Logic-Based Optimization Methods for Engineering Design

John Hooker
Carnegie Mellon University

Turkish Operational Research Society
Ankara, June 1999

Joint work with:

- Srinivas Bollapragada, General Electric R&D
- Omar Ghattas, Civil and Environmental Engineering, CMU.
- Ignacio Grossmann, Chemical Engineering, CMU

Idea of logic-based methods

- An alternative to integer and mixed integer/linear programming.
- Represent discrete choices with logical propositions rather than integer variables.
- Solve with branch-and-bound methods as in integer programming, but with a difference:
    - use logical inference and domain reduction methods from constraint programming as well as linear programming.
    - use new relaxations for logical constraints rather than the traditional linear programming relaxation

Advantages of logic-based approach to design

- Design typically involves a combination of discrete choices and continuous parameters.
- Logic framework provides more natural modeling of discrete choices.
- Solution approach harnesses power of logical inference and removes unnecessary integer variables from the linear relaxations.
- By distinguishing special cases logically, singularities can be avoided.

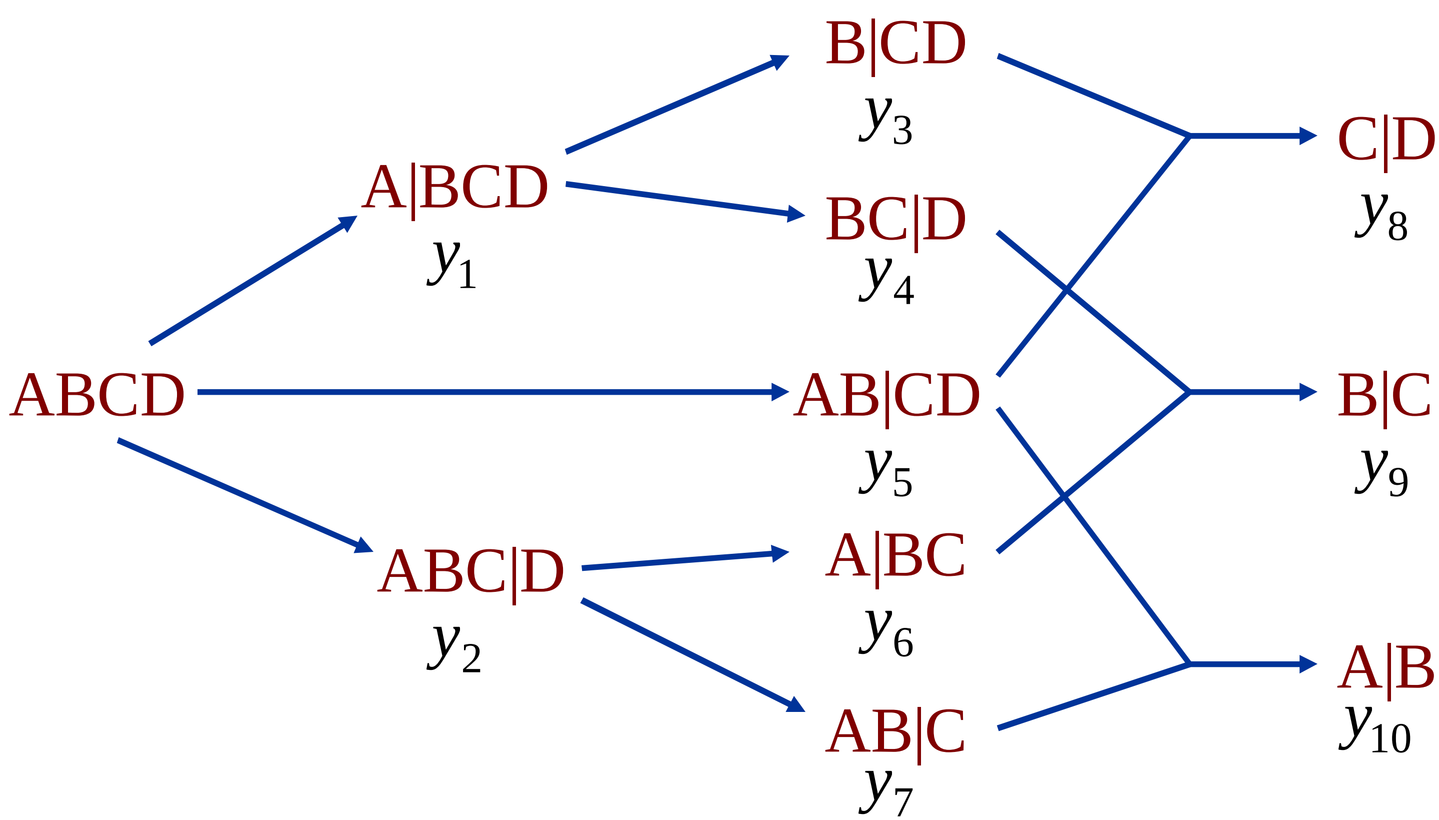
Outline

- Logic-based optimization for chemical processing network design (process synthesis).
- Logic-based optimization for truss structure design.
  - Computational results
- Integration of optimization and constraint satisfaction.

Processing network design

- Design a network of processing units, such as reactors or distillation units
- Meet demand while minimizing fixed and variable costs.
- Discrete choice is which units to install.
- Problem presented here is linear, but heat exchange and other processes give rise to nonlinear models.

4-component separation network



Model for separation problem

Unit flow cost → $\sum_{ij} c_{ij} x_{ij}$; Fixed cost → $\sum_i z_i$; Flow volume → x_{jk}

$$
\begin{aligned}
\min \quad & \sum_{ij} c_{ij} x_{ij} + \sum_i z_i \\
\text{s.t.} \quad & \sum_i x_{ij} = \sum_k x_{jk}, \quad \text{all } j \quad \} \text{ Flow balance} \\
& x_{ij} = \alpha_{ij} \sum_k x_{ki}, \quad \text{all } i, j \quad \} \text{ Unit output} \\
& 0 \le x_{ij} \le k_{ij}, \quad \text{all } i,j \quad \} \text{ Capacity} \\
& y_i \to (z_i = f_i), \quad \text{all } i \\
& \neg y_i \to \begin{pmatrix} z_i = 0 \\ \sum x_{ij} = 0 \end{pmatrix}, \quad \text{all } i
\end{aligned}
$$

Unit is installed → $y_i \to (z_i = f_i)$

Unit is not installed → $\neg y_i$

Integer programming model

Replace discrete constraints with

$$0 \le \sum_j x_{ij} \le k_i y_i, \quad \text{all } i$$

0-1 variable

A continuous (linear programming) relaxation is important for solving the problem. It can be obtained by replacing

$$y_i \in \{0,1\} \qquad \text{with} \qquad 0 \le y_i \le 1$$

Relaxation for logic constraints

The logic constraints

$$y_i \rightarrow (z_i = f_i)$$

$$\neg y_i \rightarrow \begin{pmatrix} z_i = 0 \\ \sum x_{ij} = 0 \end{pmatrix}$$

Can be relaxed:

$$\frac{z_i}{f_i} \geq \frac{1}{M_i} \sum_j x_{ij}$$

Where M_i = upper bound on output of unit i

So integer variables add needless overhead to solution of linear programming relaxation at each node.

Additional logic constraints

Constraints such as

$$y_3 \rightarrow y_1$$
$$y_4 \rightarrow y_1$$
$$y_5 \rightarrow y_8$$
$$y_5 \rightarrow y_{10}$$

can speed processing at each node by ruling out solutions that cannot be optimal.

Logic-based branch & bound

$x_1 = \text{true}$

Solve as an LP: linear part of problem, plus relaxation of conditional constraints, plus:

$$z_1 = f_1$$

$x_2 = \text{true}$

$x_2 = \text{false}$

$x_1 = \text{false}$

Solve same problem except replace $z_1 = f_1$ with:

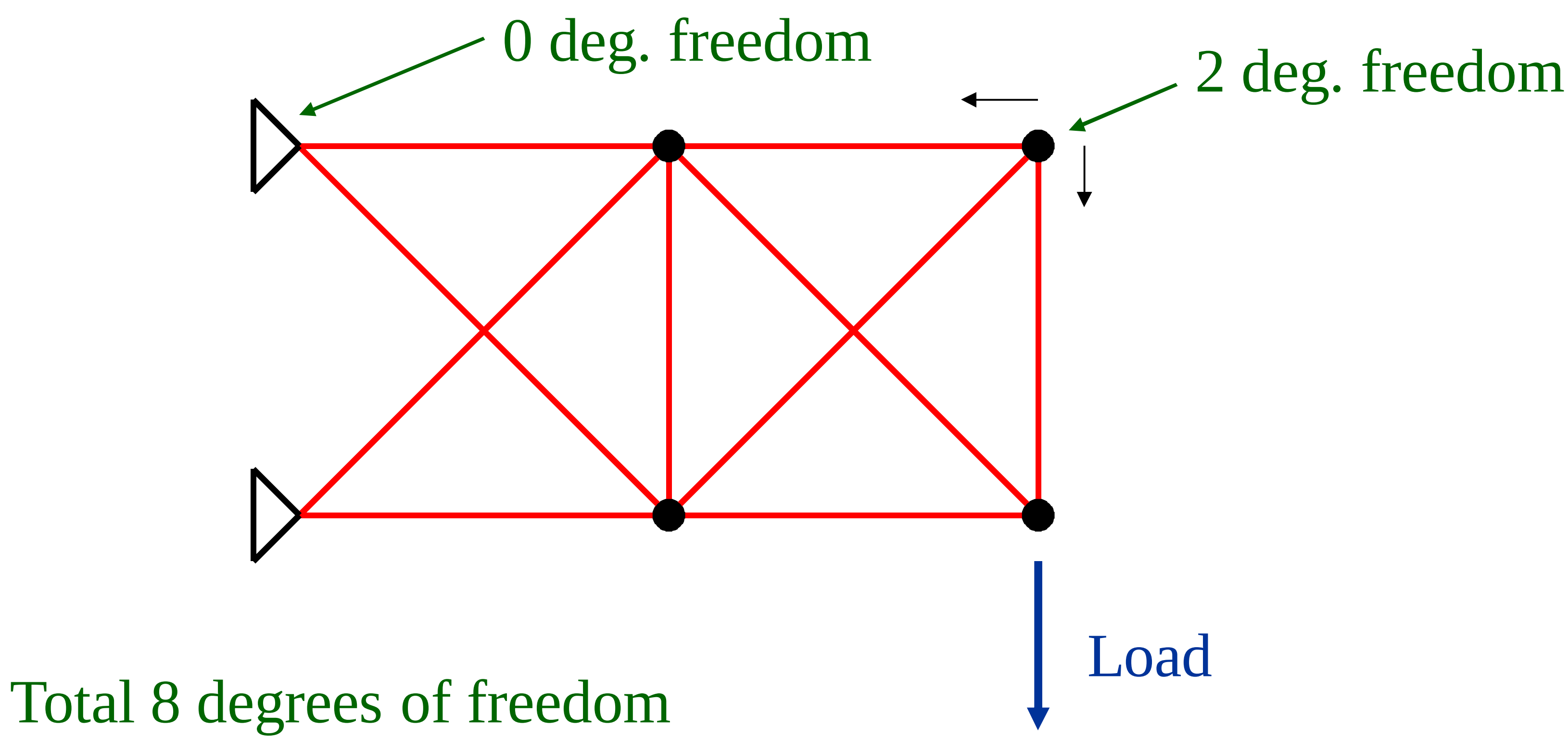
$$\begin{pmatrix} z_1 = 0 \\ \sum x_{1j} = 0 \end{pmatrix}$$

Because $x_3 \rightarrow x_1$ and $x_4 \rightarrow x_1$, fix $x_3 = x_4 = \text{false}$ and add constraints similar to above

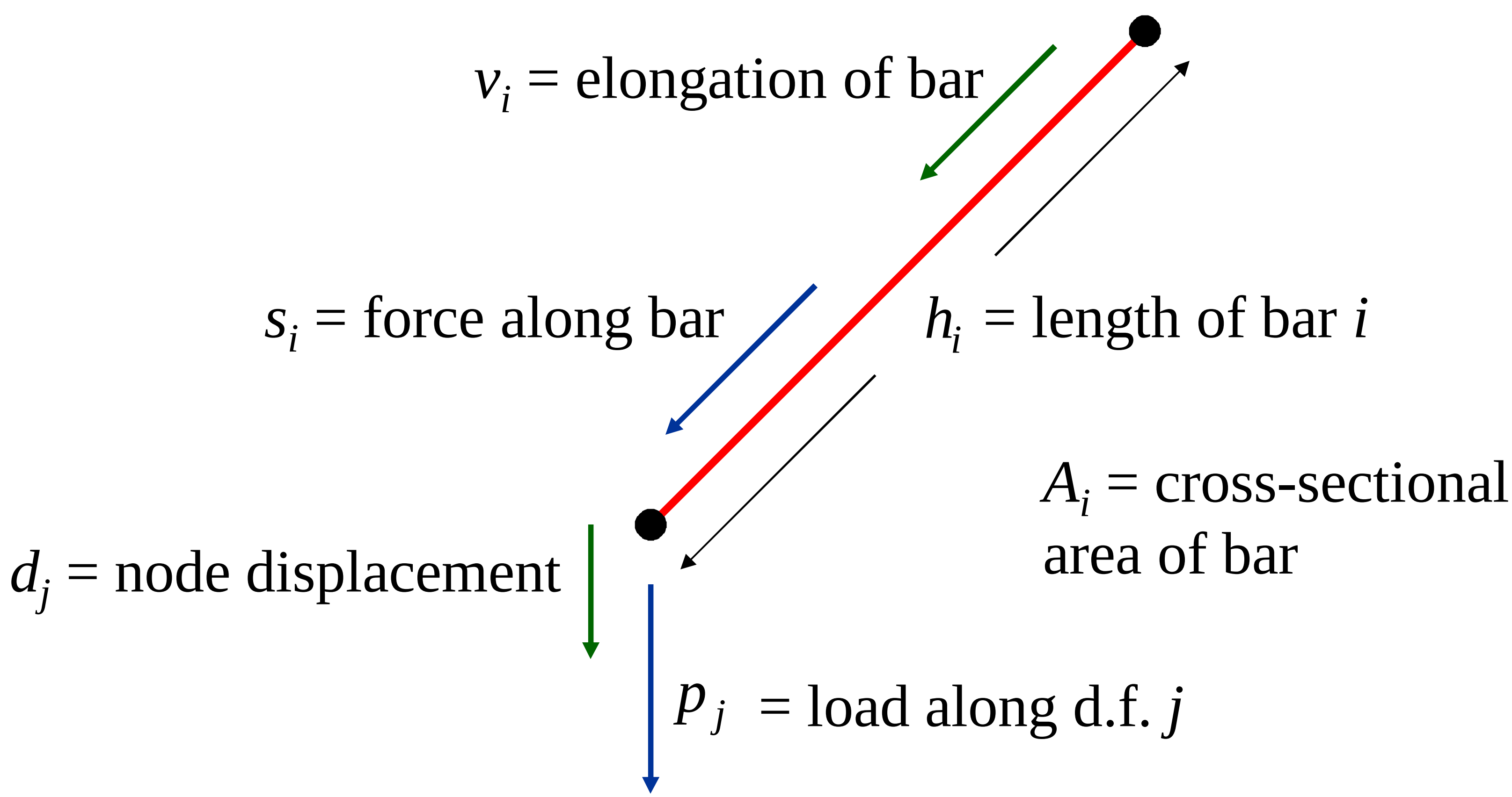
Truss structure design

- Find truss structure that supports a given load while minimizing total weight of bars.
- Any given bar may be present of absent from the truss.
- Its cross-sectional area must be one of several discrete values.
- The model is nonlinear. Logic-based modeling avoids the singularities that occur in traditional models when the bar size goes to zero.

Planar 10-bar cantilever truss



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

$$
\begin{aligned}
\min \quad & \sum_i h_i A_i && \text{\} Minimize total weight} \\
s.t. \quad & \sum_i \cos\theta_{ij} s_i = p_j, \text{ all } j && \text{\} Equilibrium} \\
& \sum_j \cos\theta_{ij} d_j = v_i, \text{ all } i && \text{\} Compatibility} \\
& \frac{E_i}{h_i} A_i v_i = s_i, \text{ all } i && \text{\} Hooke's law} \\
& v_i^L \le v_i \le v_i^U, \text{ all } i && \text{\} Elongation bounds} \\
& d_j^L \le d_j \le d_j^U, \text{ all } j && \text{\} Displacement bounds} \\
& \bigvee_k (A_i = A_{ik}) && \text{\} Logical disjunction}
\end{aligned}
$$

nonlinear → (Hooke's law)

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

Constraints can be imposed for multiple loading conditions

Logic-based branch & bound

- Rather than branch on variables, branch by splitting the range of areas A_i
- At each node, add the constraints

$$A_i^L \leq A_i \leq A_i^U$$

where the bounds are equal to one of the discrete areas bar *i* can have

Mixed integer model introduces many additional variables

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

0-1 variables indicating size of bar *i*

Elongation variable disaggregated by bar size

Hooke's law becomes linear

A useful relaxation can be obtained without the extra variables...

Linear relaxation

Use the change of variables:

Current bounds on areas

$$A_i = A_i^L y_i + A_i^U (1 - y_i)$$
$$v_i = v_{i0} + v_{i1}$$

The y_i's are not 0-1 variables but are continuous variables that vary in the interval [0,1], introduced only to form a relaxation.

The resulting "relaxation" is not a true relaxation but provides a valid bound on the optimal value...

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

Hooke's law is linearized

Elongation bounds split into 2 sets of bounds

Parallel with branch & bound

- In branch & bound, solution of the linear relaxation is often integral, which reduces branching.

- In logic-based branching, solution of "relaxation" often puts areas at endpoints of their ranges, in which case they have one of the permissible discrete values.

Computational testing

- Logic-based: Branch on upper half of interval first.
- MILP: Solve with CPLEX 4.0 with automatic SOS detection turned on.
- Problems:
  10-bar truss
  25-bar transmission tower
  rectangular building (72, 90, 108 bars)
- Bars are “linked” when they must have the same size.

Computational results

Problem	*Linking groups*	*Logic-based seconds*	*CPLEX MILP seconds*
10-bar truss	10	0.3	1.3
10-bar truss	10	0.3	1.6
10-bar truss	10	1.2	2.6
10-bar truss	10	1.4	2.6
10-bar truss + L	10	5.8	23.6
10-bar truss + D	10	68	1089
10-bar truss + D	10	1654	13744

L = 2 loading conditions
D = displacement bounds

Computational results

Problem	*Linking groups*	*Logic-based seconds*	*CPLEX MILP seconds*
25-bar tower + L	8	226	272
72-bar building + L	16	208	12693
90-bar building + L	20	169	>72000
108-bar building + L	24	329	>72000
200-bar structure	96	>36000	>72000

L = 2 loading conditions

An alternative model

Remove nonlinearities by writing Hooke's law directly as a disjunction

$$\bigvee_k \left(\frac{E_i}{h_i} A_{ik} v_i = s_i \right)$$

that can be relaxed with a single inequality for each *i*.

The disjunctive constraint $\bigvee_k (A_i = A_{ik})$

can now be dropped.

Possible enhancements

- Limit total number of bar sizes in structure.
- Enforce symmetries in topology even when bar sizes need not match.
- Enforce connectedness and other properties logically.
- Find logical characterization of stability.

Constraint programming

- Constraint programming is based primarily on logic-based methods known as *constraint satisfaction* methods.

- Discrete variables tend to be multivalued (here, they were 2-valued).

- *Global* constraints exploit problem structure and speed solution; e.g., **all-different**($y_1, \ldots, y_n$), **cumulative**.

- Inference takes the form of *domain reduction* algorithms, particularly when applied to global constraints.

Optimization + constraint programming

- Combining optimization and constraint programming is a very active area of research; e.g. OPL modeling language.
- The logic-based method described here is an example of one approach.