ZIB Guest Lecture
Mixed-Integer Nonlinear Optimization

Sven Leyffer

Mathematics & Computer Science Division
Argonne National Laboratory

ZIB Guest Lecture
October 8, 2015

Collaborators



Ashutosh Mahajan, Jeff Linderoth, and Jim Luedtke

Outline

Mixed-Integer Nonlinear Optimization

Mixed-Integer Nonlinear Program (MINLP)

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) \leq 0 \\ & x \in X \\ & x_i \in \mathbb{Z} \text{ for all } i \in I \end{aligned}$$

- $f : \mathbb{R}^n \to \mathbb{R}$, $c : \mathbb{R}^n \to \mathbb{R}^m$ smooth (often convex) functions
- $X \in \mathbb{R}^n$ bounded, polyhedral set, e.g. $X = \{x : l \leq A^T x \leq u\}$
- $I \subset \{1, \ldots, n\}$ subset of integer variables
- $x_i \in \mathbb{Z}$ for all $i \in I$ ... combinatorial problem
- Combines challenges of handling nonlinearities with combinatorial explosion of integer variables
- More general constraints possible, e.g. $l \leq c(x) \leq u$ etc.

Complexity of MINLP

Mixed-Integer Nonlinear Program (MINLP)

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) \leq 0 \\ & x \in X \\ & x_i \in \mathbb{Z} \text{ for all } i \in I \end{aligned}$$

Complexity of MINLP

- MINLP is NP-hard: includes MILP, which are NP-hard [Kannan and Monma, 1978]
- Worse: MINLP are undecidable [Jeroslow, 1973]: quadratically constrained IP for which no computing device can compute the optimum for all problems in this class
  ... but we're OK if X is compact!

Convexity of Nonlinear Functions

MINLP techniques distinguish convex and nonconvex MINLPs.
For our purposes, we define convexity as ...

Definition

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex, iff $\forall x^{(0)}, x^{(1)} \in \mathbb{R}^n$ we have:

$$f(x^{(1)}) \geq f(x^{(0)}) + (x^{(1)} - x^{(0)})^T \nabla f^{(0)}$$

In a slight abuse of notation, we say that ...

Definition

MINLP is a *convex* if the problem functions $f(x)$ and $c(x)$ are convex functions. If either $f(x)$ or any $c_i(x)$ is a nonconvex function, then MINLP is *nonconvex*.

Convexity (cont.)

We also define the convex hull of a set S as ...

Definition

For a set S, the *convex hull of* S is $\text{conv}(S)$:

$$\left\{x | x = \lambda x^{(1)} + (1-\lambda)x^{(0)},\ \forall 0 \leq \lambda \leq 1,\ \forall x^{(0)}, x^{(1)} \in S\right\}.$$

- If $X = \{x \in \mathbb{Z}^p : l \leq x \leq u\}$ and $l \in \mathbb{Z}^p$, $u \in \mathbb{Z}^p$, then $\text{conv}(X) = [l, u]^p$
- Finding convex hull is hard, even for polyhedral X.
- Convex hull important for MILP ...

Theorem

MILP can be solved as LP over the convex hull of feasible set.

MILP $\neq$ MINLP

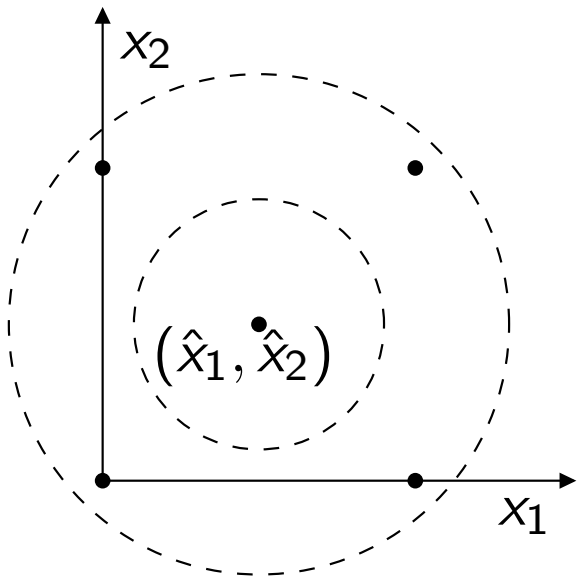
Important difference between MINLP and MILP

$$\underset{x}{\text{minimize}} \sum_{i=1}^{n} (x_i - \tfrac{1}{2})^2, \quad \text{subject to } x_i \in \{0, 1\}$$

... solution is not extreme point (lies in interior)

Remedy: Introduce objective η and a constraint $\eta \geq f(x)$

$$\begin{cases} \underset{\eta, x}{\text{minimize}} & \eta, \\ \text{subject to} & f(x) \leq \eta, \\ & c(x) \leq 0, \\ & x \in X, \\ & x_i \in \mathbb{Z}, \ \forall i \in I. \end{cases}$$



Assume wlog that MINLP objective is linear

MILP $\neq$ MINLP

Important difference between MINLP and MILP

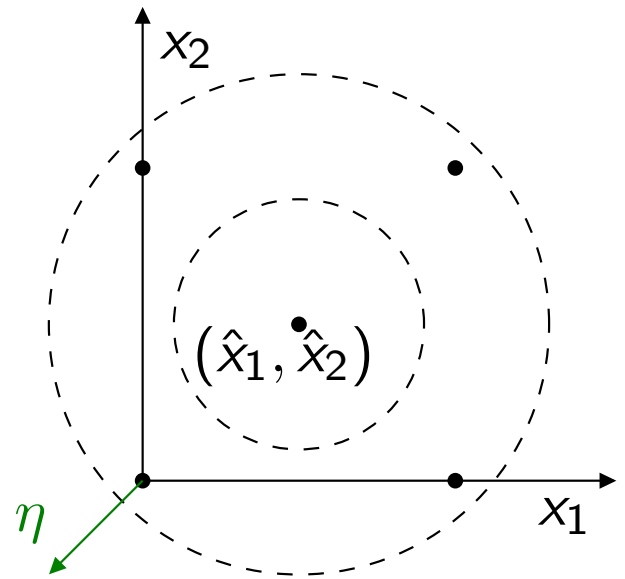
$$\underset{x}{\text{minimize}} \sum_{i=1}^{n} (x_i - \tfrac{1}{2})^2, \quad \text{subject to } x_i \in \{0, 1\}$$

... solution is not extreme point (lies in interior)

Remedy: Introduce objective η and a constraint $\eta \geq f(x)$

$$\begin{cases} \underset{\eta, x}{\text{minimize}} & \eta, \\ \text{subject to} & f(x) \leq \eta, \\ & c(x) \leq 0, \\ & x \in X, \\ & x_i \in \mathbb{Z}, \ \forall i \in I. \end{cases}$$

Assume wlog that MINLP objective is linear



Outline

MINLP Modeling Practices

Modeling plays a fundamental role in MILP see [Williams, 1999]
... even more important in MINLP

- MINLP combines integer and nonlinear formulations
- Reformulations of nonlinear relationships can be convex
- Interactions of nonlinear functions and binary variables
- Sometimes we can linearize expressions

MINLP Modeling Preference

We prefer linear over convex over nonconvex formulations.

Convexification of Binary Quadratic Programs

Consider pure binary quadratic function

$$q(x) = x^T Qx + g^T x \quad \text{where } x \in \{0,1\}^p$$

Let λ be smallest eigenvalue of Q

If $\lambda \geq 0$ then $q(x)$ is convex

Convexification of Binary Quadratics

Let $W := Q - \lambda I$ and $c := g + \lambda e$, where $e = (1, \ldots, 1)$, then $q(x) = x^T Wx + c^T x$ is convex.

Linearization of Constraints

Assume $x_2 \neq 0$. A simple transformation (a constant parameter):

$$\frac{x_1}{x_2} = a \Leftrightarrow x_1 = ax_2$$

Linearization of bilinear terms $x_1 x_2$ with:

- Binary variable $x_2 \in \{0, 1\}$
- Variable upper bound: $0 \leq x_1 \leq Ux_2$

... introduce new variable x_{12} to replace $x_1 x_2$ and add constraints

$$0 \leq x_{12} \leq x_2 U \quad \text{and} \quad -U(1 - x_2) \leq x_1 - x_{12} \leq U(1 - x_2),$$

Never Multiply a Nonlinear Function by a Binary

Previous example generalizes to nonlinear functions
Often binary variables "switch" constraints on/off

Warning

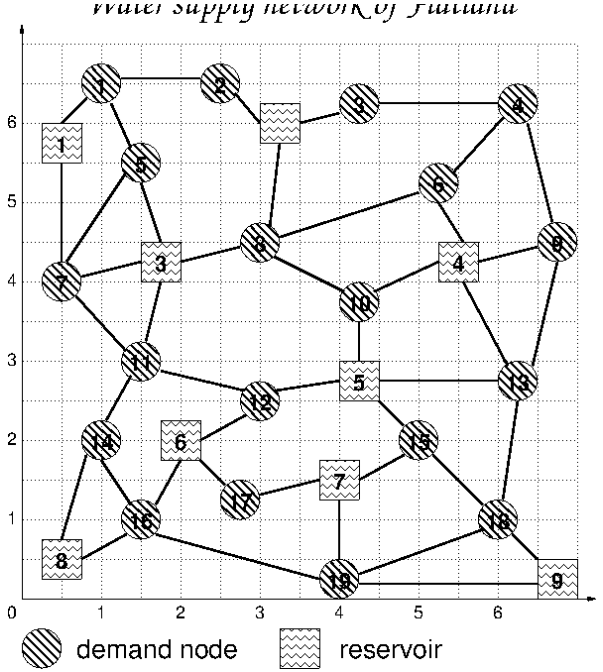
Never model on/off constraints by multiplying by a binary variable.

Three alternative approaches

- Disjunctive programming, see [Grossmann and Lee, 2003]
- Perspective formulations (not always), see [Günlük and Linderoth, 2012]
- Big-M formulation (weak relaxations)

Design of Water Distribution Networks

Goal: Design minimum cost network from discrete pipe diameters to meet water demand



Sets

- $\mathcal{N}$ nodes in network
- $\mathcal{S}$ source nodes
- $\mathcal{A}$: arcs in the network
- $\mathcal{K}$: Pipe diameters

Design of Water Distribution Networks

Variables

q_{ij}: flow pipe $(i,j) \in \mathcal{A}$

h_i: hydraulic head at node $i \in \mathcal{N}$

d_{ij}: diameter of pipe $(i,j) \in \mathcal{A}$, where $d_{ij} \in \{P_1, \ldots, P_r\}$

a_{ij}: area of cross section $(i,j) \in \mathcal{A}$

y_{ijk}: Binary variable $= 1$ if pipe of size $k \in \mathcal{K}$ chosen for $(i,j) \in \mathcal{A}$

Design of Water Distribution Networks

Variables

q_{ij}: flow pipe $(i,j) \in \mathcal{A}$

h_i: hydraulic head at node $i \in \mathcal{N}$

d_{ij}: diameter of pipe $(i,j) \in \mathcal{A}$, where $d_{ij} \in \{P_1, \ldots, P_r\}$

a_{ij}: area of cross section $(i,j) \in \mathcal{A}$

y_{ijk}: Binary variable $= 1$ if pipe of size $k \in \mathcal{K}$ chosen for $(i,j) \in \mathcal{A}$

Constraints

- Conservation of flow

$$\sum_{(i,j)\in\mathcal{A}} q_{ij} - \sum_{(j,i)\in\mathcal{A}} q_{ji} = D_i, \ \forall i \in \mathcal{N} \setminus \mathcal{S}.$$

- Flow bounds: (linear in a, but since $a_{ij} = 0.25\pi d_{ij}^2$, nonlinear in d)

$$-V_{\max} a_{ij} \leq q_{ij} \leq V_{\max} a_{ij}, \ \forall (i,j) \in \mathcal{A}.$$

Design of Water Distribution Networks

- Discrete pipe sizes $d_{ij} \in \{P_1, \ldots, P_r\}$
- Use "multiple choice" integrality to remove nonlinearity $a_{ij} = \pi d_{ij}^2/4$
- This is in general a good modeling practice

Design of Water Distribution Networks

- Discrete pipe sizes $d_{ij} \in \{P_1, \ldots, P_r\}$
- Use "multiple choice" integrality to remove nonlinearity $a_{ij} = \pi d_{ij}^2/4$
- This is in general a good modeling practice

1. Binary variables $y_{ijk} \in \{0, 1\}$ for $k \in \mathcal{K}$
2. Model discrete choice as

$$\sum_{k \in \mathcal{K}} y_{ijk} = 1, \quad \text{and} \quad \sum_{k \in \mathcal{K}} P_k y_{ijk} = d_{ij}. \; \forall (i,j) \in \mathcal{A}$$

3. Model $a_{ij} = \pi d_{ij}^2/4$ (nonlinear) as

$$a_{ij} = \sum_{k \in \mathcal{K}} \left(\frac{\pi}{4} P_k\right)^2 y_{ijk} \; \forall (i,j) \in \mathcal{A}$$

Design of Water Distribution Networks

- Nonsmooth, nonlinear pressure loss as a function of flow q_{ij} along arc $(i,j) \in \mathcal{A}$ and diameter d_{ij}:

$$h_i - h_j = \kappa_{ij} d_{ij}^{-c_1} \operatorname{sgn}(q_{ij}) |q_{ij}|^{c_2} \quad \forall (i,j) \in \mathcal{A}$$

Design of Water Distribution Networks

- Nonsmooth, nonlinear pressure loss as a function of flow q_{ij} along arc $(i,j) \in \mathcal{A}$ and diameter d_{ij}:

$$h_i - h_j = \kappa_{ij} d_{ij}^{-c_1} \operatorname{sgn}(q_{ij}) |q_{ij}|^{c_2} \quad \forall (i,j) \in \mathcal{A}$$

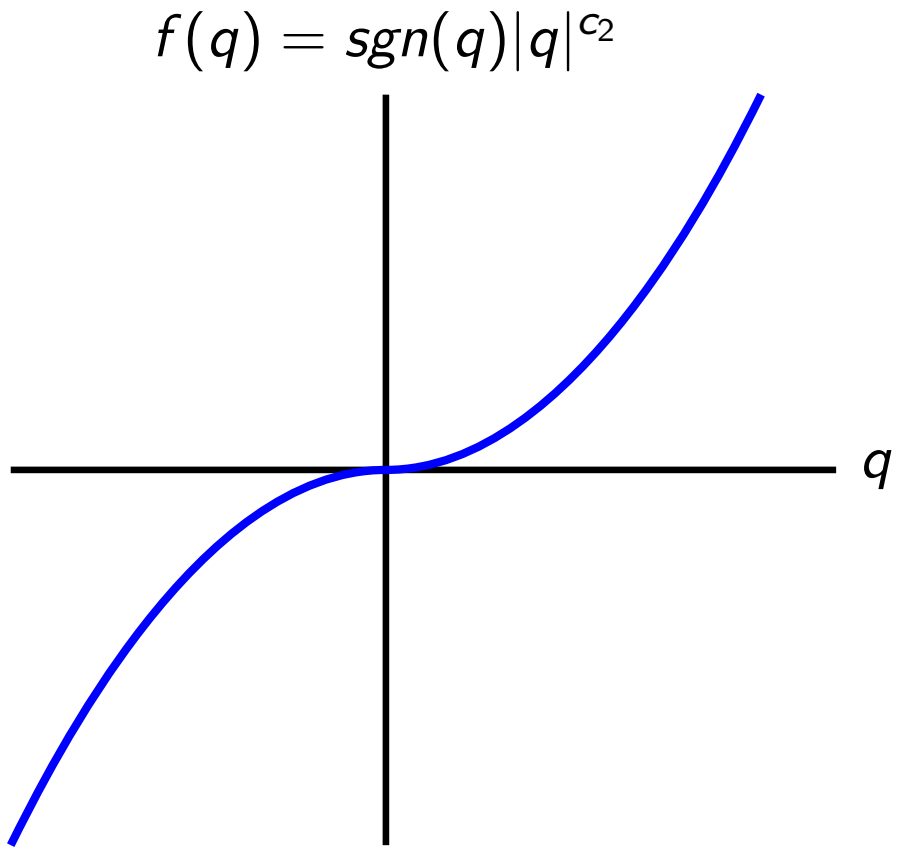
- Another trick: disaggregate flow into flow variables for each pipe size:

$$q_{ij} = \sum_{k \in \mathcal{K}} q_{ijk}$$

$$(-V_{\max} a_{ij}) y_{ijk} \leq q_{ijk} \leq (V_{\max} a_{ij}) y_{ijk}$$

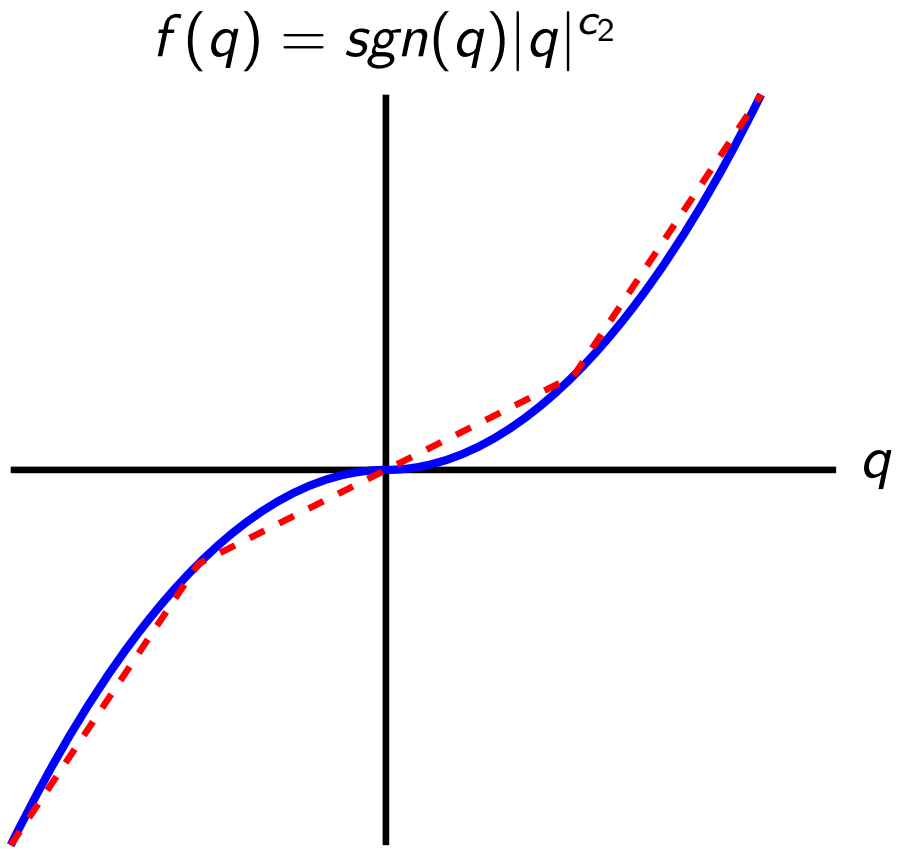
Why On Earth?

- Doing this allows us to write the (nonlinear) headloss as a function of a single variable q_{ijk}
- Often in applications, the nonlinearity is low-dimensional (low $\in \{1, 2\}$)



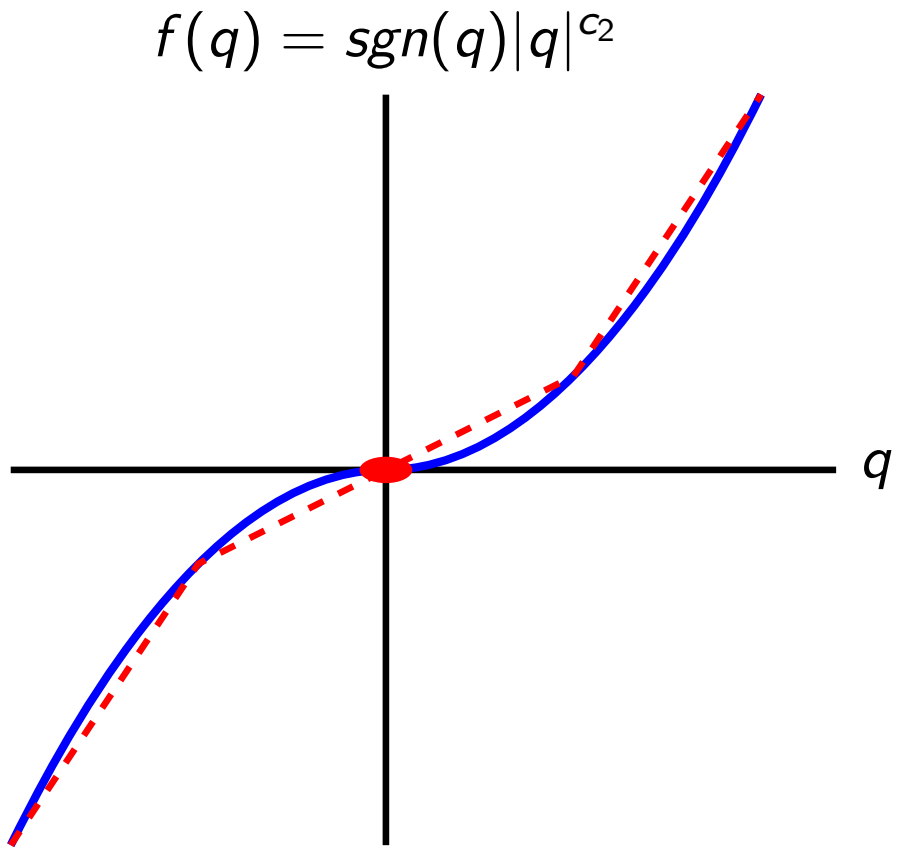
Why On Earth?

- Doing this allows us to write the (nonlinear) headloss as a function of a single variable q_{ijk}
- Often in applications, the nonlinearity is low-dimensional (low $\in \{1, 2\}$)
- Therefore a piecewise linear approximation (or relaxation) may be sufficiently accurate



Why On Earth?

- Doing this allows us to write the (nonlinear) headloss as a function of a single variable q_{ijk}
- Often in applications, the nonlinearity is low-dimensional (low $\in \{1, 2\}$)
- Therefore a piecewise linear approximation (or relaxation) may be sufficiently accurate



Luedtke's Theorem!

Note also that

$y_{ijk} = 0 \Rightarrow q_{ijk} = 0$

Optimization of IEEE 802.11 Broadband Networks

Optimize 802.11 broadband networks for resource sharing meshes

- objective: minimizing co-channel and inter-channel interference
- integrality: assign channels to basic nodes within a network
- 13 Direct Sequence Spread Spectrum (DSSS) overlapping channels
- co-channel interference: two access points with same channel
- inter-channel interference: cards with overlapping channels transmit simultaneously

$\Rightarrow$ general nonconvex MINLP

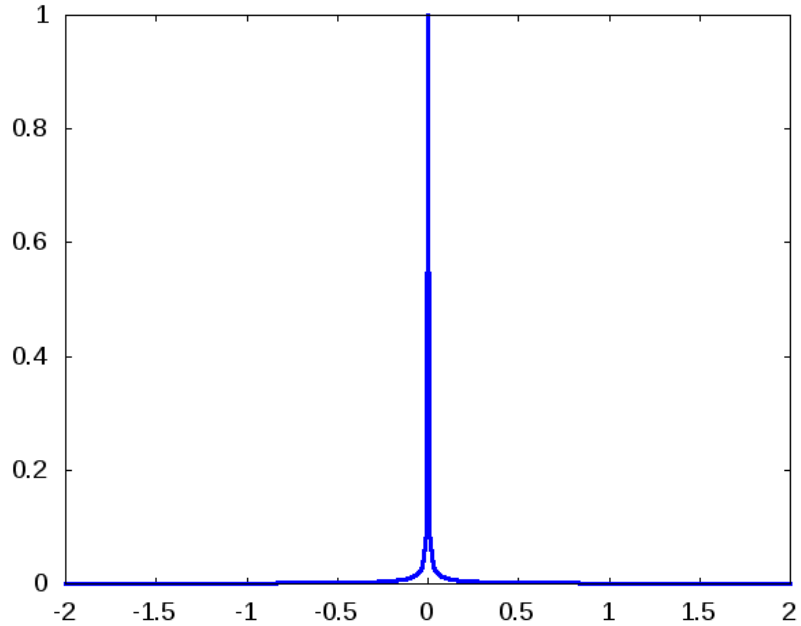
Original model has one horrible constraint ...

Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex

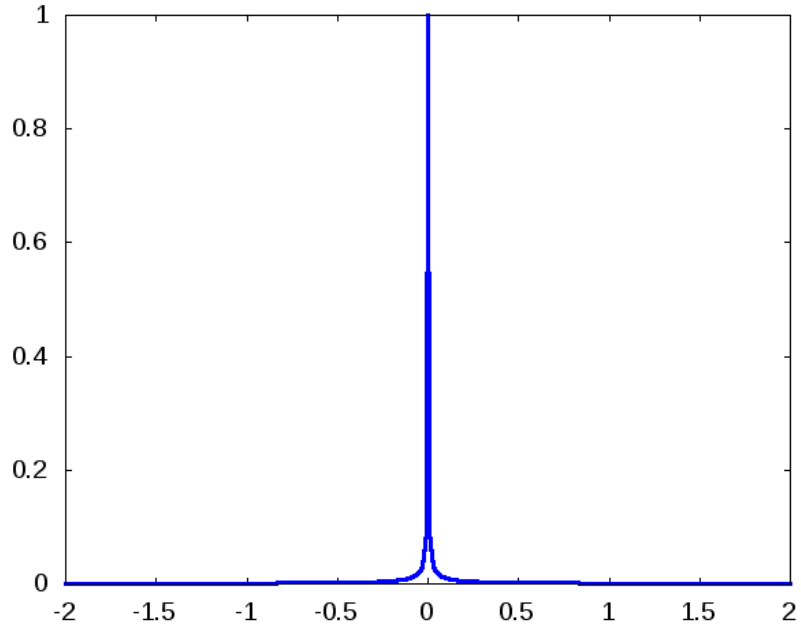


Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex
- $z = 1$, if $x = y$
- $z = 0$, if $x \neq y$

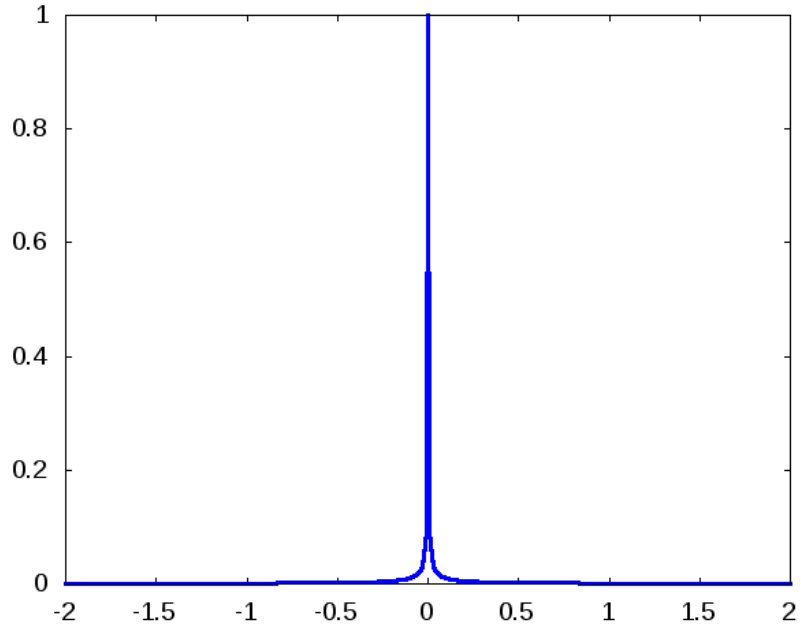


Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex
- $z = 1$, if $x = y$
- $z = 0$, if $x \neq y$
- x, y integer (channels)

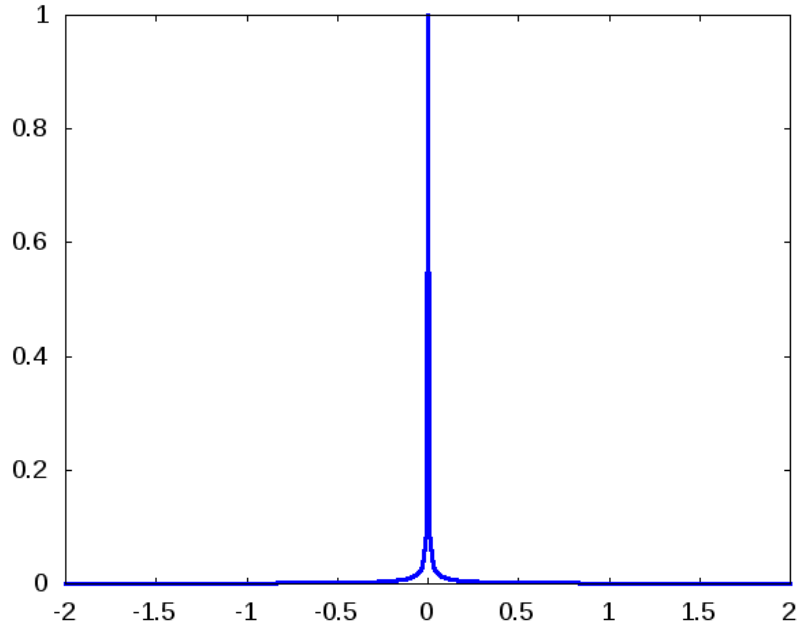


Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex
- $z = 1$, if $x = y$
- $z = 0$, if $x \neq y$
- x, y integer (channels)
- model as MIP not NLP

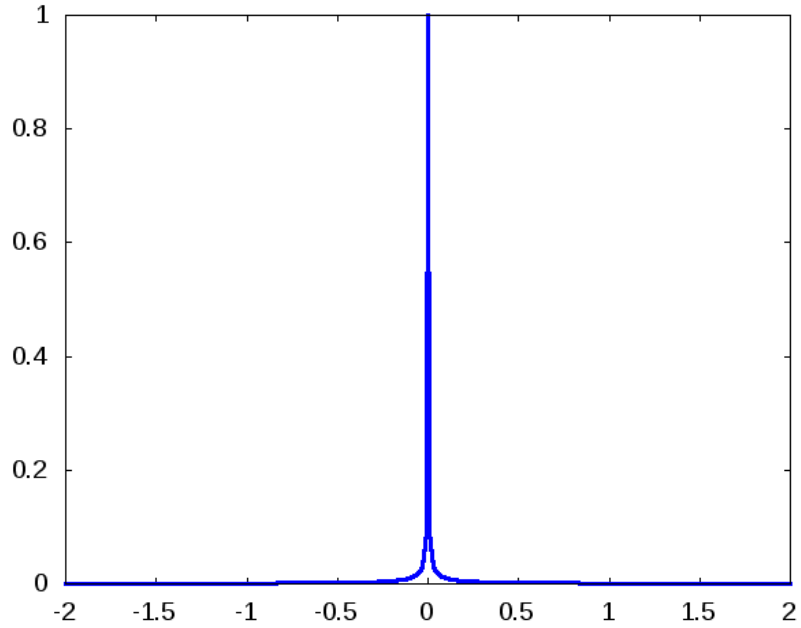


Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex
- $z = 1$, if $x = y$
- $z = 0$, if $x \neq y$
- x, y integer (channels)
- model as MIP not NLP



Exercise

Assuming that $0 \leq x, y \leq U$ are integers, derive an equivalent linear model.

Outline

Relaxation and Constraint Enforcement

Relaxation

- Used to compute a lower bound on the optimum
- Obtained by enlarging feasible set; e.g. ignore constraints
- Typically much easier to solve than MINLP

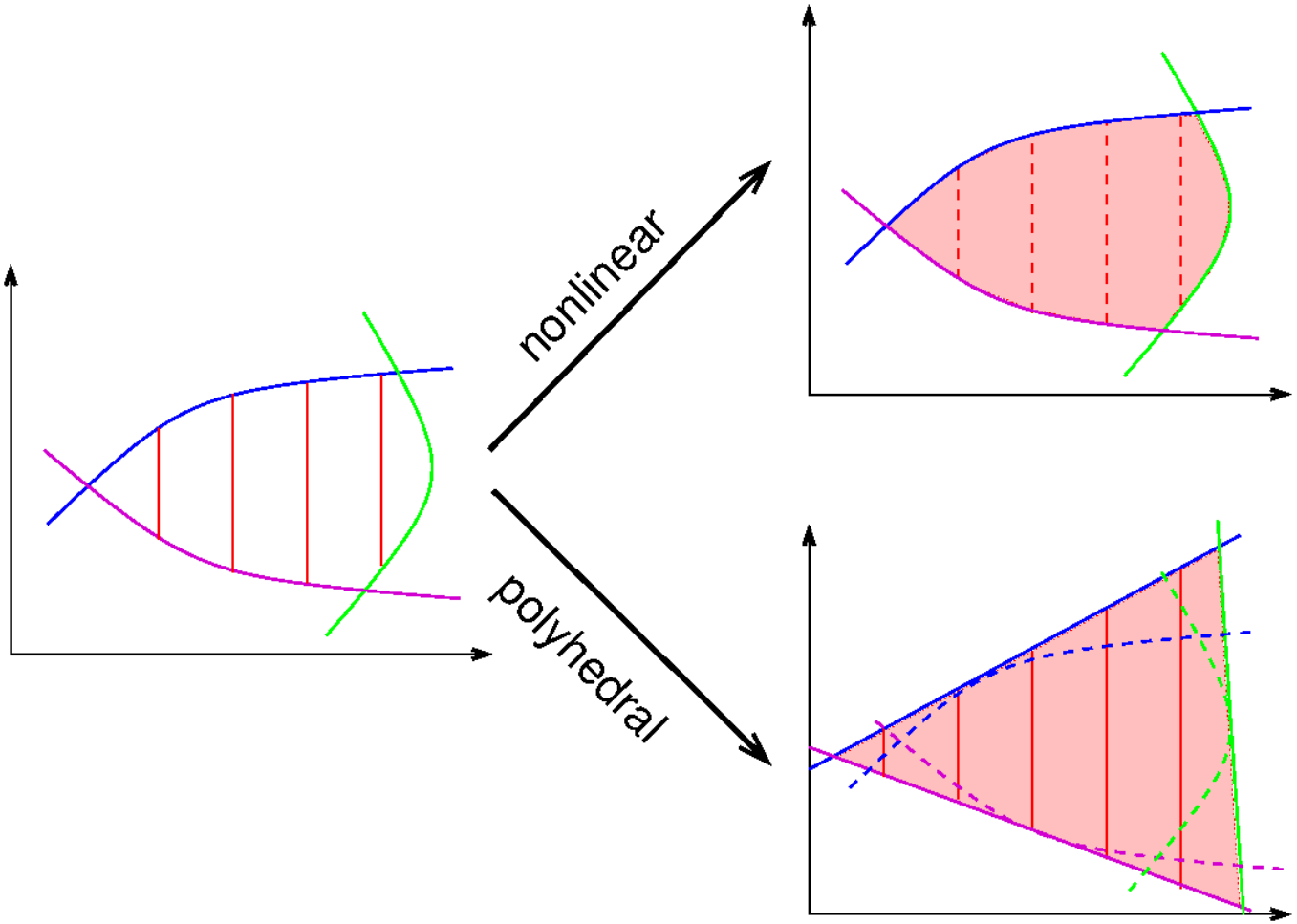
Constraint Enforcement

- Exclude solutions from relaxations not feasible in MINLP
- Refine or tighten of relaxation; e.g. add valid inequalities

Upper Bounds

- Obtained from any feasible point; e.g. solve NLP for fixed x_I

Classes of Relaxations for Convex MINLP

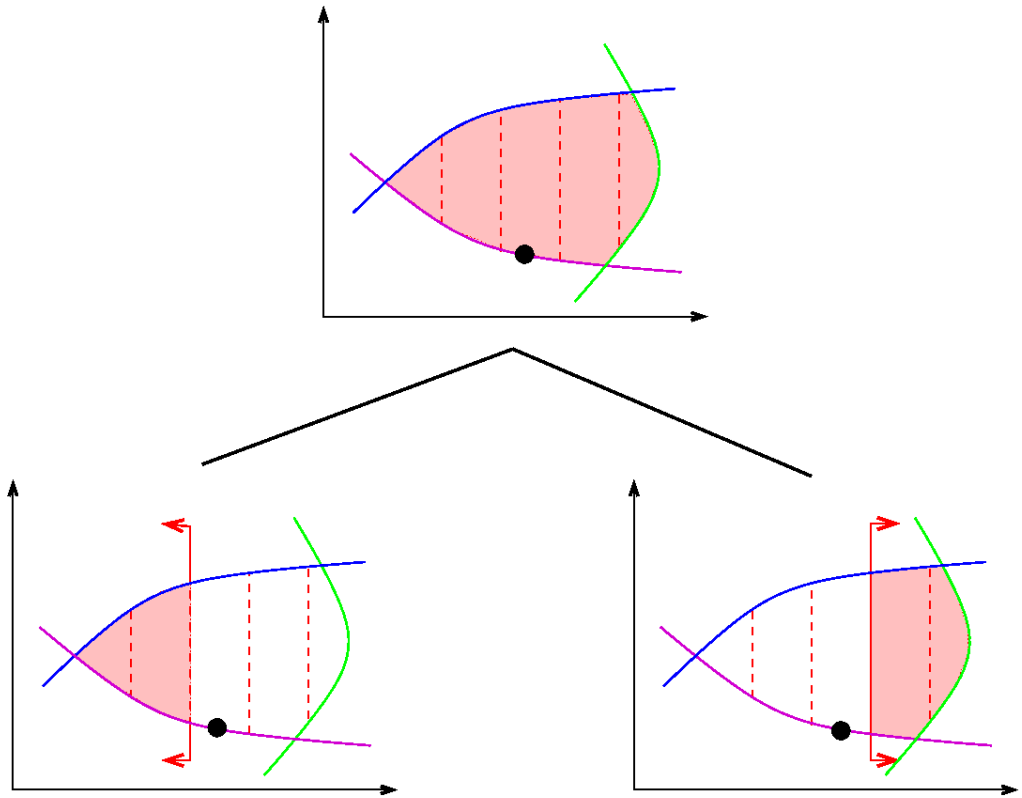


Nonlinear and polyhedral relaxation

Nonlinear Branch and Bound

Solve continuous "natural" NLP relaxations

... branch to "enforce" integrality



... creates branch-and-bound tree

Nonlinear Branch and Bound

Solve NLP relaxation (x_I continuous, not integer)

$$\underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } g(x) \leq 0, \; x \in X$$

- If $x_i \in \mathbb{Z} \; \forall \; i \in I$, then solved MINLP
- If relaxation is infeasible, then MINLP infeasible

... otherwise search tree whose nodes are NLPs:

$$\begin{cases} \underset{x}{\text{minimize}} & f(x), \\ \text{subject to} & g(x) \leq 0, \\ & x \in X, \\ & l_i \leq x_i \leq u_i, \; \forall i \in I. \end{cases} \qquad (\text{NLP}(l, u))$$

NLP relaxation is NLP$(-\infty, \infty)$

Nonlinear Branch and Bound

Branching: solution x' of (NLP(l, u)) feasible but not integral:

- Find a nonintegral variable, say $x'_i, i \in I$.
- Introduce two child nodes with bounds $(l^-, u^-) = (l^+, u^+) = (l, u)$ and setting:

$$u_i^- := \lfloor x'_i \rfloor, \quad \text{and} \quad l_i^+ := \lceil x'_i \rceil$$

$\Rightarrow$ create NLP(l^-, u^-) / NLP(l^+, u^+) (down/up branch)

Nonlinear Branch and Bound

Branching: solution x' of (NLP(l, u)) feasible but not integral:

- Find a nonintegral variable, say $x'_i, i \in I$.
- Introduce two child nodes with bounds $(l^-, u^-) = (l^+, u^+) = (l, u)$ and setting:

$$u_i^- := \lfloor x'_i \rfloor, \quad \text{and} \quad l_i^+ := \lceil x'_i \rceil$$

$\Rightarrow$ create NLP(l^-, u^-) / NLP(l^+, u^+) (down/up branch)

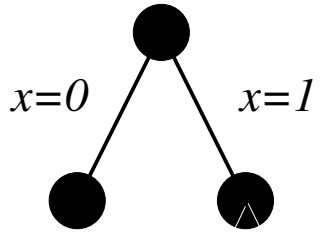
Pruning Rules: Let U upper bound on solution

1. (NLP(l, u)) infeasible $\Rightarrow$ NLPs in subtree also infeasible
2. Integer feasible solution $x^{(l,u)}$ of (NLP(l, u)):
   - If $f(x^{(l,u)}) < U$, then new $x^* = x^{(l,u)}$ and $U = f^{(l,u)}$.
   - Otherwise, prune node: no better solution in subtree
3. Optimal value of (NLP(l, u)), $f(x^{(l,u)}) \geq U$
   $\Rightarrow$ prune node: no better integer solution in subtree

Nonlinear Branch and Bound

Solve relaxed NLP ($0 \leq x \leq 1$) . . . solution gives lower bound

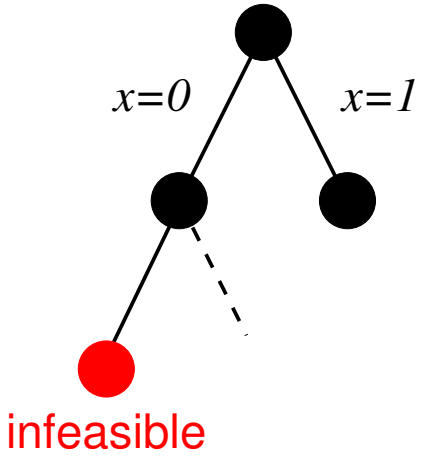
1. Solve NLPs & branch on x_i until



Nonlinear Branch and Bound

Solve relaxed NLP ($0 \leq x \leq 1$) . . . solution gives lower bound

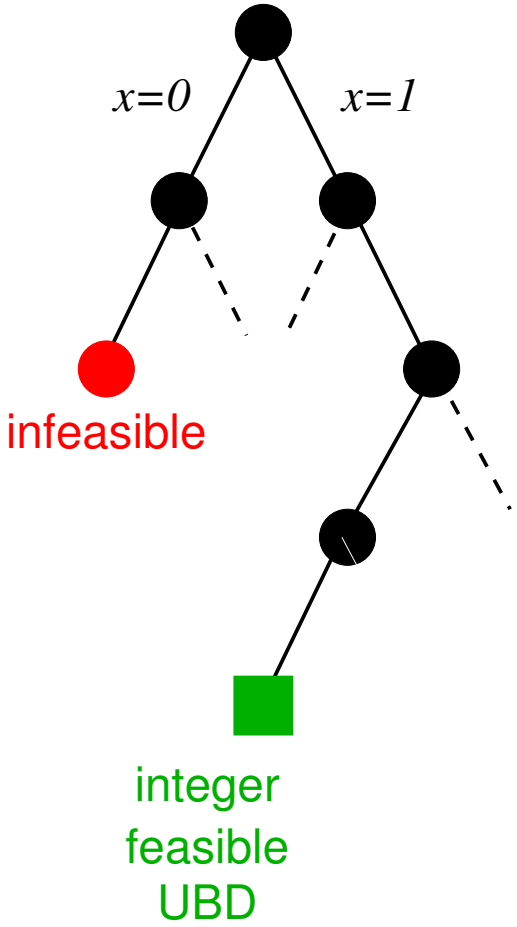
1. Solve NLPs & branch on x_i until
2. Node infeasible:



Nonlinear Branch and Bound

Solve relaxed NLP ($0 \leq x \leq 1$) . . . solution gives lower bound

1. Solve NLPs & branch on x_i until
2. Node infeasible: ●
3. Node integer feasible: □
   $\Rightarrow$ get upper bound (U)

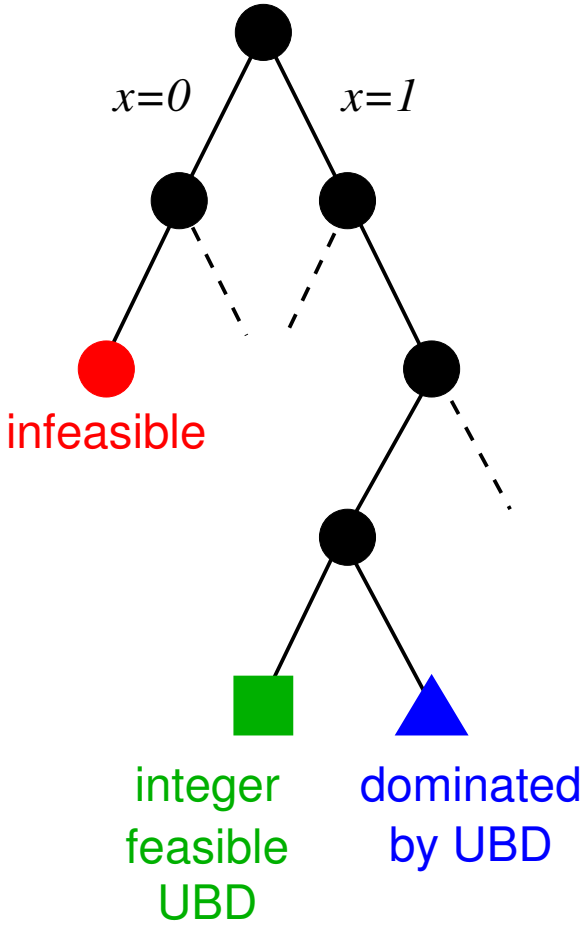


Nonlinear Branch and Bound

Solve relaxed NLP ($0 \leq x \leq 1$) . . . solution gives lower bound

1. Solve NLPs & branch on x_i until
2. Node infeasible: ●
3. Node integer feasible: □
   $\Rightarrow$ get upper bound (U)
4. Lower bound $\geq U$: ▲

Search until no unexplored nodes



Theorem (Convergence)

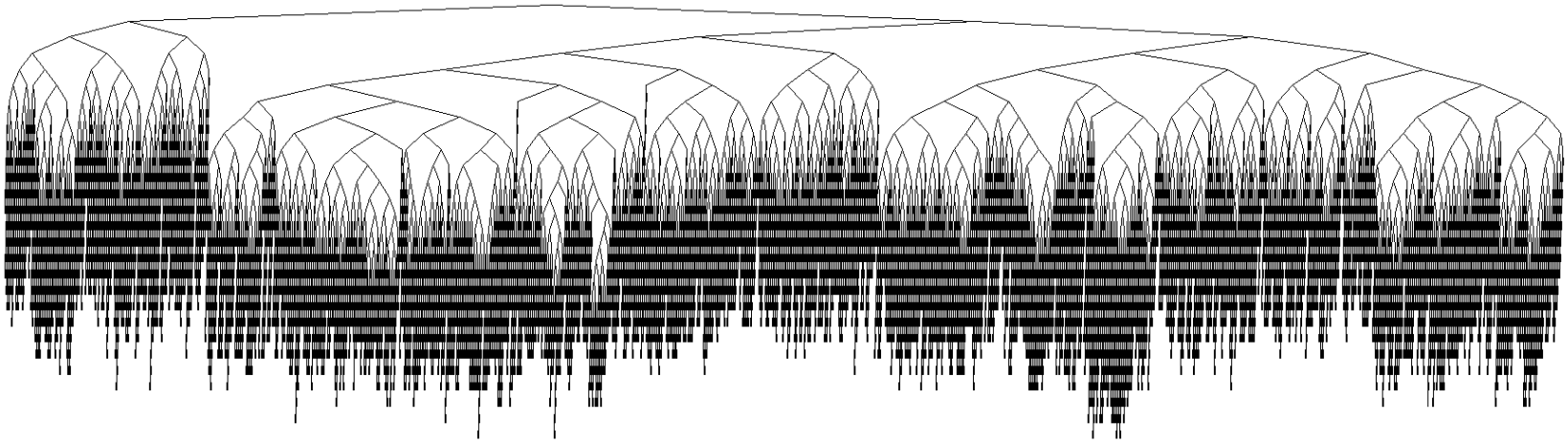
Assume that:

- *X bounded polyhedral set;*
- *NLP solver returns global min.*

$\Rightarrow$ *BnB terminates at optimal solution*

Nonlinear Branch and Bound

BnB trees can get pretty large ...

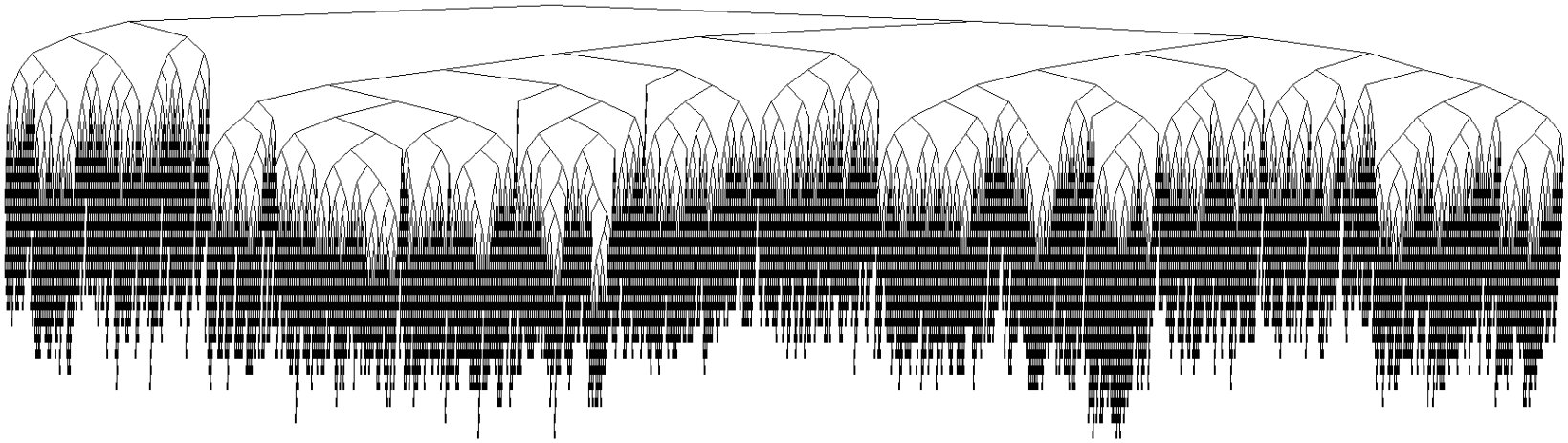


Synthesis MINLP B&B Tree: 10000+ nodes after 360s

... be smart about solving NLPs & searching tree ...

Nonlinear Branch and Bound

BnB trees can get pretty large ...



Synthesis MINLP B&B Tree: 10000+ nodes after 360s

... be smart about solving NLPs & searching tree ...

Common Theme in MINLP

Extend proven MILP techniques to MINLP

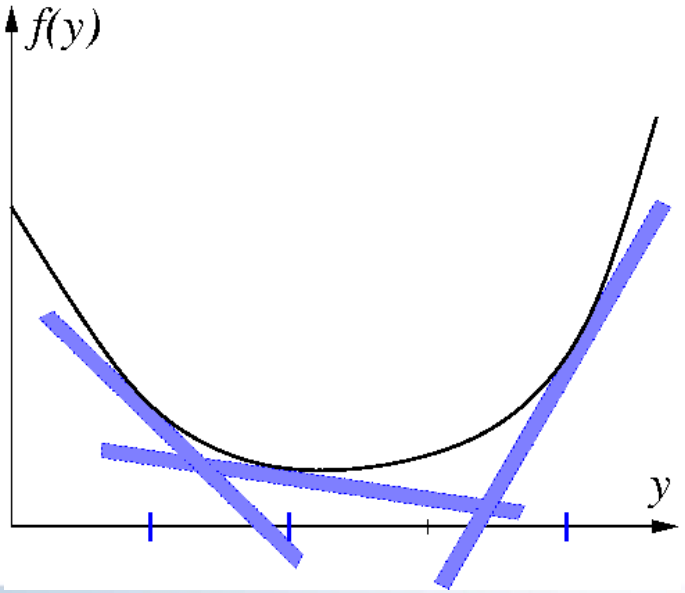
Relaxations of Nonlinear Convex Constraints

Relaxing Convex Constraints

- Convex $0 \geq c(x)$ and $\eta \geq f(x)$ relaxed by supporting hyperplanes

$$\eta \geq f^{(k)} + \nabla f^{(k)^T}(x - x^{(k)})$$
$$0 \geq c^{(k)} + \nabla c^{(k)^T}(x - x^{(k)})$$

  for a set of points $x^{(k)},\ k = 1, \ldots, K$.
- Obtain polyhedral relaxation of convex constraints.
- Used in the outer approximation methods.



MILP Master Problem & Relaxations

Mixed-Integer Nonlinear Program (MINLP)

$$\underset{x}{\text{minimize}}\ f(x)\ \text{ subject to }\ g(x) \leq 0,\ x \in X,\ \text{ and }\ x_i \in \mathbb{Z} \text{ for all } i \in I$$

Equivalent MILP master problem for f, g convex:

$$\begin{cases} \underset{\eta,x}{\text{minimize}}\ \eta \\ \text{subject to } \eta \geq f^{(k)} + \nabla f^{(k)^T}(x - x^{(k)}), & \forall k \in \mathcal{K} \\ \qquad\qquad 0 \geq g^{(k)} + \nabla g^{(k)^T}(x - x^{(k)}), & \forall k \in \mathcal{K} \\ \qquad\qquad x \in X,\ x_i \in \mathbb{Z} \text{ for all } i \in I \end{cases}$$

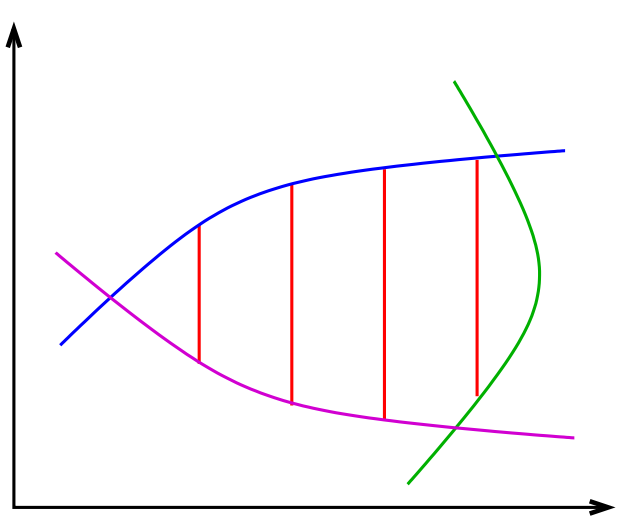
where $\exists$ finite set $\mathcal{K} = \left\{k : x^{(k)} \in X,\ x_I^{(k)} \in \mathbb{Z}^p\right\}$ and $f^{(k)} = f(x^{(k)})$ etc.

MILP relaxation use $\mathcal{J} \subset \mathcal{K}$

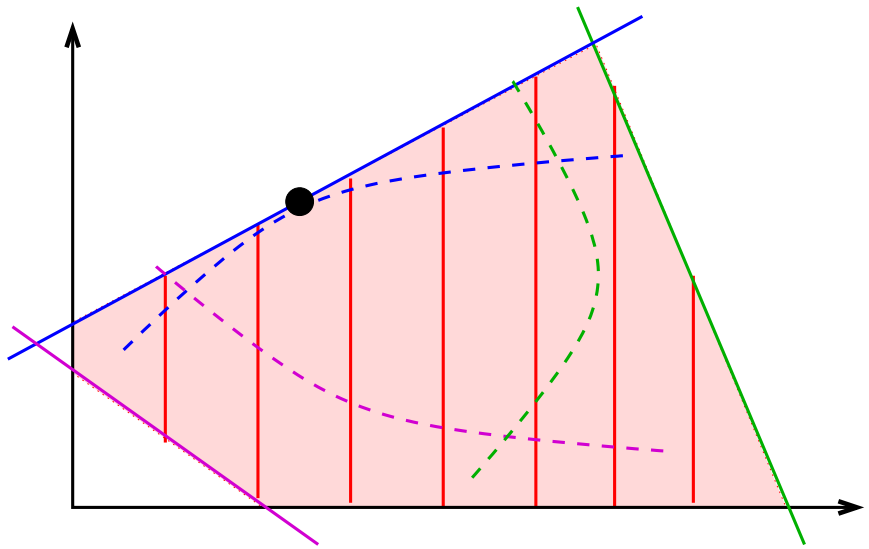
LP/NLP Branch and Bound

LP/NLP-based branch-and-bound

- Branch-and-cut algorithm with cuts from NLP solves
- Create MILP relaxation of MINLP



$0 \geq g(x)$



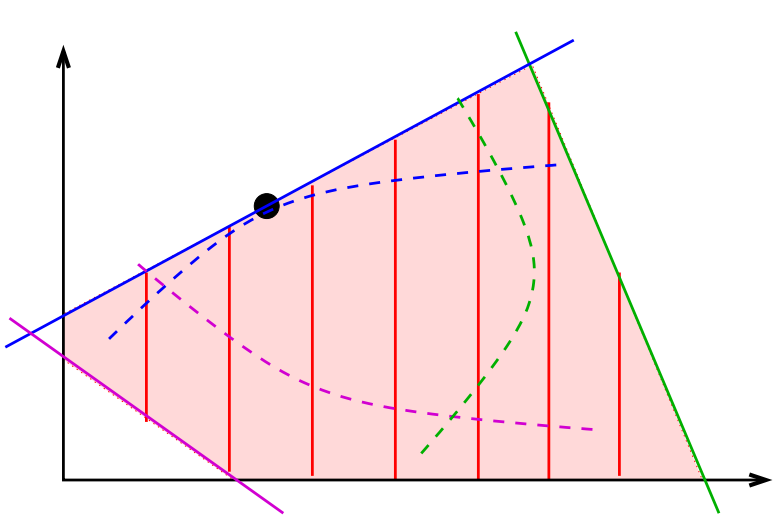
$0 \geq g^{(k)} + \nabla g^{(k)^T}(x - x^{(k)})$

- Search MILP-tree $\Rightarrow$ faster re-solves
- Interrupt MILP tree-search to create new linearizations

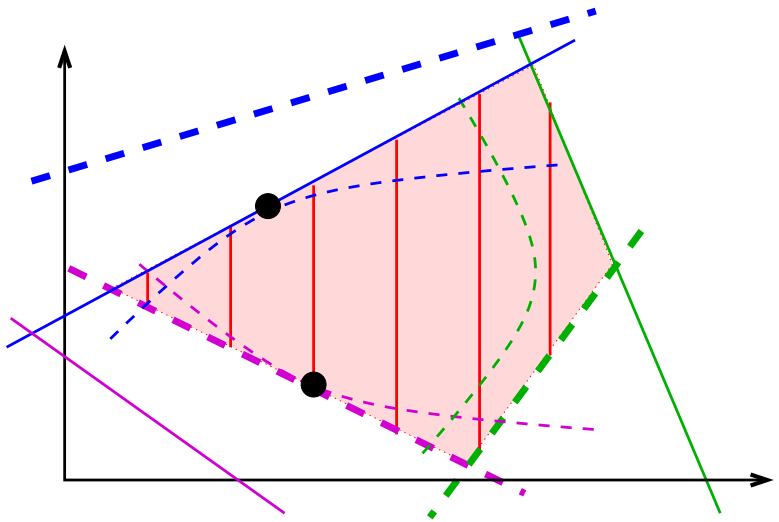
LP/NLP Branch and Bound

LP/NLP-based branch-and-bound

- Branch-and-cut algorithm with cuts from NLP solves
- Create MILP relaxation of MINLP & refine linearizations



$0 \geq g(x)$



$0 \geq g^{(k)} + \nabla g^{(k)^T}(x - x^{(k)})$

- Search MILP-tree $\Rightarrow$ faster re-solves
- Interrupt MILP tree-search to create new linearizations

LP/NLP-Based Branch and Bound

Aim: avoid perform tree-search on "cheaper" MILP tree
NLP subproblem for fixed $\hat{x}_I$:

$$\text{NLP}(\hat{x}_I) \begin{cases} \min\limits_{x} f(x) \\ \text{s.t. } g(x) \leq 0,\ x_I = \hat{x}_I \end{cases}$$

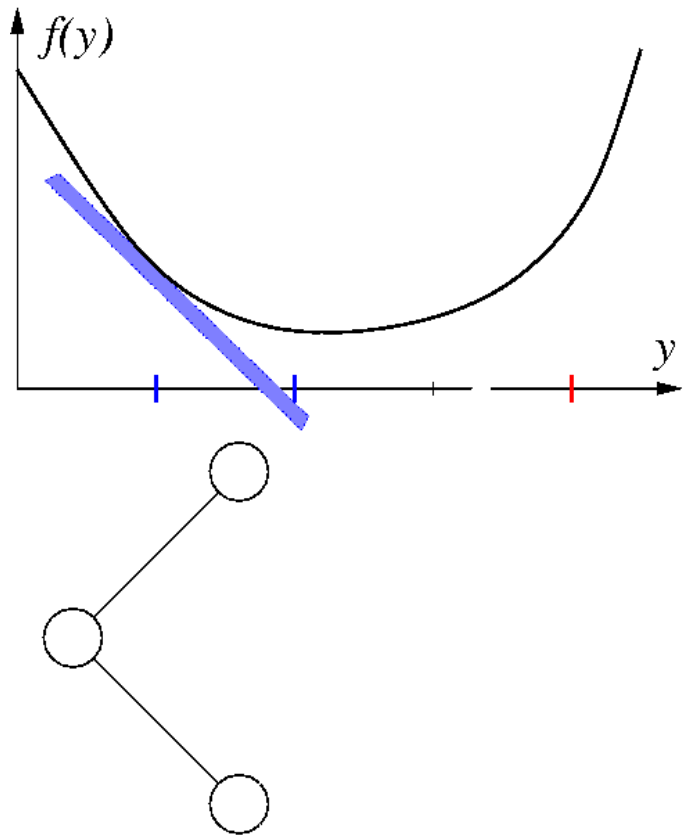
- Relax MILP master

LP/NLP-Based Branch and Bound

Aim: avoid perform tree-search on "cheaper" MILP tree
NLP subproblem for fixed $\hat{x}_I$:

$$\text{NLP}(\hat{x}_I) \begin{cases} \min\limits_{x} f(x) \\ \text{s.t. } g(x) \leq 0,\ x_I = \hat{x}_I \end{cases}$$

- Relax MILP master
- Start initial MILP tree

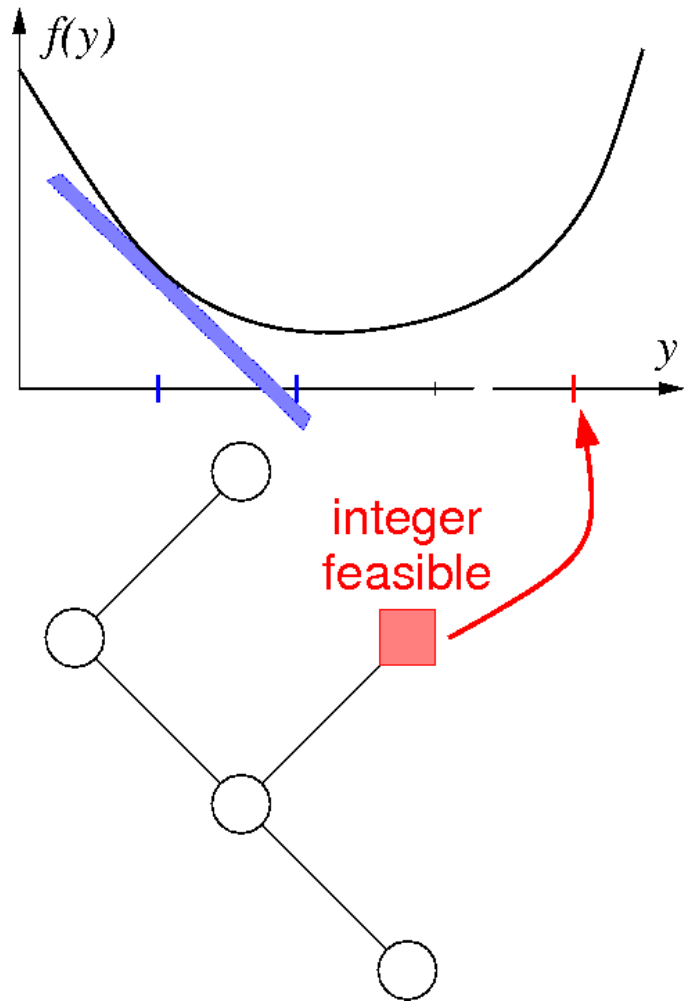


LP/NLP-Based Branch and Bound

Aim: avoid perform tree-search on "cheaper" MILP tree
NLP subproblem for fixed $\hat{x}_I$:

$$\text{NLP}(\hat{x}_I) \begin{cases} \min\limits_{x} f(x) \\ \text{s.t. } g(x) \leq 0,\ x_I = \hat{x}_I \end{cases}$$

- Relax MILP master
- Start initial MILP tree
- interrupt MILP, when $x_I^{(j)}$ integral
  - solve NLP($x_I^{(j)}$) get $x^{(j)}$
  - linearize f, c about $x^{(j)}$
  - add linearization to tree



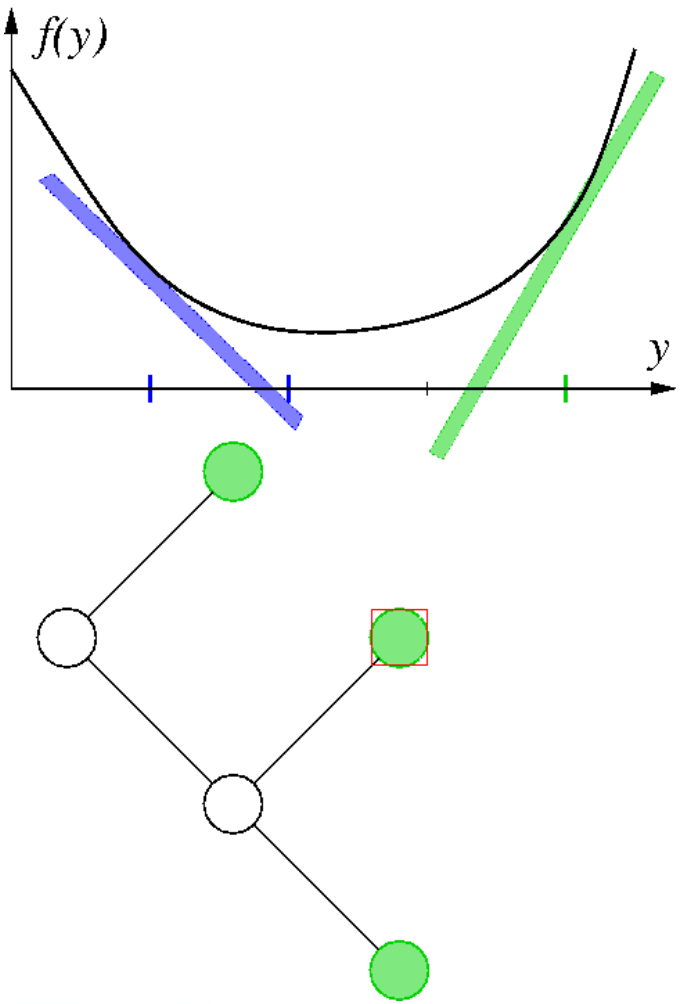
LP/NLP-Based Branch and Bound

Aim: avoid perform tree-search on "cheaper" MILP tree
NLP subproblem for fixed $\hat{x}_I$:

$$\text{NLP}(\hat{x}_I) \begin{cases} \min\limits_{x} f(x) \\ \text{s.t. } g(x) \leq 0,\ x_I = \hat{x}_I \end{cases}$$

- Relax MILP master
- Start initial MILP tree
- interrupt MILP, when $x_I^{(j)}$ integral
  - solve NLP($x_I^{(j)}$) get $x^{(j)}$
  - linearize f, c about $x^{(j)}$
  - add linearization to tree
- continue MILP tree-search

... until lower bound $\geq$ upper bound



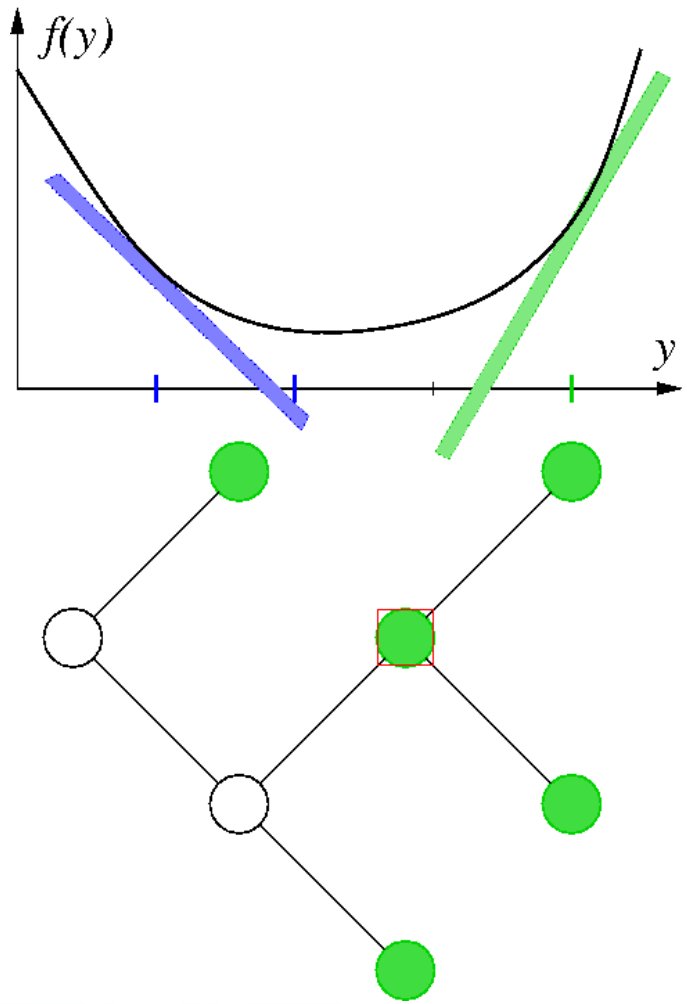
LP/NLP-Based Branch and Bound

Aim: avoid perform tree-search on "cheaper" MILP tree
NLP subproblem for fixed $\hat{x}_I$:

$$\text{NLP}(\hat{x}_I) \begin{cases} \min\limits_{x} f(x) \\ \text{s.t. } g(x) \leq 0,\ x_I = \hat{x}_I \end{cases}$$

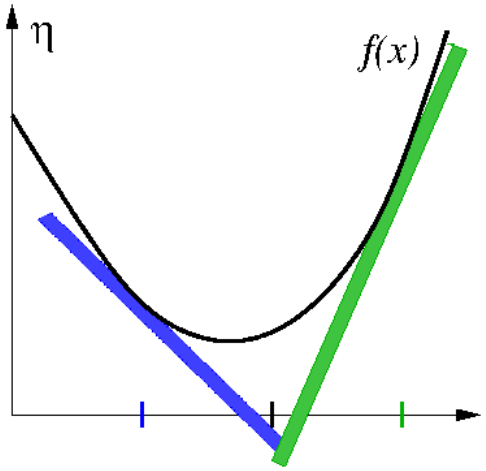
- Relax MILP master
- Start initial MILP tree
- interrupt MILP, when $x_I^{(j)}$ integral
  - solve NLP($x_I^{(j)}$) get $x^{(j)}$
  - linearize f, c about $x^{(j)}$
  - add linearization to tree
- continue MILP tree-search

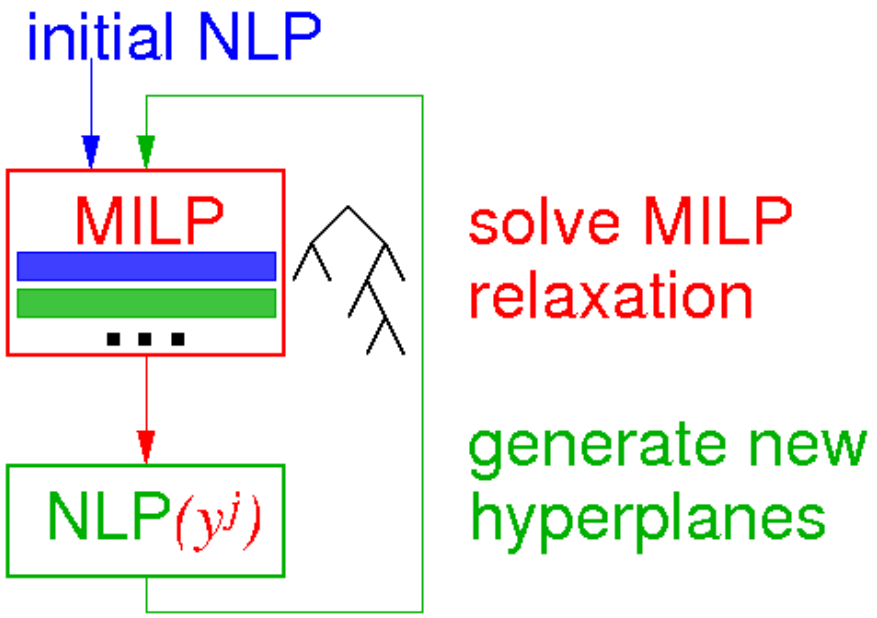
... until lower bound $\geq$ upper bound



Outer Approximation, Benders Decomposition & ECP

Alternate between solve of NLP(x_I) and MILP relaxation





MILP $\Rightarrow$ lower bound; NLP $\Rightarrow$ upper bound

Snag: Solve multiple MILPs ... no hot-starts

- Outer approximation [Duran and Grossmann, 1986]
- Benders decomposition [Geoffrion, 1972]
- Extended cutting plane [Westerlund and Pettersson, 1995]

Solvers for MINLP

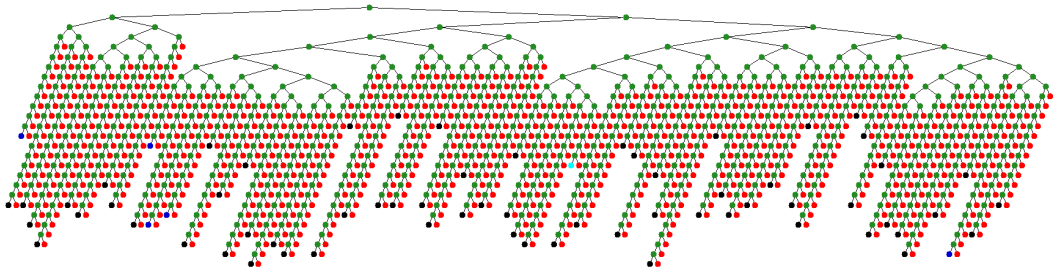
Convex		**Nonconvex**
α-ECP		α-BB
BONMIN		ANTIGONE
DICOPT		BARON
FilMINT		COCONUT
GUROBI†		COUENNE
KNITRO		CPLEX‡
MILANO		LGO
MINLPBB		LindoGlobal
MINOPT		SCIP
MINOTAUR		
SBB		
Xpress†		
†MIQP		‡MIQP/convex MIQP

Open-source and commercial solvers

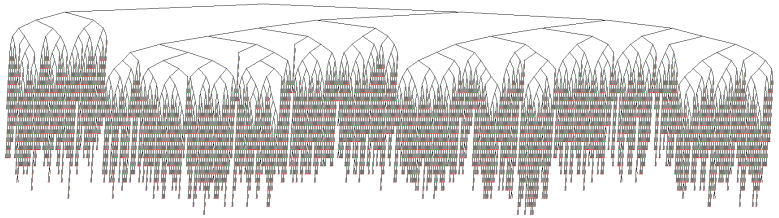
What Could Go Wrong in MINLP?

Syn20M04M : a synthesis design problem in chemical engineering

Problem size: 160 Integer Variables, 56 Nonlinear constraints

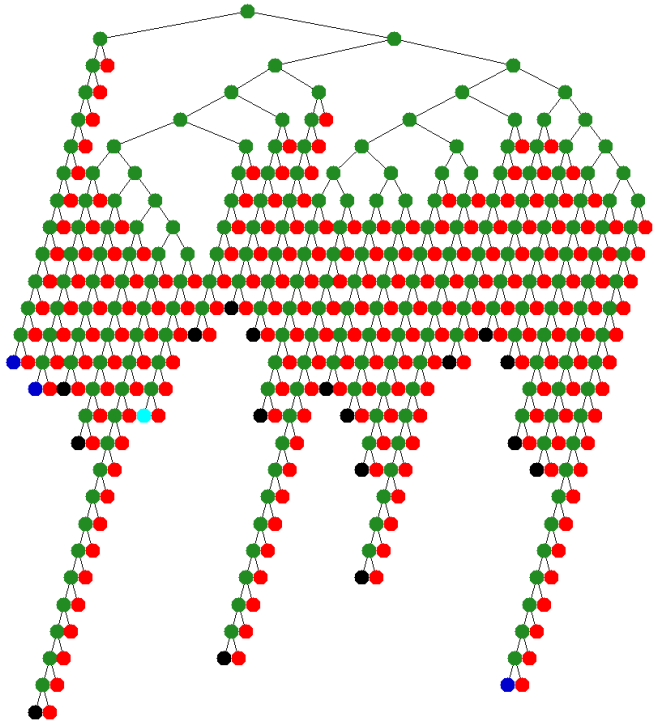


1000+ nodes after solving for 75s



5000+ nodes after solving for 200s

See solution this afternoon!



250+ nodes after solving for 45s

Solver	CPU	Nodes
Bonmin	>2h	>149k
MINLPBB	>2h	>150k
Minotaur	>2h	>264k

Improving Coefficients: An Example

$$(1) \quad x_1 + 21x_2 \leq 30$$

$$0 \leq x_1 \leq 14$$

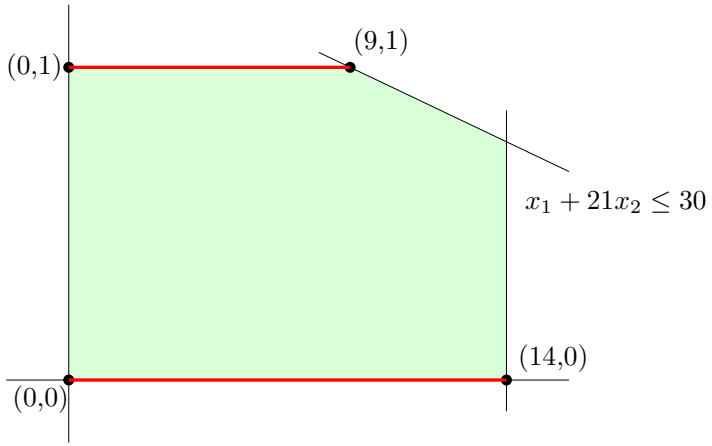
$$x_2 \in \{0, 1\}$$

Improving Coefficients: An Example

(1) $x_1 + 21x_2 \leq 30$

$0 \leq x_1 \leq 14$

$x_2 \in \{0, 1\}$



If $x_2 = 0$

$x_1 + 0 \leq 30$

(1) is loose.

If $x_2 = 1$

$x_1 \leq 9$

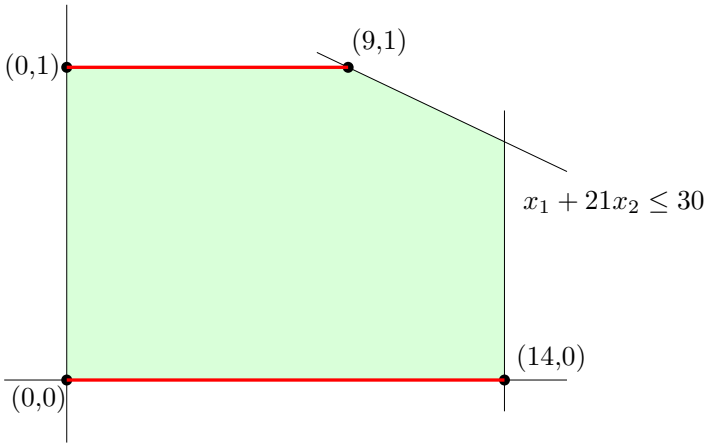
(1) is tight.

Improving Coefficients: An Example

(1) $x_1 + 21x_2 \leq 30$

$0 \leq x_1 \leq 14$

$x_2 \in \{0, 1\}$



If $x_2 = 0$

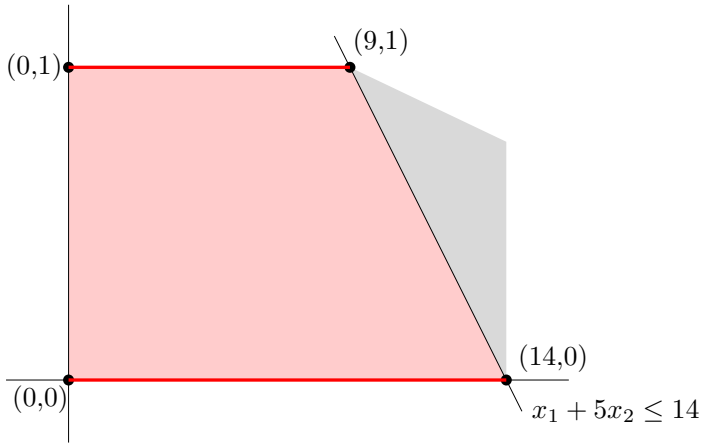
$x_1 + 0 \leq 30$

(1) is loose.

If $x_2 = 1$

$x_1 \leq 9$

(1) is tight.



Improving Coefficients: An Example

(1) $x_1 + 21x_2 \leq 30$

$0 \leq x_1 \leq 14$

$x_2 \in \{0, 1\}$

If $x_2 = 0$

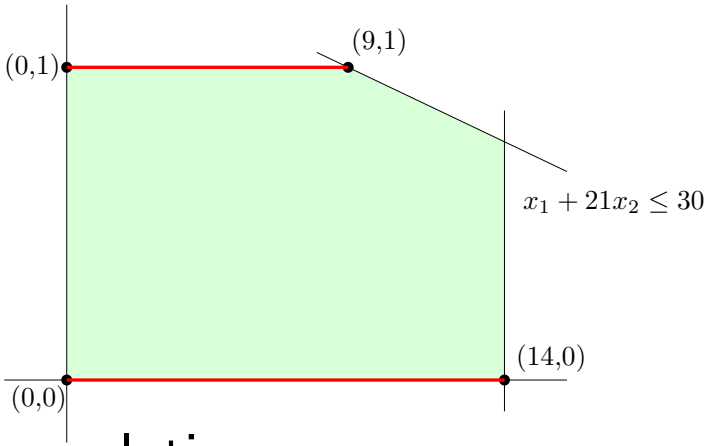
$x_1 + 0 \leq 30$

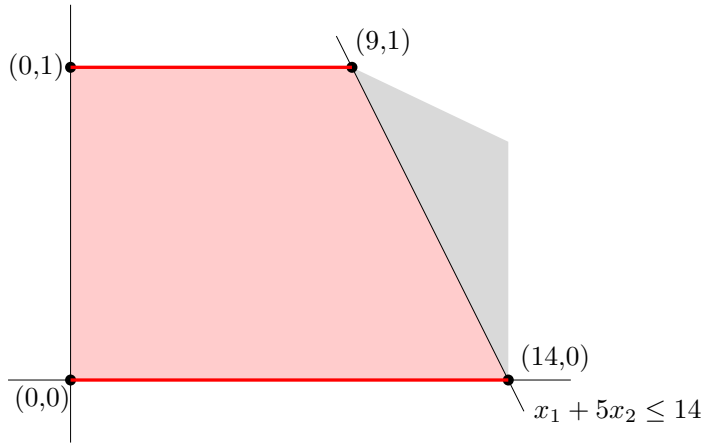
(1) is loose.

If $x_2 = 1$

$x_1 \leq 9$

(1) is tight.





Reformulation:

(2) $x_1 + 5x_2 \leq 14$

$0 \leq x_1 \leq 14$

$x_2 \in \{0, 1\}$

If $x_2 = 0$

$x_1 + 0 \leq 14$

(2) is tight.

If $x_2 = 1$

$x_1 \leq 9$

(2) is tight.

(1) and (2) equivalent. But relaxation of (2) is tighter.

Improving Coefficients: Using Implications

Common MINLP structure: $x_0 \in \{0, 1\}$, and $x_i \in \mathbb{R}$:

$$
\begin{aligned}
c(x_1, x_2, \ldots, x_k) &\leq M(1 - x_0) \\
0 \leq x_1 &\leq M_1 x_0 \\
0 \leq x_2 &\leq M_2 x_0 \\
&\ldots \\
0 \leq x_k &\leq M_k x_0 \\
x_0 &\in \{0, 1\}
\end{aligned}
$$

Improving Coefficients: Using Implications

Common MINLP structure: $x_0 \in \{0,1\}$, and $x_i \in \mathbb{R}$:

$$\begin{aligned} c(x_1, x_2, \ldots, x_k) &\leq M(1 - x_0) \\ 0 \leq x_1 &\leq M_1 x_0 \\ 0 \leq x_2 &\leq M_2 x_0 \\ &\ldots \\ 0 \leq x_k &\leq M_k x_0 \\ x_0 &\in \{0,1\} \end{aligned}$$

- $x_0 = 0 \Rightarrow x_1 = x_2, \ldots = x_k = 0$. (Implications)
- If $c^u := c(0, \ldots, 0) < M$, then we can improve coefficients

$$c(x_1, \ldots, x_k) \leq c^u(1 - x_0)$$

harder without $0 \leq x_i \leq M_i x_0$ (need global minimization)

Presolve for MINLP

Advanced functions of presolve (Reformulating):

- Improve coefficients.
- Disaggregate constraints.
- Derive implications and conflicts.

Basic functions of presolve (Housekeeping):

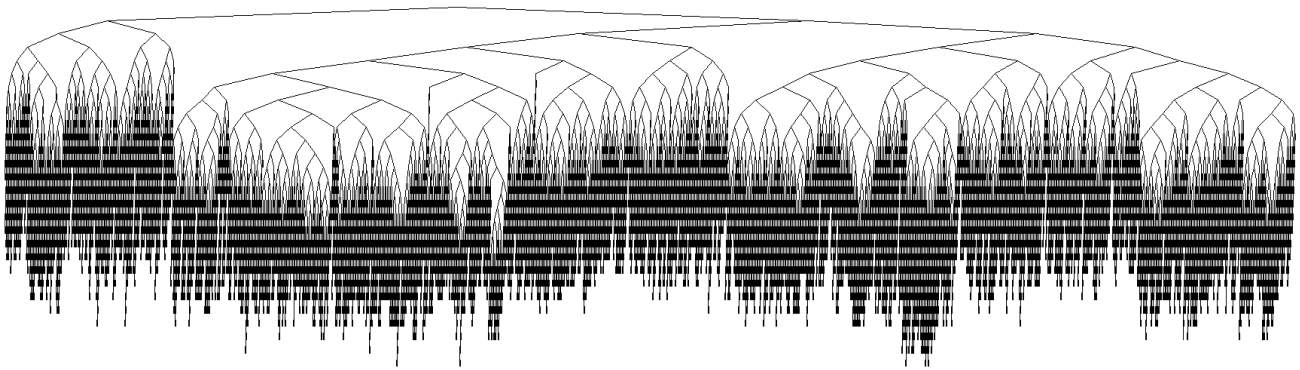
- Tighten bounds on variables and constraints.
- Fix/remove variables.
- Identify and remove redundant constraints.
- Check duplicacy.

Popular in Mixed-Integer Linear Optimization [Savelsbergh, 1994]

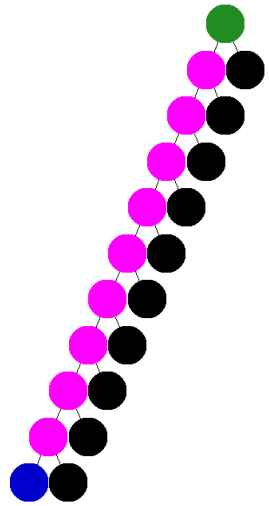
Presolve for MINLP: Computational Results

`Syn20M04M` from `egon.cheme.cmu.edu`

	No Presolve	Basic Presolve	Full Presolve
Variables:	420	328	292
Binary Vars:	160	144	144
Constraints:	1052	718	610
Nonlin. Constr:	56	56	56
Bonmin(sec):	>7200	NA	NA
Minotaur(sec):	>7200	>7200	2.3

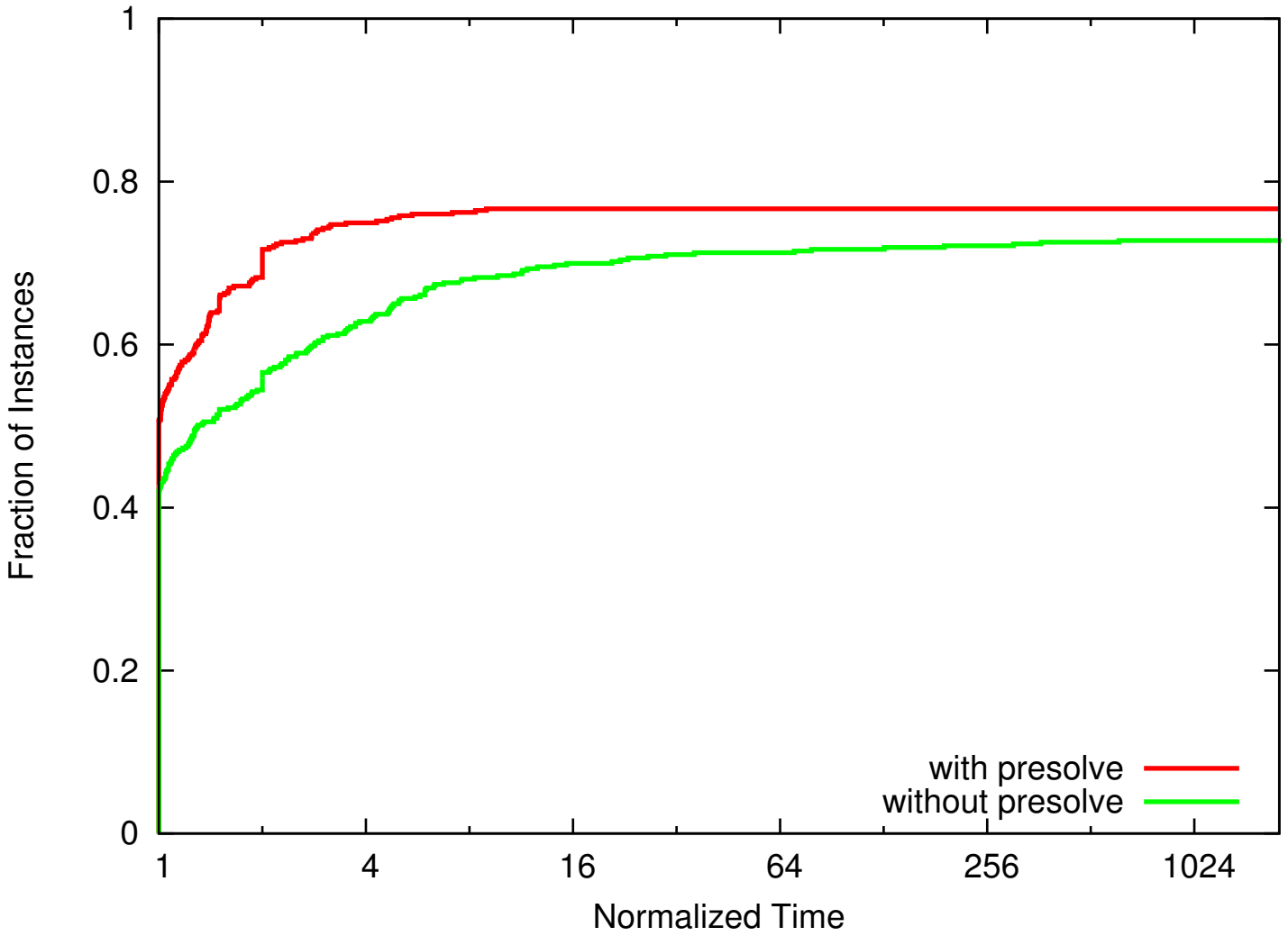


Minotaur, no presolve: 10000+ nodes after solving for 360s



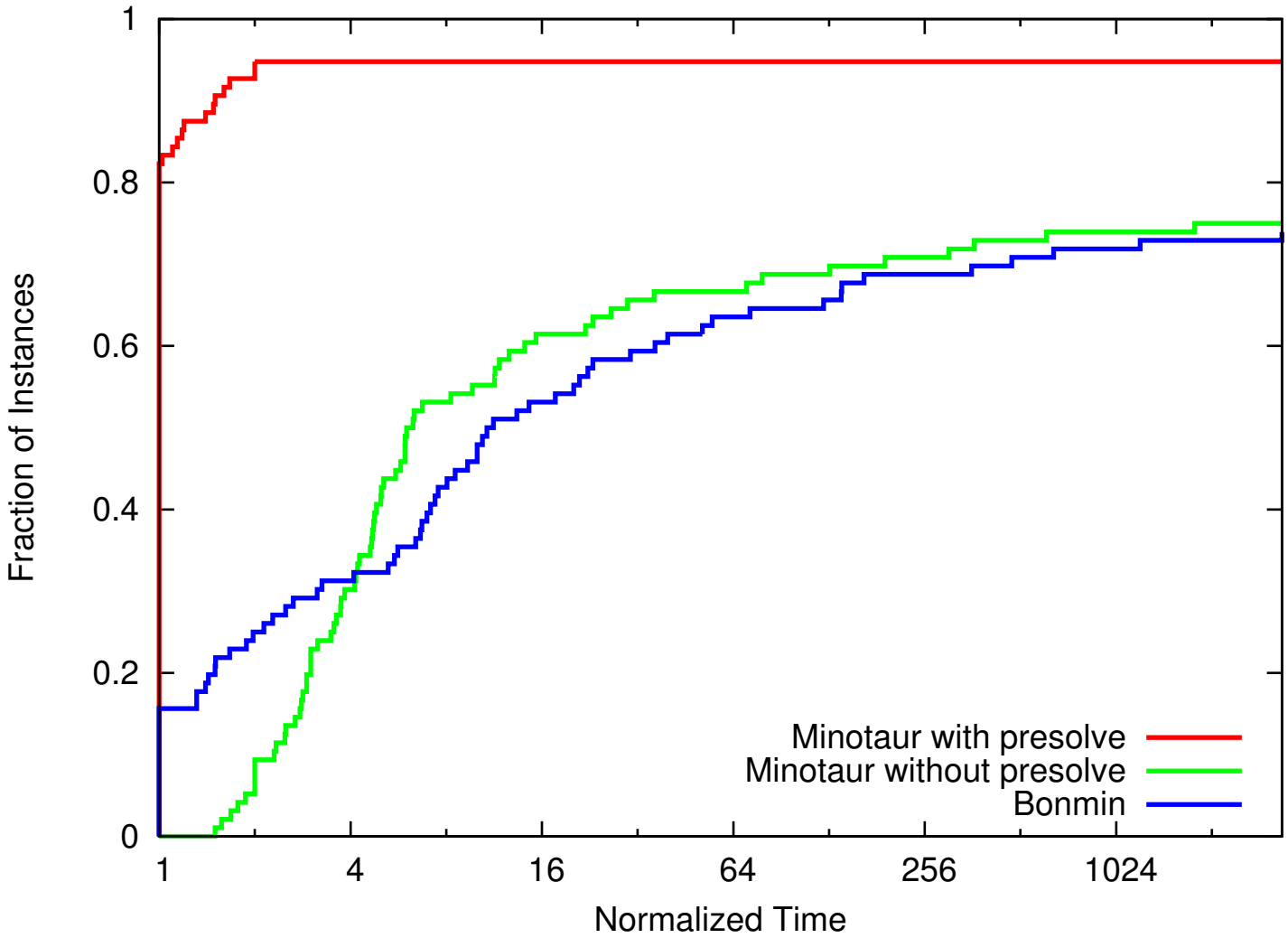
Full Presolve

Presolve for MINLP: Results



Time taken in Branch-and-Bound on all 463 instances.

Presolve for MINLP: Results



Time for B&B on 96 RSyn-X and Syn-X instances.

Take Home Messages

- Modeling is extremely important for MINLP
  ... strong relaxations are key
- Effective MILP techniques useful for MINLP
  ... more work can be done!
- MINLP software maturing,
  ... but lags behind commercial MILP software
- Linear better than convex better than nonconvex
- Exploit structure whenever possible

MINOTAUR's Soft-Wear Stack

... available at `www.mcs.anl.gov/minotaur`


Duran, M. A. and Grossmann, I. (1986).
An outer-approximation algorithm for a class of mixed-integer nonlinear programs.
Mathematical Programming, 36:307–339.


Geoffrion, A. M. (1972).
Generalized Benders decomposition.
Journal of Optimization Theory and Applications, 10(4):237–260.


Grossmann, I. and Lee, S. (2003).
Generalized convex disjunctive programming: Nonlinear convex hull relaxation.
Computational Optimization and Applications, pages 83–100.


Günlük, O. and Linderoth, J. T. (2012).
Perspective reformulation and applications.
In *IMA Volumes*, volume 154, pages 61–92.


Jeroslow, R. G. (1973).
There cannot be any algorithm for integer programming with quadratic constraints.
Operations Research, 21(1):221–224.


Kannan, R. and Monma, C. (1978).
On the computational complexity of integer programming problems.
In Henn, R., Korte, B., and Oettli, W., editors, *Optimization and Operations Research,*, volume 157 of *Lecture Notes in Economics and Mathematical Systems*, pages 161–172. Springer.


Savelsbergh, M. W. P. (1994).

Preprocessing and probing techniques for mixed integer programming problems.
ORSA Journal on Computing, 6:445–454.

Westerlund, T. and Pettersson, F. (1995).
A cutting plane method for solving convex MINLP problems.
Computers & Chemical Engineering, 19:s131–s136.

Williams, H. P. (1999).
Model Building in Mathematical Programming.
John Wiley & Sons.