Logic-Based Methods for Global Optimization

J. N. Hooker

GSIA, Carnegie Mellon University, Pittsburgh, USA,
jh38@andrew.cmu.edu

November 2003

Abstract. Logic-based methods provide a strategy for applying convex nonlinear programming to nonconvex global optimization. Such methods assume that the problem becomes convex when selected variables are fixed. The selected variables must be discrete, or else discretized if they are continuous. We provide a tutorial survey of disjunctive programming with convex relaxations, logic-based outer approximation, and logic-based Benders decomposition. We then introduce a branch-and-bound method with convex quasi-relaxations (BBCQ) that can be effective when the discrete variables take a large number of real values. The BBCQ method generalizes work of Bollapragada, Ghattas and Hooker on structural design problems. It applies when the constraint functions are concave in the discrete variables and have a weak homogeneity property in the continuous variables.

We address global optimization problems that become convex when selected variables are fixed. If these variables are discrete, the constraints can be reformulated as logical disjunctions of convex constraints. If some of the selected variables are not discrete, we discretize them in order to obtain an approximate global solution.

The motivation for this approach is to take advantage of highly developed nonlinear programming methods for convex problems, as well as branch-and-bound methods for discrete problems. A branch-and-bound method chooses the appropriate disjunct in each constraint. Nonlinear programming is applied to the convex subproblem that results when the disjuncts are chosen.

We present four variations of this general approach. Two of them are most practical when the discrete variables do not take a large number of possible values: (a) disjunctive programming with convex relaxations, and (b) logic-based outer approximation. The disjunctive programming model can also be solved as a mixed integer/nonlinear programming (MINLP) problem. When there are a large number of discrete values, as when some discrete variables represent discretized continuous variables, one can turn to methods that do not require explicit representation of the disjunctions: (c) logic-based Benders decomposition, and (d) branch and bound with convex quasi-relaxations (BBCQ). The convergence rate of the Benders method depends heavily on the problem structure, however. BBCQ is intended for problems in which the discrete variables are real-valued. It does not rely on decomposition but requires that the constraint functions satisfy certain properties.

This paper begins with a tutorial summary of the first three methods, which are developed elsewhere. It then introduces the BBCQ method as a formalization and generalization of a technique applied by Bollapragada, Ghattas and Hooker to structural design problems [1]. This application is presented at the end of the paper as an illustration of disjunctive programming and BBCQ.

1 General Form of the Problem

We solve problems of the form

$$\begin{array}{ll} \min & x_0 \\ \text{subject to} & g^j(x, y_j) \leq 0, \;\; j \in J \\ & L(y) \\ & x \in \mathbb{R}^n, \;\; y_j \in Y_j, \; j \in J \end{array} \tag{1}$$

where $g^j(x, y_j)$ is a vector of functions and $L(y)$ is a logical constraint on possible values of the discrete variables y_j. If some of the y_j are continuous, we discretize them by converting Y_j to a finite set. We assume that when each y_j is fixed to some $\bar{y}_j \in Y_j$ we obtain the convex subproblem:

$$\begin{array}{ll} \min & x_0 \\ \text{subject to} & g^j(x, \bar{y}_j) \leq 0, \;\; j \in J \\ & x \in \mathbb{R}^n \end{array} \tag{2}$$

It is convex in the sense that each $g^j(x, \bar{y}_j)$ is a vector of convex functions of x.

We assume without loss of generality that the objective function is a single variable x_0, since x_0 can be defined in the constraints. We also suppose that each constraint contains only one discrete variable y_j. Many problems naturally occur in this form. Problems that do not can in principle be put into this form by a change of variables. Thus a constraint $g^j(x, y_1, \ldots, y_m) \leq 0$ can be written $g^j(x, y^j) \leq 0$, where $y^j = (y_1^j, \ldots, y_m^j)$ is regarded as a single variable. The variables y^j can now be related by the logical constraints $y^j = y^1$ for all $j \in J$. For instance, the constraints $x + y_1 + y_2 \geq b$ and $x + y_2 + y_3 \geq b$ can be rewritten $x + y_1^1 + y_2^1 \geq b$ and $x + y_2^2 + y_3^2 \geq b$ by adding the constraint $y_2^1 = y_2^2$.

2 Structural Design Example

We use a simple structural design problem to illustrate all four solution methods. The two pillars of Fig. 1 support the two horizontal platforms shown. Pillar 1 bears a weight of 10 and pillar 2 a weight of 20. The weight on pillar 1 causes it to compress an amount x_1 given by Hooke's law, $x_1 y_1 = 10$, where y_1 is the thickness (cross-sectional area) of pillar 1, and similarly for pillar 2. Thus the

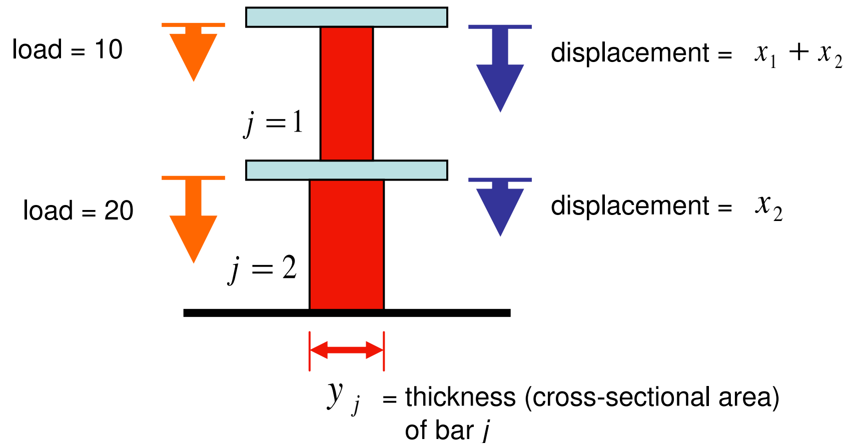

Fig. 1. A simple structural design problem.

top platform is displaced downward a distance $x_1 + x_2$, and the bottom platform a distance x_2. The cost of steel in pillar j is $300y_j$, and we impose a penalty $(x_1 + x_2)^2 + x_2^2$ for displacement. The objective is to choose thicknesses for the two pillars so as to minimize the sum of the cost and the penalty:

$$\begin{aligned} &\text{minimize} \;\; 300y_1 + 300y_2 + (x_1 + x_2)^2 + x_2^2 \\ &\text{subject to } x_1 y_2 = 10, \;\; x_2 y_2 = 20 \\ &\qquad\qquad\;\; x_j \geq 0, \; y_j \in \{1, 2\} \text{ for } j = 1, 2 \end{aligned} \tag{3}$$

To simplify the example we consider two discrete thicknesses (1 and 2). Near the end of the paper we also consider the case in which y_1 and y_2 are continuous variables.

The model (3) is chosen for the sole purpose of illustrating the algorithms described below. *It is not intended to be a realistic or proper engineering model for a structural design problem.* In particular, the nonlinear penalty terms are inserted to show how they are treated in the algorithms, not because they necessarily represent the best way to limit displacement. A more realistic structural design model is presented in Section 8.

To verify that the model (3) has the general form (1), note that it can be written

$$\begin{aligned} &\text{minimize} \;\; z_0 \\ &\text{subject to } z_1 + z_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \leq 0 \\ &\qquad \begin{bmatrix} 300y_1 - z_1 \\ 10 - x_1 y_1 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ &\qquad \begin{bmatrix} 300y_2 - z_2 \\ 20 - x_2 y_2 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ &\qquad x_j \geq 0, \;\; y_j \in \{0, 1\} \;\; \text{for } j = 1, 2 \end{aligned} \tag{4}$$

The model is clearly convex when the y_js are fixed.

3 Disjunctive Formulation

A straightforward but generally impractical way to solve (1) is by a branch-and-bound method that branches on the y_j and solves a continuous relaxation of the problem at each node of the branching tree. The difficulty is that these continuous problems are in general nonconvex.

To obtain convex relaxations, we write (1) as a *disjunctive programming problem* by creating a disjunct for each possible value of y_j.

$$\begin{array}{ll} \min & x_0 \\ \text{subject to} & \displaystyle\bigvee_{v \in Y_j} \begin{bmatrix} y_j = v \\ g^j(x, v) \le 0 \end{bmatrix}, \quad j \in J \\ & L(y) \\ & x \in \mathbb{R}^n \end{array} \tag{5}$$

The functions $g^j(x, v)$ are convex because the second argument is fixed. They may also simplify in form. In some cases singularities disappear, as for example when

$$g^j(x, y_j) = \begin{bmatrix} x_1 - 1/y_1 \\ x_1 - x_2 0 \end{bmatrix} \le \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

can be written simply $x_1 - x_2 \le 0$ for $y_j = 0$.

Example. The disjunctive formulation for the structural design example of the previous section is

$$\begin{array}{ll} \text{minimize} & x_0 \\ \text{subject to} & z_1 + z_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \le 0 \\ & \begin{bmatrix} y_1 = 1 \\ 300 - z_1 \le 0 \\ 10 - x_1 \le 0 \end{bmatrix} \vee \begin{bmatrix} y_1 = 2 \\ 600 - z_1 \le 0 \\ 10 - 2x_1 \le 0 \end{bmatrix} \\ & \begin{bmatrix} y_2 = 1 \\ 300 - z_2 \le 0 \\ 20 - x_2 \le 0 \end{bmatrix} \vee \begin{bmatrix} y_2 = 2 \\ 600 - z_2 \le 0 \\ 20 - 2x_2 \le 0 \end{bmatrix} \\ & x_1, x_2 \ge 0 \end{array} \tag{6}$$

Note that the first disjunction contains only one disjunct, since no y_j appears in the corresponding constraint. (For convenience we regard the first constraint function as $g^0(x, y_0)$, where y_0 does not actually appear.) All of the disjuncts are convex, and in fact all but the first are linear. The subproblem (2) becomes

$$\begin{aligned} \text{minimize } & x_0 \\ \text{subject to } & z_1 + z_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \le 0 \\ & 300\bar{y}_1 - z_1 \le 0 \\ & 10 - x_1\bar{y}_1 \le 0 \\ & 300\bar{y}_2 - z_2 \le 0 \\ & 20 - x_2\bar{y}_2 \le 0 \\ & x_1, x_2 \ge 0 \end{aligned} \tag{7}$$

4 Disjunctive Programming with Convex Relaxations

A branch-and-bound method can be practical for the disjunctive programming problem (5) when it is possible to devise a convex relaxation at each node of the search tree. Two such relaxations, based on big-M and convex hull formulations, are presented here.

Branch and bound proceeds by branching on the alternatives in the disjunctions of (5). At each node of the search tree, some disjuncts have been selected by prior branching, and these are imposed as constraints. The disjunctions on which the algorithm has not yet branched are relaxed. A lower bound is obtained by solving a convex problem that minimizes x_0 subject to the imposed disjuncts and the relaxed disjunctions. The lower bound is used to prune the search as is normally done in branch-and-bound search (see [8, 10] for details).

A closely related approach is to apply an MINLP method to a 0-1 model of the disjunctive model (5), which results from imposing an integrality condition on either the big-M or the convex hull relaxation of (5).

The *big-M relaxation* introduces a variable β_{jv} for each $v \in Y_j$, where $\beta_{jv} = 1$ is interpreted as indicating $y_j = v$. It is assumed that there are bounds $x^L \le x \le x^U$ on x. Let $L(\beta)$ be an inequality encoding of the logical constraints $L(y)$ [2]. The big-M relaxation of (5) is:

$$\begin{aligned} \min \quad & x_0 \\ \text{subject to} \quad & g^j(x, v) \le M^{jv}(1 - \beta_{jv}), \;\; \text{all } v \in Y_j,\; j \in J \\ & \sum_{v \in Y_j} \beta_{jv} = 1,\; \beta_{jv} \ge 0, \;\; \text{all } v \in Y_j,\; j \in J \\ & L(\beta),\; x^L \le x \le x^U \\ & 0 \le \beta_{jv} \le 1, \;\; \text{all } v \in Y_j, j \in J \end{aligned} \tag{8}$$

where M^{jv} is a vector of valid upper bounds on the component functions of $g^j(x, v)$, given that $x^L \le x \le x^U$. This relaxation is clearly convex.

One can solve (5) by using relaxation (8) at each node, where J in (8) corresponds to the set of disjunctions on which the algorithm has not yet branched. Alternatively, one can apply an MINLP algorithm to the 0-1 model obtained

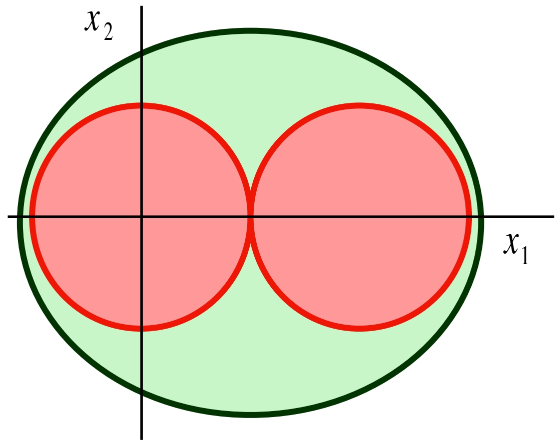

Fig. 2. A big-M relaxation of a union of two discs.

by replacing $\beta_{jv} \in [0,1]$ in (8) with $\beta_{jv} \in \{0,1\}$, where J corresponds to the original set of disjunctions.

The bounds M^{jv} should be the tightest that can be practicably obtained. One valid bound is

$$M_i^{jv} = \max_{x^L \le x \le x^U} \left\{ g_i^j(x,v) \right\} \tag{9}$$

but the tightest bound is

$$M_i^{jv} = \max_{v' \in Y_j \setminus \{v\}} \left\{ \max_{x^L \le x \le x^U} \left\{ g_i^j(x,v) \mid g^j(x,v') \le 0 \right\} \right\}$$

Example. Consider the disjunction

$$\begin{bmatrix} y_j = 1 \\ x_1^2 + x_2^2 - 1 \le 0 \end{bmatrix} \vee \begin{bmatrix} y_j = 2 \\ (x_1 - 2)^2 + x_2^2 - 1 \le 0 \end{bmatrix} \tag{10}$$

with $x_1 \in [-1,3]$ and $x_2 \in [-1,1]$. The feasible set for (10) is the union of the discs in Fig. 2. Setting $M^{j1} = M^{j2} = 9$ as given by (9), we get the big-M relaxation

$$\begin{aligned} &x_1^2 + x_2^2 - 1 \le 9(1-\beta) \\ &(x_1-2)^2 + x_2^2 \le 9\beta \\ &\beta \in [0,1] \end{aligned}$$

The elliptical area in Fig. 2 depicts the projection of the relaxation onto the x-space. The projection is described by $x_1^2 + (x_1-2)^2 + 2x_2^2 \le 11$.

A second convex relaxation for (5), based on convex hull descriptions of the disjunctions, was developed by Stubbs and Mehrotra [13] and Grossmann and

Lee [6]. It is generally tighter than the big-M relaxation but requires that we introduce for each disjunction j a new continuous variable x^{jv} for each $v \in Y_j$.

The *convex hull relaxation* for a disjunction

$$\bigvee_{v \in Y_j} g^j(x, v) \le 0 \tag{11}$$

can be derived as follows. We assume that x and g^j are bounded; that is, $x \in [x^L, x^U]$, and $g^j(x) \in [-L, L]$ for $x \in [x^L, x^U]$. We wish to characterize all points x that can be written as a convex combination of points $\hat{x}^{jv}$ that respectively satisfy the disjuncts of (11). Thus we have

$$\begin{aligned}
& x = \sum_{v \in Y_j} \beta_{jv} \hat{x}^{jv} \\
& g^j(\hat{x}^j, v) \le 0, \ \ \text{all } v \in Y_j \\
& x^L \le \hat{x}^j \le x^U \\
& \sum_{v \in Y_j} \beta_{jv} = 1, \ \ \beta_{jv} \ge 0, \ \ \text{all } v \in Y_j
\end{aligned}$$

Using the change of variable $x^{jv} = \beta_{jv} \hat{x}^{jv}$, we obtain the relaxation

$$\begin{aligned}
& x = \sum_{v \in Y_j} x^{jv} \\
& g^j\left(\frac{x^{jv}}{\beta_{jv}}, v\right) \le 0, \ \ \text{all } v \in Y_j \\
& \beta_{jv} x^L \le x^{jv} \le \beta_{jv} x^U, \ \ \text{all } v \in Y_j \\
& \sum_{v \in Y_j} \beta_{jv} = 1, \ \ \beta_{jv} \ge 0, \ \text{all } v \in Y_j
\end{aligned} \tag{12}$$

The function $g^j(x^{jv}/\beta_{jv}, v)$ is in general nonconvex, but a classical result of convex analysis (e.g. [7]) implies that one can restore convexity by multiplying the second constraint of (12) by β_{jv}.

Theorem 1. *Consider the set S consisting of all (x, β) with $\beta \in [0, 1]$ and $x \in [\beta x^L, \beta x^U]$. If $g(x)$ is convex and bounded for $x \in [\beta x^L, \beta x^U]$, then*

$$h(x, \beta) = \begin{cases} \beta h(x/\beta) & \textit{if } \beta > 0 \\ 0 & \textit{if } \beta = 0 \end{cases}$$

is convex and bounded on S.

Proof. To show convexity of $h(x, \beta)$ we arbitrarily choose $(x^1, \beta_1), (x^2, \beta_2) \in S$. Supposing first that $\beta_1, \beta_2 > 0$, we have convexity since

$$\begin{aligned}
&h\left(\alpha x^1 + (1-\alpha)x^2, \alpha\beta_1 + (1-\alpha)\beta_2\right) \\
&= (\alpha\beta_1 + (1-\alpha)\beta_2)\, g\left(\frac{\alpha x^1 + (1-\alpha)x^2}{\alpha\beta_1 + (1-\alpha)\beta_2}\right) \\
&= (\alpha\beta_1 + (1-\alpha)\beta_2)\, g\left(\frac{\alpha\beta_1}{\alpha\beta_1 + (1-\alpha)\beta_2}\frac{x^1}{\beta_1} + \frac{(1-\alpha)\beta_1}{\alpha\beta_1 + (1-\alpha)\beta_2}\frac{x^2}{\beta_2}\right) \\
&\le (\alpha\beta_1 + (1-\alpha)\beta_2)\left[\frac{\alpha\beta_1}{\alpha\beta_1 + (1-\alpha)\beta_2} g\left(\frac{x^1}{\beta_1}\right) + \frac{(1-\alpha)\beta_1}{\alpha\beta_1 + (1-\alpha)\beta_2)} g\left(\frac{x^2}{\beta_2}\right)\right] \\
&= \alpha h\left(x^1, \beta_1\right) + (1-\alpha) h\left(x^2, \beta_2\right)
\end{aligned}$$

for any $\alpha \in [0, 1]$, where the inequality is due to the convexity of $g(x)$. If $\beta_1 = \beta_2 = 0$, then

$$h\left(\alpha x^1 + (1-\alpha)x^2, \alpha\beta_1 + (1-\alpha)\beta_2\right) = h(0,0) = \alpha h\left(x^1, \beta_1\right) + (1-\alpha) h\left(x^2, \beta_2\right)$$

since $-\beta_j L \le x^j \le \beta_j L$ implies $x^j = 0$. If $\beta_1 = 0$ and $\beta_2 > 0$, we have

$$\begin{aligned}
&h\left(\alpha x^1 + (1-\alpha)x^2, \alpha\beta_1 + (1-\alpha)\beta_2\right) \\
&= h\left((1-\alpha)x^2, (1-\alpha)\beta_2\right) = (1-\alpha) g\left(\frac{x^2}{\beta_2}\right) \\
&= \alpha h(0,0) + (1-\alpha) h\left(x^2, \beta_2\right)
\end{aligned}$$

Finally, $h(x, \beta) = \beta g(x/\beta)$ is bounded because $\beta \in [0, 1]$, $x/\beta \in [x^L, x^U]$, and $g(x)$ is bounded for $x \in [x^L, x^U]$.

We now obtain the following convex relaxation for (5):

$$\begin{array}{lll}
\min & x_0 & \\
\text{subject to} & x = \displaystyle\sum_{v \in Y_j} x^{jv}, \;\; \text{all } j \in J & \\
& \beta_{jv} g^j\left(\dfrac{x^{jv}}{\beta_{jv}}, v\right) \le 0, \;\; \text{all } v \in Y_j,\ j \in J & \\
& \beta_{jv} x^L \le x^{jv} \le \beta_{jv} x^U, \;\; \text{all } v \in Y_j,\ j \in J & \\
& \displaystyle\sum_{v \in Y_j} \beta_{jv} = 1,\ \beta_{jv} \ge 0, \;\; \text{all } v \in Y_j,\ j \in J & \\
& L(\beta), \;\; x, x^{jv} \in \mathbb{R}^n, \text{ all } v \in Y_j,\ j \in J &
\end{array} \tag{13}$$

This is not a convex hull relaxation for (5) as a whole, but it provides a convex hull relaxation of each disjunction of (5).

Since β_{jv} can vanish, it is common in practice to use the constraint

$$(\beta_{jv} + \epsilon) g^j\left(\frac{x^{jv}}{\beta_{jv} + \epsilon}, v\right) \le 0, \;\; \text{all } v \in Y_j,\ j \in J$$

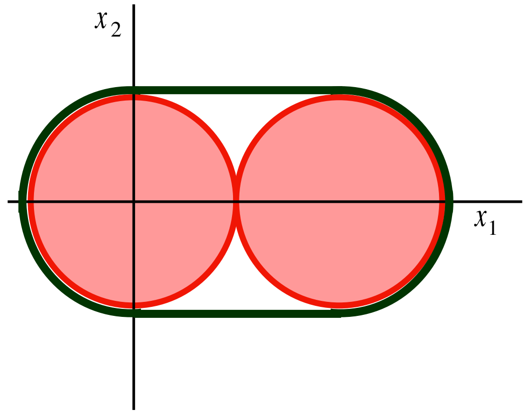

Fig. 3. The convex hull relaxation of a union of two discs.

The introduction of ϵ preserves convexity. Grossmann and Lee [6] suggest using $\epsilon = 10^{-4}$.

Example. The convex hull relaxation for the disjunction (10) is

$$
\begin{aligned}
&\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix} + \begin{bmatrix} x_{12} \\ x_{22} \end{bmatrix} \\
&\frac{x_{11}^2 + x_{21}^2}{\beta + \epsilon} \le \beta + \epsilon \\
&\frac{x_{12}^2 + x_{22}^2}{1 - \beta + \epsilon} - 4x_{12} + 3(1 - \beta + \epsilon) \le 0 \\
&\beta \begin{bmatrix} -1 \\ -1 \end{bmatrix} \le \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix} \le \beta \begin{bmatrix} 3 \\ 1 \end{bmatrix} \\
&(1 - \beta) \begin{bmatrix} -1 \\ -1 \end{bmatrix} \le \begin{bmatrix} x_{12} \\ x_{22} \end{bmatrix} \le (1 - \beta) \begin{bmatrix} 3 \\ 1 \end{bmatrix} \\
&0 \le \beta \le 1
\end{aligned}
$$

The bounds on x^{jv} are redundant in this case and can be dropped. Figure 3 shows the projection of the feasible set of this relaxation onto the original x-space.

Example. A big-M relaxation for the structural design problem is

$$
\begin{array}{ll}
\text{minimize} & x_0 \\
\text{subject to} & z_1 + z_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \le 0 \\
& \begin{array}{ll}
300 - z_1 \le 0 & 600 - z_1 \le 300\beta_1 \\
10 - x_1 \le 5(1 - \beta_1) & 10 - 2x_1 \le 0 \\
300 - z_2 \le 0 & 600 - z_2 \le 300\beta_2 \\
20 - x_2 \le 10(1 - \beta_2) & 20 - 2x_2 \le 0 \\
5 \le x_1 \le 10 & 10 \le x_2 \le 20 \\
\beta_1, \beta_2 \in [0, 1] &
\end{array}
\end{array}
\tag{14}
$$

The convex hull relaxation is

$$
\begin{array}{ll}
\text{minimize} & x_0 \\
\text{subject to} & \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_{11} + x_{12} \\ x_{21} + x_{22} \end{bmatrix} \; \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} z_{11} + z_{12} \\ z_{21} + z_{22} \end{bmatrix} \\
& z_1 + z_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \le 0 \\
& \begin{array}{ll}
z_{11} \ge 300\beta_1 & z_{12} \ge 600(1 - \beta_1) \\
x_{11} \ge 10\beta_1 & x_{12} \ge 10(1 - \beta_1) \\
z_{21} \ge 300\beta_2 & z_{22} \ge 600(1 - \beta_2) \\
x_{21} \ge 20\beta_2 & x_{22} \ge 20(1 - \beta_2) \\
\beta_1, \beta_2 \in [0, 1] &
\end{array}
\end{array}
\tag{15}
$$

The disjunctive programming problems (14) and (15) can be solved directly by branch and bound, or by MINLP with $\beta_j \in \{0, 1\}$ replacing $\beta_j \in [0, 1]$. The optimal solution is $(y_1, y_2) = (1, 2)$, meaning that the upper bar has size 1 and the lower bar size 2.

5 Logic-based Outer Approximation

One can use linear rather than convex nonlinear relaxations by modifying the outer approximation method for MINLP [3] to solve disjunctive programming problems, as shown by Türkay and Grossmann [14]. The drawback is that the linear relaxations must be updated and solved repeatedly.

Logic-based outer approximation solves a master problem containing first-order approximations of the disjuncts of (5) to obtain a value $\bar{y}$ for y. It then solves the nonlinear but convex subproblem (2) to obtain a corresponding value for x. The first-order approximations are computed about the values of x obtained in previous iterations. The process continues until optimal value of the master problem approximates the largest optimal subproblem value found so far.

Let (x^k, y^k) for $k = 1, \ldots, K$ be the solutions obtained by solving the master problem and subproblem in previous iterations. The master problem in iteration

$K+1$ can be written

$$
\begin{array}{ll}
\min & x_0 \\
\text{subject to} & \displaystyle\bigvee_{v \in Y_j} \left[\begin{array}{c} y_j = v \\ g^j(x^k, v) + \nabla g^j(x^k, v)(x - x^k) \le 0, \\ \text{all } k \in \{1, \ldots, K\} \text{ with } y_j^k = v \end{array} \right], \ \ \text{all } j \in J \\
& L(y), \ \ x \in \mathbb{R}^n
\end{array} \tag{16}
$$

Since the disjuncts in (16) are linear, the relaxations (8) and (13) are likewise linear. One can therefore solve (16) by applying a mixed integer programming method to a 0-1 formulation of (16). Again, either (8) or (13) can serve as a 0-1 formulation if the variables β_{jv} are treated as 0-1 variables. The solution y of (16) becomes y^{K+1}, and x^{K+1} is an optimal solution of the subproblem (2) with $\bar{y} = y^{K+1}$.

In practice it is advantageous to obtain a warm start by solving the subproblem for several values of $\bar{y}$ before solving the first master problem.

Example. The master problem for the structural design problem in iteration $K+1$ is

$$
\begin{array}{ll}
\text{minimize} & x_0 \\
\text{subject to} & z_1 + z_2 + (x_1^k + x_2^k)^2 + (x_2^k)^2 + 2(x_1^k + x_2^k)(x_1 - x_1^k) \\
& \quad +2(x_1^k + 2x_2^k)(x_2 - x_2^k) - x_0 \le 0, \ \ k = 1, \ldots K \\
& \left[\begin{array}{c} y_1 = 1 \\ 300 - z_1 \le 0 \\ 10 - x_1 \le 0 \end{array} \right] \vee \left[\begin{array}{c} y_1 = 2 \\ 600 - z_1 \le 0 \\ 10 - 2x_1 \le 0 \end{array} \right] \\
& \left[\begin{array}{c} y_2 = 1 \\ 300 - z_2 \le 0 \\ 20 - x_2 \le 0 \end{array} \right] \vee \left[\begin{array}{c} y_2 = 2 \\ 600 - z_2 \le 0 \\ 20 - 2x_2 \le 0 \end{array} \right] \\
& x_1, x_2 \ge 0
\end{array} \tag{17}
$$

Since the disjuncts other than the first are already linear, no first order approximation is needed for them. We solve (17) by solving its big-M formulation as an MILP:

$$
\begin{array}{lll}
\text{minimize} & x_0 & \\
\text{subject to} & \multicolumn{2}{l}{z_1 + z_2 + (x_1^k + x_2^k)^2 + (x_2^k)^2 + 2(x_1^k + x_2^k)(x_1 - x_1^k)} \\
& \multicolumn{2}{l}{\quad +2(x_1^k + 2x_2^k)(x_2 - x_2^k) - x_0 \le 0, \ \ k = 1, \ldots K} \\
& 300 - z_1 \le 0 & 600 - z_1 \le 300\beta_1 \\
& 10 - x_1 \le 5(1 - \beta_1) & 10 - 2x_1 \le 0 \\
& 300 - z_2 \le 0 & 600 - z_2 \le 300\beta_2 \\
& 20 - x_2 \le 10(1 - \beta_2) & 20 - 2x_2 \le 0 \\
& 5 \le x_1 \le 10 & 10 \le x_2 \le 20 \\
& \beta_1, \beta_2 \in \{0, 1\} &
\end{array} \tag{18}
$$

To obtain a warm start, we solve the subproblem (7) with $(\bar{y}_1, \bar{y}_2)$ set to $y^1 = (1,1)$ and $y^2 = (2,2)$,which yield $x^1 = (20,20)$ and $x^2 = (5,10)$. This results in the master problem

$$
\begin{array}{lll}
\text{minimize} & x_0 & \\
\text{subject to} & z_1 + z_2 + 2000 + 80(x_1 - 20) + 120(x_2 - 20) - x_0 \leq 0 & \\
& z_1 + z_2 + 325 + 30(x_1 - 5) + 50(x_2 - 10) - x_0 \leq 0 & \\
& 300 - z_1 \leq 0 \qquad\qquad 600 - z_1 \leq 300\beta_1 & \\
& 10 - x_1 \leq 5(1 - \beta_1) \qquad 10 - 2x_1 \leq 0 & \\
& 300 - z_2 \leq 0 \qquad\qquad 600 - z_2 \leq 300\beta_2 & \\
& 20 - x_2 \leq 10(1 - \beta_2) \qquad 20 - 2x_2 \leq 0 & \\
& 5 \leq x_1 \leq 10 \qquad\qquad 10 \leq x_2 \leq 20 & \\
& \beta_1, \beta_2 \in \{0,1\} &
\end{array}
\tag{19}
$$

The master problem has solution $\beta = (1,0)$ and $x_0 = 1375$, which corresponds to $y^3 = (1,2)$. This yields $x^3 = (10,10)$ and $x_0 = 1400$ in the subproblem. The first order expansion of the nonlinear constraint about x^3 is

$$z_1 + z_2 + 500 + 40(x_1 - 10) + 60(x_2 - 10) - x_0 \leq 0$$

This is added to the master problem, which now has solution $\beta = (1,0)$ corresponding to $y^4 = (1,2)$, and $x_0 = 1400$. Since the master problem and subproblems converge, the algorithm terminates with $y = (1,2)$.

6 Logic-based Benders Decomposition

When a constraint in the disjunctive programming formulation contains many disjuncts, the number of variables in the relaxations (8) and (13) can become quite large. This can be avoided by applying logic-based Benders decomposition to (5), which in effect uses a discrete relaxation of the problem and does not require an explicit formulation of the disjunctions [8, 11]. However, the convergence rate is unpredictable.

In logic-based Benders, the master problem consists of *Benders cuts* that contain only the discrete variables y_j. At any point in the algorithm, the Benders cuts partially describe the projection of the original problem's feasible set onto the y-space.

In iteration K the subproblem is (2) with $\bar{y}$ set to the solution y^K of the current master problem. Let λ^{Kj} be the vector of Lagrange multipliers associated with constraint j in the optimal solution of (2), and let x_0^K be the optimal value of (2). Since constraints with vanishing Lagrange multipliers are inactive in the subproblem, we can state the following: whenever $\bar{y}_j$ is set to y_j^K for all constraints j with $\lambda^{Kj} \neq 0$, the optimal value of the subproblem is still x_0^K. We generate a Benders cut that states this fact, and add it to the master problem for iteration $K+1$:

$$\begin{array}{ll} \min & z \\ \text{subject to} & \displaystyle\bigwedge_{\substack{j \\ \lambda^{kj} \neq 0}} (y_j = y_j^k) \Longrightarrow (z \geq x_0^k), \quad k = 1, \ldots, K \\ & L(y) \end{array} \tag{20}$$

where $\Longrightarrow$ means "implies." For each k the implication in (20) is the Benders cut generated in iteration k. The master problem is solved for y^{K+1}, and the process continues until the optimal value of (20) approximates the best subproblem value found so far.

The master problem can be solved by finite-domain constraint programming techniques or by converting it to an integer programming problem for an MILP solver.

In general, logic-based Benders cuts are obtained by solving the *inference dual* of the subproblem. This approach has been successfully applied to planning and scheduling problems in which the master problem is solved by integer programming and the subproblem by constraint programming [8, 9, 12]. There is little experience to date with continuous nonlinear subproblems, but decomposition is clearly more effective when most of the Lagrange multipliers vanish, since this results in stronger Benders cuts. When none of the multipliers vanish, the method reduces to exhaustive enumeration.

It is useful in practice to enhance the master problem with any known information about the y_js, both valid constraints and "don't be stupid" constraints that exclude feasible but no optimal solutions. Such constraints can often be deduced from a practical understanding of the problem domain.

Example. The initial master problem for the structural design problem is

$$\begin{array}{ll} \text{minimize} & z \\ \text{subject to} & y_1 \leq y_2 \\ & y_1, y_2 \in \{1, 2\} \end{array} \tag{21}$$

Note the don't-be-stupid constraint $y_1 \leq y_2$, which is based on the physical intuition that the top pillar will be no larger than the bottom pillar in any optimal solution. One optimal solution of (21) is $y^1 = (1, 1)$, with $z = -\infty$. Solving the subproblem (7) with $\bar{y} = (1, 1)$, we obtain Lagrange multipliers $\lambda^1 = (1, 60)$ and $\lambda^2 = (1, 100)$, with $x_0^1 = 1900$. Since both multipliers are nonzero, the master problem becomes

$$\begin{array}{ll} \text{minimize} & z \\ \text{subject to} & y_1 \leq y_2 \\ & y = (1, 1) \Longrightarrow (z \geq 1900) \\ & y_1, y_2 \in \{1, 2\} \end{array} \tag{22}$$

The next two iterations produce the additional Benders cuts

$$\begin{aligned} y = (1,2) &\Longrightarrow (z \ge 1400) \\ y = (2,2) &\Longrightarrow (z \ge 1525) \end{aligned}$$

When these are added to the master problem (22), we get $y^4 = (1,2)$ with $z = 1400$. The algorithm therefore terminates with $y = (1,2)$. In this small example the Benders approach is inefficient, since none of the Lagrange multiplers vanish, and each Benders cut excludes only one solution.

7 Branch and Bound with Convex Quasi-Relaxations

In the methods presented so far, the discrete variables need have no particular domain. However, in many applications the discrete variables are real-valued, as for example when they are discretized continuous variables. In such cases it may be advantageous to have a relaxation in both the x and y variables, so that one can branch on y_j's by splitting intervals. The solution of the relaxation would indicate where to split. Thus for example if $y_j \in [y_j^L, y_j^U]$ and the solution value of y_j in the relaxation lies between discrete values $v, v' \in Y_j$, one would split the interval into $[y_j^L, v]$ and $[v', y_j^U]$. If the solution value of y_j lies at a bound y_j^L or y_j^U, no more splitting is necessary. The relaxation may therefore accelerate the search not only by providing bounds, but by providing split points that lead more quickly to feasible solutions.

This strategy is practical, however, only when a *convex* relaxation involving the y variables is available. Such a relaxation normally cannot be obtained by relaxing y_j's domain Y_j to a continuous interval, since the resulting problem is in general nonconvex.

Even when a convex relaxation is unavailable, however, it may be possible to construct a convex *quasi-relaxation* that is equally useful for obtaining lower bounds. A quasi-relaxation of a problem $\min\{f(x) \mid x \in S\}$ is a problem $\min\{f'(x) \mid x \in S'\}$ with the property that for any $x \in S$, there exists an $x' \in S'$ for which $f(x') \le f(x)$. It is clear that the optimal value of the quasi-relaxation, if it exists, provides a valid lower bound on the optimal value of the original problem.

The following theorem provides conditions under which one may construct a convex quasi-relaxation for problem (1). Let function $g(x, y_j)$ be convex in x when $g(x, v)$ is convex for any $v \in Y_j$. Also let $g(x, y_j)$ be *semihomogeneous* in x if

$$\begin{aligned} & g(\alpha x, v) \le \alpha g(x, v) \;\text{ for all } \alpha \in [0,1], x \in \mathbb{R}^n, v \in Y_j && (a) \\ & g(0, y_j) = 0 \;\text{ for all } y_j \in Y_j && (b) \end{aligned} \qquad (23)$$

Theorem 2. *Suppose each $g_i^j(x, y_j)$ in (1) is convex in x and satisfies at least one of the following conditions:*

1. *$g_i^j(x, y_j)$ is convex.*

2. $g_i^j(x, y_j)$ *is semihomogeneous in* x *and concave in* y_j*.*

Let (i, j) *belong to* J_1 *when* g_i^j *satisfies condition 1 and* J_2 *otherwise. Suppose also that* $x^L \leq x \leq x^U$ *and* $y^L \leq y \leq y^U$*. Then the following is a convex quasi-relaxation of (1):*

$$\begin{array}{lll}
\textit{minimize } x_0 & & \\
\textit{subject to } g_i^j\left(x, \alpha_j y_j^L + (1-\alpha_j) y_j^U\right) \leq 0, \;\; \textit{all } (i,j) \in J_1 & (a) & \\
\qquad g_i^j(x^{j1}, y_j^L) + g_j^i(x^{j2}, y_j^U) \leq 0, \;\; \textit{all } (i,j) \in J_2 & (b) & \\
\qquad \alpha_j x^L \leq x^{j1} \leq \alpha_j x^U, \;\; \textit{all } j \in J & (c) & (24)\\
\qquad (1-\alpha_j) x^L \leq x^{j2} \leq (1-\alpha_j) x^U \;\; \textit{all } j \in J & (d) & \\
\qquad x = x^{j1} + x^{j2}, \;\; \textit{all } j \in J & (e) & \\
\qquad x^{j1}, x^{j2} \in \mathbb{R}^n, \;\; \alpha_j \in [0,1], \;\; \textit{all } j \in J & &
\end{array}$$

Furthermore, if each α_j *is 0 or 1 in the optimal solution of (24), then (24) has the same optimal value as (1).*

Proof. We first observe that (24) is convex. Constraint (a) is convex because $g_i^j(x, y_j)$ is convex for $(i, j) \in J_1$, and a convex function composed with an affine function is convex. Constraint (b) is convex because $g_i^j(x, y_j)$ is convex when y_j is fixed. The remaining constraints are linear.

To show that (24) is a quasi-relaxation, take any feasible solution $(\bar{x}, \bar{y})$ of (1) and construct a feasible solution for (24) as follows. For each $j \in J$ choose $\alpha_j \in [0, 1]$ so that $\bar{y}_j = \alpha_j y_j^L + (1-\alpha_j) y_j^U$. Set $x^{j1} = \alpha_j \bar{x}_j$, $x^{j2} = (1-\alpha_j)\bar{x}$, and $x = x^{j1} + x^{j2}$. To see that this produces a feasible solution of (24), note first that constraints (a) and (c)-(e) are satisfied by construction. Constraint (b) is also satisfied, since for $(i, j) \in J_2$ we have

$$\begin{aligned}
&g_i^j(x^{j1}, y_j^L) + g_i^j(x^{j2}, y_j^U) = g_i^j\left(\alpha_j \bar{x}, y_j^L\right) + g_i^j\left((1-\alpha_j)\bar{x}, y_j^U\right) \\
&\leq \alpha_j g_i^j\left(\bar{x}, y_j^L\right) + (1-\alpha_j) g_i^j\left(\bar{x}, y_j^U\right) \leq g_i^j\left(\bar{x}, \alpha_j y_j^L\right) + g_i^j\left(\bar{x}, (1-\alpha_j) y_j^U\right) \\
&= g_i^j(\bar{x}, \bar{y}^j) \leq 0
\end{aligned}$$

where the first inequality is due to the semihomogeneity of $g_i^j(x, y_j)$ in x, the second to the concavity of $g_i^j(x, y_j)$ in y_j, and the third to the feasibility of $(\bar{x}, \bar{y}_j)$ in (1). Also the objective function value of (24) is less than or equal to (in fact equal to) that of (1), since $x_0 = \bar{x}_0$. Thus (24) is a convex quasi-relaxation of (1).

Finally, when $\alpha_j = 1$ we have $x^{j1} = x$ and $x^{j2} = 0$, and similarly if $\alpha_j = 0$. It easy to verify, using the semihomogeneity of $g_i^j(x, y_j)$ in x, that (24) reduces to (1) when each $\alpha_j \in \{0, 1\}$ and therefore has the same optimal value. This completes the proof.

Let $g(x, y_j)$ be homogeneous in x when $g(\alpha x, y_j) = \alpha g(x, y_j)$ for all $\alpha \in [0, 1], y_j \in Y_j$.

Corollary 1. *Theorem 2 holds in particular when each* $g_i^j(x, y_j)$ *is either (a) convex or (b) homogeneous in* x *and concave in* y_j.

If the global optimization problem (1) satisfies the conditions of Theorem 2, it can be solved by branch and bound as follows. Each node of the search tree is processed as in the algorithm below, where z^U is the value of the best feasible solution found so far (initially $z^U = \infty$), and $[y_j^L, y_j^U]$ is the interval in which y_j is currently contrained to lie (where $y_j^L, y_j^U \in Y_j$). Initially the only unprocessed node is the root node, which is processed first.

1. Compute an optimal solution $\bar{x}$, $\bar{x}^{j1}$, $\bar{x}^{j2}$, $\bar{\alpha}_j$ (for $j \in J$) of the convex quasi-relaxation (24) at the current node. Set $\bar{y}_j = \bar{\alpha}_j y_j^L + (1 - \bar{\alpha}_j) y_j^U$.
2. If $\bar{x}_0 \geq z^U$, go to an unprocessed node and begin with step 1.
3. If some $\bar{\alpha}_j \notin \{0, 1\}$, let v, v' be the values in $Y_j \cap [y_j^L, y_j^U]$ on either side of $\bar{y}_j$ that are closest to $\bar{y}_j$. (Possibly v or v' is identical to $\bar{y}_j$.) Branch on y_j by creating an unprocessed node at which $y_j \in [y_j^L, v]$ and a second unprocessed node at which $y_j \in [v', y_j^U]$. Go to an unprocessed node and begin with step 1.
4. The solution $(\bar{x}, \bar{y})$ is feasible in (1). Set $z^U = \min\{\bar{x}_0, z^U\}$. Go to an unprocessed node and start with step 1.

The algorithm terminates when no unprocessed nodes remain. To ensure termination, one should fix α_j at 0 or 1 (either yields the same reuslt) whenever $y_j^L = y_j^U$.

Example. We return to the structural design problem (4) and solve it for continuous y_j rather than over the finite set $\{1, 2\}$. For illustrative purposes we discretize y_j by setting each $Y_j = \{0, 0.1, 0.2, \ldots, 3.0\}$, although a finer resolution could be used.

We first check that (4) satisfies the conditions of Theorem 2. All the constraint functions are convex except $g_2^1(x, y_1) = 10 - x_1 y_1$ and $g_2^2(x, y_2) = 20 - x_2 y_2$. These are not semihomogeneous in x, since $g_2^j(0, y_j) \neq 0$, for example. However we can make them semihomogeneous (indeed, homogeneous) in x by replacing the constant term with s_j and adding the bounds $s_1 \in [10, 10]$ and $s_2 \in [20, 20]$. We also use the bounds on x_j introduced earlier. To simplify the problem we substitute $z_j = 300 y_j$ into the objective function. The problem is now

$$\begin{aligned}
\text{minimize } & z_0 \\
\text{subject to } & 300y_1 + 300y_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \leq 0 \\
& s_1 - x_1 y_1 \leq 0 \\
& s_2 - x_2 y_2 \leq 0 \\
& x_1 \in [5, 10],\ x_2 \in [10, 20] \\
& s_1 \in [10, 10],\ s_2 \in [20, 20] \\
& y_1, y_2 \in \{0, 0.1, \ldots, 3\}
\end{aligned} \tag{25}$$

The convex quasi-relaxation (24) is

$$
\begin{array}{ll}
\text{minimize} & z_0 \\
\text{subject to} & 300y_1 + 300y_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \leq 0 \\
& \left(s_{11} - x_{11}y_1^L\right) + \left(s_{12} - x_{12}y_1^U\right) \leq 0 \\
& \left(s_{21} - x_{21}y_2^L\right) + \left(s_{22} - x_{22}y_2^U\right) \leq 0 \\
& 5\alpha_1 \leq x_{11} \leq 10\alpha_1, \qquad 5(1-\alpha_1) \leq x_{12} \leq 10(1-\alpha_1) \\
& 10\alpha_2 \leq x_{21} \leq 20\alpha_1, \qquad 10(1-\alpha_2) \leq x_{12} \leq 20(1-\alpha_2) \\
& 10\alpha_1 \leq s_{11} \leq 10\alpha_1, \qquad 10(1-\alpha_1) \leq s_{12} \leq 10(1-\alpha_1) \\
& 20\alpha_2 \leq s_{21} \leq 20\alpha_1, \qquad 20(1-\alpha_2) \leq s_{12} \leq 20(1-\alpha_2) \\
& \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_{11} + x_{12} \\ x_{21} + x_{22} \end{bmatrix}, \quad \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} s_{11} + s_{12} \\ s_{21} + s_{22} \end{bmatrix} \\
& y_j = \alpha_j y_j^L + (1-\alpha_j)y_j^U, \ \alpha_j \in [0,1] \ \ j = 1,2
\end{array}
$$

At this point we can reaggregate s_j, which allows it to be eliminated.

$$
\begin{array}{ll}
\text{minimize} & z_0 \\
\text{subject to} & 300y_1 + 300y_2 + (x_1 + x_2)^2 + x_2^2 - x_0 \leq 0 \\
& 10 - x_{11}y_1^L - x_{12}y_1^U \leq 0 \\
& 20 - x_{21}y_1^L - x_{22}y_1^U \leq 0 \\
& 5\alpha_1 \leq x_{11} \leq 10\alpha_1, \qquad 5(1-\alpha_1) \leq x_{12} \leq 10(1-\alpha_1) \\
& 10\alpha_2 \leq x_{21} \leq 20\alpha_1, \quad 10(1-\alpha_2) \leq x_{12} \leq 20(1-\alpha_2) \\
& \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_{11} + x_{12} \\ x_{21} + x_{22} \end{bmatrix} \\
& y_j = \alpha_j y_j^L + (1-\alpha_j)y_j^U, \ \alpha_j \in [0,1] \ \ j = 1,2
\end{array}
$$

Figure 4 shows the first few nodes of the branching tree. A globally optimal solution $y = (1.1, 2.0)$ with objective value $z_0 = 1394.5$ is found after processing 63 nodes, out of a possible 31×31 solutions. A more precise solution is $y = (1.126, 1.972)$ with $z_0 = 1394.1$.

8 Truss Structure Design

We conclude with a more realistic truss structure design problem. The model presented here is a simplified version of that described in [1].

The notation is illustrated in Fig. 5. A truss structure consists of a number of bars j joined at nodes, each bar having length h_j and a cost of c_j per unit volume. Each node can move in a specified number of directions. Thus if the problem is solved in three dimensions, there are at most three degrees of freedom at each node. Each degree of freedom i is associated with a load ℓ_i. The decision variables are the thickness (cross-sectional area) y_j of the bars. Other variables are the elongation s_j of bar j, the tension (pulling force) f_j on bar j, and the displacement x_i along degree of freedom i. The objective is to minimize the cost

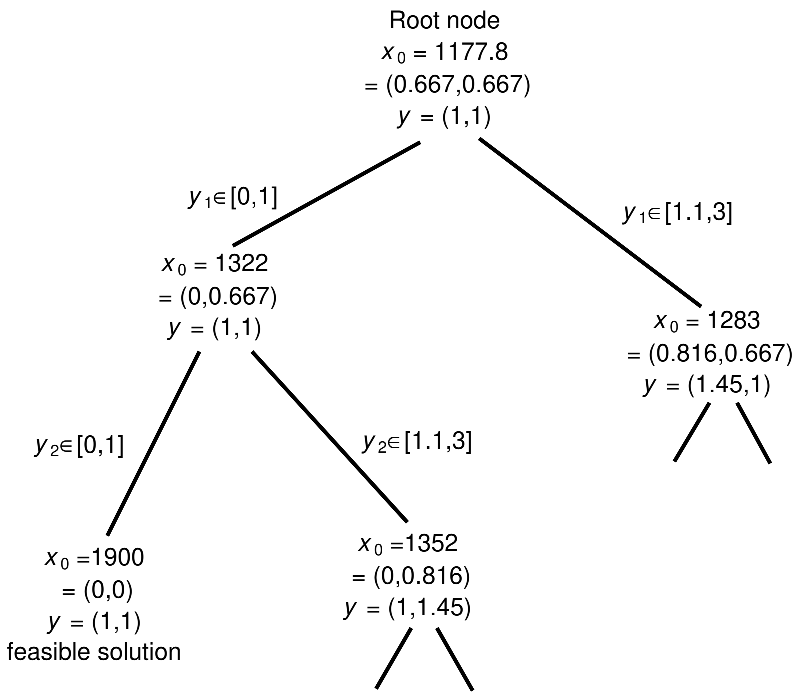

Fig. 4. Beginning of a BBCQ branch-and-bound tree for the structural design example.

of the bars subject to bounds on elongation and displacement. Stress bounds also exist and are factored into the elongation bounds. The model is

$$
\begin{array}{lll}
\text{minimize} \ \displaystyle\sum_j c_j h_j y_j & & \text{cost of bars} \\
\text{subject to} \ \dfrac{E_j}{h_j} y_j s_j = f_j, \ \ \text{all } j & & \text{Hooke's law} \\
\qquad \displaystyle\sum_j f_j \cos\theta_{ij} = \ell_i, \ \ \text{all } i & & \text{equilibrium equations} \\
\qquad \displaystyle\sum_i x_i \cos\theta_{ij} = s_j, \ \ \text{all } j & & \text{compatibility equations} \\
\qquad s_j^L \le s_j \le s_j^U, \ \ \text{all } j & & \text{elongation bounds} \\
\qquad x_j^L \le x_j \le x_j^U, \ \ \text{all } j & & \text{displacement bounds} \\
\qquad y_j \in Y_j, \ \ \text{all } j & & \text{discrete thicknesses}
\end{array}
\tag{26}
$$

where E_j in Hooke's law is the modulus of elasticity for bar j. Since structural bars are generally available only in certain thicknesses, the variables y_j can be regarded as discrete.

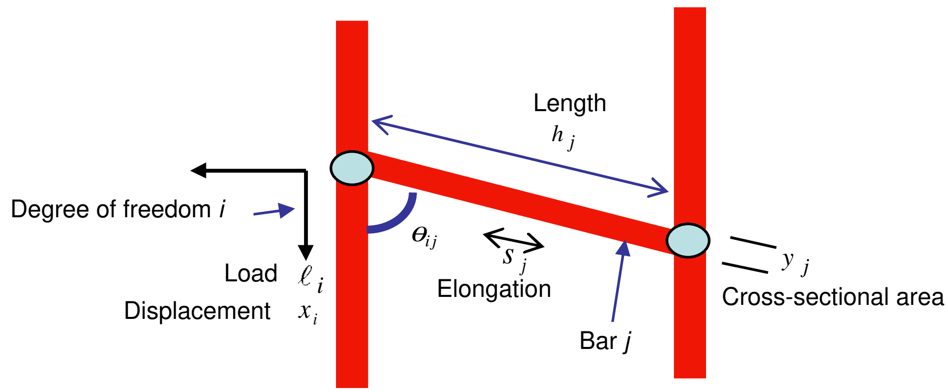

Fig. 5. Notation for a truss structure design problem.

Since the problem becomes convex (in fact, linear) when variables y_j are fixed, it is amenable to a logic-based method. We will apply disjunctive programming and BBCQ.

First we develop the disjunctive programming approach, using convex hull relaxations. A disjunctive representation of (26) is

$$\begin{array}{lll}
\text{minimize} & \displaystyle\sum_j z_j & \text{cost of bars} \\
\text{subject to} & \displaystyle\bigvee_{v \in Y_j} \begin{bmatrix} y_j = v \\ z_j \geq c_j h_j v \\ \dfrac{E_j}{h_j} v s_j = f_j \end{bmatrix}, \ \text{all } j & \text{cost, Hooke's law} \\
& \displaystyle\sum_j f_j \cos\theta_{ij} = \ell_i, \ \text{all } i & \text{equilibrium equations} \\
& \displaystyle\sum_i x_i \cos\theta_{ij} = s_j, \ \text{all } j & \text{compatibility equations} \\
& s_j^L \leq s_j \leq s_j^U, \ \text{all } j & \text{elongation bounds} \\
& x_j^L \leq x_j \leq x_j^U, \ \text{all } j & \text{displacement bounds}
\end{array} \tag{27}$$

Using convex hull relaxations of the disjunctions, we obtain the following convex relaxation of (27):

$$
\begin{aligned}
\text{minimize } & \sum_j z_j \\
\text{subject to } & z_j = \sum_{v \in Y_j} z_{jv}, \quad s_j = \sum_{v \in Y_j} s_{jv}, \quad f_j = \sum_{v \in Y_j} f_{jv}, \ \text{ all } j \\
& z_{jk} \geq c_j h_j v \beta_{jv}, \ \text{ all } v \in Y_j, \text{ all } j \\
& \frac{E_j}{h_j} v s_{jv} = f_{jv}, \ \text{ all } v \in Y_j, \text{ all } j \\
& \sum_j f_j \cos\theta_{ij} = \ell_i, \ \text{ all } i \\
& \sum_i x_i \cos\theta_{ij} = s_j, \ \text{ all } j \\
& \beta_{jv} s_j^L \leq s_{jv} \leq \beta_{jv} s_j^U, \ \text{ all } j \\
& x_j^L \leq x_j \leq x_j^U, \ \text{ all } j \\
& \sum_{v \in Y_j} \beta_{jv} = 1, \quad \beta_{jv} \geq 0 \ \text{ all } v \in Y_j, \text{ all } j
\end{aligned}
\tag{28}
$$

The relaxation can be simplified, in part by summing each instance of Hooke's law over all $v \in Y_j$.

$$
\begin{aligned}
\text{minimize } & \sum_j \sum_{v \in Y_j} c_j h_j v \beta_{jv} \\
\text{subject to } & \frac{E_j}{h_j} \sum_{v \in Y_j} v s_{jv} = f_j, \ \text{ all } j \\
& \sum_j f_j \cos\theta_{ij} = \ell_i, \ \text{ all } i \\
& \sum_i x_i \cos\theta_{ij} = s_j, \ \text{ all } j \\
& \beta_{jv} s_j^L \leq s_{jv} \leq \beta_{jv} s_j^U, \ \text{ all } j \\
& x_j^L \leq x_j \leq x_j^U, \ \text{ all } j \\
& \sum_{v \in Y_j} \beta_{jv} = 1, \quad \beta_{jv} \geq 0 \ \text{ all } v \in Y_j, \text{ all } j
\end{aligned}
\tag{29}
$$

The disjunctive problem (27) can be solved as an MINLP by solving (29) with the integrality condition $\beta_{jv} \in \{0, 1\}$. This MINLP model was in fact proposed by Ghattas, Voudouris and Grossmann [4, 5].

We now develop a BBCQ approach to solving (27). Note first that the model (26) satisfies the conditions of Theorem 2, since all of the constraints are convex (in fact, linear) except Hooke's law, which is convex (in fact, linear) when the

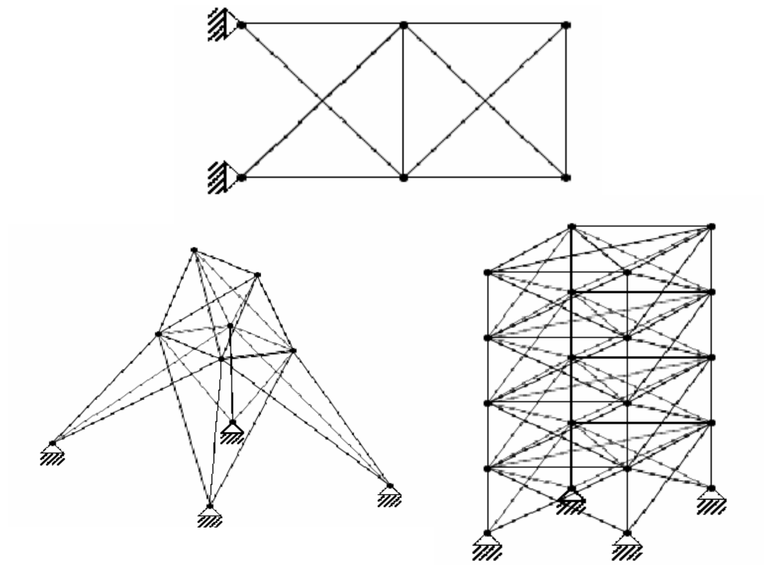

Fig. 6. A 10-bar cantilever truss, 25-bar electrical treansmission tower, and 72-bar building.

y_js are fixed. In addition, the constraint function in Hooke's law is homogeneous in the continuous variables s_j, f_j and concave (in fact, linear) in the discrete variable y_j. The convex quasi-relaxation (24) therefore becomes

$$\begin{aligned}
\text{minimize} \;\; & \sum_j c_j h_j y_j \\
\text{subject to} \;\; & \frac{E_j}{h_j}\left(y_j^L s_{j1} + y_j^U s_{j2}\right) = f_j, \;\; \text{all } j \\
& \sum_j f_j \cos\theta_{ij} = \ell_i, \;\; \text{all } i \\
& \sum_i x_i \cos\theta_{ij} = s_j, \;\; \text{all } j \\
& \alpha_j s_j^L \le s_{j1} \le \alpha_j s_j^U, \;\; \text{all } j \\
& (1-\alpha_j) s_j^L \le s_{j2} \le (1-\alpha_j) s_j^U, \;\; \text{all } j \\
& x_j^L \le x_j \le x_j^U, \;\; \text{all } j \\
& x_j = x_{j1} + x_{j2}, \;\; \text{all } j \\
& y_j = \alpha_j y_j^L + (1-\alpha_j) y_j^U, \;\; \text{all } j \\
& \alpha_j \in [0,1], \;\; \text{all } j
\end{aligned} \tag{30}$$

Table 1. Summary of solution times in seconds for MILP and BBCQ applied to truss structure design problems. When there two "loads" (i.e., two sets of loads applied to each degree of freedom), the structure is required to withstand each of the two loads, and a constraint set is written for each one. BBCQ was enhanced with some simple cutting planes when solving the cantilever and tower problems.

Problem Instance		*MILP*	*BBCQ*
10-bar	1 load	1.3	0.3
cantilever	1 load, wider stress bounds	1.6	0.3
truss	1 load, still wider stress bounds	2.6	1.2
	1 load, still wider stress bounds	2.6	1.4
	2 loads	23.6	5.8
	1 load, displacement bounds	1089.4	67.5
	2 loads, displacement bounds	13743.9	1654.0
25-bar transmission tower	2 loads	271.7	225.8
Building	72 bars, 2 loads	12692.7	207.9
	90 bars, 2 loads	*	168.9
	108 bars, 2 loads	*	329.4

*No solution after 20 hours (72,000 seconds).

Bollapragada et al. [1] applied both the MILP and BBQC methods to the structural design problems illustrated in Fig. 6. Each structural bar had 11 possible thicknesses. Symmetries in the transmission tower and buildings were exploited to reduce the number of variables. Computational results are summarized in Table 1. MILP was implemented in CPLEX, and BBCQ in C with calls to the CPLEX linear programming solver. All problems were solved on a Sun Sparc Ultra work station.

These results suggest that BBCQ can carry a substantial advantage over a disjunctive approach when the constraint functions satisfy the conditions of Theorem 2.

References

1. S. Bollapragada, O. Ghattas and J. N. Hooker, Optimal design of truss structures by mixed logical and linear programming, *Operations Research* **49** (2001) 42-51.
2. V. Chandru and J. N. Hooker, *Optimization Methods for Logical Inference*, John Wiley & Sons (New York, 1999).
3. M. A. Duran and I. E. Grossmann, An outer-approximation algorithm for a class of mixed-integer nonlinear programs, *Mathematical Programming* **36** (1986) 307.
4. O. Ghattas and I. E. Grossmann, MINLP and MILP strategies for discrete sizing structural optimization problems, *Proceeedings of the ASCE 10th Conference on Electronic Communication*, Indianapolis (1991).
5. I. E. Grossmann, V. T. Voudouris, and O. Ghattas, Mixed-integer linear programming formulations of some nonlinear discrete design optimization problems, in C. A.

Floudas and P. M. Pardalos, eds., *Recent Advances in Global Optimization*, Princeton University Press (1992).
6. I. E. Grossmann and S. Lee, Generalized disjunctive programming: Nonlinear convex hull relaxation, Carnegie Mellon University (2001) submitted.
7. J. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms*, Vol. 1 (Springer-Verlag, 1993).
8. J. N. Hooker, *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*, John Wiley & Sons (2000).
9. J. N. Hooker, Logic-based Benders decomposition for planning and scheduling, manuscript, GSIA, Carnegie Mellon University 2003).
10. J. N. Hooker and M. A. Osorio, Mixed logical/linear programming, *Discrete Applied Mathematics* **96-97** (1999) 395-442.
11. J. N. Hooker and G. Ottosson, Logic-based Benders decomposition, *Mathematical Programming* **96** (2003) 33-60.
12. Jain, V., and I. E. Grossmann, Algorithms for hybrid MILP/CP models for a class of optimization problems, *INFORMS Jorunal on Computing* **13** (2001) 258–276.
13. R. Stubbs and S. Mehrotra, A branch-and-cut method for 0-1 mixed convex programming, *Mathematical Programming* **86** (1999) 515-532.
14. M. Türkay and I. E. Grossmann, Logic-based outer-approximation algorithm for MINLP optimization of process flowsheets, *Computers and Chemical Engineering* **19** (1996) S131-S136.