An Integrated Approach to Truss Structure Design

J. N. Hooker
Tallys Yunes

CPAIOR Workshop on Hybrid Methods for Nonlinear Combinatorial Problems
Bologna, June 2010

How to Solve Nonlinear Combinatorial Problems?

- One answer: basically the same way as other problems.

How to Solve Nonlinear Combinatorial Problems?

- One answer: basically the same way as other problems.
- We will look at the solution approach of **SIMPL**
  - An integrated solver.
  - Combines ideas from MIP, CP, and global optimization.
- Application to **truss structure design**

Outline

- Overview of an integrated solver – SIMPL
- Lagrangean-based domain reduction
- Global optimization in SIMPL
- The truss structure design problem
- Quasi-relaxations
- SIMPL model and computational results

Overview of an integrated solver

- Integration principles.
    - Search-infer-relax
    - Classical solution methods.
- Some references

Integration Principles

- Integrate MILP, constraint programming, global optimization in a **unified** approach.

Integration Principles

- Integrate MILP, constraint programming, global optimization in a **unified** approach.
- **Low-level** integration with **high-level** modeling.

Integration Principles

- Integrate MILP, constraint programming, global optimization in a **unified** approach.
- **Low-level** integration with **high-level** modeling.
- Succinct modeling with **meta-constraints**.
  - Model communicates **problem structure** to the solver.

Integration Principles

- Integrate MILP, constraint programming, global optimization in a **unified** approach.
- **Low-level** integration with **high-level** modeling.
- Succinct modeling with **meta-constraints**.
  - Model communicates **problem structure** to the solver.
- General **search-infer-relax** solution algorithm.
  - Enumerate problem restrictions.
  - Branching or logic-based Benders.
  - Underlying search/inference and search/relaxation dualities.

Integration Principles

- Integrate MILP, constraint programming, global optimization in a **unified** approach.
- **Low-level** integration with **high-level** modeling.
- Succinct modeling with **meta-constraints**.
  - Model communicates **problem structure** to the solver.
- General **search-infer-relax** solution algorithm.
  - Enumerate problem restrictions.
  - Branching or logic-based Benders.
  - Underlying search/inference and search/relaxation dualities
- **Constraint-based** control.
  - Filtering, relaxation, branching.

Search-Infer-Relax

- **Search:** Enumerate problem restrictions.
  - Branching tree nodes, Benders subproblems, local search neighborhoods, etc.
- **Infer:** Deduce constraints from current restriction
  - Nogoods, cutting planes, filtering, etc.
- **Relax:** Solve relaxation of current restriction
  - LP, Lagrangean, domain store, Benders master, etc.

Classical solution methods

- CP solver
  - **Search:** Branching
  - **Inference:** FIltering
  - **Relaxation:** Domain store

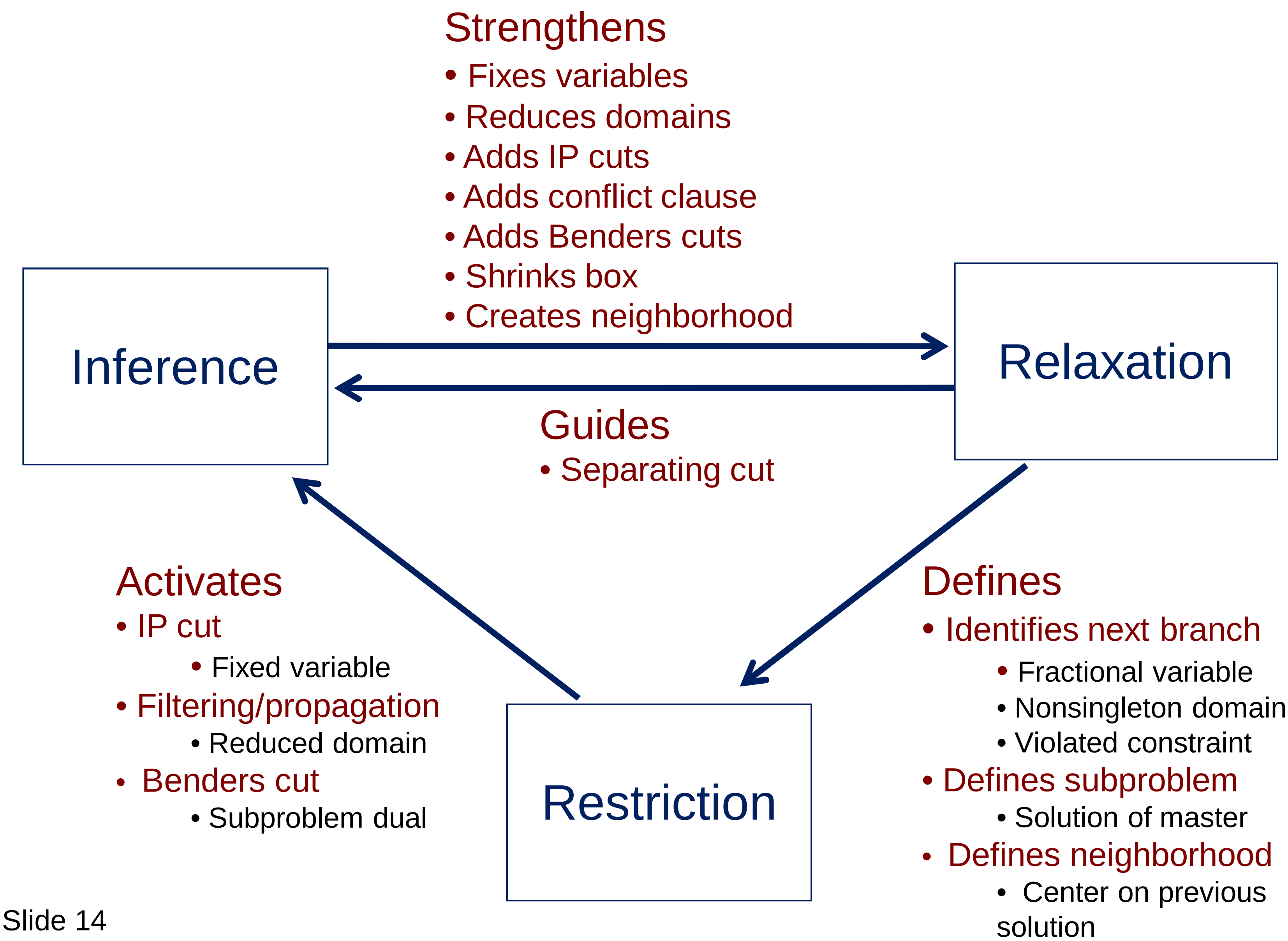
- MILP solver
  - **Search:** Branching
  - **Inference:** Cutting planes, presolve, reduced cost variable fixing
  - **Relaxation:** LP

- Benders
  - **Search:** Enumerate subproblems.
  - **Inference:** Benders cuts
  - **Relaxation:** Master problem

Classical solution methods

- Global optimization
  - **Search:** Enumerate boxes
  - **Inference:** Domain reduction, dual-based variable bounding
  - **Relaxation:** Convexification
- SAT
  - **Search:** Branching
  - **Inference:** Conflict clauses
  - **Relaxation:** Same as restriction
- Local search
  - **Search:** Enumerate neighborhoods.
  - **Inference:** Tabu list, etc.
  - **Relaxation:** Same as restriction

Interaction



Some references

- Basic Framework
  - JNH. **Logic-based methods for optimization**, *CP 1994*
  - JNH, M. A. Osorio. **Mixed logical/linear programming**, *Discrete Applied Mathematics 1999.*
  - JHH, G. Ottosson, E. S. Thorsteinsson, H.-J. Kim. **On integrating CP and LP for combinatorial optimization**. *AAAI 1999.*
  - G. Ottosson, E. Thorsteinsson, JNH. **Mixed global constraints and inference in hybrid IP-CLP solvers**. *CP 1999*
  - JNH, G. Ottosson, E. S. Thorsteinsson, H.-J. Kim. **A scheme for unifying optimization and constraint satisfaction methods**. *Knowledge Engineering Review, 2000.*

Some references

- JNH. **A framework for integrating solution methods**. *ICS 2003.*
- I. Aron, JNH, T. H. Yunes. **SIMPL: A system for integrating**
- **optimization techniques**. *CPAIOR 2004.*
- T. Yunes, I. Aron, JNH. **An integrated solver for optimization problems**. *Operations Research, to appear.*

Some references

- Search
  - JNH. **Unifying local and exhaustive search**. *ENC 2005.*
  - JNH, **A search-infer-and-relax framework for integrating solution methods**. *CPAIOR 2005*
- Modeling
  - JNH, H.-J. Kim, G. Ottosson. **A declarative modeling framework that integrates solution methods**. *Annals of OR* 2001.
  - JNH. **Hybrid modeling**. *Ten Years of CPAIOR,* to appear.

Some references

- Theory and background
  - JNH. ***Logic-Based Methods for Optimization.*** 2000.
  - JNH. **Logic, optimization and constraint programming**. *INFORMS Journal on Computing, 2002.*
  - A. Bockmayr, JNH. **Constraint programming**. *Handbook of Discrete Optimization,* 2005.
  - JNH. **Operations research methods in constraint programming**, *Handbook of Constraint Programming, 2006.*
  - J. N. Hooker. **Duality in optimization and constraint satisfaction**. *CPAIOR 2006*
  - JNH. ***Integrated Methods for Optimization****. 2007.*

Some references

- Nogood-based search (Logic-based Benders)

  - JNH, H. Yan. **Logic circuit verification by Benders decomposition**. CP 1995.
  - JNH. ***Logic-Based Methods for Optimization.*** 2000.
  - E. Thorsteinsson. **Branch and check: A hybrid framework for integrating MIP and CLP**. *CP 2001.*
  - JNH, G. Ottosson. **Logic-based Benders decomposition.** *Mathematical Programming, 2003.*
  - JNH. **A hybrid method for planning and scheduling**. *Constraints 2005.*
  - JNH. **Planning and scheduling to minimize tardiness**. *CP 2005.*
  - JNH. **Planning and scheduling by logic-based Benders decomposition**. *Operations Research,* 2007.

Some references

- Relaxation methods
  - I. E. Grossmann, JNH, R. Raman, H. Yan. **Logic cuts for processing networks with fixed charges**. *Computers and OR,* 1994.
  - E. Thorsteinsson and G. Ottosson. **Linear relaxations and reduced-cost based propagation of continuous variable subscripts**. *Annals of OR, 2001.*
  - S. Bollapragada, O. Ghattas, JNH. **Optimal design of truss structures by mixed logical and linear programming**. *Operations Research* 2001.
  - JNH. **Convex programming methods for global optimization**, *COCOS 2003*
  - L. Genc-Kaya and JNH. **The circuit polytope**. 2008.

Nonlinear Domain Filtering

Suppose we have a **continuous relaxation** of an optimization problem :

$$\begin{aligned} &\min \ f(x) \\ &g(x) \geq 0 \\ &x \in S \end{aligned}$$

Nonlinear Domain Filtering

Suppose we have a **continuous relaxation** of an optimization problem :

$$\begin{aligned} &\min\ f(x) \\ &g(x) \geq 0 \\ &x \in S \end{aligned}$$

A **Lagrangean relaxation** dualizes the inequality constraints and provides a lower bound on the optimal value:

$$\theta(\lambda) = \min_{x \in S} \left\{ f(x) + \lambda^T g(x) \right\}$$

Nonlinear Domain Filtering

Suppose we have a **continuous relaxation** of an optimization problem :

$$\begin{aligned} &\min \ f(x) \\ &g(x) \geq 0 \\ &x \in S \end{aligned}$$

A **Lagrangean relaxation** dualizes the inequality constraints and provides a lower bound on the optimal value:

$$\theta(\lambda) = \min_{x \in S} \left\{ f(x) + \lambda^T g(x) \right\}$$

The **Lagrangean dual** finds the tightest lower bound:

$$\max_{\lambda \geq 0} \left\{ \theta(\lambda) \right\}$$

Can be solved by subgradient optimization, etc.

Nonlinear Domain Filtering

Suppose we have a **continuous relaxation** of an optimization problem :

$$\begin{aligned} &\min\ f(x) \\ &g(x) \geq 0 \\ &x \in S \end{aligned}$$

Suppose it has optimal solution x^*, optimal value v^*, and optimal Lagrangean dual solution λ^*.

Nonlinear Domain Filtering

Suppose we have a **continuous relaxation** of an optimization problem :

$$\begin{aligned} &\min\ f(x) \\ &g(x) \geq 0 \\ &x \in S \end{aligned}$$

Suppose it has optimal solution x^*, optimal value v^*, and optimal Lagrangean dual solution λ^*.

…and $\lambda_i^* > 0$, which means the i-th constraint is tight (complementary slackness);

Nonlinear Domain Filtering

Suppose we have a **continuous relaxation** of an optimization problem :

$$\begin{aligned} &\min\ f(x) \\ &g(x) \geq 0 \\ &x \in S \end{aligned}$$

Suppose it has optimal solution x^*, optimal value v^*, and optimal Lagrangean dual solution λ^*.

…and $\lambda_i^* > 0$, which means the i-th constraint is tight (complementary slackness);

…and we have a feasible solution of the original problem with value U, so that U is an upper bound on the optimal value.

Supposing $\begin{array}{l} \min\ f(x) \\ g(x) \geq 0 \\ x \in S \end{array}$ has optimal solution x^*, optimal value v^*, and optimal Lagrangean dual solution λ^*:

If x were to change to a value other than x^*, the LHS of i-th constraint $g_i(x) \geq 0$ would change by some amount Δ_i.

Since the constraint is tight, this would increase the optimal value as much as changing the constraint to $g_i(x) - \Delta_i \geq 0$.

So it would increase the optimal value at least $\lambda_i^* \Delta_i$.

Supposing

$$\begin{aligned} &\min\ f(x) \\ &g(x) \ge 0 \\ &x \in S \end{aligned}$$

has optimal solution x^*, optimal value v^*, and optimal Lagrangean dual solution λ^*:

We have found: a change in x that changes $g_i(x)$ by Δ_i increases the optimal value at least $\lambda_i^* \Delta_i$.

Since optimal value of relaxation $\le$ true optimal value $\le U$, we have $\lambda_i^* \Delta_i \le U - v^*$, or

$$\Delta_i \le \frac{U - v^*}{\lambda_i^*}$$

Supposing
$$\begin{aligned} \min\ & f(x) \\ & g(x) \ge 0 \\ & x \in S \end{aligned}$$
has optimal solution x^*, optimal value v^*, and optimal Lagrangean dual solution λ^*:

We have found: a change in x that changes $g_i(x)$ by Δ_i increases the optimal value at least $\lambda_i^*\Delta_i$.

Since optimal value of relaxation $\le$ true optimal value $\le U$, we have $\lambda_i^*\Delta_i \le U - v^*$, or

$$\Delta_i \le \frac{U - v^*}{\lambda_i^*}$$

Since $\Delta_i = g_i(x) - g_i(x^*) = g_i(x)$, this implies the inequality

$$g_i(x) \le \frac{U - v^*}{\lambda_i^*}$$

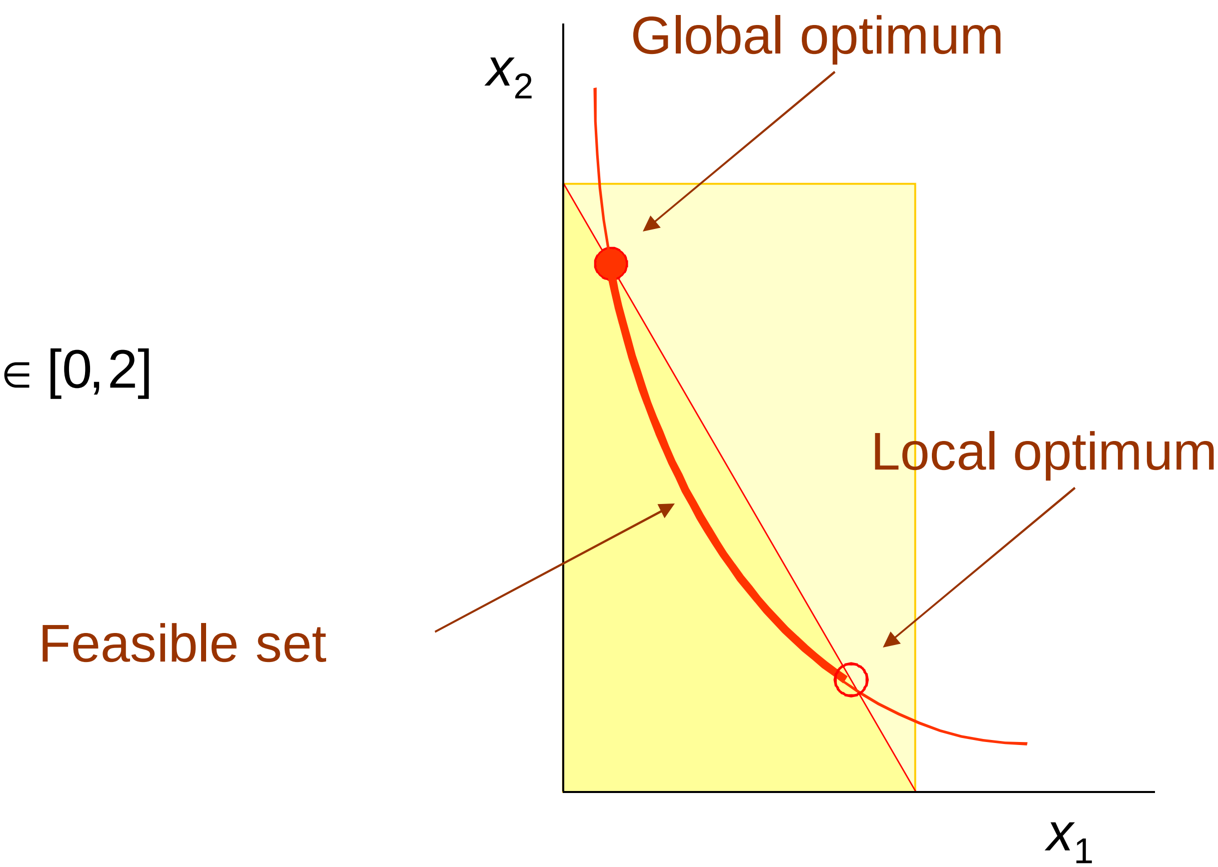
…which can be propagated to reduce domains.

Global Optimization in SIMPL

- Combine OR-style relaxation with CP-style interval arithmetic and domain filtering.
  - This much is similar to some other global solvers.
- Also some additional features:
  - Lagrangean-based propagation (just described)
  - Branching on noninteger discrete variables
  - Convex quasi-relaxations

Global Optimization in SIMPL

$$\begin{aligned}
&\max \;\; x_1 + x_2 \\
&4x_1x_2 = 1 \\
&2x_1 + x_2 \le 2 \\
&x_1 \in [0,1], \;\; x_2 \in [0,2]
\end{aligned}$$

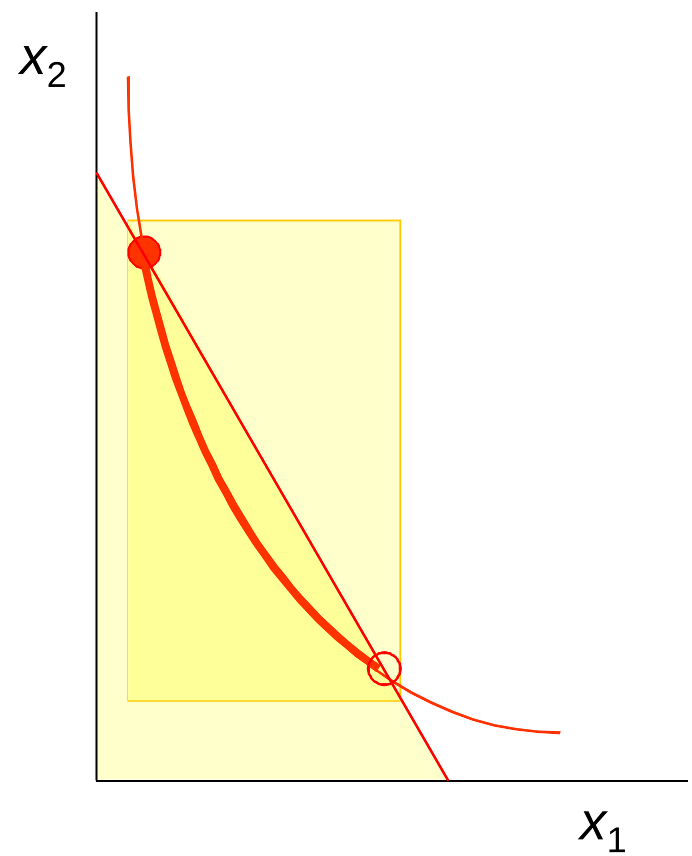


To solve it:

- Search: split interval domains of x_1, x_2.
  - Each node of search tree is a problem restriction.
- Propagation: Interval propagation, domain filtering.
  - Use Lagrange multipliers to infer valid inequality for propagation.
  - Reduced-cost variable fixing is a special case.
- Relaxation: Use function factorization to obtain linear continuous relaxation.

Interval propagation

Propagate intervals
[0,1], [0,2]
through constraints
to obtain
[1/8,7/8], [1/4,7/4]



Relaxation (function factorization)

Factor complex functions into elementary functions that have known linear relaxations (**McCormick factors**).

Write $4x_1x_2 = 1$ as $4y = 1$ where $y = x_1x_2$.

This factors $4x_1x_2$ into linear function $4y$ and bilinear function x_1x_2.

Linear function $4y$ is its own linear relaxation.

Relaxation (function factorization)

Factor complex functions into elementary functions that have known linear relaxations (**McCormick factors**).

Write $4x_1x_2 = 1$ as $4y = 1$ where $y = x_1x_2$.

This factors $4x_1x_2$ into linear function $4y$ and bilinear function x_1x_2.

Linear function $4y$ is its own linear relaxation.

Bilinear function $y = x_1x_2$ has relaxation:

$$\underline{x}_2x_1 + \underline{x}_1x_2 - \underline{x}_1\underline{x}_2 \le y \le \underline{x}_2x_1 + \bar{x}_1x_2 - \bar{x}_1\underline{x}_2$$
$$\bar{x}_2x_1 + \bar{x}_1x_2 - \bar{x}_1\bar{x}_2 \le y \le \bar{x}_2x_1 + \underline{x}_1x_2 - \underline{x}_1\bar{x}_2$$

where domain of x_j is $[\underline{x}_j, \bar{x}_j]$

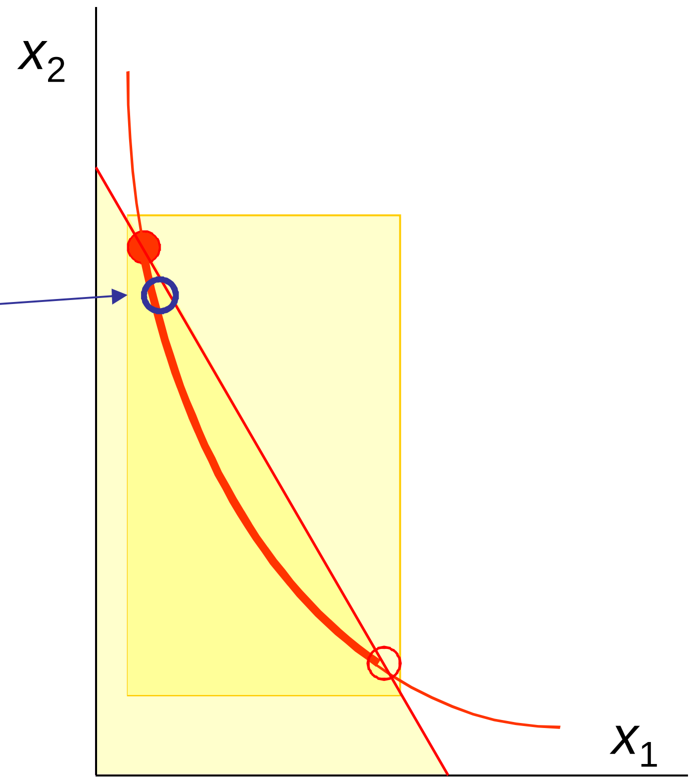
Relaxation (function factorization)

The linear relaxation becomes:

$$
\begin{aligned}
&\min\ x_1 + x_2 \\
&4y = 1 \\
&2x_1 + x_2 \le 2 \\
&\underline{x}_2 x_1 + \underline{x}_1 x_2 - \underline{x}_1 \underline{x}_2 \le y \le \underline{x}_2 x_1 + \overline{x}_1 x_2 - \overline{x}_1 \underline{x}_2 \\
&\overline{x}_2 x_1 + \overline{x}_1 x_2 - \overline{x}_1 \overline{x}_2 \le y \le \overline{x}_2 x_1 + \underline{x}_1 x_2 - \underline{x}_1 \overline{x}_2 \\
&\underline{x}_j \le x_j \le \overline{x}_j,\ \ j = 1,2
\end{aligned}
$$

Relaxation (function factorization)

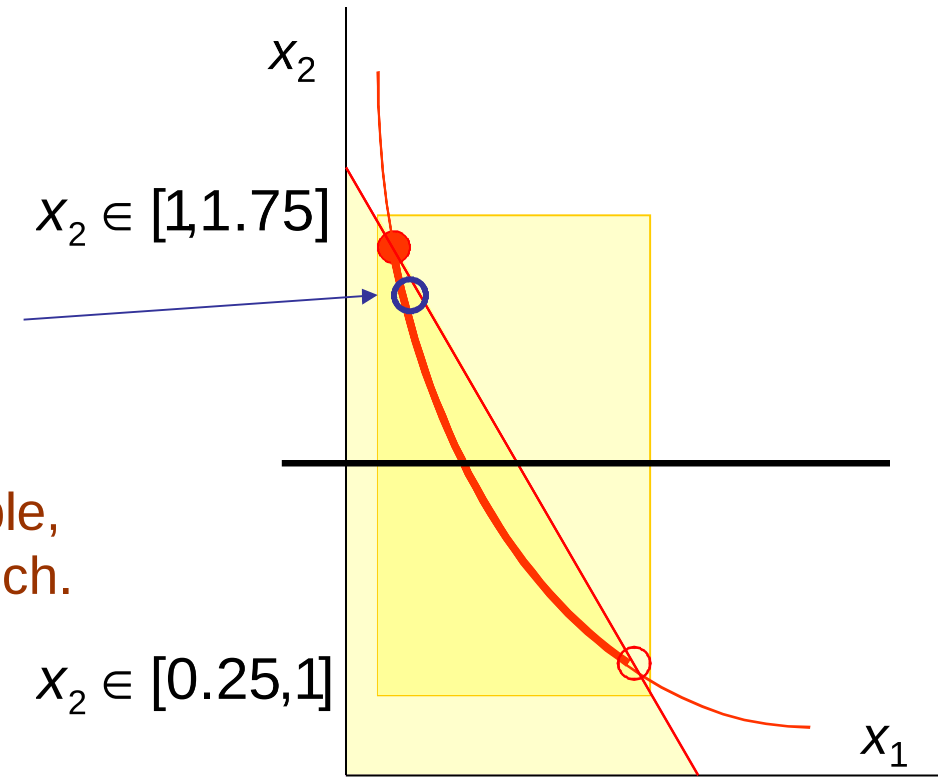
Solve linear relaxation.

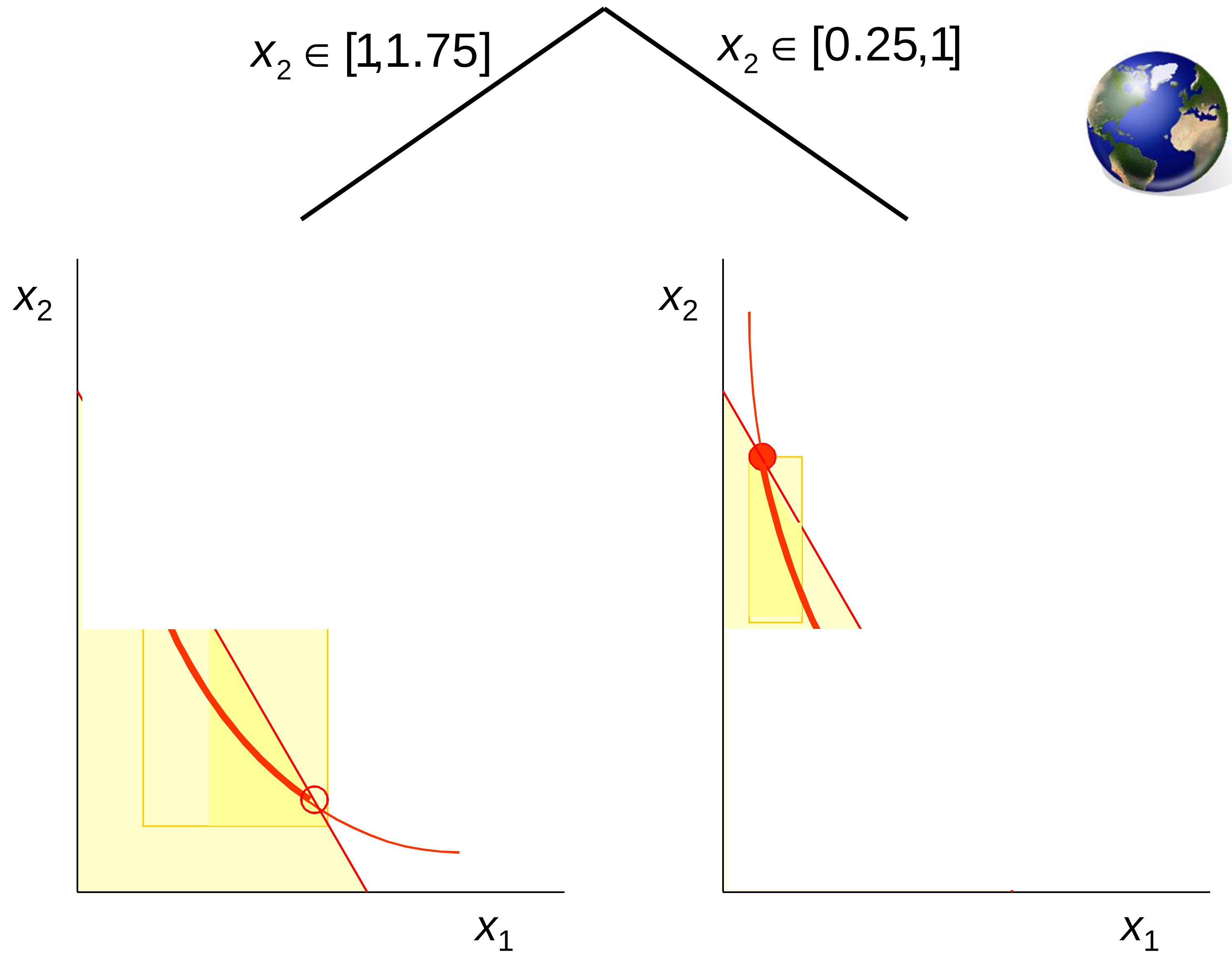


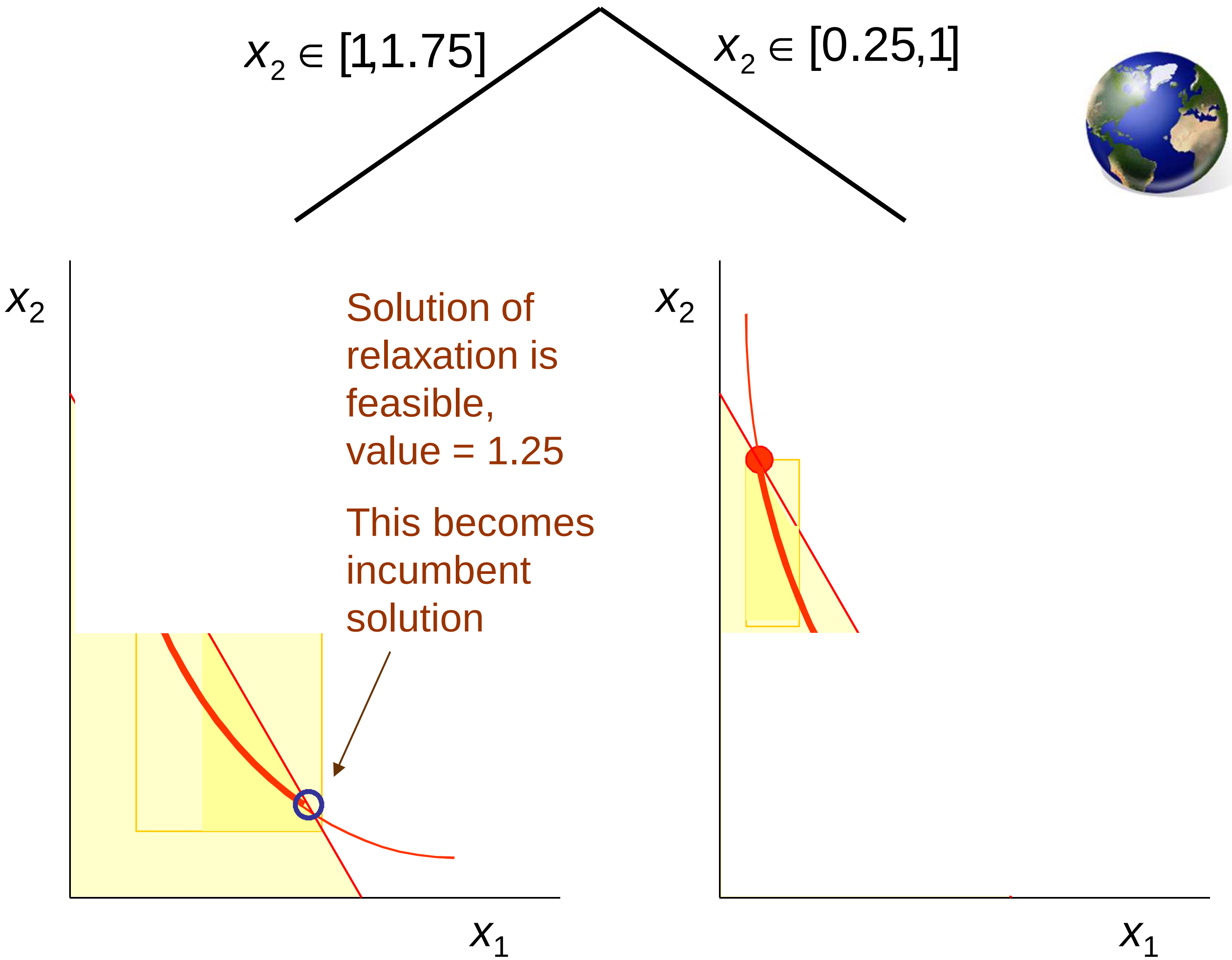
Relaxation (function factorization)

Solve linear relaxation.

Since solution is infeasible, split an interval and branch.



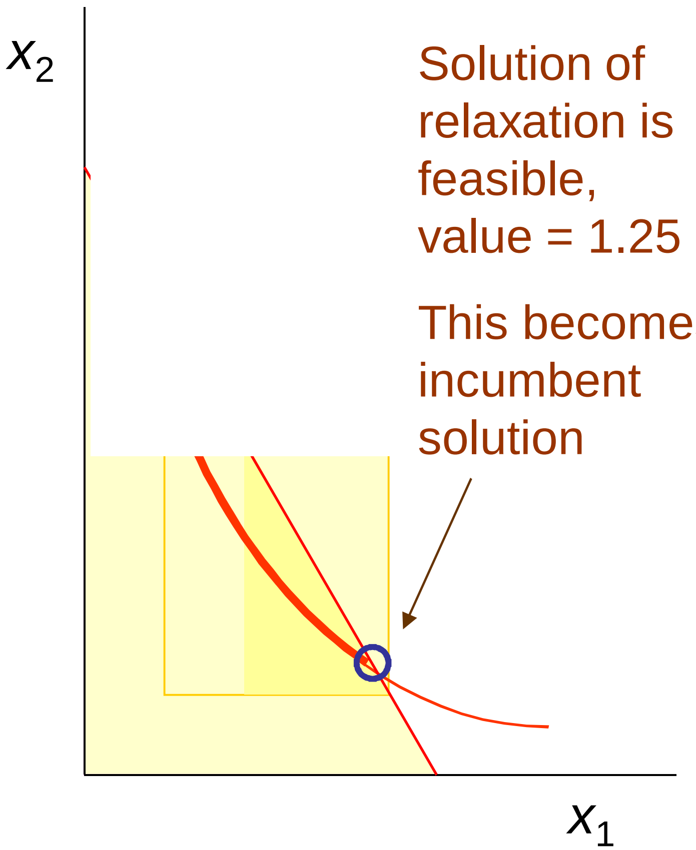

$x_2 \in [1,1.75]$
$x_2 \in [0.25,1]$
x_2
x_1
x_2
x_1



Solution of relaxation is feasible, value = 1.25

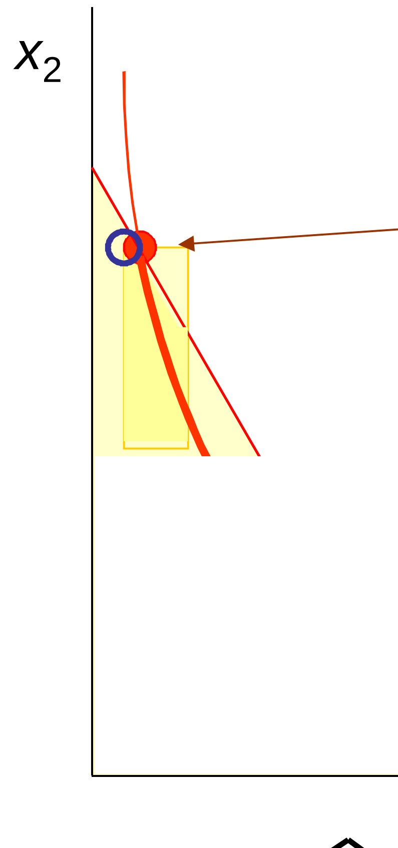
This becomes incumbent solution

$x_2 \in [1,1.75]$ $x_2 \in [0.25,1]$



Solution of relaxation is feasible, value = 1.25

This becomes incumbent solution



Solution of relaxation is not quite feasible, value = 1.854

Also use Lagrange multipliers for domain filtering…

Relaxation (Lagrange multipliers)

$$
\begin{aligned}
&\min\ x_1 + x_2 \\
&4y = 1 \\
&2x_1 + x_2 \le 2 \\
&\underline{x}_2 x_1 + \underline{x}_1 x_2 - \underline{x}_1 \underline{x}_2 \le y \le \underline{x}_2 x_1 + \bar{x}_1 x_2 - \bar{x}_1 \underline{x}_2 \\
&\bar{x}_2 x_1 + \bar{x}_1 x_2 - \bar{x}_1 \bar{x}_2 \le y \le \bar{x}_2 x_1 + \underline{x}_1 x_2 - \underline{x}_1 \bar{x}_2 \\
&\underline{x}_j \le x_j \le \bar{x}_j, \quad j = 1,2
\end{aligned}
$$

Associated Lagrange multiplier in solution of relaxation is $\lambda_2 = 1.1$

Relaxation (Lagrange multipliers)

$$\min \ x_1 + x_2$$
$$4y = 1$$
$$2x_1 + x_2 \le 2$$
$$\underline{x}_2 x_1 + \underline{x}_1 x_2 - \underline{x}_1 \underline{x}_2 \le y \le \underline{x}_2 x_1 + \overline{x}_1 x_2 - \overline{x}_1 \underline{x}_2$$
$$\overline{x}_2 x_1 + \overline{x}_1 x_2 - \overline{x}_1 \overline{x}_2 \le y \le \overline{x}_2 x_1 + \underline{x}_1 x_2 - \underline{x}_1 \overline{x}_2$$
$$\underline{x}_j \le x_j \le \overline{x}_j, \quad j = 1,2$$

Associated Lagrange multiplier in solution of relaxation is $\lambda_2 = 1.1$

This yields a valid inequality for propagation:

$$2x_1 + x_2 \ge 2 - \frac{\boxed{1.854} - \boxed{1.25}}{\boxed{1.1}} = 1.451$$

Value of relaxation

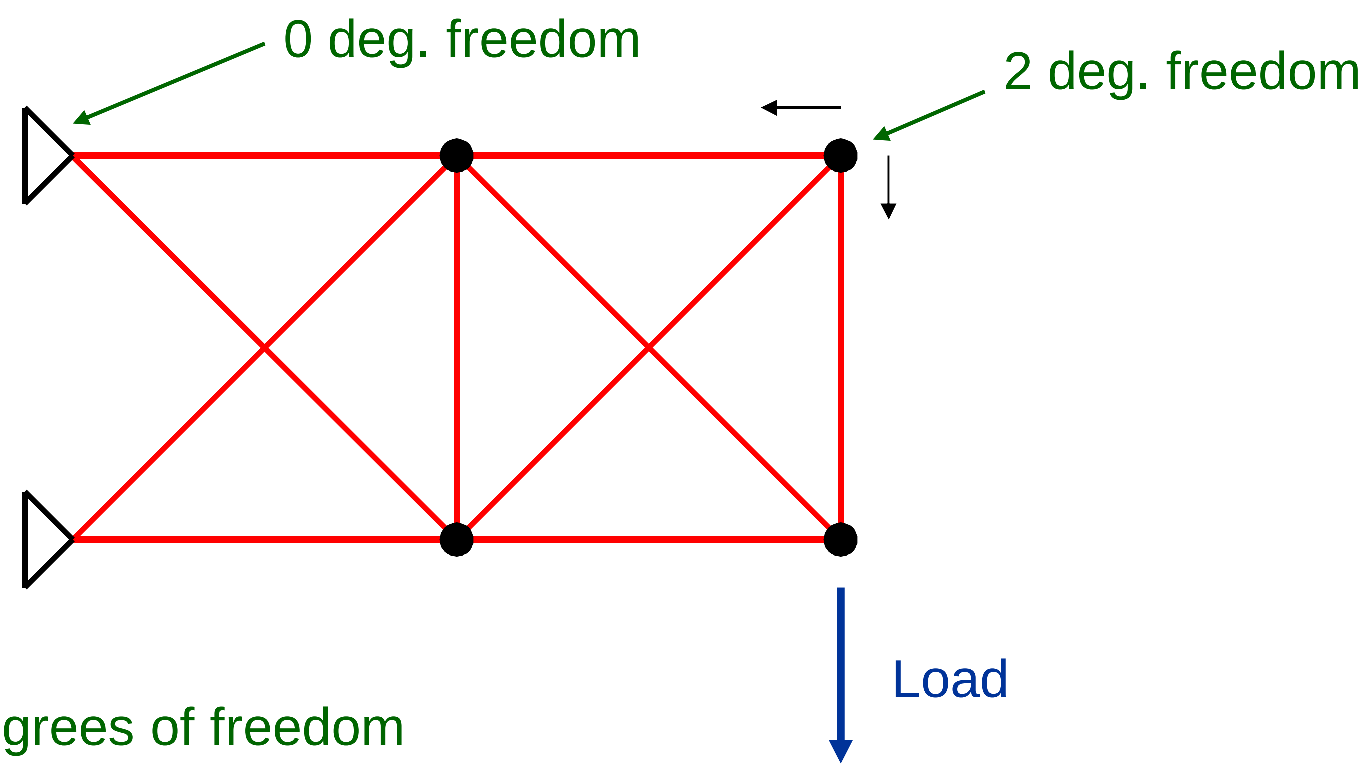
Lagrange multiplier

Value of incumbent solution

Truss Structure Design

Select size of each bar (possibly zero) to support the load while minimizing weight. Bar sizes are **discrete**.

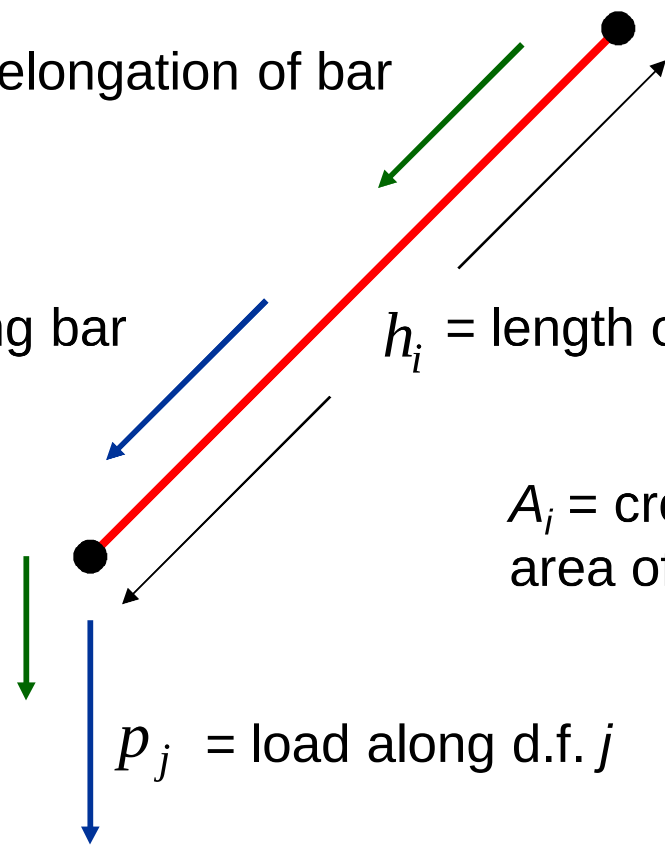
10-bar cantilever truss



Total 8 degrees of freedom

Truss Structure Design

Notation



v_i = elongation of bar

s_i = force along bar

h_i = length of bar i

A_i = cross-sectional area of bar

d_j = node displacement

p_j = load along d.f. j

Truss Structure Design

$$\min \quad \sum_i h_i A_i \qquad \text{\} Minimize total weight}$$

$$s.t. \quad \sum_i \cos\theta_{ij} s_i = p_j, \text{ all } j \qquad \text{\} Equilibrium}$$

$$\sum_j \cos\theta_{ij} d_j = v_i, \text{ all } i \qquad \text{\} Compatibility}$$

nonlinear

$$\frac{E_i}{h_i} A_i v_i = s_i, \text{ all } i \qquad \text{\} Hooke's law}$$

$$v_i^L \le v_i \le v_i^U, \text{ all } i \qquad \text{\} Elongation bounds}$$

$$d_j^L \le d_j \le d_j^U, \text{ all } j \qquad \text{\} Displacement bounds}$$

$$\bigvee_k (A_i = A_{ik}) \qquad \text{\} Logical disjunction}$$

Area must be one of several discrete values A_{ik}

Constraints can be imposed for multiple loading conditions

Truss Structure Design

Introducing new variables linearizes the problem but makes it much larger.

MILP model

$$
\begin{aligned}
\min \quad & \sum_i h_i \sum_k A_{ik} y_{ik} \\
s.t. \quad & \sum_i \cos\theta_{ij} s_i = p_j, \text{ all } j \\
& \sum_j \cos\theta_{ij} d_j = \sum_k v_{ik}, \text{ all } i \\
& \frac{E_i}{h_i} \sum_k A_{ik} v_{ik} = s_i, \text{ all } i \\
& v_i^L \le v_i \le v_i^U, \text{ all } i \\
& d_j^L \le d_j \le d_j^U, \text{ all } j \\
& \sum_k y_{ik} = 1, \text{ all } i
\end{aligned}
$$

0-1 variables indicating size of bar *i*

Elongation variable disaggregated by bar size

Hooke's law becomes linear

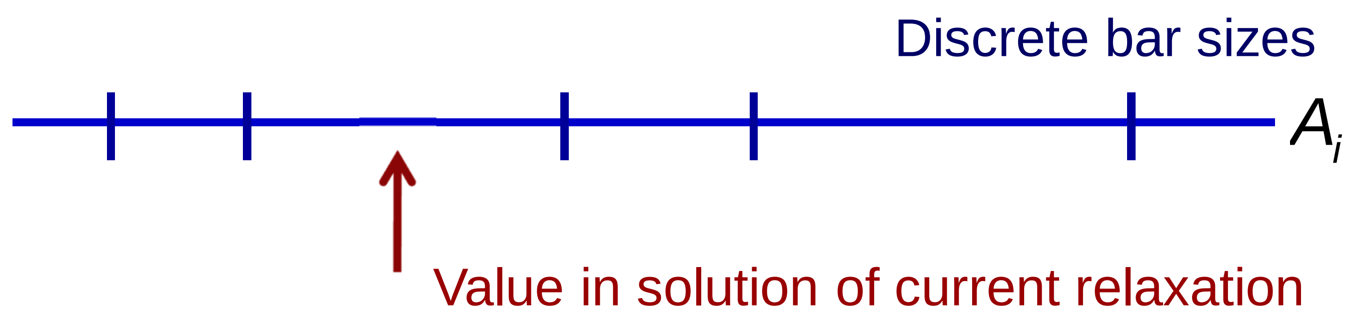
Truss Structure Design

Integrated approach

- Use the **original model** (don’t introduce new variables)
- Branch by **splitting** the range of areas A_i
  - No 0-1 or integer variables!
- Generate a linear **quasi-relaxation**, which is much smaller than MILP model.
- Use **logic cuts**.

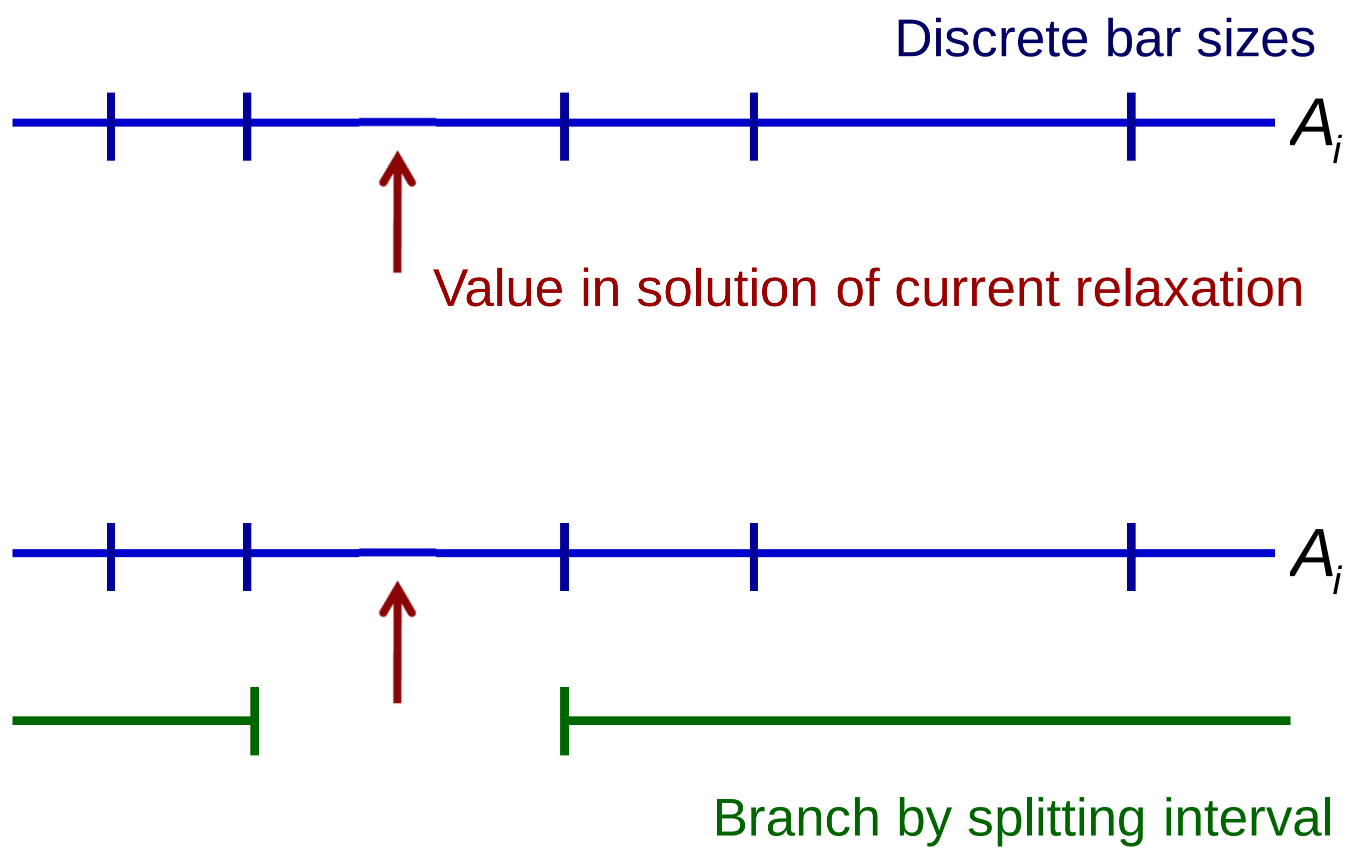
Original hand-coded method: Bollapragada, Ghattas, and JNH 2001.

Branching

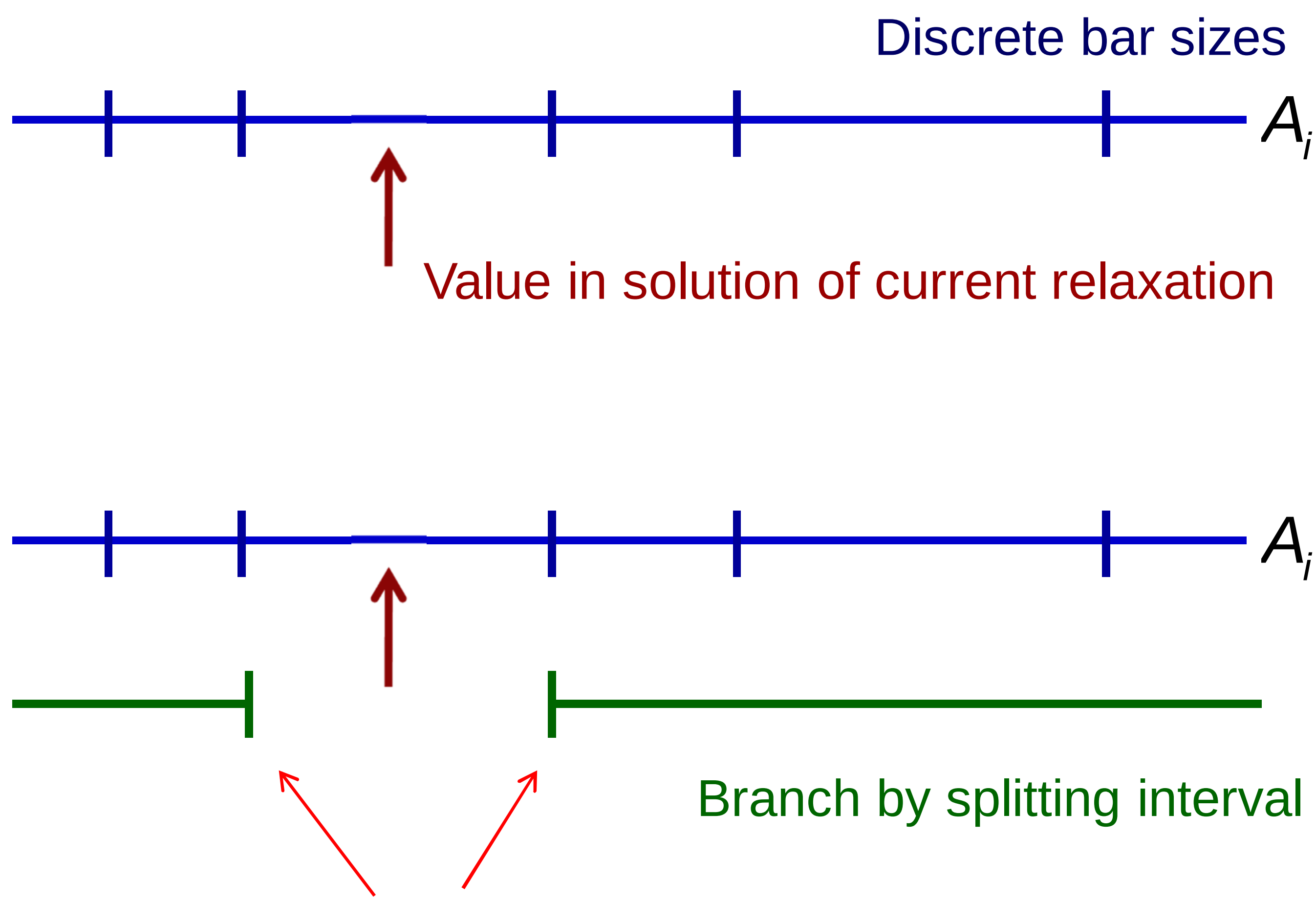


Discrete bar sizes
A_i
Value in solution of current relaxation

Branching


Discrete bar sizes
A_i
Value in solution of current relaxation
A_i
Branch by splitting interval

Branching



Solution of next relaxation likely to be at an endpoint.
This branching intelligence unavailable in 0-1 model.

Quasi-relaxation

Given problem $\min_{x \in S} \{ f(x) \}$

The problem $\min_{x \in S'} \{ f'(x) \}$ is a **quasi-relaxation** if for any $x \in S$, there is an $x' \in S'$ with $f'(x') \le f(x)$.

A quasi-relaxation need not be a valid relaxation.

But its **optimal value** is a **valid lower bound** on the optimal value of the original problem.

Quasi-relaxation

Consider the problem

$$\begin{aligned}&\min f(x)\\&g^j(x, y_j) \le 0, \text{ all } j\\&x \in \mathbb{R}^n, y_j \text{ discrete}\end{aligned}$$

Quasi-relaxation

Consider the problem

$$\begin{aligned}&\min f(x)\\&\boxed{g^j}(x, y_j) \le 0, \text{ all } j\\&x \in \mathbb{R}^n, y_j \text{ discrete}\end{aligned}$$

Each g^j is a vector of functions

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &\boxed{g^j}(x, y_j) \le 0, \ \text{all } j \\ &x \in \mathbb{R}^n, \boxed{y_j} \text{ discrete} \end{aligned}$$

Each g^j is a vector of functions

Each y_j is a scalar variable

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &\boxed{g^j}(x, y_j) \le 0, \text{ all } j \\ &x \in \mathbb{R}^n, \boxed{y_j} \text{ discrete} \end{aligned}$$

Each g^j is a vector of functions

Each y_j is a scalar variable

Relaxing the problem by making y_j continuous could result in a **nonconvex** problem.

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &\boxed{g^j}(x, y_j) \le 0, \ \text{all } j \\ &x \in \mathbb{R}^n, \boxed{y_j} \text{ discrete} \end{aligned}$$

Each g^j is a vector of functions

Each y_j is a scalar variable

Relaxing the problem by making y_j continuous could result in a **nonconvex** problem.

But suppose the problem becomes convex when each y_j is fixed to a **constant**.

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &\boxed{g^j}(x, y_j) \le 0, \ \text{all } j \\ &x \in \mathbb{R}^n, \boxed{y_j} \text{ discrete} \end{aligned}$$

Each g^j is a vector of functions

Each y_j is a scalar variable

Relaxing the problem by making y_j continuous could result in a **nonconvex** problem.

But suppose the problem becomes convex when each y_j is fixed to a **constant**.

Then we may be able to write a **convex quasi-relaxation**.

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &g^j(x, y_j) \le 0, \ \text{all } j \\ &x \in \mathbb{R}^n, y_j \text{ discrete} \end{aligned}$$

Theorem (JNH 2005)

If each $g^j(x,y)$ is semihomogeneous in x and concave in scalar y_j, then the following is a **quasi-relaxation**:

$$\begin{aligned} &\min f(x) \\ &g(x^1, y_L) + g(x^2, y_U) \le 0 \\ &\alpha x^L \le x^1 \le \alpha x^U \\ &(1-\alpha) x^L \le x^2 \le (1-\alpha) x^U \\ &x = x^1 + x^2 \\ &\alpha \in [0,1] \end{aligned}$$

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &g^j(x, y_j) \le 0, \ \text{all } j \\ &x \in \mathbb{R}^n, \ y_j \text{ discrete} \end{aligned}$$

Theorem (JNH 2005)

If each $g^j(x,y)$ is semihomogeneous in x and concave in scalar y_j, then the following is a **quasi-relaxation**:

$$\begin{aligned} &\min f(x) \\ &g(x^1, y_L) + g(x^2, y_U) \le 0 \\ &\alpha x^L \le x^1 \le \alpha x^U \\ &(1-\alpha) x^L \le x^2 \le (1-\alpha) x^U \\ &x = x^1 + x^2 \\ &\alpha \in [0,1] \end{aligned}$$

$g(\alpha x, y) \le \alpha g(x, y)$ for all x, y and $\alpha \in [0,1]$

$g(0, y) = 0$ for all y

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &g^j(x, y_j) \le 0, \ \text{all } j \\ &x \in \mathbb{R}^n, \ y_j \ \text{discrete} \end{aligned}$$

Theorem (JNH 2005)

If each $g^j(x,y)$ is semihomogeneous in x and concave in scalar y_j, then the following is a **quasi-relaxation**:

$$\begin{aligned} &\min f(x) \\ &g(x^1, \boxed{y_L}) + g(x^2, \boxed{y_U}) \le 0 \\ &\alpha x^L \le x^1 \le \alpha x^U \\ &(1-\alpha)x^L \le x^2 \le (1-\alpha)x^U \\ &x = x^1 + x^2 \\ &\alpha \in [0,1] \end{aligned}$$

Bounds on y

Quasi-relaxation

Consider the problem

$$\begin{aligned} &\min f(x) \\ &g^j(x, y_j) \le 0, \text{ all } j \\ &x \in \mathbb{R}^n, y_j \text{ discrete} \end{aligned}$$

Theorem (JNH 2005)

If each $g^j(x,y)$ is semihomogeneous in x and concave in scalar y_j, then the following is a **quasi-relaxation**:

$$\begin{aligned} &\min f(x) \\ &g(x^1, y_L) + g(x^2, y_U) \le 0 \\ &\alpha \boxed{x^L} \le x^1 \le \alpha \boxed{x^U} \\ &(1-\alpha)x^L \le x^2 \le (1-\alpha)x^U \\ &x = x^1 + x^2 \\ &\alpha \in [0,1] \end{aligned}$$

Bounds on x

$$
\begin{aligned}
&\min f(x)\\
&g(x^1, y_L) + g(x^2, y_U) \leq 0\\
&\alpha x^L \leq x^1 \leq \alpha x^U\\
&(1-\alpha)x^L \leq x^2 \leq (1-\alpha)x^U\\
&x = x^1 + x^2\\
&\alpha \in [0,1]
\end{aligned}
$$

Quasi-relaxation

Why?

Take any feasible solution $\bar{x}, \bar{y}$

Quasi-relaxation

$$
\begin{aligned}
&\min f(x)\\
&g(x^1, y_L) + g(x^2, y_U) \le 0\\
&\alpha x^L \le x^1 \le \alpha x^U\\
&(1-\alpha)x^L \le x^2 \le (1-\alpha)x^U\\
&x = x^1 + x^2\\
&\alpha \in [0,1]
\end{aligned}
$$

Why?

Take any feasible solution $\bar{x}, \bar{y}$

Choose α so that

$$\bar{y}_j = \alpha_j y_j^L + (1-\alpha_j) y_j^U$$

Set $x^{j1} = \alpha_j \bar{x}, \quad x^{j2} = (1-\alpha_j)\bar{x}$

Quasi-relaxation

$$\begin{aligned}
&\min f(x)\\
&g(x^1, y_L) + g(x^2, y_U) \le 0\\
&\alpha x^L \le x^1 \le \alpha x^U\\
&(1-\alpha)x^L \le x^2 \le (1-\alpha)x^U\\
&x = x^1 + x^2\\
&\alpha \in [0,1]
\end{aligned}$$

Why?

Take any feasible solution $\bar{x}, \bar{y}$

Choose α so that

$$\bar{y}_j = \alpha_j y_j^L + (1-\alpha_j) y_j^U$$

Set $x^{j1} = \alpha_j \bar{x}, \quad x^{j2} = (1-\alpha_j)\bar{x}$

Then for each component *i* of g^j we have

$$\begin{aligned}
&g_i^j(x^{j1}, y_j^L) + g_i^j(x^{j2}, y_j^U) = g_i^j\left(\alpha_j \bar{x}, y_j^L\right) + g_i^j\left((1-\alpha_j)\bar{x}, y_j^U\right)\\
&= \alpha_j g_i^j(x, y_j^L) + (1-\alpha_j) g_i^j(x, y_j^U) \le g_i^j\left(x, \alpha_j y_j^L + (1-\alpha_j) y_j^U\right) = g_i^j(x, y_j)
\end{aligned}$$

homogeneity

concavity

Quasi-relaxation

$$\begin{aligned}&\min f(x)\\&g^j(x, y_j) \le 0, \ \text{all } j\\&x \in \mathbb{R}^n, y_j \text{ discrete}\end{aligned}$$

So we have a feasible solution of the quasi-relaxation with value that is less than or equal to (in fact equal to) that of the original problem.

satisfied, by above argument

$$\begin{aligned}&\min f(x)\\&g(x^1, y_L) + g(x^2, y_U) \le 0\\&\alpha x^L \le x^1 \le \alpha x^U\\&(1-\alpha)x^L \le x^2 \le (1-\alpha)x^U\\&x = x^1 + x^2\end{aligned}$$

satisfied, by construction

Quasi-relaxation

$$\begin{aligned} &\min f(x) \\ &g^j(x, y_j) \le 0, \ \text{all } j \\ &x \in \mathbb{R}^n, \ y_j \text{ discrete} \end{aligned}$$

$\dfrac{E_i}{h_i} A_i v_i = s_i$ has the form $g(x,y) = 0$ with g semihomogenous in x and concave (linear) in y because we can write it

$$\frac{E_i}{h_i} A_i v_i - s_i = 0$$

with $x = (A_i, s_i)$, $y = v_i$.

Truss Structure Design

So we have a quasi-relaxation of the truss problem:

$$
\begin{aligned}
\min \quad & \sum_i h_i [A_i^L y_i + A_i^U (1-y_i)] \\
s.t. \quad & \sum_i \cos\theta_{ij} s_i = p_j, \text{ all } j \\
& \sum_j \cos\theta_{ij} d_j = v_{i0} + v_{i1}, \text{ all } i \\
& \frac{E_i}{h_i}(A_i^L v_{i0} + A_i^U v_{i1}) = s_i, \text{ all } i \\
& v_i^L y_i \le v_{i0} \le v_i^U y_i, \text{ all } i \\
& v_i^L (1-y_i) \le v_{i1} \le v_i^U (1-y_i), \text{ all } i \\
& d_j^L \le d_j \le d_j^U, \text{ all } j \\
& 0 \le y_i \le 1, \text{ all } i
\end{aligned}
$$

Hooke's law is linearized

Elongation bounds split into 2 sets of bounds

Truss Structure Design

Logic cuts

v_{i0} and v_{i1} must have same sign in a feasible solution.

If not, we branch by adding logic cuts

$$v_{i0}, v_{i1} \le 0, \qquad v_{i0}, v_{i1} \ge 0$$

Truss Structure Design

In general, we can have a **metaconstraint** to represent the semihomogeneous constraint $g(x,y) \leq 0$.

This tells the solver to generate a quasi-relaxation.

Truss Structure Design

In general, we can have a **metaconstraint** to represent the semihomogeneous constraint $g(x,y) \le 0$.

This tells the solver to generate a quasi-relaxation.

Since a bilinear constraint $xy = \alpha$ is always semihomogeneous, we will use a **bilinear** metaconstraint with a quasi-relaxation option.

Truss Structure Design

SIMPL model

Recognized as
linear systems

```
OBJECTIVE
  maximize sum i of c[i]*h[i]*A[i]
CONSTRAINTS
  equilibrium means {
    sum i of b[i,j]*s[i,l] = p[j,l] forall j,l
    relaxation = { lp } }
  compatibility means {
    sum j of b[i,j]*d[j,l] = v[i,l] forall i,l
    relaxation = { lp } }
  hooke means {
    E[i]/h[i]*A[i]*v[i,l] = s[i,l] forall i
    relaxation = { lp:quasi } }
SEARCH
  type = { bb:bestdive }
  branching = { hooke:first:quasicut, A:splitup }
```

Truss Structure Design

SIMPL model

```
OBJECTIVE
  maximize sum i of c[i]*h[i]*A[i]
CONSTRAINTS
  equilibrium means {
    sum i of b[i,j]*s[i,l] = p[j,l] forall j,l
    relaxation = { lp } }
  compatibility means {
    sum j of b[i,j]*d[j,l] = v[i,l] forall i,l
    relaxation = { lp } }
  hooke means {
    E[i]/h[i]*A[i]*v[i,l] = s[i,l] forall i
    relaxation = { lp:quasi } }
SEARCH
  type = { bb:bestdive }
  branching = { hooke:first:quasicut, A:splitup }
```

Recognized as bilinear system

Truss Structure Design

SIMPL model

```
OBJECTIVE
  maximize sum i of c[i]*h[i]*A[i]
CONSTRAINTS
  equilibrium means {
    sum i of b[i,j]*s[i,l] = p[j,l] forall j,l
    relaxation = { lp } }
  compatibility means {
    sum j of b[i,j]*d[j,l] = v[i,l] forall i,l
    relaxation = { lp } }
  hooke means {
    E[i]/h[i]*A[i]*v[i,l] = s[i,l] forall i
    relaxation = { lp:quasi } }
SEARCH
  type = { bb:bestdive }
  branching = { hooke:first:quasicut, A:splitup }
```

Generate quasi-relaxation for semihomogenous function

Truss Structure Design

SIMPL model

```
OBJECTIVE
  maximize sum i of c[i]*h[i]*A[i]
CONSTRAINTS
  equilibrium means {
    sum i of b[i,j]*s[i,l] = p[j,l] forall j,l
    relaxation = { lp } }
  compatibility means {
    sum j of b[i,j]*d[j,l] = v[i,l] forall i,l
    relaxation = { lp } }
  hooke means {
    E[i]/h[i]*A[i]*v[i,l] = s[i,l] forall i
    relaxation = { lp:quasi } }
SEARCH
  type = { bb:bestdive }
  branching = { hooke:first:quasicut, A:splitup }
```

Branch first on violated logic cuts for quasi-relaxation

Truss Structure Design

SIMPL model

```
OBJECTIVE
  maximize sum i of c[i]*h[i]*A[i]
CONSTRAINTS
  equilibrium means {
    sum i of b[i,j]*s[i,l] = p[j,l] forall j,l
    relaxation = { lp } }
  compatibility means {
    sum j of b[i,j]*d[j,l] = v[i,l] forall i,l
    relaxation = { lp } }
  hooke means {
    E[i]/h[i]*A[i]*v[i,l] = s[i,l] forall i
    relaxation = { lp:quasi } }
SEARCH
  type = { bb:bestdive }
  branching = { hooke:first:quasicut, A:splitup }
```

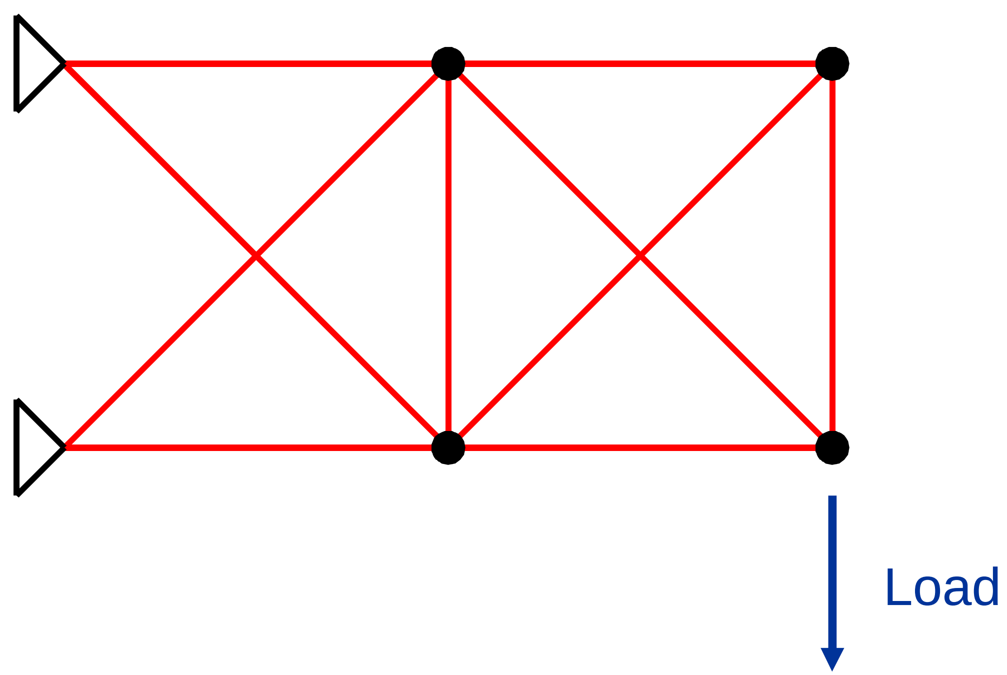
Then branch on A_i in-domain constraint.

Violated when A_i is not one of the discrete bar sizes.

Take upper branch first.

Truss Structure Design

10-bar cantilever truss



Truss Structure Design

Hand-coded
integrated method

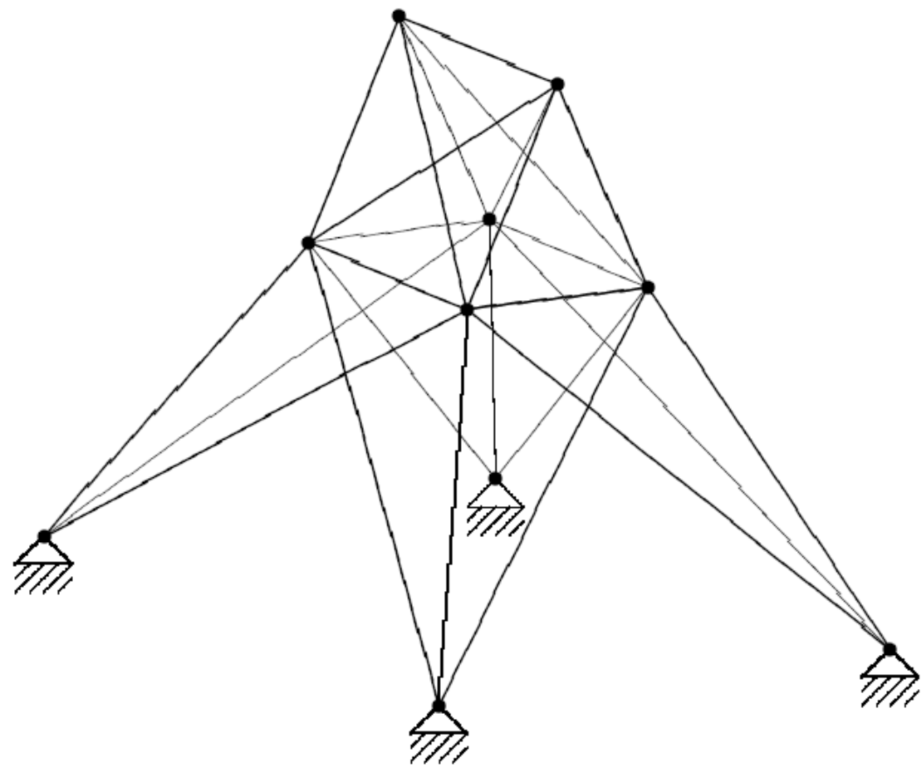
Computational results (seconds)

No. bars	Loads	BARON	CPLEX 11	Hand coded	SIMPL
10	1	5.3	0.40	0.03	0.08
10	1	3.8	0.26	0.02	0.07
10	1	8.1	0.83	0.16	0.49
10	1	8.8	1.2	0.22	0.63
10	2	24	4.9	0.64	1.84
10	2*	327	146	145	65
10	2*	2067	1087	600	651

*plus displacement bounds

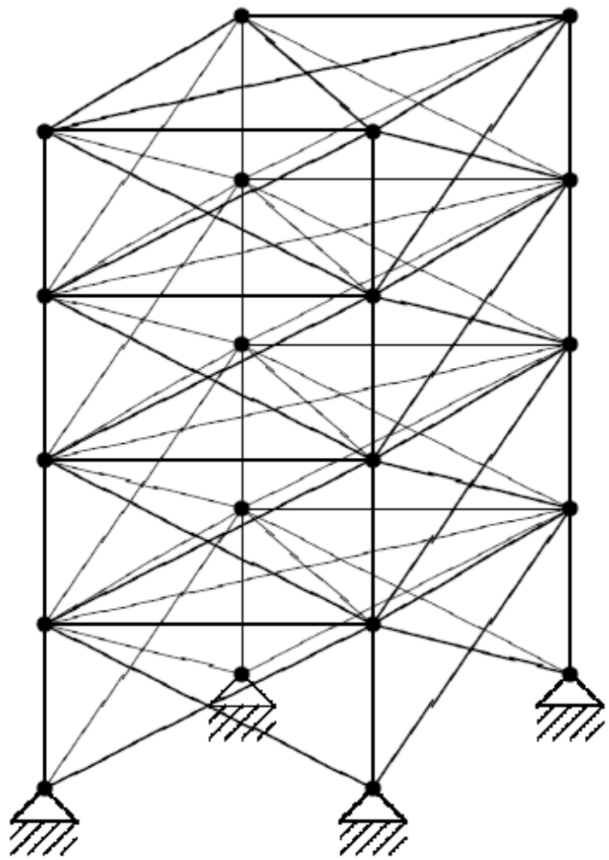
Truss Structure Design

25-bar problem



Truss Structure Design

72-bar problem



Truss Structure Design

Hand-coded
integrated method

Computational results (seconds)

No. bars	Loads	BARON	CPLEX 11	Hand coded	SIMPL
25	2	3,302	44	44	20
72	2	3,376	208	33	28
90	2	21,011	570	131	92
108	2	> 24 hr*	3208	1907	1720
200	2	> 24 hr*	> 24 hr*	> 24 hr**	> 24 hr***

* no feasible solution found

** best feasible solution has cost 32,748

*** best feasible solution has cost 32,700

Current Version of SIMPL

- To download:
  - Click the link to SIMPL on John Hooker’s website.
  - See readme file for complete instructions.
  - Download executable and associated files
- Operational on GNU/Linux only
- Requires subsidiary solvers
  - CPLEX (version 9, 10, or 11)
  - Eclipse (any version 5.8.80 or later), free download
- Download problem instances
  - Including all reported in this talk.