An Improved Branch-and-Cut Algorithm for Mixed-Integer Nonlinear Systems Optimization Problem

Yushan Zhu, Yijiong Hu, and Hao Wu
Dept. of Chemical Engineering, Tsinghua University, Beijing 100084, China

Masaru Nakaiwa
Energy-Efficient Chemical Systems Group, National Institute of Advanced Industrial Science and Technology, Tsukuba Centra-5, Tsukuba, Japan

DOI 10.1002/aic.11616
Published online November 3, 2008 in Wiley InterScience (www.interscience.wiley.com).

Recent advances for global optimization and dynamic optimization of the mixed-integer systems have created an increasing demand for efficient and robust numerical algorithms for mixed-integer nonlinear programming (MINLP) problem. In this article, an improved branch-and-cut algorithm for 0-1 MINLP problems has been proposed by using our former critical finding (Zhu and Kuno, Ind Eng Chem Res. 2006;45:187–196) of the disjunctive cutting plane for 0-1 MINLP. By virtue of the polyhedral outer approximation of the mixed-integer nonlinear set of the original MINLP problem at each enumeration node, a lift-and-project cutting planes that is valid for the original MINLP problem and cuts the fractional solution away can be generated by using the cut generating, lifting, and strengthening approach for MILP problem of Balas et al. (Math Program. 1993;58:295–324). The specific implementation issues of the cutting planes are discussed and incorporated into the algorithmic development of a branch-and-cut algorithm. The efficiency of the improved disjunctive cutting plane is demonstrated by the computational results for 11 systems optimization problems, and it is implied that the proposed branch-and-cut algorithm is very promising for practical MINLP problems as the interior-point solvers for nonlinear programming problems with many constraints are becoming mature gradually. © 2008 American Institute of Chemical Engineers *AIChE J,* 54: 3239–3247, 2008
Keywords: systems optimization, MINLP, branch-and-cut, cutting plane

Introduction

During the last two decades, mixed-integer optimization has achieved wide and successful applications in process systems engineering, such as the process or product design problems and the batch process scheduling problems.[1–5] Mixed-integer optimization provides a powerful superstructure framework for mathematically modeling many optimization problems that involve both discrete and continuous variables, and such problems can be classified into mixed-integer linear programming (MILP) or mixed-integer nonlinear programming (MINLP) ones according to their linear or nonlinear characteristics respectively. Mixed-integer linear programming methods and computer codes have been available and applied to many practical problems for more than 30 years,[6] but those methods were mainly developed by operations researchers. The major

Correspondence concerning this article should be addressed to Y. Zhu at yszhu@tsinghua.edu.cn.

contribution of chemical engineers in this area has been to discover problems and build modeling in the form of MILP, such as the batch process scheduling problems.[7] As the nonlinearity is deeply rooted in the chemical processes, MINLP provides much greater flexibility for tacking a large variety of problems. Different from the MILP methods, chemical engineers have played a prominent role in the development of MINLP methods for modeling and algorithmic approaches.[1,8]

The most basic form of an MINLP problem when represented in algebraic form is as follows[3]:

$$\begin{aligned} \min \quad & Z = f(x,y) \\ \text{s.t.} \quad & g_j(x,y) \le 0 \quad j \in J \\ & x \in X, \quad y \in Y \end{aligned} \tag{MINLP1}$$

where $f(\cdot)$, $g(\cdot)$ are convex, differentiable functions, and at least one of them is nonlinear, J is the index set of inequalities, and x and y are the continuous and discrete variables, respectively. The set X is commonly assumed to be a convex compact set, e.g., $X = \{x | x \in \Re^n, Ax \le b, x^L \le x \le x^U\}$; The discrete set Y corresponds to a polyhedral set of integer points, $Y = \{y | y \in Z^q, Gy \le d\}$, which in most applications is restricted to 0-1 values, $y \in \{0,1\}^q$. Duran and Grossmann[9] pioneered the development of practical algorithms for MINLP problems whose main characteristics defining the mathematical structures are linearity of the binary variables and convexity of the nonlinear functions that involve only continuous variables, their well-known out-approximation approach was further extended by Fletcher and Leyffer[10] to solve the more general MINLP problems represented by (MINLP1). Generally, the out-approximation approach solves the MINLP problem by computing a nonlinear program (NLP) primal subproblem at some particular integer combination and a MILP master problem alternatively, to get the upper and lower bounds of the MINLP original problem, respectively. The generalized Benders decomposition (GBD) method[1,11] for MINLP problems is similar to the out-approximation approach. The difference arises in the definition of the MILP master problem. At each round, the GBD method just generates one cut to update the constraint set of the MILP master problem, and then it always yields a loose lower bound compared with that of the outer approximation approach. But, the computational cost of the outer approximation approach for solving the MILP master problem is greater since the number of constraints added per iteration is equal to the number of nonlinear constraints plus the nonlinear objective. The extended cutting plane method[12] for MINLP problems is an extension of Kelly's cutting plane algorithm for convex NLP,[13] which does not rely on the use of NLP subproblems. Note that since the discrete and continuous variables are converged simultaneously, the extended cutting plane method may require a large number of iterations.

The popular branch-and-bound approach for MILP problems can be extended to solve MINLP problems directly by replacing the linear programming (LP) subproblem by a NLP subproblem at each node in an enumeration tree.[14–16] It is now understood that the branch-and-bound method can benefit considerably from the strong relaxations of the NLP subproblem since the integral gap for MINLP problem is always greater than that for MILP problems due to the nonlinearity.[17,18] Stubbs and Mehrotra[19] generalized the disjunctive cutting plane method[20] or the lift-and-project cutting plane method[21,22] for 0-1 MILP problems and extended their method into a branch-and-cut algorithm to solve the MINLP problems. But, the disjunctive cutting plane developed by Stubbs and Mehrotra[19] is obtained by solving a convex projection problem, which is computationally expensive and always encounters nondifferential problems. Recently, Zhu and Kuno[23] developed a new disjunctive cut generation approach where the disjunctive cut is produced by solving a linear programming problem rather than a nonlinear programming problem. The critical finding of Zhu and Kuno[23] approach is that a lift-and-project cutting plane that is valid for the original MINLP problem and cuts the fractional solution away can be generated based on the polyhedral outer approximation of the mixed-integer nonlinear set of the original MINLP problem at the current node. In this article, the cut generating, lifting, and strengthening approach proposed by Balas et al.[20] for MILP problem is used to replace the cut generating linear programming in our former work, and the specific implementation issues are discussed and incorporated into the algorithmic development to handle large-scale MINLP problems. In terms of the computational results, it is efficient to incorporate general cuts within branch-and-cut for MINLP problems, which is similar to the conclusion claimed by Balas et al.[24] for MILP problem.

Basic Elements of 0-1 MINLP

The 0-1 MINLP problem considered in this article can be formulated by

$$\begin{aligned} \min \quad & Z = c^T x \\ \text{s.t.} \quad & g_i(x,y) \le 0 \quad i = 1,...,l \\ & Ax + Gy \le b \\ & x \in \Re^n, \quad y \in \{0,1\}^q \end{aligned} \tag{MINLP2}$$

where the constant vectors and matrices for the linear constraint set are defined as $c \in \Re^n, A \in \Re^{m \times n}, G \in \Re^{m \times q}$, $b \in \Re^m$, and the convex compact set of the continuous variables and the polyhedral set of the integer variables are all incorporated into this linear inequality set. The general 0-1 MINLP problem described by (MINLP1) can be transformed into (MINLP2) by minimizing an additional continuous variable and transforming the mixed-integer objective function to become an additional mixed-integer inequality.[23] Denote by $F_0, F_1 \subseteq \{1,...,q\}$, the sets of binary variables that have been fixed at 0 and 1, respectively, then the MINLP subproblem that can be considered for problem (MINLP2) at node (F_0, F_1) in a branch-and-bound enumeration tree can be formulated by

$$\begin{aligned} \min \quad & Z_{BB} = c^T x \\ \text{s.t.} \quad & g_i(x,y) \le 0 \quad i = 1,...,l \\ & Ax + Gy \le b \\ & y_j = 0, \quad j \in F_0 \\ & y_j = 1, \quad j \in F_1 \\ & y_j \in \{0,1\}, \quad j \in \{1,...,q\} \backslash \{F_0 \cup F_1\} \\ & x \in \Re^n, \quad y \in \Re^q \end{aligned} \tag{MINLP(F_0, F_1)}$$

The basic NLP continuous relaxation subproblem for problem (MINLP(F_0,F_1)) can be formulated as follows:

$$\begin{aligned}
\min \quad & Z_{\mathrm{LBB}} = c^T x \\
\text{s.t.} \quad & g_i(x,y) \le 0 \quad i = 1,...,l \\
& Ax + Gy \le b \\
& y_j = 0, \quad j \in F_0 \qquad (\mathrm{NLP}(F_0,F_1)) \\
& y_j = 1, \quad j \in F_1 \\
& 0 \le y_j \le 1, \quad j \in \{1,...,q\}\backslash\{F_0 \cup F_1\} \\
& x \in \Re^n, \quad y \in \Re^q
\end{aligned}$$

While $F_0 = \phi, F_1 = \phi$, i.e., the root node of the enumeration tree, the feasible region of the continuous relaxation problem (NLP(F_0,F_1)) is defined as

$$C = \left\{ (x,y) \in \Re^{n+q} \left| \begin{array}{l} g_i(x,y) \le 0 \quad i = 1,...,l \\ Ax + Gy \le b \\ 0 \le y_j \le 1, \quad j = 1,...,q \end{array} \right. \right\} \qquad (1)$$

To solve the continuous relaxation problem (NLP(F_0,F_1)), some assumptions are made as follows:

(A1) The continuous variables are bounded, i.e., $|x_i| \le L, i = 1,...,n$, which can be taken as a part of the constraints in problem (MINLP2) and L is a large positive number.

(A2) For any $(x,y) \in C, g_i(x,y) \ge -L, i = 1,...,l$.

(A3) A constraint qualification holds for the resulting NLP problem at each enumeration node (F_0,F_1).

With these assumptions, we know that problem (NLP(F_0,F_1)) is well-defined in the sense that if it is feasible and does not have unbounded optimal value, then it has an optimal solution and provides a valid lower bound for problem (MINLP(F_0,F_1)). If the solution to problem (NLP(F_0,F_1)) is satisfied with one of the following fathoming rules, then no branching is required, the current node has been fully explored and can be eliminated, as

(FR1) Problem (NLP(F_0,F_1)) is detected to be infeasible. In this case, the whole subtree starting at this node is infeasible and the node has been fathomed.

(FR2) Problem (NLP(F_0,F_1)) is feasible and all binary variables in the solution are taken values at either 0 or 1, then an upper bound on the optimum of problem (MINLP2) is obtained and no further branching is needed at this node and the node has been fathomed.

(FR3) The optimal value of problem (NLP(F_0,F_1)) is greater than or equal to the current best upper bound. In this case, the current node is fathomed.

The efficiency of the branch-and-bound approach can be greatly enhanced if the solution quality of problem (NLP(F_0,F_1)) can be improved by using some valid cutting planes at the current node. This leads to the combination of the cutting plane approach with the branch-and-bound or the branch-and-cut algorithm for MINLP. Let ($\bar{x}$, $\bar{y}$) be a solution to problem (NLP(F_0,F_1)). If any of the components of the binary variables $\bar{y}$ are not in {0, 1}, then we can add a valid inequality, such as $\alpha^T x + \beta^T y \le \gamma$, into the current feasible set such that this inequality is violated by ($\bar{x}$, $\bar{y}$). The key to the success of the branch-and-cut algorithm for MILP problem[20] is that the cut generated that cuts away ($\bar{x}$, $\bar{y}$) is not only valid at the current node but also can be lifted to be valid throughout the enumeration tree. We update C, the family of inequalities, i.e., $C = C \cup \{(x,y) \in \Re^{n+q} | \alpha^T x + \beta^T y \le \gamma\}$, to describe the current feasible set and a set of cutting planes that is added into C at the current node. Let

$$K(C,F_0,F_1) = \left\{ (x,y) \in C \left| \begin{array}{ll} y_j = 0 & \text{for} \quad j \in F_0 \\ y_j = 1 & \text{for} \quad j \in F_1 \end{array} \right. \right\} \qquad (2)$$

and let NLP(C,F_0,F_1) denote the continuous relaxation problem with the cutting planes in a branch-and-cut algorithm, as

$$\begin{aligned}
\min \quad & Z_{\mathrm{LBC}} = c^T x \\
\text{s.t.} \quad & (x,y) \in K(C,F_0,F_1) \qquad (\mathrm{NLP}(C,F_0,F_1))
\end{aligned}$$

This NLP is assumed to be either bounded from below with a finite minimum or infeasible at the current node by virtue of the aforestated assumptions. The solution quality of problem NLP(C,F_0,F_1) will be greatly enhanced compared with problem NLP(F_0,F_1) if the cutting planes are tight and their effects to improve the searching efficiency of the branch-and-bound framework can be implied by the aforementioned fathoming rules. The following sections will contribute to find such cutting planes for problem MINLP2.

Polyhedral Approximation of 0-1 MINLP

The convexity of the nonlinear functions in problem MINLP(F_0,F_1) is exploited by replacing them with supporting hyperplanes that are derived at ($\bar{x}$, $\bar{y}$), which is the solution to problem NLP(F_0,F_1). So, a linear approximation problem at ($\bar{x}$, $\bar{y}$) for problem MINLP(F_0,F_1) can be obtained by

$$\begin{aligned}
\min \quad & Z_{\mathrm{MILP}} = c^T x \\
\text{s.t.} \quad & g_i(\bar{x},\bar{y}) + \nabla g_i(\bar{x},\bar{y})^T \begin{pmatrix} x - \bar{x} \\ y - \bar{y} \end{pmatrix} \le 0 \quad i = 1,...,l \\
& Ax + Gy \le b \\
& y_j = 0, \quad j \in F_0 \qquad (\mathrm{MILP}(F_0,F_1)) \\
& y_j = 1, \quad j \in F_1 \\
& y_j \in \{0,1\}, \quad j \in \{1,...,q\}\backslash\{F_0 \cup F_1\} \\
& x \in \Re^n, \quad y \in \Re^q
\end{aligned}$$

where the original convex and differentiable functions are replaced by their first-order Taylor approximation at ($\bar{x}$, $\bar{y}$). Accordingly, a linear programming continuous relaxation problem corresponding to problem MILP(F_0,F_1) can be described by

$$\begin{aligned}
\min \quad & Z_{\mathrm{MILP}} = c^T x \\
\text{s.t.} \quad & g_i(\bar{x},\bar{y}) + \nabla g_i(\bar{x},\bar{y})^T \begin{pmatrix} x - \bar{x} \\ y - \bar{y} \end{pmatrix} \le 0 \quad i = 1,...,l \\
& Ax + Gy \le b \\
& y_j = 0, \quad j \in F_0 \qquad (\mathrm{LP}(F_0,F_1)) \\
& y_j = 1, \quad j \in F_1 \\
& 0 \le y_j \le 1, \quad j \in \{1,...q\}\backslash\{F_0 \cup F_1\} \\
& x \in \Re^n, \quad y \in \Re^q
\end{aligned}$$

The following theorem[23] defines the relationship between problem NLP(F_0,F_1) and problem LP(F_0,F_1), as

Theorem 1. (Ref. 23.) *Assume that problem* NLP(F_0,F_1) *achieves its optimal solution at* ($\overline{x}$, $\overline{y}$). *Then,* ($\overline{x}$, $\overline{y}$) *is also an optimal solution to problem* LP(F_0,F_1).

On the basis of Theorem 1 and the polyhedral approximation property for convex function, we have the following theorem, as

Theorem 2. *If there exists a valid cutting plane for* MILP(F_0,F_1) described by $\alpha^T x + \beta^T y \le \gamma$ *that cuts away* ($\overline{x}$, $\overline{y}$), *then, this cutting plane is also valid for* MINLP(F_0,F_1) *and cuts away* ($\overline{x}$, $\overline{y}$).

Proof. By virtue of the convexity property of the nonlinear function, we have $g_i(\bar{x},\bar{y}) + \nabla g_i(\bar{x},\bar{y})^T \begin{pmatrix} x-\bar{x} \\ y-\bar{y} \end{pmatrix} \le g_i(x,y)$, then the problem MILP($F_0$,$F_1$) is a valid polyhedral approximations of problem MINLP(F_0,F_1), this implies that the feasible points of problem MINLP(F_0,F_1) are also feasible for problem MILP(F_0,F_1), then a valid cutting plane for problem MILP(F_0,F_1) is also valid for problem MINLP(F_0,F_1). According to Theorem 1, this cutting plane cuts away ($\overline{x}$, $\overline{y}$) from the feasible region of problem MINLP(F_0,F_1). □

According to Theorem 1, ($\overline{x}$, $\overline{y}$) is a fractional solution to MILP(F_0,F_1). By virtue of Theorem 2, the various cutting planes that is valid for 0-1 MILP and cuts off ($\overline{x}$, $\overline{y}$),[6] such as the Gomory mixed-integer cutting planes, can be used to strengthen the bounding property of NLP(F_0,F_1) either in a book-keeping way that is only valid in the subtree starting from the current node, or they can be lifted to be valid throughout the enumeration tree.[24] In this article, we will focus on the disjunctive cutting plane since for each Gomory mixed-integer cutting plane there exists a disjunctive cutting plane that dominates it.[20]

Disjunctive Cutting Plane Generation Approach

In this section, the lift-and-project cut generating linear program (CGLP) proposed by Balas et al.[20,24] is briefly introduced for MILP(F_0,F_1) to produce valid disjunctive cutting plane that cuts away ($\overline{x}$, $\overline{y}$). To describe the CGLP clearly, we assume that the constraint set of LP(C,F_0,F_1) that is the polyhedral approximation form of NLP(C,F_0,F_1) in a branch-and-cut tree can be formulated by

$$\left\{ (x,y) \in \Re^{n+q} \middle| \begin{array}{l} \overline{A}x + \overline{G}y \le \overline{b} \\ y_j = 0, \quad j \in F_0 \\ y_j = 1, \quad j \in F_1 \end{array} \right\} \tag{3}$$

Note, the linear constraint set $\overline{A}x + \overline{G}y \le \overline{b}$ contains (i) the linear approximation inequalities of the nonlinear constraints; (ii) the original linear inequalities including the lower and upper bounds on continuous variables; (iii) the inequalities describing the lower and upper bounds of the binary variables; (iv) the valid inequalities added at the current node. So, we have $\overline{A} \in \Re^{(l+m+2q+p)\times n}$, $\overline{G} \in \Re^{(l+m+2q+p)\times q}$, and $\overline{b} \in \Re^{l+m+2q+p}$, where p is the number of valid inequalities added. The F_0 and F_1 are defined here to contain the binary variables fixed at the current node and those taken at the particular 0 or 1 values in the solution to NLP(C,F_0,F_1). The following subsections will contribute to the specific generating process for the lift-and-project cut.

Complementation process

Let $F = \{1, \ldots q\} \backslash (F_0 \cup F_1)$ denote the set of free variables at node (F_0,F_1). Without loss of generality, we assume F_1 is empty. If this is not true, each binary variable y_j in F_1 can be replaced by $1-y_j$, which amounts to replacing the column $\overline{G}_j$ and the right-hand side $\overline{b}$ with $-\overline{G}_j$ and $\overline{b} - \overline{G}_j$, respectively. Note the reformulated set F_0 will be the union of former sets F_0 and F_1, and the reformulated set F_1 becomes empty. After doing the complementation, the linear constrain set becomes

$$\left\{ (x,y) \in \Re^{n+q} \middle| \begin{array}{l} \tilde{A}x + \tilde{G}y \le \tilde{b} \\ y_j = 0, \quad j \in (F_0 \cup F_1) \end{array} \right\} \tag{4}$$

where, $\tilde{A}x + \tilde{G}y \le \tilde{b}$ is a compact form for

$$\overline{A}x + \sum_{j\in F} \overline{G}_j y_j + \sum_{j\in F_0} \overline{G}_j y_j + \sum_{j\in F_1} (-\overline{G}_j) y_j \le \overline{b} - \sum_{j\in F_1} \overline{G}_j \tag{5}$$

By virtue of the assumption that F_1 is empty, a reduced form of the linear constraint set can be obtained as

$$\left\{ (x,y) \in \Re^{n+|F|} \middle| A^F x + G^F y^F \le b^F \right\} \tag{6}$$

where G^F is the matrix obtained from $\tilde{G}$ by removing the columns $\tilde{G}_j$ for $j \in F_0$ and the rows corresponding to the lower and upper bounding inequalities for fixed binary variables, i.e., $-y_j \le 0$, and $y_j \le 1$ for $j \in F_0$. The right-hand side vector b^F is obtained from $\tilde{b}$ by removing the components corresponding to the rows removed from $\tilde{G}$, and similarly, the matrix A^F is obtained from $\tilde{A}$ by removing the rows corresponding to the rows removed from $\tilde{G}$. So, we have $A^F \in \Re^{(l+m+2|F|+p)\times n}$, $G^F \in \Re^{(l+m+2|F|+p)\times |F|}$, and $b^F \in \Re^{l+m+2|F|+p}$.

Cut generating linear program

The lift-and-project cuts can be derived by imposing the 0-1 condition on a single binary variable y_k such that $0 < \bar{y}_k < 1$. These cuts are valid inequalities for the convex hull of a disjunction of the form[25]

$$\begin{pmatrix} A^F x + G^F y \le b^F \\ y_k \le 0 \end{pmatrix} \vee \begin{pmatrix} A^F x + G^F y \le b^F \\ -y_k \le -1 \end{pmatrix} \tag{7}$$

for the variable y_k. A lift-and-project cut $(\alpha^F)^T x + (\beta^F)^T y^F \le \gamma^F$ from this disjunction is obtained by solving the cut generating linear program[26]

$$\begin{array}{ll} \max & (\alpha^F)^T \bar{x} + (\beta^F)^T \bar{y}^F - \gamma^F \\ \text{s.t.} & \alpha^F - u^T A^F \le 0 \\ & \alpha^F - v^T A^F \le 0 \\ & \beta^F - u^T G^F - u_0 e_k^F \le 0 \\ & \beta^F - v^T G^F - v_0 e_k^F \le 0 \\ & \gamma^F - u^T b^F = 0 \\ & \gamma^F - v^T b^F + v_0 = 0 \\ & \sum_i u_i + u_0 + \sum_i v_i + v_0 = 1 \\ & u, u_0, v, v_0 \ge 0 \end{array} \tag{CGLP}$$

where $\alpha^F \in \Re^n$, $\beta^F \in \Re^{|F|}$, $\gamma^F \in \Re$, $u \in \Re^{l+m+2|F|+p}$, $v \in \Re^{l+m+2|F|+p}$, $u_0 \in \Re$, $v_0 \in \Re$. Solving CGLP, yields the cut $(\alpha^F)^T x + (\beta^F)^T y^F \le \gamma^F$, where

$$\begin{aligned}
\alpha_j^F &= \min\left\{u^T A_j^F, v^T A_j^F\right\} && \text{for } j = 1,\ldots,n \\
\beta_j^F &= \min\left\{u^T G_j^F, v^T G_j^F\right\} && \text{for } j \in F, j \neq k \\
\beta_k^F &= \min\left\{u^T G_k^F + u_0, v^T G_k^F - v_0\right\} && j = k \\
\gamma^F &= \min\{u^T b^F, v^T b^F + v_0\}
\end{aligned} \tag{8}$$

The objective function of CGLP is chosen so as to maximize the amount by which $(\bar{x}, \bar{y})$ is cut off, such that the cut quality can be measured in this way. The last equation of CGLP is a normalization constraint, meant to truncate the polyhedral cone defined by the remaining inequalities. Various normalizations for CGLP are proposed by Balas et al.,[24] and their strengths and weaknesses are discussed extensively by Balas and Perregaard.[27]

Cut lifting

The disjunctive cutting plane $(\alpha^F)^T x + (\beta^F)^T y \leq \gamma^F$ obtained by solving CGLP is only valid at the current node (F_0,F_1) and its descendants. Such a cut can be lifted to be valid throughout the enumeration tree by using the multipliers u, u_0, v, v_0, obtained along with the cut vectors $\alpha^F, \beta^F, \gamma$ and the lifting cut coefficients β_j for the variables $j \in F_0 \cup F_1$, which are assumed to be at their lower bound, are then given by

$$\beta_j = \min\left\{u^T \tilde{G}_j^F, v^T \tilde{G}_j^F\right\} \quad \text{for } j \in F_0 \cup F_1 \tag{9}$$

where $\tilde{G}_j^F$ denotes the subvector of $\tilde{G}_j$ with the same row set as G^F. Then, a valid disjunctive cutting plane $a^T x + \beta^T y \leq \gamma$ that is valid for the whole enumeration tree can be calculated by

$$\begin{aligned}
\alpha_j &= \alpha_j^F && \text{for } j = 1, \ldots, n \\
\beta_j &= \beta_j^F && \text{for } j \in F \\
\beta_j &= \min\left\{u^T \tilde{G}_j^F, v^T \tilde{G}_j^F\right\} && \text{for } j \in F_0 \cup F_1 \\
\gamma &= \gamma^F
\end{aligned} \tag{10}$$

It should be noted that the lift-and-project cut $(\alpha^F)^T x + (\beta^F)^T y^F \leq \gamma^F$ described by Eq. 8 is derived from the disjunction on the 0-1 variable y_k in the form of Eq. 7. The integrality conditions on $y_j, j \in F, j \neq k$ can be used to strengthen their cut coefficients, and β^F can be replaced by $\overline{\beta}^F$, where

$$\overline{\beta}_j^F = \max\left\{u^T G_j^F + u_0 \lceil m_j \rceil, v^T G_j^F - v_0 \lfloor m_j \rfloor\right\} \quad \text{for } j \in F, \ j \neq k \tag{11}$$

with

$$m_j = \frac{u^T G_j^F - v^T G_j^F}{u_0 + v_0} \tag{12}$$

Remember that we always assume that the index set F_1 is empty. If this is not true for the current node (F_0,F_1), complementing is done on this set to change it to be empty. A simple reverse linear transformation can get the cut that is valid for the original problem MINLP2 before complementing, as

$$\alpha^T x + \sum_{j \in F \cup F_0} \beta_j y_j + \sum_{j \in F_1} (-\beta_j) y_j \leq \gamma + \sum_{j \in F_1} (-\beta_j) \tag{13}$$

Finally, this cut can be added into the feasible sets of the NLP or CGLP at any node (F_0,F_1) in a branch-and-cut searching procedure.

Algorithmic Procedures

The general branch-and-cut algorithm for problem (MINLP2) can now be formally stated, which is similar to that for MILP problem.[24,28] The active nodes of the enumeration tree are represented by a list of S with ordered pairs (F_0, F_1). Let UBD stand for the current upper bound, i.e., the value of the best-known solution to problem (MINLP2).

For an input of c, n, q, A, G, b, m, and g_i (for $i = 1,\ldots,l$):

Step 1. Initialization: Set $S = \{(F_0 = \phi, F_1 = \phi)\}$ and UBD $= \infty$.

Step 2. Node selection: If $S = \phi$, stop. Otherwise, choose an ordered pair $(F_0, F_1) \in S$ and remove it from S.

Step 3. Lower bounding: Solve problem NLP(C, F_0, F_1). If the problem is infeasible, go to Step 2. Else, let $(\bar{x}, \bar{y})$ denote its optimal solution, if $c\bar{x} \geq$ UBD, go to Step 2; else if $\bar{y}_j \in \{0,1\}, j = 1, \ldots, q$, let $(x^*, y^*) = (\bar{x}, \bar{y})$, UBD $= c\bar{x}$, and go to Step 2; else, go to Step 4.

Step 4. Branching versus cutting decision: Should cutting planes be generated? If yes, go to Step 5, else go to Step 6.

Step 5. Cut generation: Solve problem CGLP, yields $\alpha x + \beta y \leq \gamma$ that is valid for problem MINLP2, but violated by $(\bar{x}, \bar{y})$. Add the cuts into C and go to Step 3.

Step 6. Branching step: Pick an index $j \in \{1, \ldots, q\}$ such that $0 < \bar{y}_j < 1$. Generate two subproblems corresponding to $(F_0 \cup \{j\}, F_1)$ and $(F_0, F_1 \cup \{j\})$, add them into the node set S. Go to Step 2.

When the algorithm terminates, if UBD $< \infty$, (x^*, y^*) is an optimal solution to (MINLP2), otherwise (MINLP2) is infeasible.

The specific implementation issues in the aforestated algorithm are presented in the next section, such as the answer to the question proposed in Step 4, the node selection strategies in Step 2, the branching rules in Step 6, and the cutting plane management in Step 5.

Implementation Issues

The branch-and-cut procedures described in the former section for 0-1 MINLP have been implemented in an algorithmic package entitled MINO,[29] i.e., mixed-integer nonlinear optimizer, written in standard C language, where the problem NLP(C,F_0,F_1) is solved by an NLP solver LSGRG,[30,31] and problem CGLP is solved by a LP solver ILOG CPLEX 10.0.[32] Some specific implementation issues are discussed in the following subsections.

Weak relaxation

For problem CGLP, the number of constraints is $2n + 2|F| + 3$, and the number of variables is $n + 5|F| + 2l + 2m + 2p + 3$. So, clearly the effort involved in solving problem CGLP is substantial for large-scale 0-1 MINLP problems. Balas et al.[24] suggested that a relaxation $\overline{K}$

of $K(C, F_0, F_1)$ could be used to obtain lift-and-project cuts. In our implementation, we chose the relaxation $\overline{K}$ that contains the active constraints at $(\bar{x}, \bar{y})$ as well as the inequalities of $y_j \geq 0$ and $y_j \leq 1$ for $j \in F$. As $\overline{K}$ contains fewer constraints, the size of the resulting CGLP can be reduced while cuts with good quality can still be generated. In Figure 1, the dot line represents the cut of Stubbs and Mehrotra[21] generated directly from the convex hull of the mixed-integer set, the short dashed line stands for the cut of Zhu and Kuno[23] produced from the linear approximation of the mixed-integer set, and the long dashed line is the cut in this paper which is generated from the active linear approximation of the mixed-integer set at the relaxed solution. The cut of this paper is relatively weaker, but it still cuts the fraction solution away and very amenable for large-scale MINLP problems.

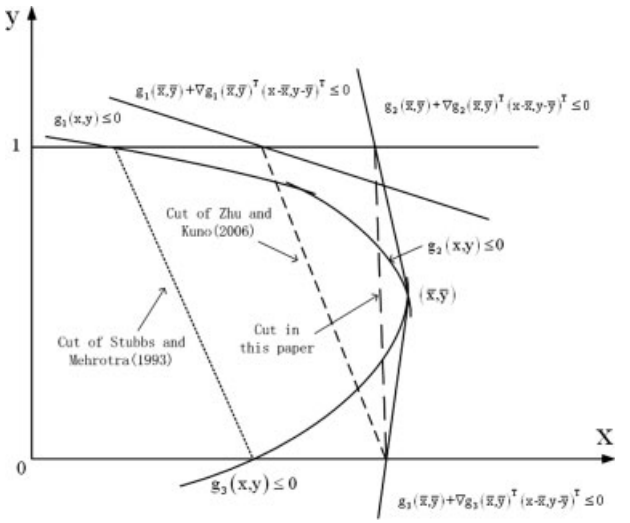

Figure 1. Cut proposed in this article was based on the active linear approximation of the mixed-integer set, where the mixed-integer convex set is described by three nonlinear functions, and two of them are active at the relaxed solution.

Cut generating frequency

For the use of lift-and-project cutting planes in a branch-and-cut procedure, Balas et al.[24] concluded that it is better to generate cuts in large rounds rather than one at a time or in small rounds. Based on this, our implementation generates rounds of cuts for all 0-1 variables that are fractional in the current solution, or some upper bound, whichever is smaller. Balas et al.[24] recommended two strategies to avoid generating cuts too often, one is the fixed strategy that generates one round of cuts every *SK* nodes of the enumeration tree, for some fixed integer *SK* (called skip factor), the other is the automatic strategy in which the skip factor is chosen automatically by the algorithm, depending on the instance to be solved. In our implementation, we use the fixed strategy that generates a round of cuts at the root node and every *SK* nodes after. The skip factor *SK* was chosen between 2 and 16 in the test problems.

Cut management

For large-scale 0-1 MINLP problems, if all cuts generated in the course of the algorithm are kept active in the problem formulation, the nonlinear programs NLP(C,F_0,F_1) would become too large and bring about heavy computational burden for NLP solvers. To solve this problem, the generated cuts are kept in two separate lists, the active list and the pool, which were firstly proposed by Padberg and Rinaldi[33] and used by Balas et al.[24] The active list keeps the cuts that are currently active at the selected node, whereas the pool contains all the other cuts. When a node (F_0,F_1) is retrieved from the node queue, all cuts are evaluated by the solution $(\bar{x}, \bar{y})$ retrieved at the new node and the violated inequalities are added to the active list and the left to the pool. If the total number of cuts became larger than an upper limit, cuts that are inactive at every node in the remaining node queue were removed from the pool. For the test problems, the upper limit was set to 1000.

During a round of cut generation, it is very likely that the same cuts will be generated for different $y_k, k \in F$. So, to avoid adding the same cuts, a new cut is added to the active list only if the cosine of the angle between its normal vector and each previously added cuts is less than θ, where θ is a parameter and we took $\theta = 0.99$ for the test problems.

Node selection

There are three popular node selection strategies for node selection in a standard branch-and-bound algorithm, i.e., "best bound," "depth first," and "breadth first," here "best bound" is chosen to strengthen the effects of the valid cutting planes in the branch-and-cut procedure. The "best bound strategy" is implemented by listing the nodes in the queue in increasing order of objective function value and always picking the first one.

Branching rules

If the current node cannot be pruned, a 0-1 variable whose current solution is fractional is selected for branching. In our implementation, we simply choose the binary variable $y_k, k \in F$ whose current value is the closest to 0.5.

Numerical Experiments

The computational performance of the general MINLP algorithm based on lift-and-project cutting planes is demonstrated by eleven 0-1 MINLP test problems for systems optimization. All test problems can be downloaded from MacMINLP website, i.e., an AMPL collection of mixed-integer nonlinear programs maintained by Leyffer,[34] and CMU-IBM website for open source MINLP project.[35] Test problems named Synthes1, Synthes2, and Synthes3 are simplified process synthesis problems described by Duran and Grossmann,[9] where the goal is to simultaneously determine the optimal structure and operating conditions of a process that will meet certain given design parameters. Test problem named Optprloc is an optimal positioning problem of a new product in a multiattribute space.[9] Test problems named Batch4, Batch5, and Batch12 are arising from the optimal design of multiproduct batch plants.[36,37] Test problems named Trimloss2

Table 1. Characteristics of the Test Problems

Test Problem	Number of Variables		Number of Constraints	
	Continues	Binary	Linear	Nonlinear
Synthes1	4	3	10	3
Synthes2	7	5	22	4
Synthes3	10	8	39	5
Optprloc	6	25	15	26
Batch4	23	24	74	2
Batch5	33	40	142	2
Batch12	41	60	218	2
Trimloss2	7	31	22	2
Trimloss4	21	85	56	4
BatchS101006M	260	120	1009	11
Syn20M04M	260	160	996	56

and Trimloss4 are trim loss minimization problems in the paper industry.[38] Test problem named BatchS101006M is a multiproduct batch plant design problem with multiple units in parallel and intermediate storage tanks.[39] Test problem named Syn20M04M is a synthesis problem and corresponds to the synthesis portions of the retrofit-synthesis problems, in which one would like to simultaneously redesign part of an existing plant and synthesize part of a new one.[40] The characteristics of the test problems are given in Table 1 and the computational results for the aforementioned problems obtained by the general branch-and-cut algorithm, i.e., MINO, and its counterpart, i.e., the pure branch-and-bound algorithm with the same node selection strategy and variable branching rules, are presented in Table 2, where the time seconds are computed by running MINO on a personal computer with an Intel Centrino Duo 1.83 GHz computer processing unit (CPU) and 1.5 GB of random access memory (RAM).

The effect of the disjunctive cutting planes generated based on the polyhedral approximation is implied by the computational results shown in Table 2, the enumerated nodes, i.e., NLPs solved by the branch-and-cut method are much less than those by the branch-and-bound method for the medium size MINLP problems except for the first two small ones. The branch-and-cut method has converged on the optimum solutions for all 11 problems, which indicates that the disjunctive cutting planes generated are valid and robust. Compared with the method developed by Stubbs and Mehrotra,[19] fewer NLPs are solved for problem optprloc by using the current approach and there is no observation that the optimal solution was cut away. Different from our former approach,[23] the current CGLP does not guarantee to produce an effective cutting plane, which is shown by the differences between the number of LPs and number of cuts given in Table 2, since on one side, the weak relaxation of the constraint set is used to construct the CGLP, which may be too loose to generate an effective cutting plane, on the other side, the normalization used in CGLP cannot guarantee that such LP is always bounded. It is interesting to observe that the CPU time spent on medium size test problems by branch-and-cut method is always longer than that by branch-and-bound method, this seems paradoxical since less NLPs are solved for branch-and-cut method and a LP is always computationally cheaper than an NLP. This phenomenon can be extricated by the difference between the efficiencies of the LP and NLP solvers and the other direct reason is that the added cuts in NLP(C,F_0,F_1) cause extra computational burden for the used NLP solver, which is still not so robust like the LP solver. But, the striking bounding property of the proposed cutting planes is indicated by the large test problem trimloss4, the pure branch-and-bound method cannot converge on the solution until failure was caused by numerical difficulties in NLP solver LSGRG after running 12 h. But, the branch-and-cut algorithm found the optimum after generating 3579 cuts with 11,929 NLP solved. The commercially available MINLP solvers DICOPT and SBB have spent more than 3 CPU hours on solving trimloss4 problem reported by Bonami et al.[35] To further confirm this characteristic property of the branch-and-cut algorithm, MINO was used to solve two medium-scale industrial examples, i.e., BatchS101006M and Syn20M04M, with binary variables up to 120 and 160, respectively. Although the calculation results obtained by MINO are just comparable with those obtained by DICOPT as reported by Bonami et al.,[35] SBB has spent more than 3 CPU hours on solving test problem Syn20M04M. The sharp

Table 2. Computational Results of the Test Problems

Test Problem	Branch and Bound		Branch and Cut						
						CPU time (s)			
	NLP	CPU Time (s)	NLP	LP	Cuts	Total (s)	NLP (s)	LP (s)	Skip Factor
Synthes1	5	0.001	6	2	2	0.001	0.001	0.0*	2
Synthes2	11	0.015	13	6	6	0.031	0.031	0.0*	4
Synthes3	25	0.094	10	21	14	0.062	0.047	0.015	2
Optprloc	97	0.140	57	101	80	0.297	0.203	0.094	2
Batch4	29	0.078	18	48	35	0.156	0.124	0.032	2
Batch5	79	0.391	62	44	31	1.297	1.172	0.125	6
Batch12	85	0.718	12	59	46	0.750	0.689	0.061	2
Trimloss2	655	0.516	265	99	77	1.625	1.501	0.124	8
Trimloss4	†	†	11929	6870	3579	74.625	72.035	2.590	8
BatchS101006M‡	57470	6 hr§	1870	537	365	802.19	320.51	481.68	4
Syn20M04M‡	¶	¶	2566	137	137	883.42	766.45	116.97	16

*CPU time is less than 0.001 s.
†Failure was caused by numerical difficulties in NLP solver LSGRG after running 12 h.
‡The NLP solver used for these two problems is IPOPT.[41]
§hr here refers to CPU time in hours.
¶No convergence was observed after running for 1 week.

difference between the branch-and-bound and branch-and-cut type of approaches led by the valid cutting planes for these two problems can be observed from the calculation results shown in Table 2, such as for BatchS101006M problem, MINO has converged after solving just 1870 NLPs after producing 365 cuts, but the branch-and-bound approach has solved 57,470 NLPs to converge. Such results further corroborate that the branch-and-cut method based on the general cutting planes proposed in this article is perspective for large-scale MINLP problems.

Conclusion

An improved branch-and-cut algorithm based on the general disjunctive cutting planes for mixed-integer nonlinear systems optimization problems has been proposed. Based on the polyhedral approximation of the mixed-integer nonlinear set of the MINLP problem at each enumerated node, an MILP problem can be produced, which is an outer approximation of the original MINLP problem at that node. According to the critical finding in our former paper,[23] a disjunctive cutting plane in terms of the MILP problem can be generated, which is valid for the original MINLP problem and cuts the incumbent fractional solution away. By virtue of the approximate MILP problem, the cut generating, lifting, and strengthening process proposed by Balas et al.[20,24] is used to construct the disjunctive cutting planes in a branch-and-bound framework for the MINLP problem. Some implementation issues are also incorporated into the algorithmic development to enhance the efficiency of the branch-and-cut method for large MINLP problems. Eleven systems optimization problems formulated by 0-1 MINLP with the size from small to medium are used to evaluate the effectiveness of the disjunctive cutting plane and the efficiency of the branch-and-cut algorithm. The computational results state that the produced disjunctive cutting planes can greatly strengthen the bounding property of the branch-and-bound method, which in turn helps to accelerate the branching process. Compared with the approach proposed by Stubbs and Mehrotra[19] and our former work,[23] the current approach is prospective for practical 0-1 MINLP problems.

Acknowledgments

The authors gratefully appreciate the financial support from the National Science Foundation of China (Nos. 20506013 and 20776075), the National High Technology Research and Development (863) Program of China (Nos. 2006AA02Z337 and 2007AA02Z223).

Literature Cited

1. Floudas CA. *Nonlinear and Mixed-Integer Optimization, Fundamentals and Application*. New York: Oxford University Press, 1995.
2. Grossmann IE. Mixed-integer optimization techniques for algorithmic process synthesis. In: Denn MM, Stephanopoulos G, Seinfeld JH, Wei J, editors. *Advances in Chemical Engineering, Vol. 23: Process Synthesis*. New York: Academic Press. 1996:171–246.
3. Grossmann IE. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optim Eng*. 2002;3:227–252.
4. Biegler LT, Grossmann IE. Retrospective on optimization. *Comput Chem Eng*. 2004;28:1169–1192.
5. Zhu Y, Kuno T. Global optimization of nonconvex MINLP by a hybrid branch-and-bound and revised general Benders decomposition approach. *Ind Eng Chem Res*. 2003;42:528–539.
6. Nemhauser GL, Wolsey LA. *Integer and Combinatorial Optimization*. New York: Wiley-Interscience, 1988.
7. Mendez CA, Cerda J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng*. 2006;30:913–946.
8. Biegler LT, Grossmann IE, Westerberg AW. *Systematic Methods for Chemical Process Design*. Englewood Cliffs, NJ: Prentice-Hall, 1997.
9. Duran MA, Grossmann IE. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math Program*. 1986;36:307–339.
10. Fletcher R, Leyffer S. Solving mixed-integer nonlinear programs by out approximation. *Math Program*. 1994;66:327–349.
11. Geoffrion AM. Generalized Benders decomposition. *J Opt Theory Appl*. 1972;10:237–260.
12. Westerlund T, Pettersson F. A cutting plane method for solving convex MINLP problems. *Comput Chem Eng*. 1995;19:S131–S136.
13. Kelly JE Jr. The cutting-plane method for solving convex programs. *J SIAM*. 1960;8:703–712.
14. Gupta OK, Ravindran V. Branch and bound experiments in convex nonlinear integer programming. *Manage Sci*. 1985;31:1533–1546.
15. Borchers B, Mitchell JE. An improved branch-and-bound algorithm for mixed-integer nonlinear programming. *Comput Oper Res*. 1994;21:359–367.
16. Leyffer S. Integrating SQP and branch and bound for mixed integer nonlinear programming. *Comput Optim Appl*. 2001;18:295–309.
17. Tawarmalani M, Sahinidis NV. Global optimization of mixed-integer nonlinear programs: a theoretical and computational study. *Math Program Ser A*. 2004;99:563–591.
18. Tawarmalani M, Sahinidis NV. A polyhedral branch-and-cut approach to global optimization. *Math Program Ser B*. 2005;103:225–249.
19. Stubbs RA, Mehrotra S. A branch-and-cut method for 0-1 mixed convex programming. *Math Program*. 1999;86:515–532.
20. Balas E, Ceria S, Cornuejols G. A lift-and-project cutting plane algorithm for mixed-integer 0-1 programs. *Math Program*. 1993;58:295–324.
21. Sherali H, Adams W. A hierarchy of relaxations between the continuous and convex hull representation for zero-one programming problems. *SIAM J Discrete Math*. 1990;3:411–430.
22. Lovasz L, Schrijver A. Cones of matrices and set functions and 0-1 optimization. *SIAM J Optim*. 1991;1:166–190.
23. Zhu Y, Kuno T. A disjunctive cutting plane based branch-and-cut algorithm for 0-1 mixed-integer convex nonlinear programs. *Ind Eng Chem Res*. 2006;45:187–196.
24. Balas E, Ceria S, Cornuejols G. Mixed 0-1 programming by lift-and-project in a branch-and-cut framework. *Manage Sci*. 1996;42:1229–1246.
25. Balas E. Disjunctive programming. *Ann Discrete Math*. 1979;5:3–51.
26. Balas E, Perregaard M. A precise correspondence between lift-and-project cuts, simple disjunctive cuts, and mixed integer gomory cuts for 0-1 programming. *Math Program Ser B*. 2003;94:221–245.
27. Balas E, Perregaard M. Lift-and-project for mixed 0-1 programming: recent progress. *Discrete Appl Math*. 2002;123:129–154.
28. Wolsey LA. *Integer Programming*. New York: Wiley-Interscience, 1998.
29. Zhu Y. Mixed-integer nonlinear optimization: a disjunctive cutting plane approach. In: Floudas CA, Pardalos PM, editors. *Encyclopedia of Optimization*, 2nd ed. Springer-Verlag, 2008.
30. Smith S, Lasdon L. Solving large sparse nonlinear programs using GRG. *ORSA J Comput*. 1992;4:2–15.
31. Lasdon L. *LSGRG Version 3.0 Release Notes*. Texas: MSIS Department, College of Business Administration, the University of Texas at Austin, 2000.
32. ILOG Company. ILOG CPLEX 10.0. 2006.
33. Padberg M, Rinaldi G. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Rev*. 1991;33:60–100.
34. Leyffer S. *MacMINLP: AMPL collection of mixed integer nonlinear programs*. Dundee: University of Dundee, 2000. Available at http://www.mcs.anl.gov/~leyffer/MacMINLP/
35. Bonami P, Biegler LT, Conn AR, Cornuejols GC, Grossmann IE, Laird CD, Lee J, Lodi A, Margot F, Sawaya N, Wachter A. An

algorithmic framework for convex mixed-integer nonlinear programs. *Discrete Optimization*. 2008;5:186–204.
36. Kocis GR, Grossmann IE. Global optimization of nonconvex mixed integer nonlinear programming (MINLP) problems in process synthesis. *Ind Eng Chem Res*. 1988;27:1407–1421.
37. Grossmann IE. Mixed-integer nonlinear programming techniques for the synthesis of engineering systems. *Res Eng Des*. 1990;1:205–228.
38. Harjunkoski I, Westerlund T, Porn R, Skrifvars H. Different transformations for solving non-convex trim loss problems by MINLP. *Eur J Oper Res*. 1998;105:594–603.
39. Vechietti A, Grossmann IE. LOGMIP: a disjunctive 0-1 nonlinear optimizer for process systems models. *Comput Chem Eng*. 1999; 23:555–565.
40. Turkay M, Grossmann IE. Logic-based MINLP algorithms for the optimal synthesis of process networks. *Comput Chem Eng*. 1996;20: 959–978.
41. Wachter A, Biegler LT. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Math Program*. 2006;106:25–57.

Manuscript received Nov. 13, 2007, and revision received July 13, 2008.