Workshop on Integer Programming and Continuous Optimization

Chemnitz, November 2004

Integer Nonlinear Optimization

Sven Leyffer

leyffer@mcs.anl.gov

Mathematics and Computer Science Division, Argonne National Laboratory

Integer Nonlinear Optimization

Sven Leyffer

1. Introduction & Applications
2. Classical MINLP Methods
3. Modern MINLP Methods
4. Conclusions & Future Work

Integer Nonlinear Optimization

Sven Leyffer

1. Introduction & Applications
2. Classical MINLP Methods
3. Modern MINLP Methods
4. Conclusions & Future Work

Do not trust this expert!

1. Introduction & Applications

Mixed Integer Nonlinear Programming (MINLP)

$$\left\{ \begin{array}{ll} \underset{x,y}{\text{minimize}} & f(x,y) \\ \text{subject to} & c(x,y) \le 0 \\ & x \in X,\ y \in Y \text{ integer} \end{array} \right.$$

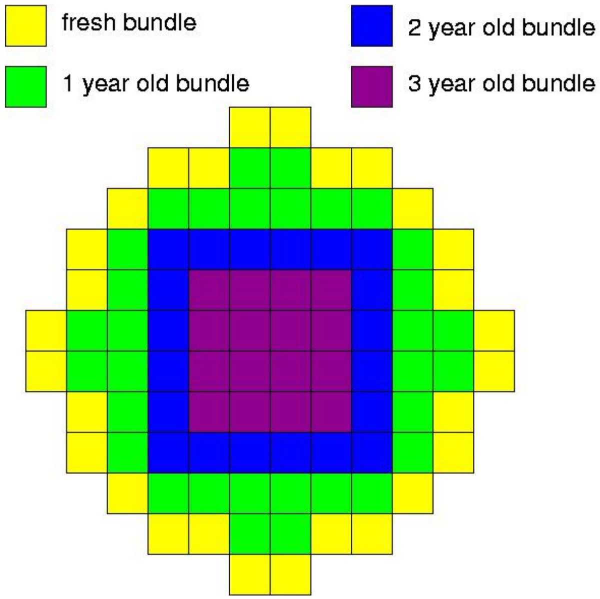
- f, c smooth (convex) functions
- X, Y polyhedral sets, e.g. $Y = \{0, 1\}$
- $y \in Y$ integer $\Rightarrow$ hard problem

1.1. Core Reload Operation [Quist:97]

- maximize reactor efficiency after reload subject to diffusion PDE & safety

- approx. diffusion by nonlinear equation
  ⇒ integer & nonlinear model

- avoid reactor becoming sub-critical

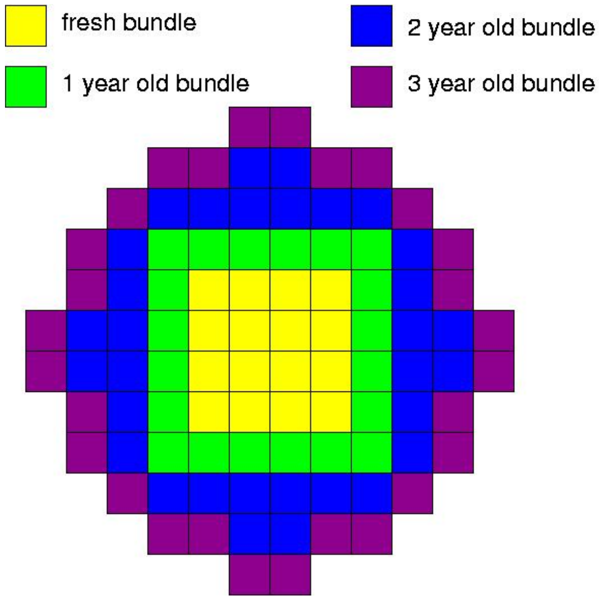


1.1. Core Reload Operation [Quist:97]

- maximize reactor efficiency after reload subject to diffusion PDE & safety

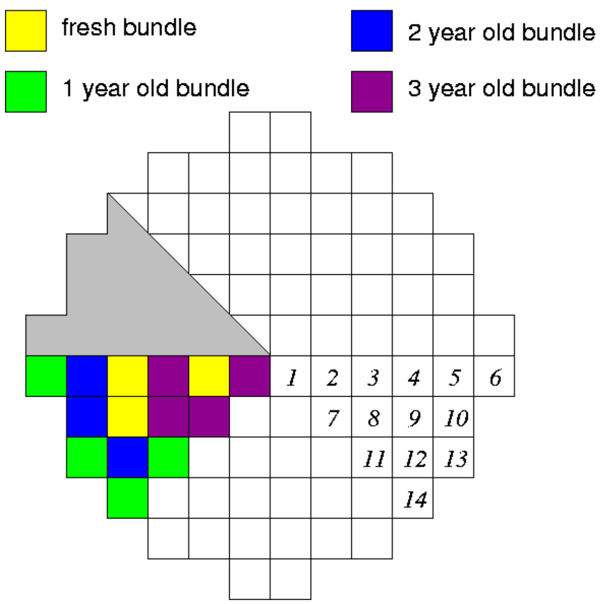
- approx. diffusion by nonlinear equation
  $\Rightarrow$ integer & nonlinear model

- avoid reactor becoming overheated



1.1. Core Reload Operation [Quist:97]

- look for cycles for moving bundles:
  e.g. $4 \quad \rightarrow \quad 6 \quad \rightarrow \quad 8 \quad \rightarrow \quad 10$
  means bundle moved from 4 to 6 to ...
- model with integer variables $x_{ilm} \in \{0,1\}$
  $x_{ilm} = 1$: node i has bundle l of cycle m



1.2. Other Applications

- Chemical Engineering Applications:
  - process synthesis [Kocis&Grossmann:88]
  - batch plant design [Grossmanno&Sargent:79]
  - cyclic scheduling [Jain&Grossmann:98]
  - design of distillation columns [Viswanathan:93]
  - pump configuration optimization [Westerlund:94]
- trimloss minimization in paper industry [Westerlund:98]
- topology optimization [Sigurd:00]
  - finite element structural optimization
  - 0-1 to model presence/absence of material

2. Classical Methods for MINLP

Basic Methods:

1. Branch-and-Bound
2. Outer Approximation, Benders Decomposition et al.

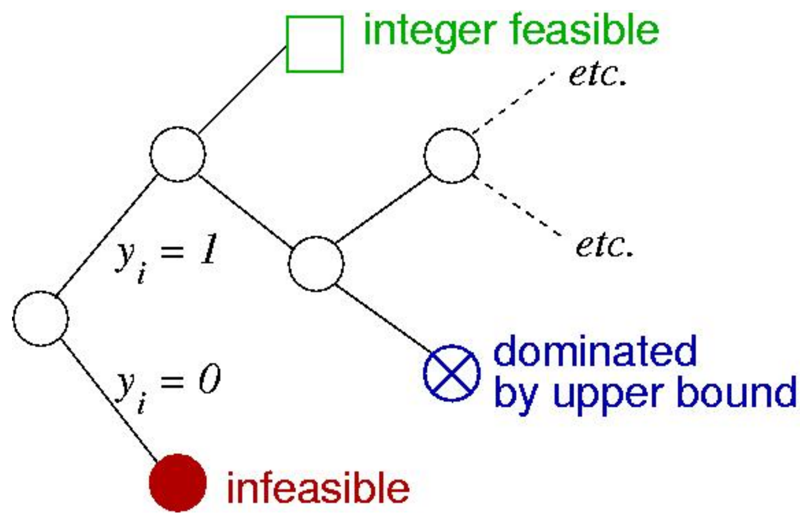
Hybrid Methods:

3. LP/NLP Based Branch-and-Bound
4. Integrating SQP with Branch-and-Bound

2.1. Branch-and-Bound

Solve relaxed NLP ($0 \leq y \leq 1$ continuous relaxation)

- Branch on y_i non-integral
- Solve NLPs & branch until ...
  1. Node infeasible ... ●
  2. Node integer feasible ... □
     $\longrightarrow$ upper bound (U)
  3. Lower bound $\geq U$ ... $\otimes$



Search until no unexplored nodes left on tree

2.2. Outer Approximation [Duran & Grossmann]

Motivation: avoid *huge number* of NLPs

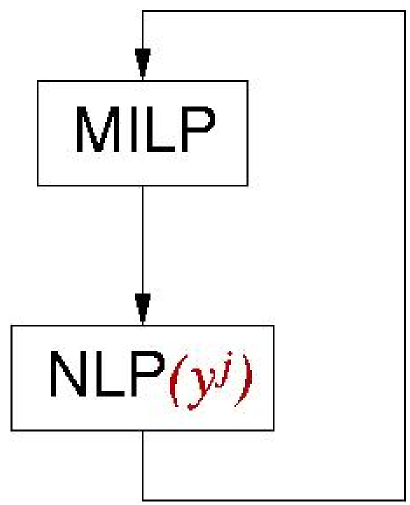
- Take advantage of MILP codes: decompose integer & nonlinear part

Key idea: reformulate MINLP as MILP (implicit)

- Solve alternating sequence of MILP & NLP

NLP subproblem y^j fixed:

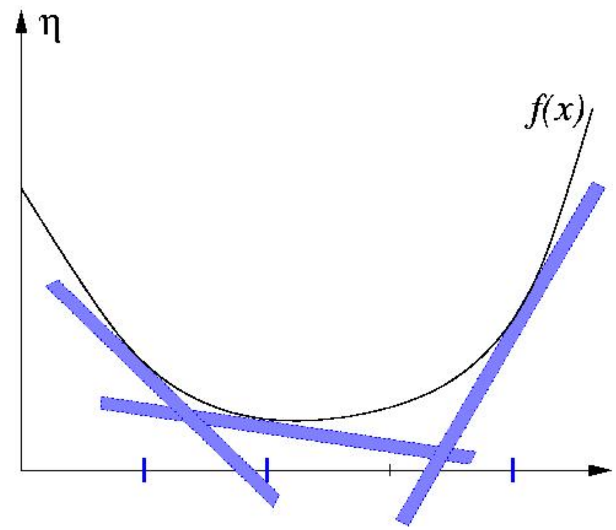
$$\text{NLP}(y^j) \left\{ \begin{array}{ll} \underset{x}{\text{minimize}} & f(x, y^j) \\ \text{subject to} & c(x, y^j) \le 0 \\ & x \in X \end{array} \right.$$



Main Assumption: f, c are convex

2.2. Outer Approximation [Duran & Grossmann]

- let (x^j, y^j) solve NLP(y^j)
- linearize f, c about $(x^j, y^j) =: z^j$
- new objective variable $\eta \geq f(x,y)$
- MINLP $(P) \equiv$ MILP (M)

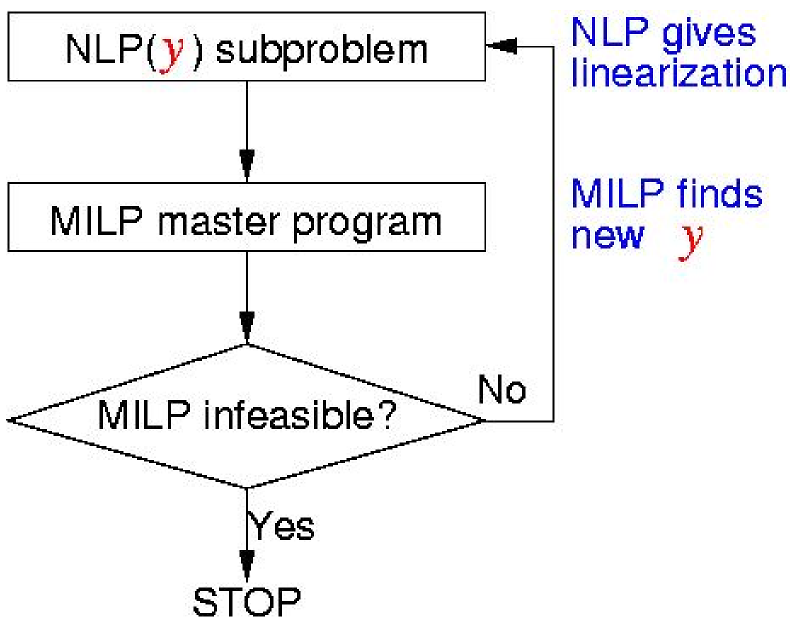


$$(M) \left\{ \begin{array}{lll} \underset{z=(x,y),\eta}{\text{minimize}} & \eta & \\ \text{subject to} & \eta \geq f^j + \nabla f^{j^T}(z - z^j) & \forall y^j \in Y \\ & 0 \geq c^j + \nabla c^{j^T}(z - z^j) & \forall y^j \in Y \\ & x \in X,\ y \in Y \text{ integer} & \end{array} \right.$$

SNAG: need *all* $y^j \in Y$ linearizations

2.2. Outer Approximation [Duran & Grossmann]

(M^k): lower bound (underestimate convex f, c)
NLP(y^j): upper bound U (fixed y^j)



$\Rightarrow$ stop, if lower bound $\geq$ upper bound

2.2. OA & Benders Decomposition

Take OA master ... $z := (x, y)$

$$
(M) \begin{cases}
\underset{z=(x,y),\eta}{\text{minimize}} & \eta \\
\text{subject to} & \eta \geq f^j + \nabla f^{j^T}(z - z^j) \quad \forall y^j \in Y \\
& 0 \geq c^j + \nabla c^{j^T}(z - z^j) \quad \forall y^j \in Y \\
& x \in X,\ y \in Y \text{ integer}
\end{cases}
$$

sum constraints $0 \geq c^j$... weighted with multipliers $\lambda^j \ \forall j$

$$
\Rightarrow \quad \eta \geq f^j + \lambda^{j^T} c^j + \left(\nabla f^j + \nabla c^j \lambda^j \right)^T (z - z^j) \qquad \forall y^j \in Y
$$

... valid inequality.

2.2. OA & Benders Decomposition

Valid inequality from OA master; $z = (x, y)$:

$$\eta \geq f^j + {\lambda^j}^T c^j + \left(\nabla f^j + \nabla c^j \lambda^j\right)^T \left(z - z^j\right)$$

use KKT conditions of NLP(y^j) ...

$$\nabla_x f^j + \nabla_x c^j \lambda^j = 0$$

... to eliminate x components from valid ineqaulity

$$\begin{aligned} \Rightarrow \quad & \eta \geq f^j + {\lambda^j}^T c^j + \left(\nabla_y f^j + \nabla_y c^j \lambda^j\right)^T \left(y - y^j\right) \\ \Leftrightarrow \quad & \eta \geq \mathcal{L}^j + \left(\mu^j\right)^T \left(y - y^j\right) \end{aligned}$$

where $\mathcal{L}^j$ Lagrangian ...

... $\mu^j = \nabla_y f^j + \nabla_y c^j \lambda^j$ multiplier of $y = y^j$ in NLP(y^j)

2.2. OA & Benders Decomposition

$\Rightarrow$ remove x from master problem & obtain Benders master problem

$$(M_B)\left\{\begin{array}{ll}\underset{y,\eta}{\text{minimize}} & \eta \\ \text{subject to} & \eta \geq \mathcal{L}^j + \left(\mu^j\right)^T \left(y - y^j\right) \quad \forall y^j \in Y \\ & y \in Y \text{ integer}\end{array}\right.$$

where $\mathcal{L}^j$ Lagrangian & μ^j multiplier of $y = y^j$ in NLP(y^j)

- (M_B) has less constraints & variables (no x!)
- (M_B) almost ILP (except for η)
- (M_B) weaker than OA (from derivation)

2.2. OA & Similar Methods

Extended Cutting Plane Method [Westerlund:95]:

- no NLP(y^j) solves; Kelley's cutting plane method instead
- linearize about $(\hat{x}^j, y^j)$, solution of (M^k)
- add most violated linearization to master (M^k)

  $\Rightarrow$ slow nonlinear convergence; > 1 evaluation per y

Drawbacks of OA, GBD & ECP:

- MILP tree-search can be bottle-neck
- potentially large number of iterations [FL:94]

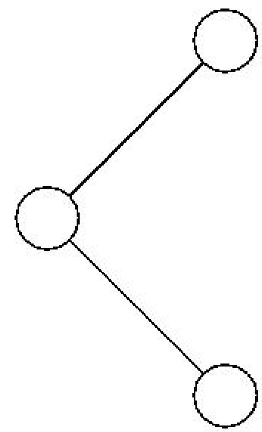
Second order master (MIQP) [FL:94]:

- add Hessian term to MILP (M) $\Rightarrow$ MIQP
  - ∘ solve MIQP by B&B; similar to MILP

2.3. LP/NLP Based Branch-and-Bound [Quesada & Grossmann]

AIM: avoid re-solving MILP master (M)

Consider MILP branch-and-bound



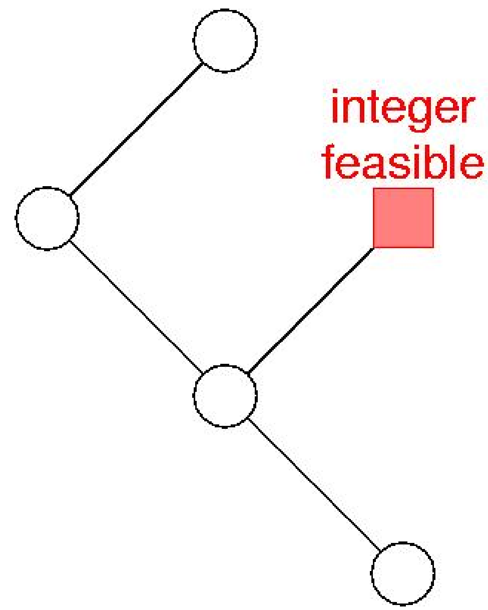
2.3. LP/NLP Based Branch-and-Bound [Quesada & Grossmann]

AIM: avoid re-solving MILP master (M)

Consider MILP branch-and-bound

interrupt MILP, when new y^j found

$\rightarrow$ solve NLP(y^j) get x^j



2.3. LP/NLP Based Branch-and-Bound [Quesada & Grossmann]

AIM: avoid re-solving MILP master (M)

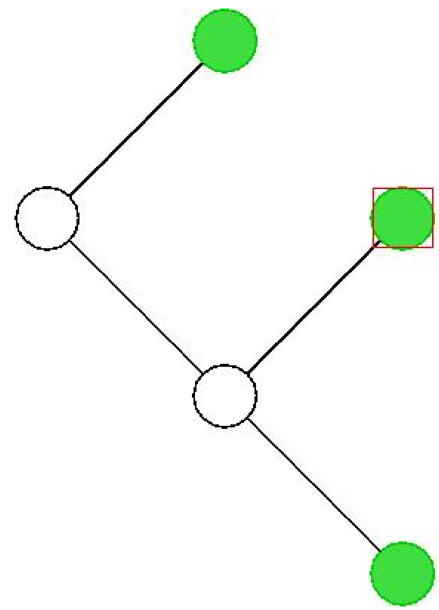
Consider MILP branch-and-bound

interrupt MILP, when new y^j found

$\longrightarrow$ solve NLP(y^j) get x^j

$\longrightarrow$ linearize f, c about (x^j, y^j)

$\longrightarrow$ add linearization to MILP tree



2.3. LP/NLP Based Branch-and-Bound [Quesada & Grossmann]

AIM: avoid re-solving MILP master (M)

Consider MILP branch-and-bound

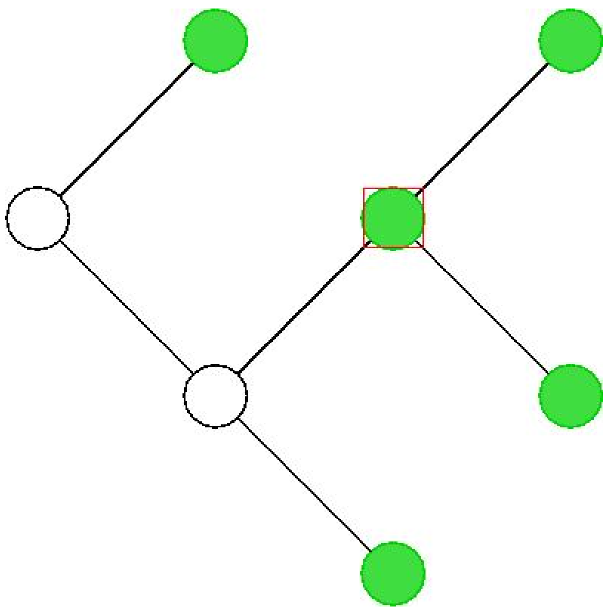
interrupt MILP, when new y^j found

$\longrightarrow$ solve NLP(y^j) get x^j;

$\longrightarrow$ linearize f, c about (x^j, y^j)

$\longrightarrow$ add linearization to MILP tree

$\longrightarrow$ continue MILP tree-search



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

2.3. LP/NLP Based Branch-and-Bound [Quesada & Grossmann]

- need access to MILP solver ... call back
  - exploit good MILP (branch-cut-price) solver
  - [Akrotirianakis&Rustem:00] use Gomory cuts in tree-search

- no commercial implementation of this idea
- preliminary results: order of magnitude faster than OA
  - same number of NLPs, but only one MILP
- similar ideas for Benders & Cutting Plane methods

... see [Quesada/Grossmann:92]

2.4. Integrating SQP & Branch-and-Bound

AIM: Avoid solving NLP node to convergence.

- Sequential Quadratic Programming (SQP)
  $\longrightarrow$ solve sequence (QP^k) at every node

$$(QP^k) \left\{ \begin{array}{ll} \underset{d}{\text{minimize}} & f^k + \nabla f^{k^T} d + \frac{1}{2} d^T W^k d \\ \text{subject to} & c^k + \nabla c^{k^T} d \leq 0 \\ & x^k + d_x \in X, \; y^k + d_y \in \hat{Y}. \end{array} \right.$$

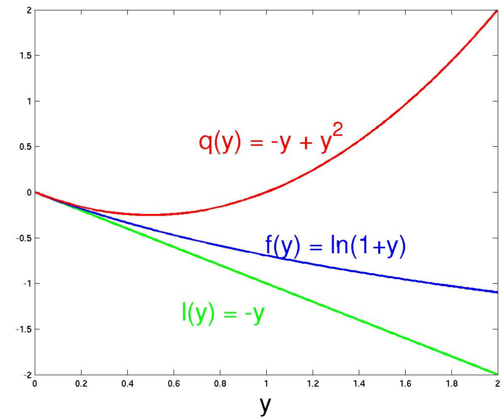
- Early branching rule [Borchers & Mitchell:94]; after QP step:
  $\longrightarrow$ choose non-integral y_i^{k+1} to branch on
  $\longrightarrow$ branch and continue SQP on branch

2.4. Integrating SQP & Branch-and-Bound

SNAG: (QP^k) not lower bound

$\Rightarrow$ no fathoming from upper bound $\Rightarrow$ less efficient B&B

$$\begin{array}{ll} \underset{d}{\text{minimize}} & f^k + \nabla f^{k^T} d + \frac{1}{2} d^T W^k d \\ \text{subject to} & c^k + \nabla c^{k^T} d \le 0 \\ & x^k + d_x \in X, \; y^k + d_y \in \hat{Y}. \end{array}$$

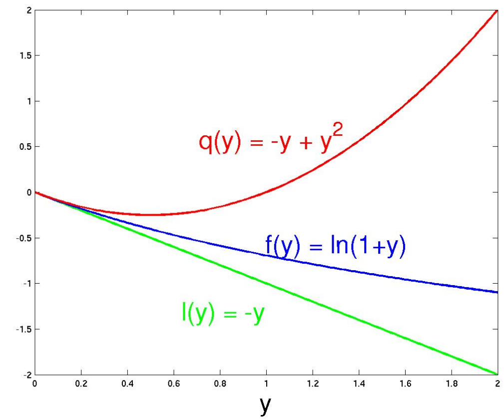


2.4. Integrating SQP & Branch-and-Bound

Snag: (QP^k) not lower bound

$\Rightarrow$ no fathoming from upper bound $\Rightarrow$ less efficient B&B

$$\begin{array}{ll} \underset{d}{\text{minimize}} & f^k + \nabla f^{k^T} d + \frac{1}{2} d^T W^k d \\ \text{subject to} & c^k + \nabla c^{k^T} d \le 0 \\ & x^k + d_x \in X,\; y^k + d_y \in \hat{Y}. \end{array}$$



Remedy: Exploit OA underestimating [L:01]:

- add objective cut $f^k + \nabla f^{k^T} d \le U - \epsilon$ to (QP^k)
- fathom node, if (QP^k) inconsistent

  $\Rightarrow$ convergence for *convex* MINLP

3. Modern Methods for MINLP

1. Branch-and-Cut
   - nonlinear cuts [Stubbs&Mehrotra:99]
   - linear cuts from OA [Akrotirianakis&Rustem:00]

2. Disjunctive Programming [Lee&Grossmann:99]

3. Parallel Tree Search Strategies

3.1. Nonlinear Branch-and-Cut [Mehrotra:99]

Consider MINLP

$$\begin{cases} \underset{x,y}{\text{minimize}} & f_x^T x + f_y^T y \\ \text{subject to} & c(x,y) \le 0 \\ & y \in \{0,1\},\ 0 \le x \le U \end{cases}$$

Linear objective

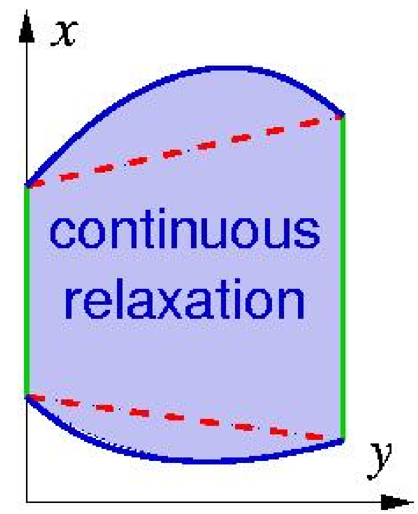
- important to exploit convex hull of constraints
- reformulate nonlinear objectives ...

$$\min \ f(x,y) \qquad \Leftrightarrow \qquad \min \ \eta \ \text{ s.t. } \ \eta \ge f(x,y)$$

3.1. Nonlinear Branch-and-Cut [Mehrotra:99]

Continuous relaxation ($z := (x, y)$):

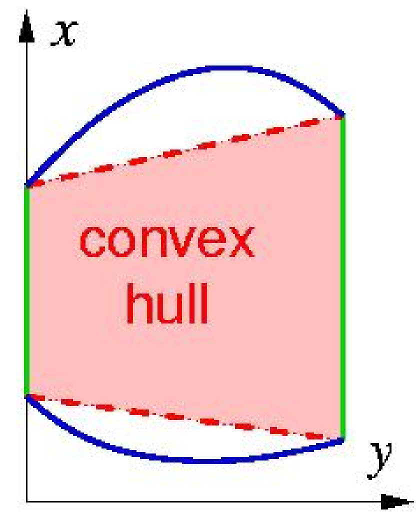
$$
\begin{aligned}
C &:= \{z | c(z) \leq 0,\ 0 \leq y \leq 1,\ 0 \leq x \leq U\} \\
\mathcal{C} &:= \text{conv}(C) \ \text{convex hull}
\end{aligned}
$$



3.1. Nonlinear Branch-and-Cut [Mehrotra:99]

Continuous relaxation ($z := (x, y)$):

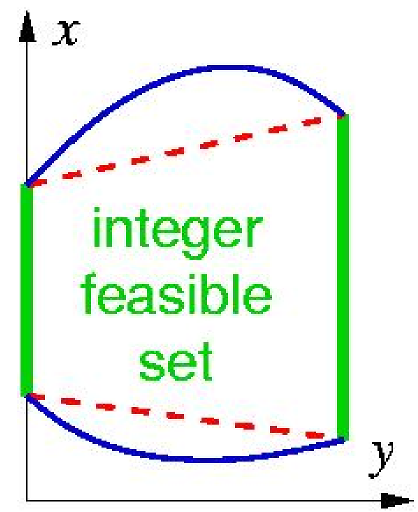
$$
\begin{aligned}
C &:= \{z | c(z) \leq 0,\ 0 \leq y \leq 1,\ 0 \leq x \leq U\} \\
\mathcal{C} &:= \text{conv}(C) \ \text{convex hull}
\end{aligned}
$$



3.1. Nonlinear Branch-and-Cut [Mehrotra:99]

Continuous relaxation ($z := (x, y)$):

$$
\begin{aligned}
C &:= \{z | c(z) \leq 0,\ 0 \leq y \leq 1,\ 0 \leq x \leq U\} \\
\mathcal{C} &:= \text{conv}(C) \ \text{ convex hull} \\
C_j^{0/1} &:= \{z \in C | y_j = 0/1\}
\end{aligned}
$$



$$
\text{let } \mathcal{M}_j(C) := \left\{ \begin{array}{l} z = \lambda_0 u_0 + \lambda_1 u_1 \\ \lambda_0 + \lambda_1 = 1,\ \lambda_0, \lambda_1 \geq 0 \\ u_0 \in C_j^0,\ u_1 \in C_j^1 \end{array} \right\}
$$

$\Rightarrow \mathcal{P}_j(C) :=$ projection of $\mathcal{M}_j(C)$ onto z

$= \text{conv}\,(C \cap y_j \in \{0, 1\})$ and $\mathcal{P}_{1\ldots p}(C) = \mathcal{C}$

3.1. Nonlinear Branch-and-Cut [Mehrotra:99]

Given $\hat{z}$ with $\hat{y}_j \notin \{0,1\}$ find separating hyperplane

$$\Rightarrow \left\{ \begin{array}{ll} \underset{z}{\text{minimize}} & \|z - \hat{z}\|_\infty \\ \text{subject to} & z \in \mathcal{P}_j(C) \end{array} \right.$$

convex reformulation of $\mathcal{M}_j(C)$ with $\mathcal{M}_j(\tilde{C})$, where

$$\tilde{C} := \left\{ (z,\mu) \;\middle|\; \begin{array}{l} \mu c_i(z/\mu) \le 0 \\ 0 \le \mu \le 1 \\ 0 \le x \le \mu U, \ 0 \le y \le \mu \end{array} \right\}$$

where $c(0/0) = 0 \Rightarrow$ convex representation

$\Rightarrow$ separating hyperplane: $\psi^T(z - \hat{z})$, where $\psi \in \partial \|z - \hat{z}\|_\infty$

3.1. Nonlinear Branch-and-Cut [Mehrotra:99]

- at each (?) node of Branch&Bound tree:
  - ◦ generate cutting planes

- generalize disjunctive approach from MILP
  $\Rightarrow$ solve one convex NLP per cut

- generalizes Sherali/Adams and Lovacz/Schrijver cuts
- tighten cuts by adding semi-definite constraint

3.2. Disjunctive Programming [Grossmann]

Consider disjunctive NLP

$$\left\{\begin{array}{ll} \underset{x,Y}{\text{minimize}} & \sum f_k \; + \; f(x) \\ \text{subject to} & \begin{bmatrix} Y_i \\ c_i(x) \le 0 \\ f_i = \gamma_i \end{bmatrix} \bigvee \begin{bmatrix} \neg Y_i \\ B_i x = 0 \\ f_i = 0 \end{bmatrix} \forall i \in I \\ & 0 \le x \le U, \; \Omega(Y) = \text{true}, \; Y \in \{\text{true}, \text{false}\}^p \end{array}\right.$$

Application: porocess synthesis

- Y_i represents presence/absence of units
- $B_i x = 0$ eliminates variables if unit absent

Exploit disjunctive structure

- special branching ... OA/GBD algorithms

3.2. Disjunctive Programming [Grossmann]

Consider disjunctive NLP

$$
\left\{
\begin{array}{ll}
\underset{x,Y}{\text{minimize}} & \sum f_k \; + \; f(x) \\
\text{subject to} & \begin{bmatrix} Y_i \\ c_i(x) \le 0 \\ f_i = \gamma_i \end{bmatrix} \bigvee \begin{bmatrix} \neg Y_i \\ B_i x = 0 \\ f_i = 0 \end{bmatrix} \forall i \in I \\
 & 0 \le x \le U, \; \Omega(Y) = \text{true}, \; Y \in \{\text{true}, \text{false}\}^p
\end{array}
\right.
$$

Big-M formulation (notoriously bad), $M > 0$:

$$c_i(x) \le M(1 - y_i)$$

$$-My_i \le B_i x \le My_i$$

$$f_i = y_i \gamma_i \qquad \Omega(Y) \text{ converted to linear inequalities}$$

3.2. Disjunctive Programming [Grossmann]

Consider disjunctive NLP

$$\left\{\begin{array}{lll} \underset{x,Y}{\text{minimize}} & \sum f_k \ + \ f(x) & \\ \text{subject to} & \begin{bmatrix} Y_i \\ c_i(x) \le 0 \\ f_i = \gamma_i \end{bmatrix} \bigvee \begin{bmatrix} \neg Y_i \\ B_i x = 0 \\ f_i = 0 \end{bmatrix} \forall i \in I & \\ & 0 \le x \le U, \ \Omega(Y) = \text{true}, \ Y \in \{\text{true}, \text{false}\}^p & \end{array}\right.$$

convex hull representation ...

$$x = v_{i1} + v_{i0}, \qquad \lambda_{i1} + \lambda_{i0} = 1$$

$$\lambda_{i1} c_i(v_{i1}/\lambda_{i1}) \le 0, \qquad B_i v_{i0} = 0$$

$$0 \le v_{ij} \le \lambda_{ij} U, \qquad 0 \le \lambda_{ij} \le 1, \qquad f_i = \lambda_{i1}\gamma_i$$

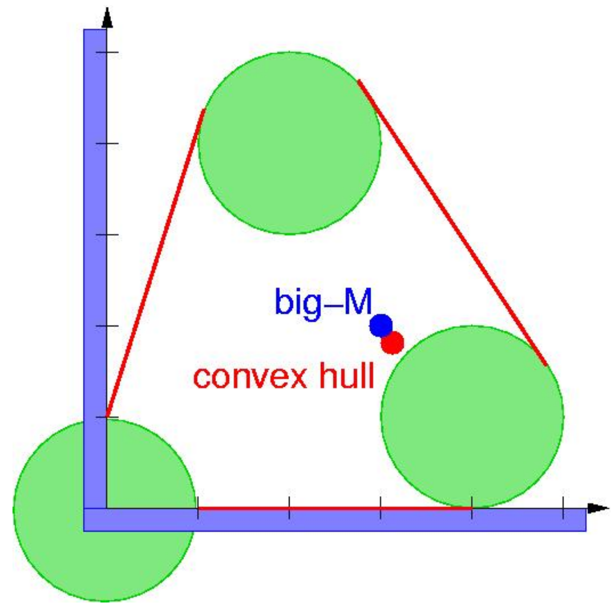
3.2. Disjunctive Programming: Example

$$\begin{bmatrix} Y_1 \\ x_1^2 + x_2^2 \le 1 \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_2 \\ (x_1 - 4)^2 + (x_2 - 1)^2 \le 1 \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_3 \\ (x_1 - 2)^2 + (x_2 - 4)^2 \le 1 \end{bmatrix}$$

$\Rightarrow$



3.2. Disjunctive Programming & MPECs

Consider Fourier Transform Infrared (FTIR) Spectroscopy

Disjunction modeled with large $P_{\max}$ parameter

$$0 \leq P \leq Y P_{\max} \quad Y \in \{0,1\}^{M \times N}$$

Either $P_{i,j} = 0$, or "count" parameter in objective

$$f(P,Y) = \sum e_k^T R^{-1} e_k + 2 \sum Y_{i,j}$$

Alternative model avoids integrality of Y

$$1 \geq Y_{i,j} \quad \perp \quad P_{i,j} \geq 0$$

where $\perp$ means orthogonality, i.e.

$$(1 - Y_{i,j}) P_{i,j} \leq 0 \quad \forall (i,j)$$

$\Rightarrow$ nonlinear constraint ... use NLP solvers (SQP)

3.2. Disjunctive Programming & MPECs

Small FTIR example: initial MPEC solution $f = 25.98$

Progress of MINLP solver

NLPs	lower bound	upper bound
1	8.4	∞
30	8.4	250.0
75	9.9	99.2
100	11.2	26.8
155	12.3	14.0

$\Rightarrow$ MPECs give good upper bound on MINLPs!

$0 \leq y \quad \perp \quad y \leq 1$ **not** always good idea! $\longrightarrow$ need structure ...

3.3. Parallel Branch-and-Bound

meta-computing platforms:

- set of distributed heterogeneous computers, e.g.
  - pool of workstations
  - group of supercomputers or anything

$\Rightarrow$
- low quality with respect to bandwidth, latency, availability
- low cost: *i*t's free !!!
- potentially huge amount of resources

… use *Condor* to "build" MetaComputer

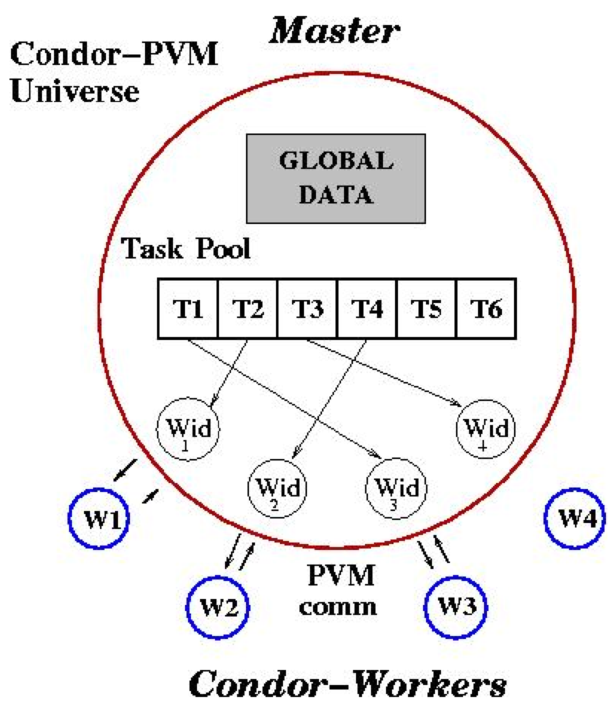
… high-throughput computing

3.3. Parallel Branch-and-Bound

Master Worker Paradigm (`MWdriver`)

Object oriented C++ library

Runs on top of Condor-PVM



Fault tolerance via master *check-pointing*

3.3. Parallel Branch-and-Bound

First Strategy: 1 worker $\equiv$ 1 NLP

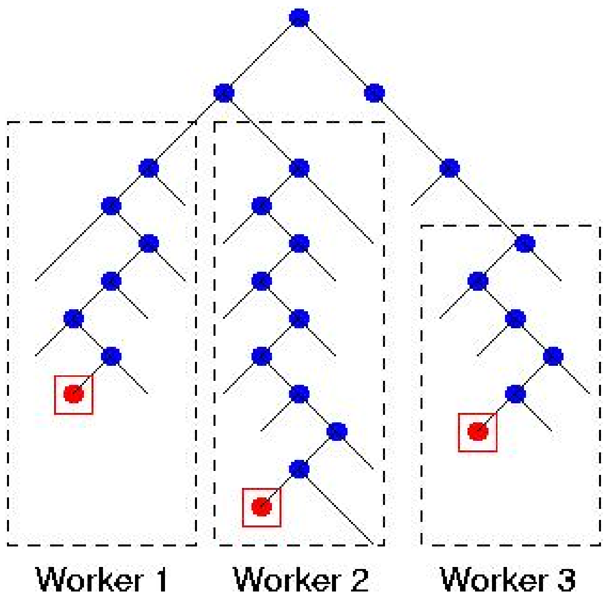
$\Rightarrow$ grain-size *too small*

... NLPs solve in seconds

New Strategy:

1 worker $\equiv$ 1 subtree (MINLP)

... "streamers" running down tree



3.3. Parallel Branch-and-Bound

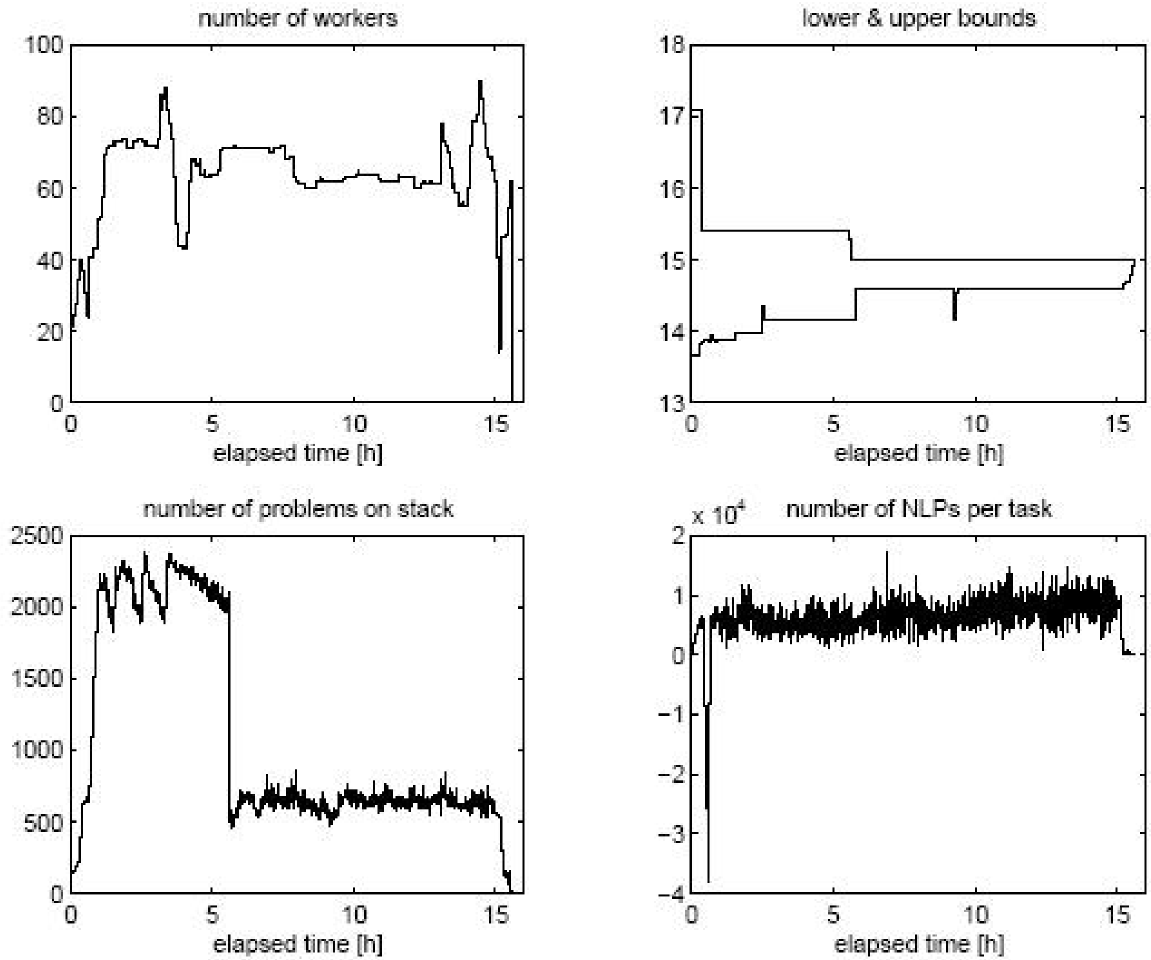
Trimloss optimization with 56 general integers
$\Rightarrow$ solve 96,408 MINLPs on 62.7 workers
$\Rightarrow$ 600,518,018 NLPs

Wall clock time $= 15.5$ hours
Cummulative worker CPU time $= 752.7$ hours $\simeq 31$ days

$$\text{efficiency} := \frac{\text{work-time}}{\text{work} \times \text{job-time}} = \frac{752.7}{62.7 \times 15.5} = 80.5$$

... proportion of time workers were busy

3.3. Parallel Branch-and-Bound: Results



4.1. Conclusions

- MINLP important modeling paradigm; many applications
  - MINLP most used solver on NEOS
- Outer Approximation et al.
  - rely heavily on convexity
  - readily exploit MILP structure in branch-and-cut
- Branch-and-Bound
  - works OK'ish for nonconvex problems (e.g. reload operation)
  - harder to exploit branch-and-cut ideas

4.1. Challenges

- Global solution of nonconvex MINLP, see Mohit's talk
  - automatic code generation for underestimators ($\equiv$ AD)

- Connection to MPECs, recall Stefan's talk
  - generate upper bounds along tree ...
  - global solution of MPECs using branch-and-cut

- PDE constraints & surrogate models
  - e.g. core reload operation
  - multi-model ... trust-regions ...