^{K+1} := \text{argmin } ||x - x^K||_1 \\
\text{s.t. } \\
\quad x_I \in \mathbb{Z}^{|I|}
$$

**NLP relaxation**

$$
x^{K+1} := \text{argmin } ||x - \hat{x}^K||_2 \\
\text{s.t. } g_j(x) \leq 0, \quad j \in J \\
\quad x \in X
$$

- **Convergence for convex problems:** finds feasible point $x^K$ or proofs infeasibility
- **Heuristic** for nonconvex problems
- **Rounding variant** finds solution for 93% of the instances; 2.7% improvement in mean branch-and-bound time

[Bonami and Gonçalves, 2008]
Perspective Relaxation: [Günlük et al., 2007, Hijazi et al., 2009, Günlük and Linderoth, 2010]
1 Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2 Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques
Solvers that implement NLP based Branch–and–Bound

<table>
<thead>
<tr>
<th>SBB = Simple Branch and Bound</th>
<th>[GAMS Development Corp., 2010]</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ developed by A. Drud (ARKI Consulting)</td>
<td></td>
</tr>
<tr>
<td>▶ available within GAMS only</td>
<td></td>
</tr>
</tbody>
</table>
### Solvers that implement NLP based Branch–and–Bound

<table>
<thead>
<tr>
<th>SBB = Simple Branch and Bound</th>
<th>[GAMS Development Corp., 2010]</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ developed by A. Drud (ARKI Consulting)</td>
<td></td>
</tr>
<tr>
<td>▶ available within GAMS only</td>
<td></td>
</tr>
</tbody>
</table>

#### Bonmin = Basic Open-source Nonlinear Mixed Integer Programming

<table>
<thead>
<tr>
<th>[Bonami et al., 2008]</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ COIN-OR solver developed by P. Bonami, A. Wächter, et.al.</td>
</tr>
<tr>
<td>▶ uses CBC, CPLEX for MIP and IPOPT, FilterSQP for NLP</td>
</tr>
<tr>
<td>▶ option B-BB = NLP-based Branch-and-Bound</td>
</tr>
<tr>
<td>▶ several heuristics, branching and node selection techniques</td>
</tr>
</tbody>
</table>
Solvers that implement NLP based Branch–and–Bound

**SBB = Simple Branch and Bound**
- developed by A. Drud (ARKI Consulting)
- available within GAMS only

**Bonmin = Basic Open-source Nonlinear Mixed Integer Programming**
- COIN-OR solver developed by P. Bonami, A. Wächter, et.al.
- uses CBC, CPLEX for MIP and IPOPT, FilterSQP for NLP
- option B-BB = NLP-based Branch-and-Bound
- several heuristics, branching and node selection techniques

**MINLP_BB = Mixed Integer NonLinear Programming Branch&Bound**
- developed by S. Leyffer
- based on FilterSQP and bqpd
- advanced (strong) branching and node selection techniques

LP/MIP based techniques for solving MINLPs
DICOPT = Discrete and Continuous Optimizer

[Viswanathan and Grossmann, 1990]
- developed by J. Viswanathan and I.E. Grossmann (CMU)
- available within GAMS only
- implements augmented penalty / equation relaxation approach to cope with nonconvexities and equalities
### Solvers that implement Outer Approximation Algorithm

<table>
<thead>
<tr>
<th>Solver</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DICOPT = Discrete and Continuous Optimizer</strong></td>
<td>Developed by J. Viswanathan and I.E. Grossmann (CMU). Available within GAMS only. Implements augmented penalty / equation relaxation approach to cope with nonconvexities and equalities.</td>
</tr>
<tr>
<td><strong>AOA = AIMMS Outer Approximation</strong></td>
<td>Developed by Paragon Decision Technology. Available as “open solver” inside AIMMS. Augmented penalty approach to cope with nonconvexities.</td>
</tr>
</tbody>
</table>
### Solvers that implement Outer Approximation Algorithm

<table>
<thead>
<tr>
<th><strong>DICOPT</strong> = Discrete and Continuous Optimizer</th>
<th>[Viswanathan and Grossmann, 1990]</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ developed by J. Viswanathan and I.E. Grossmann (CMU)</td>
<td>▶ available within GAMS only</td>
</tr>
<tr>
<td>▶ implements augmented penalty / equation relaxation approach to cope with nonconvexities and equalities</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>AOA = AIMMS Outer Approximation</strong></th>
<th>[Roelofs and Bisschop, 2009]</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ developed by Paragon Decision Technology</td>
<td>▶ available as “open solver” inside AIMMS</td>
</tr>
<tr>
<td>▶ augmented penalty approach to cope with nonconvexities</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Bonmin = Basic Open-source Nonlinear Mixed Integer Programming</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ <strong>COIN-OR solver</strong> developed by P. Bonami, A. Wächter, et.al.</td>
</tr>
<tr>
<td>▶ option B-OA = Outer Approximation Algorithm</td>
</tr>
<tr>
<td>Solver</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>AlphaECP</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

FICO Xpress-SLP = Sequential Linear Programming [FICO, 2008]
- developed by Dash Optimization (acquired by FICO)
- option "MIP within SLP" = solve MIPs as subproblems in SLP algorithm

Bonmin = Basic Open-source Nonlinear Mixed Integer Programming
- COIN-OR solver developed by P. Bonami, A. Wächter, et al.
- option B-ECP = Extended Cutting Plane experimental

LP/MIP based techniques for solving MINLPs
**Solvers implementing Extended Cutting Plane Alg.**

<table>
<thead>
<tr>
<th>Solver</th>
<th>Description</th>
<th>Developed by</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AlphaECP</strong></td>
<td>$\alpha$ Extended Cutting Plane</td>
<td>T. Westerlund and T. Lastusilta at Åbo Akademi University, Finland</td>
<td></td>
</tr>
<tr>
<td></td>
<td>may also solve NLP subproblems occasionally</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>FICO Xpress-SLP</strong></td>
<td>Sequential Linear Programming</td>
<td>Dash Optimization (acquired by FICO)</td>
<td>“MIP within SLP” = solve MIPs as subproblems in SLP algorithm</td>
</tr>
</tbody>
</table>
### Solvers implementing Extended Cutting Plane Alg.

<table>
<thead>
<tr>
<th>Solver</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AlphaECP = $\alpha$ Extended Cutting Plane</strong></td>
<td>[Westerlund and Pörn, 2002]</td>
</tr>
<tr>
<td>- developed by T. Westerlund and T. Lastusilta at Åbo Akademi University, Finland</td>
<td></td>
</tr>
<tr>
<td>- may also solve NLP subproblems occasionally</td>
<td></td>
</tr>
</tbody>
</table>

| **FICO Xpress-SLP = Sequential Linear Programming** | [FIC, 2008] |
| - developed by Dash Optimization (acquired by FICO) |
| - option “MIP within SLP” = solve MIPs as subproblems in SLP algorithm |

<p>| <strong>Bonmin = Basic Open-source Nonlinear Mixed Integer Programming</strong> |
| - COIN-OR solver developed by P. Bonami, A. Wächter, et.al. |
| - option B-ECP = Extended Cutting Plane |
| - experimental |</p>
<table>
<thead>
<tr>
<th>Solvers implementing LP/NLP-based Branch&amp;Bound</th>
</tr>
</thead>
</table>

**FilMINT** = Filter-Mixed INTeger optimizer

- developed by K. Abhishek, J.T. Linderoth (Lehigh), and S. Leyffer (Argonne)
- combines MINTO (Nemhauser et.al.) with filterSQP (Fletcher)
- includes feasibility pump and strong-branching disjunctive cuts

[Abhishek et al., 2010]
FilMINT = Filter-Mixed INTeger optimizer
- developed by K. Abhishek, J.T. Linderoth (Lehigh), and S. Leyffer (Argonne)
- combines MINTO (Nemhauser et. al.) with filterSQP (Fletcher)
- includes feasibility pump and strong-branching disjunctive cuts

Bonmin = Basic Open-source Nonlinear Mixed Integer Programming
- COIN-OR solver developed by P. Bonami, A. Wächter, et.al.
- option B-QG = Quesada-Grossmann LP/NLP-based Branch-and-Bound
- option B-Hyb = alternate between LP and NLP relaxation in nodes
- implemented as extension of CBC
KNITRO

- developed by Ziena Optimization, Inc.
- can use both LPs and NLPs for bounding
<table>
<thead>
<tr>
<th><strong>Solvers implementing LP/NLP-based Branch&amp;Bound</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KNITRO</strong></td>
</tr>
<tr>
<td>[Byrd et al., 2006]</td>
</tr>
<tr>
<td>▶ developed by Ziena Optimization, Inc.</td>
</tr>
<tr>
<td>▶ can use both LPs and NLPs for bounding</td>
</tr>
</tbody>
</table>

| **FICO Xpress-SLP = Sequential Linear Programming** |
| [FIC, 2008]                                    |
| ▶ developed by Dash Optimization (acquired by FICO) |
| ▶ option “SLP within MIP” = solve NLPs as subproblems in MIP B&B |
Solvers implementing LP/NLP-based Branch&Bound

**KNITRO**
[Byrd et al., 2006]
- developed by Ziena Optimization, Inc.
- can use both LPs and NLPs for bounding

**FICO Xpress-SLP = Sequential Linear Programming**
[FIC, 2008]
- developed by Dash Optimization (acquired by FICO)
- option “SLP within MIP” = solve NLPs as subproblems in MIP B&B

**SCIP = Solving Constraint Integer Programs**
[Achterberg, 2007, Berthold et al., 2009a]
- since 1.2.0: support for MIQQP
- general MINLP in development
- close to extended cutting plane
Benchmark

Experimental Environment:

- test set from [Bonami et al., 2009b]: 48 convex MINLP instances from various applications
- solvers:
  - **AlphaECP 1.75.04**
    CPLEX for MIPs and CONOPT for NLPs
    option ECPstrategy 1
  - **BONMIN 1.3** in variants B-BB, B-QG, and B-Hyb
    CBC for MIPs and IPOPT for NLPs
  - **DICOPT 2x-C**
    CPLEX for MIPs and CONOPT for NLPs
    options stop 1, maxcycles 10000
  - **SBB**
    CONOPT for NLPs
    option memnodes 9999999
  - **SCIP 1.2.0.7** with experimental support for convex nonlinear constraints
    CPLEX for LPs and IPOPT for NLPs
- timelimit 1 hour, gap tolerance $10^{-4}$
- 2.5GHz Intel Core2 Duo, 4GB RAM

LP/MIP based techniques for solving MINLPs
Benchmark on 48 convex MINLPs

LP/MIP based techniques for solving MINLPs

- Benchmark on 48 convex MINLPs
- 100% instances solved
- Time factor w.r.t. fastest solver
- % instances solved
Overview: Nonconvex MINLP

1 Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2 Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques

Some key papers:

- P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter, Branching and bounds tightening techniques for non-convex MINLP, 2009
1 Convex MINLP
- Algorithms
- Extensions
- Solvers

2 Nonconvex MINLP
- Spatial Branch-and-Bound Algorithm
- Solvers
- Application: Mine Production Scheduling
- Techniques
How to solve Nonconvex MINLPs?

minimize \( f(x) \)

such that \( g_j(x) \leq 0, \quad j \in J \)

\( x \in X, \)

\( x_I \in \mathbb{Z}^{|I|} \)

Now: some \( g_j : \mathbb{R}^n \rightarrow \mathbb{R} \) may be nonconvex
How to solve Nonconvex MINLPs?

minimize \( f(x) \)

such that \( g_j(x) \leq 0, \quad j \in J \)

\( x \in X, \)

\( x_I \in \mathbb{Z}^{|I|} \)

Now: some \( g_j : \mathbb{R}^n \to \mathbb{R} \) may be nonconvex

⇒ inequalities \( g_j(\hat{x}) + \nabla g_j(\hat{x})^T (x - \hat{x}) \leq 0 \) may not be valid!
How to solve Nonconvex MINLPs?

minimize $f(x)$
such that $g_j(x) \leq 0, \quad j \in J$
$x \in X,$
$x_I \in \mathbb{Z}^{|I|}$

Now: some $g_j : \mathbb{R}^n \rightarrow \mathbb{R}$ may be nonconvex

$\Rightarrow$ inequalities $g_j(\hat{x}) + \nabla g_j(\hat{x})^T(x - \hat{x}) \leq 0$ may not be valid!
$\Rightarrow$ use techniques from global optimization of NLPs
$\Rightarrow$ use convex underestimator: convex and below $g(x)$ for all $x \in X$
$\Rightarrow$ introduces convexification gap
Spatial branch-and-bound Algorithm

[Smith and Pantelides, 1999, Tawarmalani and Sahinidis, 2002]

In each node with local bounds \([\underline{x}, \overline{x}]\):

**Bounding:** generate a linear outer-approximation w.r.t. \([\underline{x}, \overline{x}]\) using convexification and linearization techniques

\[
\begin{align*}
\min & \quad \alpha \\
\text{s.t.} & \quad A x + A' \alpha \leq b, \\
& \quad x \in X, \quad x \in [\underline{x}, \overline{x}]
\end{align*}
\]
In each node with local bounds $[\underline{x}, \overline{x}]$:

**Bounding:** generate a linear outer-approximation w.r.t. $[\underline{x}, \overline{x}]$ using convexification and linearization techniques

$$\begin{align*}
\min & \quad \alpha \\
\text{s.t.} & \quad Ax + A'\alpha \leq b, \\
& \quad x \in X, \quad x \in [\underline{x}, \overline{x}] 
\end{align*}$$

**Branching:** close gap between relaxation and problem

- branch on integer variables with fractional value in LP
Spatial branch-and-bound Algorithm

[Smith and Pantelides, 1999, Tawarmalani and Sahinidis, 2002]

In each node with local bounds $[\underline{x}, \bar{x}]$:

**Bounding:** generate a linear outer-approximation w.r.t. $[\underline{x}, \bar{x}]$ using convexification and linearization techniques

$$\min \alpha$$

$$\text{s.t.} \quad Ax + A'\alpha \leq b,$$

$$x \in X, \quad x \in [\underline{x}, \bar{x}]$$

**Branching:** close gap between relaxation and problem

- branch on integer variables with fractional value in LP
- branch on continuous variables in nonconvex terms

$\Rightarrow$ smaller domain $\Rightarrow$ tighter relaxation
The linear outer-approximation of a nonconvex MINLP

**Given:** A function $g : \mathbb{R}^n \rightarrow \mathbb{R}$ defining the constraint

$$g(x) \leq 0, \quad x \in [x, \bar{x}]$$

**Seek:** A (linear) outer-approximation of

$$\text{conv}\{x \in [x, \bar{x}] : g(x) \leq 0\}$$
The linear outer-approximation of a nonconvex MINLP

Given: A function $g : \mathbb{R}^n \rightarrow \mathbb{R}$ defining the constraint

$$g(x) \leq 0, \quad x \in [\underline{x}, \bar{x}]$$

Seek: A (linear) outer-approximation of

$$\text{conv}\{x \in [\underline{x}, \bar{x}] : g(x) \leq 0\}$$

$\Rightarrow$ too difficult in general
The linear outer-approximation of a nonconvex MINLP

**Given:** A function \( g : \mathbb{R}^n \to \mathbb{R} \) defining the constraint

\[
g(x) \leq 0, \quad x \in [\underline{x}, \bar{x}]
\]

**Seek:** A convex function \( g^c(\cdot) \) that underestimates \( g(\cdot) \) on \([\underline{x}, \bar{x}]\):

\[
\{ x \in [\underline{x}, \bar{x]} : g(x) \leq 0 \} \subseteq \{ x \in [\underline{x}, \bar{x}] : g^c(x) \leq 0 \}
\]
Given: A function \( g : \mathbb{R}^n \rightarrow \mathbb{R} \) defining the constraint
\[
g(x) \leq 0, \quad x \in [\underline{x}, \overline{x}]
\]

Seek: A convex function \( g^c(\cdot) \) that underestimates \( g(\cdot) \) on \([\underline{x}, \overline{x}]\):

\[
\{x \in [\underline{x}, \overline{x}] : g(x) \leq 0\} \subseteq \{x \in [\underline{x}, \overline{x}] : g^c(x) \leq 0\} \\
\subseteq \{x \in [\underline{x}, \overline{x}] : g^c(x^k) + \nabla g^c(x^k)^T(x - x^k) \leq 0, \ k \leq K\}
\]
Given: A function \( g : \mathbb{R}^n \rightarrow \mathbb{R} \) defining the constraint
\[
g(x) \leq 0, \quad x \in [x, \bar{x}]
\]

Seek: The tightest convex function \( g^c(\cdot) \) that underestimates \( g(\cdot) \) on \( [x, \bar{x}] \):

The convex envelope \( g^e : [x, \bar{x}] \rightarrow \mathbb{R} \) is a function such that

- **analytically:**
  - \( g^e(x) \) is convex on \( [x, \bar{x}] \)
  - \( g^e(x) \leq g(x) \) for all \( x \in [x, \bar{x}] \)
  - \( \forall h : [x, \bar{x}] \rightarrow \mathbb{R}, \ h(\cdot) \) convex, \( h(\cdot) \leq g(\cdot) : \ h(\cdot) \leq g^e(\cdot) \)
Given: A function \( g : \mathbb{R}^n \rightarrow \mathbb{R} \) defining the constraint

\[
g(x) \leq 0, \quad x \in [x, \bar{x}]
\]

Seek: The tightest convex function \( g^c(\cdot) \) that underestimates \( g(\cdot) \) on \([x, \bar{x}]\):

The convex envelope \( g^e : [x, \bar{x}] \rightarrow \mathbb{R} \) is a function such that

- **analytically:**
  - \( g^e(x) \) is convex on \([x, \bar{x}]\)
  - \( g^e(x) \leq g(x) \) for all \( x \in [x, \bar{x}] \)
  - \( \forall h : [x, \bar{x}] \rightarrow \mathbb{R}, \ h(\cdot) \) convex, \( h(\cdot) \leq g(\cdot) : h(\cdot) \leq g^e(\cdot) \)

- **geometrically:** \( \text{epi}(g^e(\cdot)) = \text{conv}(\text{epi}(g(\cdot))) \)
The linear outer-approximation of a nonconvex MINLP

Given: A function $g : \mathbb{R}^n \rightarrow \mathbb{R}$ defining the constraint

$$g(x) \leq 0, \quad x \in [x, \bar{x}]$$

Seek: The tightest convex function $g^{c}(\cdot)$ that underestimates $g(\cdot)$ on $[x, \bar{x}]$:

The convex envelope $g^{e} : [x, \bar{x}] \rightarrow \mathbb{R}$ is a function such that

▷ analytically:
  ▷ $g^{e}(x)$ is convex on $[x, \bar{x}]$
  ▷ $g^{e}(x) \leq g(x)$ for all $x \in [x, \bar{x}]$
  ▷ $\forall h : [x, \bar{x}] \rightarrow \mathbb{R}$, $h(\cdot)$ convex, $h(\cdot) \leq g(\cdot)$: $h(\cdot) \leq g^{e}(\cdot)$

▷ geometrically: $\text{epi}(g^{e}(\cdot)) = \text{conv}(\text{epi}(g(\cdot)))$

In general: Convex envelopes are hard to find.
(finding the convex envelope of a function is as hard as finding its global min.)
Theorem (Tawarmalani and Sahinidis [2002])

Let $v_1, \ldots, v_k$ be the vertices of a polytope $P$. The convex envelope $g_e(x)$ of a concave function $g(x)$ over $P$ can be expressed as

$$g_e(x) = \min_{\lambda} \sum_{i=1}^{k} \lambda_i g(v_i)$$

subject to $x = \sum_{i=1}^{k} \lambda_i v_i$, $\sum_{i=1}^{k} \lambda_i = 1$, $\lambda_i \geq 0$, $i = 1, \ldots, k$.

For now: Special case $n=1, P=[x, x]$.

Convex envelope for univariate concave functions is the secant between $x$ and $x$.

⇒ log($x$), −exp($x$), $-x^2$, √$x$, . . .

In branch-and-bound: tighten by branching LP/MIP based techniques for solving MINLPs.
Theorem (Tawarmalani and Sahinidis [2002])

Let $v_1, \ldots, v_k$ be the vertices of a polytope $P$. The convex envelope $g^e(x)$ of a concave function $g(x)$ over $P$ can be expressed as

$$g^e(x) = \min_{\lambda} \sum_{i=1}^{k} \lambda_i g(v_i)$$

s.t.

$$x = \sum_{i=1}^{k} \lambda_i v_i, \quad \sum_{i=1}^{k} \lambda_i = 1, \quad \lambda_i \geq 0, \quad i = 1, \ldots, k$$
Theorem (Tawarmalani and Sahinidis [2002])

Let \( v_1, \ldots, v_k \) be the vertices of a polytope \( P \). The convex envelope \( g^e(x) \) of a concave function \( g(x) \) over \( P \) can be expressed as

\[
g^e(x) = \min_{\alpha} \sum_{i=1}^{k} \lambda_i g(v_i)
\]

\[
\text{s.t. } x = \sum_{i=1}^{k} \lambda_i v_i, \quad \sum_{i=1}^{k} \lambda_i = 1, \quad \lambda_i \geq 0, \quad i = 1, \ldots, k
\]

For now: Special case \( n = 1, \ P = [\underline{x}, \bar{x}] \)

- Convex envelope for univariate concave functions is the secant between \( \underline{x} \) and \( \bar{x} \).

\[ \Rightarrow \log(x), -\exp(x), -x^{2k}, \sqrt{x}, \ldots \]
Theorem (Tawarmalani and Sahinidis [2002])

Let \( v_1, \ldots, v_k \) be the vertices of a polytope \( P \). The convex envelope \( g^e(x) \) of a concave function \( g(x) \) over \( P \) can be expressed as

\[
g^e(x) = \min_{\lambda} \sum_{i=1}^{k} \lambda_i g(v_i)
\]

subject to

\[
x = \sum_{i=1}^{k} \lambda_i v_i, \quad \sum_{i=1}^{k} \lambda_i = 1, \quad \lambda_i \geq 0, \quad i = 1, \ldots, k
\]

For now: Special case \( n = 1, P = [x, \bar{x}] \)

Convex envelope for univariate concave functions is the secant between \( x \) and \( \bar{x} \).

\( \Rightarrow \) \( \log(x), -\exp(x), -x^{2k}, \sqrt{x}, \ldots \)

In branch-and-bound: tighten by branching
Convex envelope for monomials of odd degree

\[ g(x) = x^{2k+1}, \quad k \in \mathbb{N} \]

[Liberti and Pantelides, 2003]
Convex envelope for monomials of odd degree

\[ g(x) = x^{2k+1}, \quad k \in \mathbb{N} \]

Seek for secant from \( x \) to \( c \), where \( c \geq 0 \) is the coordinate where

\[ g'(c) = \frac{c^{2k+1} - x^{2k+1}}{c - x} \]
Convex envelope for monomials of odd degree

\[
g(x) = x^{2k+1}, \quad k \in \mathbb{N}
\]

Seek for secant from \( x \) to \( c \), where \( c \geq 0 \) is the coordinate where

\[
g'(c) = \frac{c^{2k+1} - x^{2k+1}}{c - x}
\]

\( c \) is the positive real root of the polynomial

\[
2kx^{2k+1} - x(2k + 1)x^k + x^{2k+1}
\]
Convex envelope for monomials of odd degree

\[ g(x) = x^{2k+1}, \quad k \in \mathbb{N} \]

▷ Seek for secant from \( x \) to \( c \), where \( c \geq 0 \) is the coordinate where

\[ g'(c) = \frac{c^{2k+1} - x^{2k+1}}{c - x} \]

⇒ \( c \) is the positive real root of the polynomial

\[ 2kx^{2k+1} - x(2k + 1)x^k + x^{2k+1} \]

<table>
<thead>
<tr>
<th>( k )</th>
<th>( c ) (for ( x = -1 ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5000000000</td>
</tr>
<tr>
<td>2</td>
<td>0.6058295862</td>
</tr>
<tr>
<td>3</td>
<td>0.6703320476</td>
</tr>
<tr>
<td>4</td>
<td>0.7145377272</td>
</tr>
<tr>
<td>5</td>
<td>0.7470540749</td>
</tr>
<tr>
<td>6</td>
<td>0.7721416355</td>
</tr>
<tr>
<td>7</td>
<td>0.7921778546</td>
</tr>
</tbody>
</table>
Convex envelope for monomials of odd degree

\[ g(x) = x^{2k+1}, \quad k \in \mathbb{N} \]

▷ Seek for secant from \( x \) to \( c \), where \( c \geq 0 \) is the coordinate where

\[ g'(c) = \frac{c^{2k+1} - x^{2k+1}}{c - x} \]

\[ \Rightarrow \quad c \text{ is the positive real root of the polynomial} \]

\[ 2kx^{2k+1} - x(2k+1)x^{2k} + x^{2k+1} \]

▷ obtain convex envelope

<table>
<thead>
<tr>
<th>( k )</th>
<th>( c ) ( (\text{for } x = -1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5000000000</td>
</tr>
<tr>
<td>2</td>
<td>0.6058295862</td>
</tr>
<tr>
<td>3</td>
<td>0.6703320476</td>
</tr>
<tr>
<td>4</td>
<td>0.7145377272</td>
</tr>
<tr>
<td>5</td>
<td>0.7470540749</td>
</tr>
<tr>
<td>6</td>
<td>0.7721416355</td>
</tr>
<tr>
<td>7</td>
<td>0.7921778546</td>
</tr>
</tbody>
</table>
Convex envelope for monomials of odd degree

\[ g(x) = x^{2k+1}, \quad k \in \mathbb{N} \]

▷ Seek for secant from \( x \) to \( c \), where \( c \geq 0 \) is the coordinate where

\[ g'(c) = \frac{c^{2k+1} - x^{2k+1}}{c - x} \]

⇒ \( c \) is the positive real root of the polynomial

\[ 2kx^{2k+1} - x(2k + 1)x^{2k} + x^{2k+1} \]

▷ obtain convex envelope and linear approximation

<table>
<thead>
<tr>
<th>( k )</th>
<th>( c ) (for ( x = -1 ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5000000000</td>
</tr>
<tr>
<td>2</td>
<td>0.6058295862</td>
</tr>
<tr>
<td>3</td>
<td>0.6703320476</td>
</tr>
<tr>
<td>4</td>
<td>0.7145377272</td>
</tr>
<tr>
<td>5</td>
<td>0.7470540749</td>
</tr>
<tr>
<td>6</td>
<td>0.7721416355</td>
</tr>
<tr>
<td>7</td>
<td>0.7921778546</td>
</tr>
</tbody>
</table>
Convex envelope for monomials of odd degree

\[ g(x) = x^{2k+1}, \quad k \in \mathbb{N} \]

- Seek for secant from \( x \) to \( c \), where \( c \geq 0 \) is the coordinate where

\[
g'(c) = \frac{c^{2k+1} - x^{2k+1}}{c - x}
\]

- \( c \) is the positive real root of the polynomial

\[
2kx^{2k+1} - x(2k + 1)x^{2k} + x^{2k+1}
\]

- obtain convex envelope and linear approximation

- easily generalized to \( x|x|^{n-1} = \text{sign}(x)|x|^n, \quad n \geq 1 \)

(application for flow of water and gas in pipes)

\begin{tabular}{|c|c|}
\hline
\( k \) & \( c \) (for \( x = -1 \)) \\
\hline
1 & 0.5000000000 \\
2 & 0.6058295862 \\
3 & 0.6703320476 \\
4 & 0.7145377272 \\
5 & 0.7470540749 \\
6 & 0.7721416355 \\
7 & 0.7921778546 \\
\hline
\end{tabular}
McCormick: The convex envelope of

\[ f(x) = xy \]

for \( x \in [\underline{x}, \overline{x}] \), \( y \in [\underline{y}, \overline{y}] \) is

\[ xy \geq \max \left\{ \underline{x}y + \overline{y}x - \underline{xy}, \underline{xy} + \overline{y}x - \overline{xy} \right\} \]
McCormick: The convex envelope of

\[ f(x) = xy \]

for \( x \in [x, \bar{x}], y \in [y, \bar{y}] \) is

\[ xy \geq \max \left\{ \bar{xy} + \bar{yx} - \bar{xy}, \bar{xy} + \bar{yx} - \bar{xy} \right\} \]

▷ apply McCormick underestimators to (nonconvex) quadratic forms

\[ f(x) = \sum_{i,j} a_{i,j}x_i x_j \]

▷ branching on \( x \) and \( y \) changes variable bounds

⇒ tighter underestimator ⇒ improving dual bounds
Reformulation for the general (factorable) case

[Tawarmalani and Sahinidis, 2004]

Let \( g : \mathbb{R}^m \rightarrow \mathbb{R} \) be factorable, i.e., a recursive sum and product of univariate functions

Examples:
- \( g(x) = x_1 x_2 \)
- \( g(x) = x_1 / x_2 \)
- \( g(x) = \sqrt{\exp(x_1 x_2 + x_3 \ln x_4)x_3^3} \)
Reformulation for the general (factorable) case

[Tawarmalani and Sahinidis, 2004]

Let \( g : \mathbb{R}^m \rightarrow \mathbb{R} \) be factorable, i.e., a recursive sum and product of univariate functions

Examples: \( g(x) = x_1 x_2 \), \( g(x) = x_1 / x_2 \), \( g(x) = \sqrt{\exp(x_1 x_2 + x_3 \ln x_4)} x_3^3 \)

Reformulation: replace products of functions or variables by new variables

\[
\begin{align*}
g &= \sqrt{\exp(y_1)} y_2 \\
y_1 &= x_1 x_2 + x_3 \ln(x_4) \\
y_2 &= x_3^3
\end{align*}
\]
Reformulation for the general (factorable) case

[Tawarmalani and Sahinidis, 2004]

Let \( g : \mathbb{R}^m \to \mathbb{R} \) be factorable, i.e., a recursive sum and product of univariate functions.

Examples: \( g(x) = x_1 x_2 \), \( g(x) = x_1 / x_2 \), \( g(x) = \sqrt{\exp(x_1 x_2 + x_3 \ln x_4)x_3^3} \)

Reformulation: replace products of functions or variables by new variables

\[
\begin{align*}
g & = \sqrt{\exp(y_1)y_2} \\
y_1 & = x_1 x_2 + x_3 \ln(x_4) \\
y_2 & = x_3^3
\end{align*}
\]
Reformulation for the general (factorable) case

[Tawarmalani and Sahinidis, 2004]

Let $g : \mathbb{R}^m \to \mathbb{R}$ be factorable, i.e., a recursive sum and product of univariate functions.

Examples: $g(x) = x_1x_2$, $g(x) = x_1/x_2$, $g(x) = \sqrt{\exp(x_1x_2 + x_3 \ln x_4)x_3^3}$

Reformulation: replace products of functions or variables by new variables

\begin{align*}
g &= \sqrt{y_3y_2} \\
y_1 &= x_1x_2 + x_3y_4 \\
y_2 &= x_3^3 \\
y_3 &= \exp(y_1) \\
y_4 &= \ln(x_4)
\end{align*}
Reformulation for the general (factorable) case

Let $g : \mathbb{R}^m \rightarrow \mathbb{R}$ be factorable, i.e., a recursive sum and product of univariate functions

Examples: $g(x) = x_1 x_2$, $g(x) = x_1/x_2$, $g(x) = \sqrt{\exp(x_1 x_2 + x_3 \ln x_4) x_3^3}$

Reformulation: replace products of functions or variables by new variables

$$
g = \sqrt{y_3 y_2} \\
y_1 = x_1 x_2 + x_3 y_4 \\
y_2 = x_3^3 \\
y_3 = \exp(y_1) \\
y_4 = \ln(x_4)
$$
Reformulation for the general (factorable) case

Let $g : \mathbb{R}^m \rightarrow \mathbb{R}$ be factorable, i.e., a recursive sum and product of univariate functions.

Examples: $g(x) = x_1 x_2$, $g(x) = x_1 / x_2$, $g(x) = \sqrt{\exp(x_1 x_2 + x_3 \ln x_4) x_3^3}$

Reformulation: replace products of functions or variables by new variables

\[
g = \sqrt{y_5} \\
y_1 = x_1 x_2 + x_3 y_4 \\
y_2 = x_3^3 \\
y_3 = \exp(y_1) \\
y_4 = \ln(x_4) \\
y_5 = y_3 y_2
\]
Reformulation for the general (factorable) case

Let $g : \mathbb{R}^m \rightarrow \mathbb{R}$ be factorable, i.e., a recursive sum and product of univariate functions

Examples: $g(x) = x_1 x_2$, $g(x) = x_1 / x_2$, $g(x) = \sqrt{\exp(x_1 x_2 + x_3 \ln x_4) x_3^3}$

Reformulation: replace products of functions or variables by new variables

$$g = \sqrt{y_5}$$
$$y_1 = x_1 x_2 + x_3 y_4$$
$$y_2 = x_3^3$$
$$y_3 = \exp(y_1)$$
$$y_4 = \ln(x_4)$$
$$y_5 = y_3 y_2$$

Convex outer-approximation by

- estimating univariate functions by known convex envelopes
- estimating bilinear terms by McCormick underestimators
Example for convexification via factorable reformulation

\[ g(x) = \sqrt{\exp(x_1^2) \ln(x_2)} \]

\[ x_1 \in [0, 2], \quad x_2 \in [1, 2] \]
Example for convexification via factorable reformulation

\[ g(x) = \sqrt{\exp(x_1^2) \ln(x_2)} \]

\[ x_1 \in [0, 2], \quad x_2 \in [1, 2] \]

Factorable Reformulation:

\[ g = \sqrt{y_1} \]

\[ y_1 = y_2 y_3 \]

\[ y_2 = \exp(y_4) \]

\[ y_3 = \ln(x_2) \]

\[ y_4 = x_1^2 \]
Example for convexification via factorable reformulation

\[ g(x) = \sqrt{\exp(x_1^2) \ln(x_2)} \]

\[ x_1 \in [0, 2], \quad x_2 \in [1, 2] \]

Factorable Reformulation:

\[ g = \sqrt{y_1} \]
\[ y_1 = y_2 y_3 \quad [0, \ln(2) e^4] \]
\[ y_2 = \exp(y_4) \quad [1, e^4] \]
\[ y_3 = \ln(x_2) \quad [0, \ln(2)] \]
\[ y_4 = x_1^2 \quad [0, 4] \]
Example for convexification via factorable reformulation

\[ g(x) = \sqrt{\exp(x_1^2) \ln(x_2)} \]

\[ x_1 \in [0, 2], \quad x_2 \in [1, 2] \]

**Factorable Reformulation:**

\[
\begin{align*}
    g &= \sqrt{y_1} \\
    y_1 &= y_2 y_3 \quad [0, \ln(2)e^4] \\
    y_2 &= \exp(y_4) \quad [1, e^4] \\
    y_3 &= \ln(x_2) \quad [0, \ln(2)] \\
    y_4 &= x_1^2 \quad [0, 4]
\end{align*}
\]

**Convex relaxation:**

\[
\sqrt{y_1} + \frac{y_1 - y_1}{\sqrt{y_1} + \sqrt{y_1}} \leq g \leq \sqrt{y_1}; \quad \ln x_1 + (x_2 - x_2) \frac{\ln \bar{x}_2 - \ln x_2}{x_2 - x_2} \leq y_3 \leq \ln(x_2)
\]

\[
\max \left\{ \frac{\overline{y_2}y_3 + \overline{y_3}y_2 - \overline{y_2}y_3}{y_2y_3 + \overline{y_3}y_2 - y_2y_3} \right\} \leq y_1 \leq \min \left\{ \frac{\overline{y_2}y_3 + \overline{y_3}y_2 - \overline{y_2}y_3}{y_2y_3 + \overline{y_3}y_2 - y_2y_3} \right\}
\]

\[ \ldots \]
Example for convexification via factorable reformulation

\[ g(x) = \sqrt{\exp(x_1^2) \ln(x_2)} \]

\[ x_1 \in [0, 2], \quad x_2 \in [1, 2] \]

**Factorable Reformulation:**

\[
\begin{align*}
    g & = \sqrt{y_1} \\
    y_1 & = y_2 y_3 \quad [0, \ln(2)e^4] \\
    y_2 & = \exp(y_4) \quad [1, e^4] \\
    y_3 & = \ln(x_2) \quad [0, \ln(2)] \\
    y_4 & = x_1^2 \quad [0, 4]
\end{align*}
\]

**Convex relaxation:**

\[
\begin{align*}
    \sqrt{y_1} + \frac{y_1 - y_1}{\sqrt{y_1} + \sqrt{y_1}} \leq g \leq \sqrt{y_1}; \quad \ln x_1 + (x_2 - x_2) \frac{\ln x_2 - \ln x_2}{x_2 - x_2} & \leq y_3 \leq \ln(x_2) \\
    \max \left\{ \frac{y_2 y_3 + y_3 y_2 - y_2 y_3}{y_2 y_3 + y_3 y_2 - y_2 y_3} \right\} & \leq y_1 \leq \min \left\{ \frac{y_2 y_3 + y_3 y_2 - y_2 y_3}{y_2 y_3 + y_3 y_2 - y_2 y_3} \right\}
\end{align*}
\]

LP/MIP based techniques for solving MINLPs
Example for convexification via factorable reformulation

\[ g(x) = \sqrt{\exp(x_1^2) \ln(x_2)} \]

\[ x_1 \in [0, 2], \quad x_2 \in [1, 2] \]

**Factorable Reformulation:**

\[ g = \sqrt{y_1} \]

\[ y_1 = y_2 y_3 \quad [0, \ln(2) e^4] \]

\[ y_2 = \exp(y_4) \quad [1, e^4] \]

\[ y_3 = \ln(x_2) \quad [0, \ln(2)] \]

\[ y_4 = x_1^2 \quad [0, 4] \]

**Convex relaxation:**

\[ \sqrt{y_1} + \frac{y_1 - y_1}{\sqrt{y_1} + \sqrt{y_1}} \leq g \leq \sqrt{y_1}; \quad \ln x_1 + (x_2 - x_2) \ln \frac{x_2 - \ln x_2}{x_2 - x_2} \leq y_3 \leq \ln(x_2) \]

\[ \max \left\{ \frac{y_2 y_3 + y_3 y_2 - y_2 y_3}{y_2 y_3 + y_3 y_2 - y_2 y_3} \right\} \leq y_1 \leq \min \left\{ \frac{y_2 y_3 + y_3 y_2 - y_2 y_3}{y_2 y_3 + y_3 y_2 - y_2 y_3} \right\} \]

\[ \ldots \]
1. Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2. Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques
### BARON = Branch And Reduce Optimization Navigator

[Tawarmalani and Sahinidis, 2002]

- developed by N. Sahinidis, M. Tawarmalani
- state-of-the-art convexification techniques
- winner of Beale-Orchard-Hays Prize 2006
### Solvers for nonconvex MINLP

<table>
<thead>
<tr>
<th>Solver</th>
<th>Description</th>
</tr>
</thead>
</table>
| **BARON** = Branch And Reduce Optimization Navigator | [Tawarmalani and Sahinidis, 2002]  
▶ developed by N. Sahinidis, M. Tawarmalani  
▶ state-of-the-art convexification techniques  
▶ winner of Beale-Orchard-Hays Prize 2006 |
| **Couenne** = Convex Over + Under ENvelopes for Nonlinear Estimation | [Belotti et al., 2009]  
▶ COIN-OR solver developed by P. Belotti et.al.  
▶ methodology similar to BARON  
▶ winner of COIN-OR cup competition 2010 |
### Solvers for nonconvex MINLP

#### BARON = Branch And Reduce Optimization Navigator

- [Tawarmalani and Sahinidis, 2002](#)
- developed by N. Sahinidis, M. Tawarmalani
- state-of-the-art convexification techniques
- winner of Beale-Orchard-Hays Prize 2006

#### Couenne = Convex Over + Under ENvelopes for Nonlinear Estimation

- [Belotti et al., 2009](#)
- COIN-OR solver developed by P. Belotti et.al.
- methodology similar to BARON
- winner of COIN-OR cup competition 2010

#### LindoGlobal

- [Lin and Schrage, 2009](#)
- developed by Y. Lin and L. Schrage
- some automatic reformulation techniques
  (SOC, floor, abs, max, ...)

LP/MIP based techniques for solving MINLPs

- developed by A. Mahajan, T. Munson, et.al.
- emphasis on reformulation techniques (multilinear forms, SOC)
MINOTAUR = Mixed-Integer Nonconvex Optimization Toolbox – Algorithms, Underestimators, Relaxations [Mahajan and Munson, 2010]

- developed by A. Mahajan, T. Munson, et.al.
- emphasis on reformulation techniques (multilinear forms, SOC)

COCONUT = COntinuous CONstraints - Updating the Technology [Neumaier, 2001]

- developed by H. Schichl, A. Neumaier, et.al. (Vienna)
- emphasis on rigorous computations
- many interval-arithmetic based techniques
### Solvers for nonconvex MINLP in Development

**MINOTaur** = Mixed-Integer Nonconvex Optimization Toolbox – Algorithms, Underestimators, Relaxations  
[Mahajan and Munson, 2010]

- Developed by A. Mahajan, T. Munson, et.al.
- Emphasis on reformulation techniques  
  (multilinear forms, SOC)

**COCONUT** = COntinuous CONstraints - Updating the Technology  
[Neumaier, 2001]

- Developed by H. Schichl, A. Neumaier, et.al. (Vienna)
- Emphasis on rigorous computations
- Many interval-arithmetic based techniques

**SCIP** = Solving Constraint Integer Programs  
[Berthold et al., 2009a]

- Support for MIQQP so far
- Strong in MIP part

LP/MIP based techniques for solving MINLPs
Experimental Environment:

- **test set:** 143 MINLP instances from MINLPLib [Bussieck et al., 2003]
  - convex and nonconvex, easy and hard

- **solvers:**
  - **BARON** 9.0.8
    - CPLEX for LPs and MINOS for NLPs
  - **LindoGlobal** 6.1.1
    - CPLEX for LPs and CONOPT for NLPs
  - **Couenne** 0.3
    - CLP for LPs and IPOPT for NLPs

- **timelimit** 1 hour, gap tolerance $10^{-4}$

- **3.0GHz Intel Core2 Extreme CPU X9650, 16GB RAM**
Benchmark on 143 MINLPs

LP/MIP based techniques for solving MINLPs
1. Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2. Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques
Open Pit Mine Production Scheduling with a Stockpile

[Bley et al., 2009a]
Given:
- Blocks (or aggregates) (rock, metal grade, precedences)
- Capacities, Costs (Mining, Processing; per rock ton)
- Metal Prices

Decisions:
- When to mine which block?
- Which block to process?

Goal:
max NPV

Solution Approach:
- Time-indexed MILP formulation
Practical extension:
Stockpile for interim storage
better use of capacities

Difficulty:
stockpile mixes material
Aggregated stockpile model

\[ f_{i,t}^L \in [0,1] \quad \text{% of block } i \text{ into stockpiled} \]

\[ Q_{t}^{\text{rock}}, Q_{t}^{\text{met}} \quad \text{total rock / metal tons held} \]

\[ P_{t}^{\text{rock}}, P_{t}^{\text{met}} \quad \text{total rock / metal tons out} \]

Mixing constraints:

\[ \frac{P_{t}^{\text{met}}}{Q_{t}^{\text{met}}} = \frac{P_{t}^{\text{rock}}}{Q_{t}^{\text{rock}}} \quad \text{(metal fraction out = rock fraction out)} \]

**Practical extension:**

Stockpile for interim storage

better use of capacities

**Difficulty:**

stockpile mixes material

**Mathematical model:**

Same rock / metal – mix

held and taken out of stockpile

Nonlinear constraints

LP/MIP based techniques for solving MINLPs
Aggregate MINLP formulation

\[
\text{max} \quad \sum_{t \in \text{years}} \left[ v_t \left( P^\text{met}_t + \sum_{i \in \text{blocks}} \alpha^\text{met}_i f^P_{i,t} \right) - c^P_t \left( P^\text{rock}_t + \sum_{i \in \text{blocks}} \alpha^\text{rock}_i f^P_{i,t} \right) - c^M_t \sum_{i \in \text{blocks}} \alpha^\text{rock}_i f^M_{i,t} \right]
\]

s.t.
\[
\sum_{i \in \text{blocks}} \alpha^\text{rock}_i f^M_{i,t} \leq MC_t \quad \forall t
\]

\[
P^\text{rock}_t + \sum_{i \in \text{blocks}} \alpha^\text{rock}_i f^P_{i,t} \leq PC_t \quad \forall t
\]

\[
\sum_{s=1}^{l} f^M_{i,s} \geq x_{i,t} \quad \forall t, i
\]

\[
\sum_{s=1}^{l} f^M_{i,s} \leq x_{j,t} \quad \forall t, j \in \text{Pred}(i)
\]

\[
\sum_{i \in \text{blocks}} \alpha^\text{rock}_i f^I_{i,t} + Q^\text{rock}_t - Q^\text{rock}_{t+1} - R^\text{rock}_{t+1} = 0 \quad \forall t
\]

\[
\vdots
\]

\[
P^\text{met}_t Q^\text{rock}_t - Q^\text{met}_t P^\text{rock}_t = 0 \quad \forall t
\]

\[
0 \leq f^*_{i,t} \leq 1
\]

\[
P^*_t, Q^*_t \geq 0
\]

\[
x_{i,t} \in \{0,1\}
\]
- track each block’s fractions individually through the stockpile
- best possible knowledge of the material in the stockpile

\[
\begin{align*}
    f_{i,t}^O & \in [0,1] \% \text{ of block } i \text{ out of stockpile} \\
    f_{i,t}^R & \in [0,1] \% \text{ of block } i \text{ kept in stockpile} \\
    P_{t}^{\text{rock}} & = \sum_{i \in \text{blocks}} \alpha_i^{\text{rock}} f_{i,t}^O \quad \text{total rock out}
\end{align*}
\]
- track each block’s fractions individually through the stockpile
- best possible knowledge of the material in the stockpile
- require \[ \frac{f_{i,t}^O}{f_{i,t}^O + f_{i,t}^R} = r_t \]
  for each period and block
- much tighter dual bound

\[ f_{i,t}^O \in [0,1] \text{ } \% \text{ of block } i \text{ out of stockpile} \]
\[ f_{i,t}^R \in [0,1] \text{ } \% \text{ of block } i \text{ kept in stockpile} \]

\[ P_t^{rock} = \sum_{i \in blocks} \alpha_i^{rock} f_{i,t}^O \quad \text{total rock out} \]
State-of-the-art B&B-algorithm [Bley et al., 2009a]

- ignore nonlinearities to obtain linear relaxation and enforce these by special geometric branching scheme
State-of-the-art B&B-algorithm [Bley et al., 2009a]

- ignore nonlinearities to obtain linear relaxation and enforce these by special geometric branching scheme
- application-specific heuristic for primal solutions
Problem-specific algorithm

State-of-the-art B&B-algorithm [Bley et al., 2009a]

- ignore nonlinearities to obtain linear relaxation and enforce these by special geometric branching scheme
- application-specific heuristic for primal solutions
- variable fixing and cutting planes from linear precedence constrained knapsack structure
Problem-specific algorithm

State-of-the-art B&B-algorithm [Bley et al., 2009a]

- ignore nonlinearities to obtain linear relaxation and enforce these by special geometric branching scheme
- application-specific heuristic for primal solutions
- variable fixing and cutting planes from linear precedence constrained knapsack structure

How close can generic MINLP solvers get without problem-specific knowledge? [Bley et al., 2009b]
Mixing constraints:

\[
\frac{f_{i,t}^O}{f_{i,t}^O + f_{i,t}^R} = r_t
\]
Mixing constraints:

\[ f_{i,t}^O = r_t (f_{i,t}^O + f_{i,t}^R) \]

with bounds \( r_t \in [r_t, \bar{r}_t] \), \( f_{i,t}^O + f_{i,t}^R \in [\underline{f}_{i,t}^O + \underline{f}_{i,t}^R, \bar{f}_{i,t}^O + \bar{f}_{i,t}^R] \).
Mixing constraints:

\[ f_{i,t}^O = r_t(f_{i,t}^O + f_{i,t}^R) \]

with bounds \( r_t \in [r_t, \bar{r}_t], f_{i,t}^O + f_{i,t}^R \in [\underline{f}_{i,t}^O + \underline{f}_{i,t}^R, \bar{f}_{i,t}^O + \bar{f}_{i,t}^R] \).

McCormick underestimators provide a linear outer approximation for each bilinear term \((xy)\):

\[ xy \geq xy + yx - xy \]
\[ xy \geq \bar{y}x + yx - xy \]
\[ xy \leq xy + yx - xy \]
\[ xy \leq \bar{x}y + yx - xy \]
\[ xy \leq \bar{x}y + yx - \bar{x}y \]
Mixing constraints:

\[ f_{i,t}^O = r_t(f_{i,t}^O + f_{i,t}^R) \]

with bounds \( r_t \in [r_t, \overline{r}_t] \), \( f_{i,t}^O + f_{i,t}^R \in [\underline{f}_{i,t}^O + \underline{f}_{i,t}^R, \overline{f}_{i,t} + \overline{f}_{i,t}^R] \).

McCormick underestimators provide a linear outer approximation for each bilinear term \((xy)\):

\[
\begin{align*}
xy & \geq xy + yx - xy \\
xy & \geq \overline{xy} + \overline{yx} - \overline{xy} \\
xy & \leq \underline{xy} + \underline{yx} - \underline{xy} \\
xy & \leq \overline{xy} + \overline{yx} - \overline{xy}
\end{align*}
\]

Separate dynamically if violated, otherwise perform spatial branching.
Primal solutions from: feasible LP solution, MIP heuristics, NLP local search, extended LNS heuristics

New heuristic: Undercover [Berthold and Gleixner, 2009]

- idea: fix variables such as to obtain a linear subproblem
- generic: solve an auxiliary set covering problem to automatically detect a minimal set of variables to fix
- use LP or NLP relaxation for fixing values
Primal solutions from: feasible LP solution, MIP heuristics, NLP local search, extended LNS heuristics

New heuristic: Undercover [Berthold and Gleixner, 2009]

- idea: fix variables such as to obtain a linear subproblem
- generic: solve an auxiliary set covering problem to automatically detect a minimal set of variables to fix
- use LP or NLP relaxation for fixing values

Here: Fixing only $r_1, \ldots, r_T$ linearises the mixing constraints

$$f_{i,t}^O = r_t(f_{i,t}^O + f_{i,t}^R)$$

- use fixing values from NLP relaxation solved to local optimality
- solve resulting sub-MIP with node and stall node limit
Primal solutions from: feasible LP solution, MIP heuristics, NLP local search, extended LNS heuristics

New heuristic: Undercover [Berthold and Gleixner, 2009]

▷ idea: fix variables such as to obtain a linear subproblem
▷ generic: solve an auxiliary set covering problem to automatically detect a minimal set of variables to fix
▷ use LP or NLP relaxation for fixing values

Here: Fixing only $r_1, \ldots, r_T$ linearises the mixing constraints

\[ f_{i,t}^O = r_t (f_{i,t}^O + f_{i,t}^R) \]

▷ use fixing values from NLP relaxation solved to local optimality
▷ solve resulting sub-MIP with node and stall node limit

Expensive, but effective heuristic: produces fully feasible solutions within 1% of the global optimum.
Compare the performance of

- BARON 9.0.2
- Couenne 0.2
- SCIP 2.0

...to the CPLEX-based implementation of Bley et al. [2009a],
Problem-specific vs. generic algorithms

Compare the performance of

- BARON 9.0.2
- Couenne 0.2
- SCIP 2.0

...to the CPLEX-based implementation of Bley et al. [2009a], on 2 realistic mine production scheduling instances from industry:

<table>
<thead>
<tr>
<th>Model</th>
<th>Marvin no. variables</th>
<th>Marvin no. constraints</th>
<th>Dent no. variables</th>
<th>Dent no. constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bin cont</td>
<td>linear quad</td>
<td>bin cont</td>
<td>linear quad</td>
</tr>
<tr>
<td>aggreg. model</td>
<td>1445 4403</td>
<td>7582 16</td>
<td>3125 9475</td>
<td>15750 24</td>
</tr>
<tr>
<td>ext. model</td>
<td>1445 7242</td>
<td>9044 1360</td>
<td>3125 15650</td>
<td>18900 3000</td>
</tr>
</tbody>
</table>
benchmarks algo. closes gap up to 0.02% (Marvin) and 0.33% (Dent) after 10000 secs

SCIP (with LP solver CPLEX) reaches 1.80% resp. 0.71%

difference is also due to

- underlying MIP solver (CPLEX vs. SCIP)
- problem-specific MIP cuts/variable fixings

i.e. not only/mainly due to the handling of the nonlinearities.
1 Convex MINLP
   - Algorithms
   - Extensions
   - Solvers

2 Nonconvex MINLP
   - Spatial Branch-and-Bound Algorithm
   - Solvers
   - Application: Mine Production Scheduling
   - Techniques

LP/MIP based techniques for solving MINLPs
Tighten variable bounds $[\underline{x}, \overline{x}]$ such that

- the optimal value of the problem is not changed, or
- the set of optimal solutions is not changed, or
- the set of feasible solutions is not changed
Bounds Tightening (Domain Propagation)

Tighten variable bounds $[\underline{x}, \overline{x}]$ such that
- the optimal value of the problem is not changed, or
- the set of optimal solutions is not changed, or
- the set of feasible solutions is not changed

Bound tightening allows to
- find infeasible subproblems (when bound tightening gives inconsistent bounds)
Bounds Tightening (Domain Propagation)

Tighten variable bounds $[\underline{x}, \bar{x}]$ such that

- the optimal value of the problem is not changed, or
- the set of optimal solutions is not changed, or
- the set of feasible solutions is not changed

Bound tightening allows to

- find infeasible subproblems (when bound tightening gives inconsistent bounds)
- reduce possible integral values of discrete variables $\Rightarrow$ less branching
Bounds Tightening (Domain Propagation)

Tighten variable bounds \([x, \overline{x}]\) such that

- the optimal value of the problem is not changed, or
- the set of optimal solutions is not changed, or
- the set of feasible solutions is not changed

Bound tightening allows to

- find infeasible subproblems (when bound tightening gives inconsistent bounds)
- reduce possible integral values of discrete variables \(\Rightarrow\) less branching
- very important: improve underestimators without branching \(\Rightarrow\) tighter lower bounds
Bounds Tightening (Domain Propagation)

Tighten variable bounds \([x, \bar{x}]\) such that:

- the optimal value of the problem is not changed, or
- the set of optimal solutions is not changed, or
- the set of feasible solutions is not changed

Bound tightening allows to:

- find infeasible subproblems (when bound tightening gives inconsistent bounds)
- reduce possible integral values of discrete variables \(\Rightarrow\) less branching
- very important: improve underestimators without branching \(\Rightarrow\) tighter lower bounds

Some techniques [Belotti et al., 2009]:

- **FBBT** Feasibility-Based Bounds Tightening (cheap)
- **ABT** Aggressive Feasibility-Based Bounds Tightening (expensive)
- **OBBT** Optimality-Based Bounds Tightening (expensive)
Feasibility-Based Bound Tightening for a linear constraint:

\[
\begin{align*}
    b & \leq \sum_{i: a_i > 0} a_i x_i + \sum_{i: a_i < 0} a_i x_i \leq \overline{b},
\end{align*}
\]
Feasibility-Based Bound Tightening for a linear constraint:

\[ b \leq \sum_{i:a_i>0} a_i x_i + \sum_{i:a_i<0} a_i x_i \leq \bar{b}, \]

\[ \Rightarrow \quad x_j \leq \frac{1}{a_j} \begin{cases} \bar{b} - \sum_{i:a_i<0} a_i \bar{x}_i - \sum_{i:a_i>0} a_i \bar{x}_i, & \text{if } a_j > 0 \\ b - \sum_{i:a_i<0,i\neq j} a_i \bar{x}_i - \sum_{i:a_i>0} a_i \bar{x}_i, & \text{if } a_j < 0 \end{cases} \]

\[ x_j \geq \frac{1}{a_j} \begin{cases} \bar{b} - \sum_{i:a_i<0} a_i \bar{x}_i - \sum_{i:a_i>0} a_i \bar{x}_i, & \text{if } a_j > 0 \\ b - \sum_{i:a_i<0,i\neq j} a_i \bar{x}_i - \sum_{i:a_i>0} a_i \bar{x}_i, & \text{if } a_j < 0 \end{cases} \]
FBBT for Nonlinear Constraints

[Schichl and Neumaier, 2005, Vu et al., 2008]

Represent **algebraic structure** of problem in **one** directed acyclic graph:

- nodes: variables, operations, constraints
- arcs: flow of computation
Represent **algebraic structure** of problem in one directed acyclic graph:

- **nodes**: variables, operations, constraints
- **arcs**: flow of computation

**Example:**

\[
\begin{align*}
\sqrt{x} + 2\sqrt{xy} + 2\sqrt{y} &\in [-\infty, 7] \\
x^2\sqrt{y} - 2xy + 3\sqrt{y} &\in [0, 2] \\
x, y &\in [1, 16]
\end{align*}
\]
FBBT for Nonlinear Constraints

[Schichl and Neumaier, 2005, Vu et al., 2008]

Represent algebraic structure of problem in one directed acyclic graph:

- nodes: variables, operations, constraints
- arcs: flow of computation

Example:

\[
\sqrt{x} + 2\sqrt{xy} + 2\sqrt{y} \in [-\infty, 7] \\
x^2 \sqrt{y} - 2xy + 3\sqrt{y} \in [0, 2] \\
x, y \in [1, 16]
\]

Forward propagation:

- compute bounds on intermediate nodes (top-down)
FBBT for Nonlinear Constraints

[Schichl and Neumaier, 2005, Vu et al., 2008]

Represent algebraic structure of problem in one directed acyclic graph:

- nodes: variables, operations, constraints
- arcs: flow of computation

**Example:**

\[
\sqrt{x} + 2\sqrt{xy} + 2\sqrt{y} \in [-\infty, 7] \\
x^2 \sqrt{y} - 2xy + 3\sqrt{y} \in [0, 2] \\
x, y \in [1, 16]
\]

Forward propagation:

- compute bounds on intermediate nodes (top-down)

Backward propagation:

- reduce bounds using reverse operations (bottom-up)
Aggressive Bound Tightening

Probing for MIP

- tentatively fix binary variable $x_i$ and apply domain propagation
Probing for MIP

- tentatively fix binary variable $x_i$ and apply domain propagation
- if infeasibility is detected for $x_i = a$, then fix binary variable to $1 - a$ in main problem
Aggressive Bound Tightening

Probing for MIP

- tentatively fix binary variable $x_i$ and apply domain propagation
- if infeasibility is detected for $x_i = a$, then fix binary variable to $1 - a$ in main problem
- otherwise, use unification of bound tightenings for $x_i = 0$ and $x_i = 1$, deduce implications (used later for cut generation)
Probing for MIP

- tentatively fix binary variable $x_i$ and apply domain propagation
- if infeasibility is detected for $x_i = a$, then fix binary variable to $1 - a$ in main problem
- otherwise, use unification of bound tightenings for $x_i = 0$ and $x_i = 1$, deduce implications (used later for cut generation)
- within strong branching: solve LP relaxation of tentative fixing $x_i = a$ and fix to $1 - a$ if lower bound exceeds upper bound
## Aggressive Bound Tightening

### Probing for MIP

- tentatively fix binary variable $x_i$ and apply domain propagation
- if infeasibility is detected for $x_i = a$, then fix binary variable to $1 - a$ in main problem
- otherwise, use unification of bound tightenings for $x_i = 0$ and $x_i = 1$, deduce implications (used later for cut generation)
- within strong branching: solve LP relaxation of tentative fixing $x_i = a$ and fix to $1 - a$ if lower bound exceeds upper bound

### Probing for MINLP

[Tawarmalani and Sahinidis, 2004, Belotti et al., 2009]

- tentatively reduce bounds of a nonlinear variable, e.g., $x_i \leq 0.3\bar{x}_i$, and apply FBBT
Aggressive Bound Tightening

Probing for MIP

- tentatively fix binary variable $x_i$ and apply domain propagation
- if infeasibility is detected for $x_i = a$, then fix binary variable to $1 - a$ in main problem
- otherwise, use unification of bound tightenings for $x_i = 0$ and $x_i = 1$, deduce implications (used later for cut generation)
- within strong branching: solve LP relaxation of tentative fixing $x_i = a$ and fix to $1 - a$ if lower bound exceeds upper bound

Probing for MINLP [Tawarmalani and Sahinidis, 2004, Belotti et al., 2009]

- tentatively reduce bounds of a nonlinear variable, e.g., $x_i \leq 0.3 \bar{x}_i$, and apply FBBT
- if infeasibility is detected, tighten bound in main problem, e.g., $x_i := 0.3 \bar{x}_i$
Probing for MIP

- tentatively fix binary variable \( x_i \) and apply domain propagation
- if infeasibility is detected for \( x_i = a \), then fix binary variable to \( 1 - a \) in main problem
- otherwise, use unification of bound tightenings for \( x_i = 0 \) and \( x_i = 1 \), deduce implications (used later for cut generation)
- within strong branching: solve LP relaxation of tentative fixing \( x_i = a \) and fix to \( 1 - a \) if lower bound exceeds upper bound

Probing for MINLP [Tawarmalani and Sahinidis, 2004, Belotti et al., 2009]

- tentatively reduce bounds of a nonlinear variable, e.g., \( x_i \leq 0.3 \bar{x}_i \), and apply FBBT
- if infeasibility is detected, tighten bound in main problem, e.g., \( \underline{x}_i := 0.3 \bar{x}_i \)
- within B&B: update and solve LP relaxation to obtain lower bound for tentative reduction
Consider a linear relaxation with constraints $Ax \leq b$.

Let $x^*$ be the best solution found so far.

\[
x_i := \min x_i \\
\text{s.t. } Ax \leq b \\
c^T x \leq c^T x^*
\]

\[
\bar{x}_i := \max x_i \\
\text{s.t. } Ax \leq b \\
c^T x \leq c^T x^*
\]
Effect of disabling FBBT in SCIP on 89 MIQQPs [Berthold et al., 2010]

- number of instances solved: $-1$
- mean running time: $+6%$
- time to first / optimal solution: $+13% / +7%$
- number of nodes: $+39%$
Effect of disabling FBBT in SCIP on 89 MIQQPs  [Berthold et al., 2010]

- number of instances solved: $-1$
- mean running time: $+6\%$
- time to first / optimal solution: $+13\% / +7\%$
- number of nodes: $+39\%$

Effect of bound tight. in Couenne on 21 (MI)NLPs  [Belotti et al., 2009]

<table>
<thead>
<tr>
<th>Method</th>
<th>#solved</th>
<th>#best time</th>
<th>#best nodes</th>
<th>#best depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBBT + ABT + OBBT*</td>
<td>19</td>
<td>12</td>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td>FBBT only</td>
<td>16</td>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>FBBT + ABT**</td>
<td>14</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>no bound tightening</td>
<td>16</td>
<td>10</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>FBBT + OBBT***</td>
<td>16</td>
<td>2</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

* “tuned” = apply ABT and OBBT at root or with prob. $2^{1-\text{depth}}$ and $2^{-\text{depth}}$, resp.
** apply ABT at root or with prob. $2^{3-\text{depth}}$, if depth > 3
*** apply OBBT at root or with prob. $2^{3-\text{depth}}$, if depth > 3
Branching in MINLP:

- on an integer variable to resolve fractionality
- on a nonlinear variable in a nonconvex term to allow tighter relaxation

How to select the branching variable from a set of variable candidates?

- give priority to integer variables
- apply MIP branching techniques (most fractional, reliability branching, strong branching, ...)
- extend these branching rules to continuous variables

[Belotti et al., 2009]
Branching in MINLP:

- on an integer variable to resolve fractionality
- on a nonlinear variable in a nonconvex term to allow tighter relaxation

How to select the branching variable from a set of variable candidates?

- give priority to integer variables
  apply MIP branching techniques (most fractional, reliability branching, strong branching, ...)
- extend these branching rules to continuous variables [Belotti et al., 2009]
\textbf{“Most fractional” rule for integer variables}

Branch on $x_i^*$, such that

$$i^* = \arg\max_{i \in I} \min\{\hat{x}_i - \lfloor \hat{x}_i \rfloor, \lceil \hat{x}_i \rceil - \hat{x}_i\},$$

where $\hat{x}$ is the current solution of the LP relaxation.
Branching Rule: Most violated

“Most fractional” rule for integer variables

Branch on $x_{i^*}$, such that

$$i^* = \arg \max_{i \in I} \min \{ \hat{x}_i - \lfloor \hat{x}_i \rfloor, \lceil \hat{x}_i \rceil - \hat{x}_i \},$$

where $\hat{x}$ is the current solution of the LP relaxation.

“Most violated” rule for nonlinear variables

Assign to each variable $x_i$ the set of “convexification gaps” $U_{i,j}(\hat{x})$ in $\hat{x}$ for all nonconvex terms $j$ in which $x_i$ is involved.

Branch on $x_{i^*}$, such that

$$i^* = \arg \max_{i \in I} \frac{1}{10} \sum_j U_{i,j}(\hat{x}) + \frac{13}{10} \max_j U_{i,j}(\hat{x}) + \frac{8}{10} \min_j U_{i,j}(\hat{x})$$
Branching Rule: Violation transfer

[Tawarmalani and Sahinidis, 2004, Belotti et al., 2009]

Given \((\hat{x}, \hat{\pi})\) primal-dual solution of LP relaxation \(Ax \leq b\), \(A = (a_{ij})_{i,j}\).

For \(x_i\), let \([\underline{x}_i, \overline{x}_i] \subseteq [x_i, \overline{x}_i]\), s.t.

\(\hat{x}_j \in [x_i, \overline{x}_i]\), and

\(\text{for every } x_j, x_j = f(x_i) \text{ (for some } f), \exists v \in [\underline{x}_i, \overline{x}_i]: f(v) = \hat{x}_j, \text{ and}\)

\(\text{as small as possible}\)

\([\underline{x}_i, \overline{x}_i] = [0.25, 0.7]\)
Branching Rule: Violation transfer

[Tawarmalani and Sahinidis, 2004, Belotti et al., 2009]

Given \((\hat{x}, \hat{\pi})\) primal-dual solution of LP relaxation \(Ax \leq b, A = (a_{ij})_{i,j}\).

For \(x_i\), let \([\underline{x}_i, \overline{x}_i] \subseteq [x_i, \overline{x}_i]\), s.t.

1. \(\hat{x}_j \in [x_i, \overline{x}_i]\), and
2. for every \(x_j\), \(x_j = f(x_i)\) (for some \(f\)), \(\exists v \in [\underline{x}_i, \overline{x}_i] : f(v) = \hat{x}_j\), and
3. as small as possible

\([\underline{x}_i, \overline{x}_i] = [0.25, 0.7]\)

Consider segment \(X_i := \{x : x_j = \hat{x}_j, j \neq i, x_i \in [\underline{x}_i, \overline{x}_i]\}\).

Estimate change in lower bound by range of Lagrange function of LP relaxation w.r.t. \(X_i\):

\[
\max_{x \in X^i_R} \mathcal{L}(x) - \min_{x \in X^i_R} \mathcal{L}(x) = |\overline{x}_i - \underline{x}_i| | \sum_j \hat{\pi}_j a_{ji} |.
\]
Branching Rule: Violation transfer

[Tawarmalani and Sahinidis, 2004, Belotti et al., 2009]

Given \((\hat{x}, \hat{\pi})\) primal-dual solution of LP relaxation \(Ax \leq b, A = (a_{ji})_{i,j}\).

For \(x_i\), let \([\underline{x}_i, \overline{x}_i] \subseteq [x_i, \overline{x}_i]\), s.t.

\(\hat{x}_j \in [x_i, \overline{x}_i]\), and

\(\exists v \in [\underline{x}_i, \overline{x}_i] : f(v) = \hat{x}_j\), and

as small as possible

\([\underline{x}_i, \overline{x}_i] = [0.25, 0.7]\)

Consider segment \(X_i := \{x : x_j = \hat{x}_j, j \neq i, x_i \in [\underline{x}_i, \overline{x}_i]\}\).

Estimate change in lower bound by range of Lagrange function of LP relaxation w.r.t. \(X_i\):

\[
\max_{x \in X^i_R} L(x) - \min_{x \in X^i_R} L(x) = |\overline{x}_i - \underline{x}_i| \sum_j \hat{\pi}_j a_{ji}.
\]

Branch on \(x_i^*\), such that

\(i^* = \arg \max_{i \in I} |\overline{x}_i - \underline{x}_i| \sum_j \hat{\pi}_j a_{ji}|.\)
### Branching Rule: Pseudo Costs

<table>
<thead>
<tr>
<th>Integer variables $x_i$, $i \in I$</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ when branching, memorize resulting change in lower bound</td>
</tr>
<tr>
<td>▶ average bound improvements over all down- and up-branches of $x_i$ so far</td>
</tr>
<tr>
<td>⇒ estimates $\psi_i^-$, $\psi_i^+$ for lower bound improvement per unit of change in $x_i$</td>
</tr>
</tbody>
</table>
Integer variables $x_i, i \in I$

- when branching, memorize resulting change in lower bound
- average bound improvements over all down- and up-branches of $x_i$ so far

⇒ estimates $\psi_i^-, \psi_i^+$ for lower bound improvement per unit of change in $x_i$

- branch on $x_i$ that has fractional value in LP solution $\hat{x}$ and maximizes

$$\alpha \psi_i^- (\hat{x}_i - \lfloor \hat{x}_i \rfloor) + (1 - \alpha) \psi_i^+ (\lceil \hat{x}_i \rceil - \hat{x}_i) \quad \text{or} \quad \psi_i^- (\hat{x}_i - \lfloor \hat{x}_i \rfloor) \cdot \psi_i^+ (\lceil \hat{x}_i \rceil - \hat{x}_i)$$
## Branching Rule: Pseudo Costs

### Integer variables $x_i$, $i \in I$

- when branching, memorize resulting change in lower bound
- average bound improvements over all down- and up-branches of $x_i$ so far
  ⇒ estimates $\psi_i^-$, $\psi_i^+$ for lower bound improvement per unit of change in $x_i$
- branch on $x_i$ that has fractional value in LP solution $\hat{x}$ and maximizes
  \[
  \alpha \psi_i^- (\hat{x}_i - \lfloor \hat{x}_i \rfloor) + (1 - \alpha) \psi_i^+ ([\hat{x}_i] - \hat{x}_i) \quad \text{or} \quad \psi_i^- (\hat{x}_i - \lfloor \hat{x}_i \rfloor) \cdot \psi_i^+ ([\hat{x}_i] - \hat{x}_i)
  \]

### Continuous variables in MINLP

- branching does not result in immediate change of variable’s value in LP
  ⇒ cannot estimate bound changes as $\psi_i^- (\hat{x}_i - \lfloor \hat{x}_i \rfloor)$, $\psi_i^+ ([\hat{x}_i] - \hat{x}_i)$
Branching Rule: Pseudo Costs

**Integer variables** \( x_i, \ i \in I \)

- when branching, memorize resulting change in lower bound
- average bound improvements over all down- and up-branches of \( x_i \) so far

⇒ estimates \( \psi_i^-, \ \psi_i^+ \) for lower bound improvement per unit of change in \( x_i \)

- branch on \( x_i \) that has fractional value in LP solution \( \hat{x} \) and maximizes

\[
\alpha \psi_i^- (\hat{x}_i - [\hat{x}_i]) + (1 - \alpha) \psi_i^+ ([\hat{x}_i] - \hat{x}_i) \quad \text{or} \quad \psi_i^- (\hat{x}_i - [\hat{x}_i]) \cdot \psi_i^+ ([\hat{x}_i] - \hat{x}_i)
\]

**Continuous variables in MINLP**

- branching does not result in immediate change of variable’s value in LP

⇒ cannot estimate bound changes as \( \psi_i^- (\hat{x}_i - [\hat{x}_i]), \ \psi_i^+ ([\hat{x}_i] - \hat{x}_i) \)

- Belotti et al. [2009] proposed several alternatives:
  - multiply pseudocosts by variable infeasibility (analog to fractionality):
    \[
    \psi_i^{-,+} (\frac{1}{10} \sum_j U_{i,j}(\hat{x}) + \frac{13}{10} \max_j U_{i,j}(\hat{x}) + \frac{8}{10} \min_j U_{i,j}(\hat{x}))
    \]
  - multiply by domain width after branching:
    \[
    \psi_i^- (\hat{x}_i - \underline{x}_i), \ \psi_i^+ (\overline{x}_i - \hat{x}_i) \quad \text{or} \quad \psi_i^- (\overline{x}_i - \hat{x}_i), \ \psi_i^+ (\hat{x}_i - \underline{x}_i)
    \]
  - multiply by width of intervals from violation transfer:
    \[
    \psi_i^- (\hat{x}_i - \underline{x}_i), \ \psi_i^+ (\overline{x}_i - \hat{x}_i)
    \]
Strong Branching

- at each node, compute lower bound for all (or many) possible branchings (i.e., construct two branches, update and solve relaxation) and choose the one with best bound improvement
- expensive, but best reduction in number of nodes
Branching Rule: Reliability Branching

**Strong Branching**
- at each node, compute lower bound for all (or many) possible branchings (i.e., construct two branches, update and solve relaxation) and choose the one with best bound improvement
- **expensive**, but best reduction in number of nodes

**Reliability branching**
- compute exact bound improvement **only for variables with a low number of branchings so far**
- otherwise, assume **pseudo costs are reliable** and use them to evaluate potential bound improvement
  - initialization of pseudo costs by strong branching
Effect of branching techniques in **Couenne** on 33 (MI)NLPs [Belotti et al., 2009]:

<table>
<thead>
<tr>
<th>Method</th>
<th>#solved</th>
<th>#best time</th>
<th>#best nodes</th>
<th>#largest depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable infeasibility</td>
<td>23</td>
<td>14</td>
<td>7</td>
<td>15</td>
</tr>
<tr>
<td>violation transfer</td>
<td>26</td>
<td>13</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>rel.br.-var.infeas.*</td>
<td>23</td>
<td>11</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>rel.br.-dom.width**</td>
<td>25</td>
<td>9</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>strong branching</td>
<td>25</td>
<td>1</td>
<td>17</td>
<td>11</td>
</tr>
</tbody>
</table>

* “rel.br.-infeas.” = reliability branching with variable infeasibilities as pseudo costs multipliers

** “rel.br.-width” = reliability branching with domain width after branching as pseudo costs multipliers: $\psi_i^-(\bar{x}_i - \hat{x}_i), \psi_i^+(\hat{x}_i - \bar{x}_i)$
Quadratic constraints of the form

\[
\sum_{k=1}^{N} \alpha_k x_k^2 - \alpha_{N+1} x_{N+1}^2 \leq 0
\]

with \(\alpha_1, \ldots, \alpha_{N+1} \geq 0, L_{N+1} \geq 0\) describe a convex feasible region:

\[
\sqrt{\sum_{k=1}^{N} \alpha_k x_k^2} \leq \sqrt{\alpha_{N+1} x_{N+1}}
\]

Example: \(x^2 + y^2 - z^2 \leq 0\) in \([-1, 1] \times [-1, 1] \times [0, 1]\)

feasible region

“ice cream cone”
Reformulations for MIQCPs: SOC

Quadratic constraints of the form

\[ \sum_{k=1}^{N} \alpha_k x_k^2 - \alpha_{N+1} x_{N+1}^2 \leq 0 \]

with \( \alpha_1, \ldots, \alpha_{N+1} \geq 0, L_{N+1} \geq 0 \) describe a convex feasible region:

\[ \sqrt{\sum_{k=1}^{N} \alpha_k x_k^2} \leq \sqrt{\alpha_{N+1} x_{N+1}} \]

Example: \( x^2 + y^2 - z^2 \leq 0 \) in \([-1, 1] \times [-1, 1] \times [0, 1]\)

not recognizing SOC: using termwise underestimator:

\[ \begin{cases} 
  x^2 + y^2 + w \leq 1 \\
  \frac{z^2 - \bar{z}^2}{z - \bar{z}} (z - \bar{z}) - z^2 \leq w 
\end{cases} \]
Reformulations for MIQCPs: SOC

Quadratic constraints of the form

\[ \sum_{k=1}^{N} \alpha_k x_k^2 - \alpha_{N+1} x_{N+1}^2 \leq 0 \]

with \( \alpha_1, \ldots, \alpha_{N+1} \geq 0, L_{N+1} \geq 0 \) describe a convex feasible region:

\[
\sqrt{\sum_{k=1}^{N} \alpha_k x_k^2} \leq \sqrt{\alpha_{N+1} x_{N+1}}
\]

Example: \( x^2 + y^2 - z^2 \leq 0 \) in \([-1, 1] \times [-1, 1] \times [0, 1]\)

not recognizing SOC:
using termwise underestimator:

\[
\begin{cases} 
    x^2 + y^2 + w \leq 1 \\
    \frac{z^2 - \bar{z}^2}{\bar{z} - z} (z - \bar{z}) - z^2 \leq w 
\end{cases}
\]

after branching on \( z = 0.5 \)
Quadratic constraints of the form

\[ \sum_{k=1}^{N} \alpha_k x_k^2 - \alpha_{N+1} x_{N+1}^2 \leq 0 \]

with \( \alpha_1, \ldots, \alpha_{N+1} \geq 0 \), \( L_{N+1} \geq 0 \) describe a convex feasible region:

\[ \sqrt{\sum_{k=1}^{N} \alpha_k x_k^2} \leq \sqrt{\alpha_{N+1} x_{N+1}} \]

Example: \( x^2 + y^2 - z^2 \leq 0 \) in \([-1, 1] \times [-1, 1] \times [0, 1] \)

recognizing SOC:

- initial relaxation
- no spatial branching necessary
More general [Mahajan and Munson, 2010]:

- Given a quadratic constraint \( x^T A x + c^T x + d \leq 0 \).
More general [Mahajan and Munson, 2010]:

- Given a quadratic constraint $x^T Ax + c^T x + d \leq 0$.
- Let $A = QDQ^T$ with $Q$ orthogonal and $D$ diagonal matrix.
- Let $D = RER$, $b = R^{-1} Q^T c$ with $E_{ii} = \text{sign}(D_{ii})$, $R_{ii} = \begin{cases} \sqrt{|D_{ii}|}, & \text{if } D_{ii} \neq 0, \\ 1, & \text{if } D_{ii} = 0. \end{cases}$
- Then constraint is equivalent to

\[
    y^T Ey + b^T y = d \leq 0
\]

\[
    RQ^T x = y
\]
Reformulations for MIQCPs: SOC

More general [Mahajan and Munson, 2010]:

- Given a quadratic constraint $x^T Ax + c^T x + d \leq 0$.
- Let $A = QDQ^T$ with $Q$ orthogonal and $D$ diagonal matrix.
- Let $D = RER$, $b = R^{-1} Q^T c$ with $E_{ii} = \text{sign}(D_{ii})$, $R_{ii} = \begin{cases} \sqrt{|D_{ii}|}, & \text{if } D_{ii} \neq 0, \\ 1, & \text{if } D_{ii} = 0. \end{cases}$
- Then constraint is equivalent to

$$\sum_{i : E_{ii} > 0} (y_i^2 + b_i y_i) - \sum_{i : E_{ii} < 0} (y_i^2 - b_i y_i) + \sum_{i : E_{ii} = 0} (b_i y_i) + d \leq 0$$

$$RQ^T x = y$$
More general [Mahajan and Munson, 2010]:

- Given a quadratic constraint $x^T A x + c^T x + d \leq 0$.
- Let $A = Q D Q^T$ with $Q$ orthogonal and $D$ diagonal matrix.
- Let $D = R E R$, $b = R^{-1} Q^T c$ with $E_{ii} = \text{sign}(D_{ii})$, $R_{ii} = \begin{cases} \sqrt{|D_{ii}|}, & \text{if } D_{ii} \neq 0, \\ 1, & \text{if } D_{ii} = 0. \end{cases}$
- Then constraint is equivalent to

$$
\sum_{i : E_{ii} > 0} \left( y_i + \frac{b_i}{2} \right)^2 - \sum_{i : E_{ii} < 0} \left( y_i - \frac{b_i}{2} \right)^2 + \sum_{i : E_{ii} = 0} (b_i y_i) + d - \frac{1}{4} \sum_i \text{sign}(D_{ii}) b_i^2 \leq 0
$$

$$RQ^T x = y$$
Reformulations for MIQCPs: SOC

More general [Mahajan and Munson, 2010]:

- Given a quadratic constraint \( x^T Ax + c^T x + d \leq 0 \).
- Let \( A = QDQ^T \) with \( Q \) orthogonal and \( D \) diagonal matrix.
- Let \( D = RER, b = R^{-1} Q^T c \) with \( E_{ii} = \text{sign}(D_{ii}), R_{ii} = \begin{cases} \sqrt{|D_{ii}|}, & \text{if } D_{ii} \neq 0, \\ 1, & \text{if } D_{ii} = 0. \end{cases} \)
- Then constraint is equivalent to

\[
\sum_{i:E_{ii} > 0} \left( y_i + \frac{b_i}{2} \right)^2 - \sum_{i:E_{ii} < 0} \left( y_i - \frac{b_i}{2} \right)^2 + \sum_{i:E_{ii} = 0} (b_i y_i) + d - \frac{1}{4} \sum_i \text{sign}(D_{ii}) b_i^2 \leq 0
\]

\( RQ^T x = y \)

- If \( |\{ i : E_{ii} < 0 \}| = 0 \), then quadratic function is convex.
More general [Mahajan and Munson, 2010]:

- Given a quadratic constraint $x^T Ax + c^T x + d \leq 0$.
- Let $A = QDQ^T$ with $Q$ orthogonal and $D$ diagonal matrix.
- Let $D = RER$, $b = R^{-1} Q^T c$ with $E_{ii} = \text{sign}(D_{ii})$, $R_{ii} = \{ \sqrt{|D_{ii}|}, \text{ if } D_{ii} \neq 0, 1, \text{ if } D_{ii} = 0 \}$.
- Then constraint is equivalent to
  \[
  \sum_{i : E_{ii} > 0} \left( y_i + \frac{b_i}{2} \right)^2 - \sum_{i : E_{ii} < 0} \left( y_i - \frac{b_i}{2} \right)^2 + \sum_{i : E_{ii} = 0} (b_i y_i) + d - \frac{1}{4} \sum_i \text{sign}(D_{ii}) b_i^2 \leq 0
  \]

- If $|\{ i : E_{ii} < 0 \}| = 0$, then quadratic function is convex.
- If $|\{ i : E_{ii} < 0 \}| = 1$ (w.l.o.g. $E_{11} < 0$) and $h := d - \frac{1}{4} \sum_i \text{sign}(D_{ii}) b_i^2 \geq 0$, then constraint is equivalent to
  \[\left\| \left( \begin{array}{c} E_{+}(y + \frac{b}{2}) \\ \sqrt{h} \end{array} \right) \right\|_2 \leq \left| y_1 - \frac{b_1}{2} \right|,\]
  \[
  \text{where } E \text{ is diagonal matrix with } E_{ii}^+ = \max(0, E_{ii}).
  \]
- Thus, constraint is equivalent to a union of two second order cones.
Assume \( x \in \{0, 1\}^n \) and consider the quadratic function

\[
x^T A x, \quad \text{with} \quad A \not\succeq 0.
\]

**Eigenvalue shifting:** For \( \alpha \leq \lambda_1(A) \) (minimal eigenvalue of \( A \)),

\[
(A - \alpha I) \succeq 0.
\]
Assume \( x \in \{0, 1\}^n \) and consider the quadratic function

\[
x^T A x, \quad \text{with} \quad A \not\succeq 0.
\]

**Eigenvalue shifting:** For \( \alpha \leq \lambda_1(A) \) (minimal eigenvalue of \( A \)),

\[
(A - \alpha I) \succeq 0.
\]

**Reformulate using convex quadratic function:** Since \( x_i^2 = x_i \),

\[
x^T A x = x^T (A - \alpha I) x + \alpha \sum_i x_i.
\]

▷ CPLEX uses this simple trick
Assume $x \in \{0, 1\}^n$ and consider the quadratic function

$$x^T Ax, \quad \text{with} \quad A \not\succeq 0.$$ 

**Eigenvalue shifting:** For $\alpha \leq \lambda_1(A)$ (minimal eigenvalue of $A$),

$$(A - \alpha I) \succeq 0.$$ 

**Reformulate using convex quadratic function:** Since $x_i^2 = x_i$,

$$x^T Ax = x^T (A - \alpha I)x + \alpha \sum_i x_i.$$ 

- CPLEX uses this simple trick
- it is not always the best strategy:

<table>
<thead>
<tr>
<th>instance</th>
<th>lower bound after 1 hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>qap</td>
<td>-1171771</td>
</tr>
<tr>
<td>pb302035</td>
<td>-13223300</td>
</tr>
<tr>
<td>pb351535</td>
<td>-9870063</td>
</tr>
</tbody>
</table>
Assume $x \in \{0, 1\}^n$ and consider the quadratic function

$$x^T Ax, \quad \text{with} \quad A \not\succeq 0.$$ 

**Eigenvalue shifting:** For $\alpha \leq \lambda_1(A)$ (minimal eigenvalue of $A$),

$$(A - \alpha I) \succeq 0.$$ 

**Reformulate using convex quadratic function:** Since $x_i^2 = x_i$,

$$x^T Ax = x^T (A - \alpha I) x + \alpha \sum_i x_i.$$ 

- CPLEX uses this simple trick
- it is not always the best strategy:

<table>
<thead>
<tr>
<th>instance</th>
<th>lower bound after 1 hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>qap</td>
<td>-1171771</td>
</tr>
<tr>
<td>pb302035</td>
<td>-13223300</td>
</tr>
<tr>
<td>pb351535</td>
<td>-9870063</td>
</tr>
</tbody>
</table>

**BUT:** the trivial lower bound is 0.0! ($A$ has only positive entries)
Assume $x \in \{0, 1\}^n$ and consider the quadratic function
\[ x^T Ax, \quad \text{with} \quad A \not\succeq 0. \]

**Eigenvalue shifting:** For $\alpha \leq \lambda_1(A)$ (minimal eigenvalue of $A$),
\[ (A - \alpha I) \succeq 0. \]

**Reformulate using convex quadratic function:** Since $x_i^2 = x_i$,
\[ x^T Ax = x^T (A - \alpha I)x + \alpha \sum_i x_i. \]

- CPLEX uses this simple trick
- it is not always the best strategy:

<table>
<thead>
<tr>
<th>instance</th>
<th>lower bound after 1 hour</th>
<th>SCIP lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>qap</td>
<td>-1171771</td>
<td>79071</td>
</tr>
<tr>
<td>pb302035</td>
<td>-13223300</td>
<td>1151477</td>
</tr>
<tr>
<td>pb351535</td>
<td>-9870063</td>
<td>1786035</td>
</tr>
</tbody>
</table>

**BUT:** the trivial lower bound is 0.0! ($A$ has only positive entries)
Assume \( x \in \{0, 1\}^n \) and consider the quadratic function
\[
x^T Ax, \quad \text{with} \quad A \not\succeq 0.
\]

**Eigenvalue shifting:** For \( \alpha \leq \lambda_1(A) \) (minimal eigenvalue of \( A \)),
\[
(A - \alpha I) \succeq 0.
\]

**Reformulate using convex quadratic function:** Since \( x_i^2 = x_i \),
\[
x^T Ax = x^T (A - \alpha I)x + \alpha \sum_i x_i.
\]

▷ CPLEX uses this simple trick
▷ it is not always the best strategy:

<table>
<thead>
<tr>
<th>instance</th>
<th>lower bound after 1 hour</th>
<th>SCIP lower bound</th>
<th>best upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>qap</td>
<td>-1171771</td>
<td>79071</td>
<td>388214</td>
</tr>
<tr>
<td>pb302035</td>
<td>-13223300</td>
<td>1151477</td>
<td>3379359</td>
</tr>
<tr>
<td>pb351535</td>
<td>-9870063</td>
<td>1786035</td>
<td>3599602</td>
</tr>
</tbody>
</table>

**BUT:** the trivial lower bound is 0.0! (\( A \) has only positive entries)
A quadratic term

\[ x \cdot \sum_{k=1}^{N} a_ky_k \quad \text{with} \quad x \in \{0, 1\} \]

can equivalently be replaced by

- an auxiliary variable \( w \)
- and the additional linear constraints

\[ M^L x \leq w \leq M^U x, \]
\[ \sum_{k=1}^{N} a_ky_k - M^U (1 - x) \leq w \leq \sum_{k=1}^{N} a_ky_k - M^L (1 - x), \]

where \( M^L \) and \( M^U \) are bounds on \( \sum_{k=1}^{N} a_ky_k \).

\( \Rightarrow \) linear reformulation
Reformulations for MIQCPs: Products with 0-1 Variables

A quadratic term

\[ x \cdot \sum_{k=1}^{N} a_k y_k \quad \text{with} \quad x \in \{0, 1\} \]

can equivalently be replaced by

▷ an auxiliary variable \( w \)

▷ and the additional linear constraints

\[ M^L x \leq w \leq M^U x, \]
\[ \sum_{k=1}^{N} a_k y_k - M^U (1 - x) \leq w \leq \sum_{k=1}^{N} a_k y_k - M^L (1 - x), \]

where \( M^L \) and \( M^U \) are bounds on \( \sum_{k=1}^{N} a_k y_k \).

\[ \Rightarrow \text{linear reformulation} \]

Products of \( n \) binary variables \( \Rightarrow \) Pseudo-Boolean optimization

▷ linearize dynamically [Berthold et al., 2009b]
Further topics

Tighter relaxations: E.g.,

- better handling of \textbf{quadratic} constraints: lift-and-project [Saxena et al., 2010], multiterm relaxations [Bao et al., 2009]
- convexification of \textbf{trilinear} \((x \cdot y \cdot z)\), \textbf{quadrilinear} \((w \cdot x \cdot y \cdot z)\), and \textbf{multilinear} forms: [Meyer and Floudas, 2004, Cafieri et al., 2010, Luedtke et al., 2010, Belotti et al., 2010]
- linear inequalities for orthogonal disjunctions and \textbf{bilinear covering sets} \(\sum a_i x_i y_i + b_i x_i + c_i y_i \geq r\): [Tawarmalani et al., 2008]
- convexification of \textbf{\(n-1\)-convex functions}: [Jach et al., 2008]
**Further topics**

**Tighter relaxations:** E.g.,

- better handling of **quadratic** constraints: lift-and-project [Saxena et al., 2010], multiterm relaxations [Bao et al., 2009]
- convexification of **trilinear** \((x \cdot y \cdot z)\), **quadrilinear** \((w \cdot x \cdot y \cdot z)\), and **multilinear** forms: [Meyer and Floudas, 2004, Cafieri et al., 2010, Luedtke et al., 2010, Belotti et al., 2010]
- linear inequalities for orthogonal disjunctions and **bilinear covering sets** \((\sum_i a_ix_iy_i + b_ix_i + c_iy_i \geq r)\): [Tawarmalani et al., 2008]
- convexification of \(n-1\)-convex functions: [Jach et al., 2008]

**Symmetry breaking:**

- use automorphisms in expression trees: [Liberti, 2008]
Further topics

**Tight relaxations**: E.g.,
- better handling of **quadratic** constraints: lift-and-project [Saxena et al., 2010], multiterm relaxations [Bao et al., 2009]
- convexification of **trilinear** $(x \cdot y \cdot z)$, **quadrilinear** $(w \cdot x \cdot y \cdot z)$, and **multilinear** forms: [Meyer and Floudas, 2004, Cafieri et al., 2010, Luedtke et al., 2010, Belotti et al., 2010]
- linear inequalities for orthogonal disjunctions and **bilinear covering sets** $(\sum_i a_ix_iy_i + b_ix_i + c_iy_i \geq r)$: [Tawarmalani et al., 2008]

**Convexity detection**:
- estimate spectrum of Hessian: [Nenov et al., 2004, Mönnigmann, 2008]
- walk expression tree and apply convexity rules for function compositions: not conclusive, fast [Fourer et al., 2009]
- apply quantifier elimination: conclusive, slower, only rational functions [Neun et al., 2010]
Further topics

Tighter relaxations: E.g.,
- better handling of quadratic constraints: lift-and-project [Saxena et al., 2010], multiterm relaxations [Bao et al., 2009]
- convexification of trilinear \((x \cdot y \cdot z)\), quadrilinear \((w \cdot x \cdot y \cdot z)\), and multilinear forms: [Meyer and Floudas, 2004, Cafieri et al., 2010, Luedtke et al., 2010, Belotti et al., 2010]
- linear inequalities for orthogonal disjunctions and bilinear covering sets \((\sum_i a_i x_i y_i + b_i x_i + c_i y_i \geq r)\): [Tawarmalani et al., 2008]
- convexification of \(n-1\)-convex functions: [Jach et al., 2008]

Symmetry breaking:
- use automorphisms in expression trees: [Liberti, 2008]

Convexity detection:
- estimate spectrum of Hessian: [Nenov et al., 2004, Mönnigmann, 2008]
- walk expression tree and apply convexity rules for function compositions: not conclusive, fast [Fourer et al., 2009]
- apply quantifier elimination: conclusive, slower, only rational functions [Neun et al., 2010]

Nonlinear Integer Programming: [Hemmecke et al., 2009]
LP/MIP based techniques for solving MINLPs

Stefan Vigerske
Humboldt-Universität zu Berlin

DFG Research Center MATHEON
Mathematics for key technologies

17th December 2010, Corporate Technology, Siemens AG


