An LP/NLP Based Branch and Bound Algorithm for MINLP Optimization

Ignacio Quesada and Ignacio E. Grossmann

EDRC 06-123-92

An LP/NLP based Branch and Bound Algorithm for MINLP Optimization

Ignacio Quesada and Ignacio E. Grossmann*
Department of Chemical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213

December 1991

* Author to whom correspondence should be addressed

Abstract

This paper is aimed at improving the solution efficiency of MINLP problems in which the bottleneck lies in the combinatorial search for the 0-1 variables. An LP/NLP based Branch and Bound algorithm is proposed in which the explicit solution of an MILP master problem is avoided at each major iteration. Instead, the master problem is defined dynamically during the tree search to reduce the number of nodes that need to be examined. A branch and bound search is conducted to predict lower bounds by solving LP subproblems until feasible integer solutions are found. At these nodes nonlinear programming subproblems are solved providing upper bounds and new linear approximations which are used to tighten the linear representation of the open nodes in the search tree. To reduce the size of the LP subproblems, new types of linear approximations are proposed which exploit linear substructures in the MINLP problem. The extension to nonconvex problems is also discussed. Numerical results on several test problems are reported which show that the expense of solving the MILP master problem in the outer-approximation algorithm can be greatly reduced. Typically fewer than 50% of the total number of nodes need to be enumerated, while in most cases the number of NLP subproblems to be solved remains the same.

Introduction

Design problems can be formulated as mathematical programming models which involve continuous variables for representing the operating conditions and may have binary variables to represent the set of alternative topologies. The development of these models requires postulating a superstructure that embeds a given set of alternative designs. The more general type of mathematical model of this superstructure is a mixed integer nonlinear programming (MINLP) model.

Over the last decade, mixed-integer programming techniques have been applied extensively in process synthesis. Reviews on the application of this type of design models can be found in Grossmann (1989, 1990). Their main advantage is that they provide a systematic approach for synthesis problems. In this way, as has been pointed out for instance by Sagli *et al* (1990), suboptimal solutions or topology traps, that may be obtained by decomposing the design problems, are avoided. One of the major reasons for the increased importance of mixed integer optimization is the development of more efficient algorithmic techniques and of more reliable and robust software packages for nonlinear and mixed integer linear programming which are required in the solution of these problems.

Although a number of different methods are currently available for the solving MINLP models (Beale, 1977; Gupta and Ravindran, 1985; Geoffrion, 1972; Duran and Grossmann, 1986a, 1986b; Viswanathan and Grossmann, 1990; Nabar and Schrage, 1990), a major limitation in their application to actual designs is the potentially large size of the models. Furthermore, since the space of possible configurations may grow exponentially, the computational requirements can be expensive. For this reason, improvements in the performance of the MINLP solution methods are clearly necessary.

This paper is aimed at the efficient solution of MINLP problems in which the main computational effort lies in the optimization of 0-1 variables, while the NLP subproblems for fixed 0-1 variables are significantly less expensive to solve. Examples include retrofit of heat exchanger networks (Yee and Grossmann, 1991) and synthesis of heat integrated columns

(Floudas and **Paules, 1988).** In both of these applications the optimization of the 0-1 variables at the level of the **master** problem is at least one order of magnitude more expensive than the solution of the NLP subproblems. The proposed algorithm involves a tree search in the space of the binary variables using linear approximations to bound the original problem. NLP subproblems are solved at nodes with integer solutions for the binary variables. The key feature in the algorithm is the dynamic generation of the linear approximations which are derived at integer solutions in the tree. This dynamic generation avoids the sequential solution of NLP and MILP master problems such as in the Generalized Benders Decomposition (GBD) and Outer Approximation (OA) algorithms. The objective of the proposed method is to reduce the computational work required to solve the MILP master problem which is the major bottleneck in the type of MINLP problems that are considered. Additionally, linear approximations that exploit linear substructures in the MINLP are proposed to reduce the size of the LP subproblems that must be solved in the branch and bound enumeration. Numerical results are presented to ilustrate the computational savings that can be achieved with the proposed algorithm.

Background

Consider a design problem for selecting the optimal design parameters for the equipment and the optimal topology of the system that is modelled as an MINLP problem of the following form:

$$
\begin{aligned}
&Z = \min_{Xty} c^T y + flx) \\
&\qquad st \qquad By + gM \leq 0 \\
&x \;\; e\, X = (\, x \,|\, x \;\; eR^n{}_t x^L < x \leq x^u \backslash \\
&y \;\; G\, Y = \{ y \,| y \;\; e \;\; \{0, 1\}^m . Ay \leq a \}
\end{aligned}
\qquad\qquad (MINLP)
$$

The continuous variables x represent the operating conditions, flows and design parameters of the equipment. The set of binary variables, y, defines the topology of the system, representing the existence or non-existence of the different processing units. These variables normally appear in linear form in the model. If this is not the case, the model can be transformed by introducing additional continuous variables. For simplicity in this presentation, no equalities are considered in explicit form. The equalities Ay + h(x) =0 can be included by relaxing them into inequalities according to the sign of the Lagrange multipliers (see Kocis and Grossmann, 1987). The nonlinearities of the model appear in the terms ffcc) and g(x), and it is assumed in this paper that these functions are convex. These conditions will be relaxed later in the paper.

The algorithms for solving MINLP problems can be classified in three main categories:

-Branch and Bound

-Generalized Benders Decomposition

-Outer Approximation

The Branch and Bound algorithm for MINLP problems (Gupta and Ravindran, 1985; Nabar and Schrage, 1990) is based on the same idea as the LP based Branch and Bound algorithm for MILP. The first step is to solve the problem that results from relaxing the discrete conditions in the binary variables. If this relaxed problem yields an integral solution the procedure stops and this is the optimal solution. Otherwise, the relaxed problem provides a lower bound to the optimal solution and a tree enumeration is performed where each node of the tree is a subproblem that results from relaxing a subset of the integrality conditions. When an integer solution is found, it provides an upper bound to the solution. All the nodes that exceed this bound are pruned and the search is conducted until all the nodes in the whole tree are fathomed. The main disadvantage of using this solution approach for MINLP problems is that the resulting nodes in the tree are NLP subproblems that cannot be easily updated. Additionally, the size of these subproblems can be significant since they are formulated both in the continuous and part of the binary space.

The Generalized Benders Decomposition (GBD) (Geoffrion, 1972) algorithm divides the variables into sets of complicating and npncomplicating variables. For MINLP models, the 0-1 variables are usually considiered as the complicating variables.-A sequence of NLP subproblems and MILP master problems in the space of the complicating variables is solved. The subproblems correspond to NLP's that are generated by fixing the binary variables in the original MINLP to a given value y*. An upper bound to the optimal solution of the MINLP is provided by the solution

of these subproblems- The master problem is generated by projecting the original problem in the reduced space of the complicating variables. This is accomplished by deriving at each subproblem a Lagrangian function parameterized in the discrete variables y. The formulation of the master problem is given by,

$$\begin{array}{lll} z=\min\ a & & \\ \text{st}\ \ a \geq c^{T} y + \text{fl}x^{k}) + (\ \text{AJTIBy} + g \wedge | & k = 1 \ldots \text{Kfe*} & (1) \\ \quad (\ \text{W'HBy}+\text{gfc*})\text{l} \ \leq \text{O} & k = 1 \ldots \text{K} \wedge & \\ \quad a\text{e}\ \ R^{*}, y \ \text{€}\ Y & & \end{array}$$

where z is the predicted lower bound, (x^k, X^k) are the optimal primal and dual variables of the NLP subproblem, and the index Kf_{cas} refers to the subproblems that were feasible and the index K^^ refers to subproblems that were infeasible. The solution of this MILP master problem predicts a new set of binary variables for a subsequent NLP subproblem. When the original problem is convex, the master problem provides a valid lower bound to the optimal solution which increases monotonically at each major iteration. The procedure converges when the best upper bound and the lower bound are equal, and the optimal solution is given by this bound.

Similarly, the Outer Approximation algorithm (Duran and Grossmann, 1986a, b) also consists of a sequence of NLP subproblems and MILP master problems. The difference with GBD lies in the way that the master problem is defined. In this case, outer approximations (linearizations) of the nonlinearities are dreived at the optimal solution of the NLP subproblems. The master problem is given by:

$$\begin{array}{lll} z = \min\ a & & \\ \text{st}\ \ a \pounds c^{T} y + \text{fl}x^{k}) + \ \text{Vf}(x^{k})^{T}(x-\ x^{k}) & & \\ \quad \text{By} + \text{gfr}^{k}) + \text{Vgfr}\wedge\text{tx}-\ x^{k}) \leq 0 & k = 1 \ldots K & (2) \\ \quad ae\ \ R^{1},\ y\ \text{e}\ Y,\ x\ e\ \ X & & \end{array}$$

where in this case x^k is the solution (feasible or infeasible) of the NLP subproblem for fixed y^k. For the convex case an overestimation of the feasible region is obtained, so that the OA master problem also provides a valid lower bound. The resulting master problem is an MILP problem in the full space of all variables. In the same way as GBD, the convergence criterion is based on the bounds that the subproblems and the master problems provide to the optimal solution.

A close relation between GBD and **OA** can in fact be established for the case when the nonlinear **functions** fix) **and** g(x) are convex. Specifically, as shown in the Appendix, the Lagrangian cuts in **GBD** (see (1)) correspond to surrogate constraints to each of the linearizations of the OA master problem in (2). Although the GBD master is smaller in size, the OA algorithm provides stronger lower bounds, and hence generally requires fewer major iterations to converge. A limitation is that the convexity conditions of the original problem are more stringent for the OA algorithm.

An extension of **OA** for dealing with nonconvex MINLP problems is the **OA/AP** algorithm proposed by (Viswanathan and Grossmann, 1990). Here, the master problem includes an exact penalty function that allows violation of the linearizations of nonconvex constraints. The master problem formulation is:

$$\begin{array}{llr} z = \min & a + \underset{k}{Z} co_0^k po^k + \underset{k}{Z} <\& P^k & \\ st & c^T y + flx^k) + Vflx\wedge tx - x^k) - a£ \; po^k & \\ & By + g(x^*) + Vgfcck)^T(x - x^k) \le pk \qquad k = 1 \ldots K & (3) \\ & ae \; R^J, y \; e \; Y, x \; eX, po^k, \& > \Theta \quad k = 1 \ldots K & \end{array}$$

where co_0^k, co^k are large weight factors for the slack variables p_o^k, p^k. These variables allow to consider part of the feasible region which otherwise may have been cut off by the linearizations. Although the approach is not rigorous for finding a global optimum, it has proved to be reliable in many problems.

Another extension to deal with nonconvexities is the approach proposed by Floudas *et al* (1989). This method uses Generalized Benders Decomposition by partitioning the complicating and noncomplicating variables so that the subproblems and master problems are convex. Although the computational performance has been good on a set of test problems, this method has no guarantee for finding the global optimum, and may not converge to a local minimum (see Sahinidis and Grossmann, 1991). It should be noted, however, that recently Floudas and Visweswaran (1990) have developed a new version of this method which can rigorously find the global optimum for various classes of MINLP problems.

Motivation for new algorithm

The GBD and OA algorithms have the limitation that the size of the master problem increases as the iterations proceed. This a major drawback when the original MINLP model has a large number of integer variables, since in this approach it is necessary to solve a complete MILP master problem at each iteration. The typical performance of this type of algorithm is shown in Figure 1. The time used to solve the master problem increases as iterations proceed while the time for the NLP subproblems remains of the same order of magnitude. When using simplified process models, the solution times of the NLP's can be substantially smaller than for the MILP master problem since each NLP subproblem corresponds to the optimization for a given topology of the system.

To improve the performance of GBD algorithm Nielsen and Zenios (1990) have proposed to solve the master problems as feasibility problems. In this way it is possible to reduce the expense necessary for solving the master problems.

The algorithm proposed in this paper consists of a tree search over the space of the binary variables. Here the nodes to be solved correspond to LP problems that result from relaxing some of the integrality conditions and replacing the nonlinear terms by linear approximations. These LP^{f}s are dynamically updated with respect to the representation of the nonlinear terms, by solving NLP subproblems at those nodes of the tree with integer solutions. As will be shown, the algorithm offers the flexibility of adding different types of linear approximations with which it is still possible to obtain a tight representation of the feasible region, but without greatly increasing the size of the LP subproblems.

The proposed method can be viewed as an integration of the branch and bound search for the master problem with the solution of NLP subproblems. In the tree search, the nodes remain as LP subproblems that can be easily updated and there is no need to restart the search once the new constraints are added. This is particularly relevant in design problems that involve a large number of alternative topologies (e.g. retrofit design), and where the NLP subproblems are not very expensive to solve. Additionally, by solving the NLP subproblems at succesive integer nodes.

stronger upper bounds are generated to reduce the branch and bound enumeration. The proposed algorithm is first illustrated with a small convex example in the next section.

Convex Example

Consider the problem proposed by Kocis and Grossmann (1988) of deriving the best configuration for the processes given in Figure 2. The MINLP model formulation, which can be shown to correspond to a convex problem, is given by,

$$
\begin{aligned}
\min\ & Z = -[11C - 7B1 - B2 - 1.2\,B3 - 1.8\,(A2 + A3) - 3.5y1 - y2 - 1.5y3] \\
\text{st}\ & C = 0.9B \\
& B2 = \log(1+A2) \\
& B3 = 1.2\log(1+A3) \\
& B = B1 + B2 + B3 \\
& C \le y1 \qquad\qquad (4) \\
& B2 \le 10y2 \\
& B3 \le 10y3 \\
& y2 + y3 \le 1 \\
& y1, y2, y3\ \in \{0,\ 1\}. \qquad C,B,B1,B2,B3,A2,A3 \ge 0
\end{aligned}
$$

The variables y1, y2 and y3 denote the existence/non-existence of process 1, 2 and 3, respectively. When the problem is solved using the OA algorithm, three iterations are required to obtain the optimal solution when an initial value for the binary variables of (0,1,0) is used. The value of the objective function for the NLP subproblems and master problems is reported in Table 1. The branch and bound search conducted for solving the first MILP master problem is given in Figure 3a. Five nodes are examined and the optimal solution is y=(1,1,0)

With the configuration predicted by the optimal solution of the master problem (y=(1,1,0)), an NLP subproblem is solved yielding an objective value of Z=-1.72. A new outer approximation is generated and the resulting master problem is solved that includes the constraints from both iterations. This implies that the branch and bound search for the master problem is started all over again. The second and third iteration require three and five nodes respectively to be examined, giving a total of 13 nodes at the master level (see Fig 3). Note that the size of the LP subproblems increases in terms of the number of rows at each iteration. Also, 3 NLP subproblems **had** to be solved to obtain the optimal solution, Z=-1.923.

In the proposed algorithm, instead of having to solve an MILP master problem independently of the NLP subproblems, the solution of these subproblems is embedded in the search tree. An initial representation of the feasible region is created by using an outer approximation at the solution of an initial NLP. In this example the same initial configuration (y=(O,l,O)) is considered . A search is then conducted by evaluating nodes 1 and 2 in Figure 4a. Node 2 yields the integer solution y (1,0,1) with a lower bound of z=-3 (Figure 4a). At this configuration, a second NLP subproblem is solved which yields an upper bound (Z=-1.923).

The upper bound for pruning the tree is provided by the NLP subproblems, so the feasible nodes that are below this bound are kept open. Once the second NLP subproblem is solved, new linearizations are added to these open nodes tightening the linear representation of the feasible region. Node 2 has not been branched so it is updated by adding the new approximating constraints (see node 3 in Fig 4b). This node can be pruned since its solution (lower bound z = -1.923) is equal to the current best upper bound. After backtracking, node 1 has already been branched so it is not updated, instead a new node is created (node 4). A new integer solution is found later (y=(1,1,0)) and the corresponding NLP subproblem is solved giving a higher upper bound, Z=-1.72. Hence, Z=-1.923 Is kept as the best upper bound.

Again, new outer approximations are added to the open nodes 6 and 7 whose LP's exceed the current upper bound (see Fig 4c). Therefore, the search at this point can be stopped to confirm that Z=-1.923 is the optimal MINLP solution. Note that with the proposed method the search is completed after examining 7 nodes at the branch and bound level, and having solved 3 NLP subproblems. Hence, with respect to the original algorithm only one half of the number of nodes had to be examined. Furthermore, as shown in Table 2, the computational work for solving the LP problems in terms of number of rows was also smaller.

Detailed description of the algorithm

For simplicity we will assume that the NLP subproblems are feasible (see remarks section). The algorithm for the MINLP convex case is as follows:

0. Set lower bound *z* = - oo , upper bound D^1 = oo, set of list of open nodes P= 0. Select initial value y^1 for the binary variables, this set is given or some heuristics can be used to obtain it.

1. a) The initial NLP subproblem for the binary variables at y^1 is solved:

$$Z = \min_x c^T y^1 + fix)$$

st Byi+gMsD (NLP!)

xe X

b) The above provides an upper bound *ZP* to the optimal solution of the MINLP problem.

2. An outer approximation of the original MINLP problem is obtained using the optimal solution of the NLP subproblem (x^1). The nonlinear functions are linearized at this point, yielding the following MILP problem.

z = min a

st $a \geq c^T y + fix^1) + Vf\wedge Rx - x^1)$ (MILP)

$By + gtx^1) + Vg(x^1)^T(x-x^1) \leq 0$

ae $R^!$,yG Y,x e X

3. The integrality conditions over the binary variables y are relaxed and the resulting LP is solved (Po^1). This solution provides a lower bound z_x to the optimal solution. Store the problem in P(P=PU Po^1). If the solution to this problem is integer, go to step 7.

4. Take the last problem in P (PjJ). Pick a binary variable that has a fractional value, y_{Jf} and create two new problems ($Pi+i^1$ and $P_{i+}i^2$) by adding the constraints $yj \geq 1$ and $yj \leq 0$ to subproblem P_t respectively. Delete the parent problem and include the new subproblems in P, P= (P\P|*) u Pi+i^P^2).

5. If at the end of the list there are two problems with the same parent problem do step (a); otherwise do step (b)

a) Solve P_{l+1}^2. If the objective $z^2 \geq Z^U$ delete P_{l+1}^2 from **P,** P= P\ P_{l+1}^2; otherwise if the solution is integer, go to step 7. **Solve** P_{l+1}^1, if $z^1 \geq Z^U$ delete P_{l+1}^1 otherwise if the solution is integer go to step 7. If $z^2 \geq z^1$ invert P_{l+1}^2 and P_{l+1}^1 in the list of nodes P.

b) Solve P_{l+1}^j. If the objective $z^j \geq Z^U$ delete P_{l+1}^j from P, P= P\ P_{l+1}^j . Otherwise if the solution is integer, go to step 7.

6. If the set of open nodes P= 0 go to to step 9. Otherwise go to step 4.

7. Solve an NLP subproblem fixing the binary variables of the MINLP at the level of the LP problem solution. If $Z_{NLP} < Z^U$ set $Z^U = Z_{NLP}$.

8. The solution of the NLP is used to generated additional constraints. This can be done by deriving outer approximations of the nonlinear constraints. Benders cuts or any other type of valid constraints as will be discussed later in this paper. These constraints are added to all the problems in P. Return to step 5.

9. The current upper bound Z^U is the optimal solution to the original MINLP problem.

Remarks

If no initial value y^1 for the binary variables is available, one can generate the initial linearizations by solving a relaxed NLP problem where the integrality conditions of the original MINLP model are dropped. Since in general a non-integer value will be obtained the initial upper bound is set to $Z^U = \infty$.

When the NLP subproblem is infeasible, the corresponding value found for the continuous variables can be used for generating the outer approximation constraints. Else, a feasibility NLP subproblem can be solved where penalties that allow violating the constraints are added to the objective function (see Appendix).

It is important to start with a good initial NLP subproblem to provide a tight upper bound for pruning nodes in the tree. Standard branching techniques are being used. A depth first strategy is used in which both branches are examined, and the one with the best objective function is selected to generate as fast as possible new NLP subproblems. The idea is to quickly tighten the feasible region so that the size of the tree does not become excessively large. Backtracking is done by the same path, since it can be expected that the information from the NLP subproblem is relevant to the closest nodes in the tree. Different branching strategies can be used and additional criteria can be employed to select the integer variable to branch on. These aspects have not yet been addressed in this paper.

The algorithm is finite since the possible number of different configurations is finite, and each time that an NLP subproblem is solved, the constraints that are added cut off that particular configuration, and therefore the algorithm cannot cycle.

Alternative approximation constraints

The information from the solution of the NLP subproblems can be added to the open nodes in the tree in several ways. For instance, outer approximations of the nonlinear terms as in (2), or Benders cuts projected in the space of the binary variables as in (1) can be considered. The advantage of outer approximations is that they give a tighter representation of the feasible region. However, the limitation is that the number of rows of the LP subproblems to be solved at the nodes can become very large. To circumvent this problem, one option is to use Benders cuts, but the drawback is that they in general do not provide strong cuts. For this reason, a new type of approximation constraints is proposed that take advantage of linear substructures which are frequently present in the MINLP. The basic idea is to avoid generating explicit linearizations of nonlinear functions while retaining the linear constraints in order to strengthen the cuts.

From the original problem (MINLP) consider the partition of the continuous variables in two subsets u and v such that the constraints are divided into linear and nonlinear constraints, and the continuous variables into linear and nonlinear variables,

$$\begin{aligned} Z = \min\ & c^T y + a^T w + r(v) \\ \text{st}\ & Cy + Dw + t(v) \le 0 \\ & Ey + Fw + Gv \le b \\ & y \in Y,\ w \in W,\ v \in V \end{aligned} \tag{5}$$

In this representation $fW=a^Tw + r(vK\ B^T = f C\ |EJ^T$, $g(x)= \begin{matrix} \backslash Dw + t(v)l \\ LF\backslash V + GV\ J \end{matrix}$, and $X=WxV$.

Problem (5) is reformulated by the addition of two continuous variables (a, p) to represent the linear and nonlinear parts of the objective function; that is,

$$Z= \min a$$

$$\text{st} \quad Cy + Dw + t(v) \leq 0$$

$$ifv) \leq P \qquad (6)$$

$$Ey + Fw + Gv \leq b$$

$$c^Ty + a^T w + p - a = O$$

$$y e Y, w 6 W, v e V, ae R^1, P € R^1$$

The outer approximation of (6) at the point (w^k, v*) generated by the k-NLP subproblem is

$$Cy + Dw + t(v^k) + Vt(v^k)^T(v-v^k) \leq 0 \qquad (7a)$$

$$r(v^k) + Vrfv\wedge v-v\wedge \leq p \qquad (7b)$$

$$Ey+F\backslash v + Gv \leq b \qquad (7c)$$

$$c^Ty + a^Tw + p-ct=O \qquad (7d)$$

If the Kuhn Tucker conditions of the k-NLP subproblem in (5) are considered with respect to the nonlinear variables v, one has that

$$Vitv^*) \bullet Vtfv^1') X^k + G^TM.^k=O \qquad (8)$$

Then, multiplying by (v - v*),

$$Vr(v^k)^T(v-v^k) + (|I^k)^T G(v-v^k) = -(\wedge FVt\wedge Hv-v^{*1}) \qquad (9)$$

Using the outer approximation constraints in (7a) yields,

$$Vitv\wedge ifv-v^*) + (H.^k)^T Gfv-vk) * U^k F l c y + Dw + tfv^k)l \qquad (10)$$

Finally, by substituting (10) in (7b), the outer approximations of the feasible region are reduced to a partial surrogate with the linear inequalities in an explicit-form as follows:

$$p \geq r(v^k) + U^k)^T [Cy + Dw + t(v^k)l - (M^k)^T G (v - v\wedge \quad ' \qquad (11)$$

$$Ey + Fw+Gv \leq b$$

$$c^Ty + a^Tw + \beta - \alpha = 0$$

Note that **by using the above linear** approximations only the first inequality must be **updated,** so that **effectively one only** has **to add** one **new** linear approximation constraint when updating the LP subproblems in the tree search.

Different special cases of the linear approximations in (11) are the following:

a) When t(v) =0, the constraints in (11) can be shown to reduce to,

$$P \geq itv^{k})-(\wedge^{k}FG(v-v^{k}) \qquad (12)$$

$$Ey + F\backslash v + Gv \leq b$$

$$c^{T}y + a^{T}w + \beta - \alpha = 0$$

b) When r(v) =0, the formulation in (6) does not require the variable p and the constraints in (11) reduce to,

$$(A,^{k})^{T}[Cy + Dw + t(v^{k})] - \wedge F G I v - v k) \wedge \qquad (13)$$

$$Ey + Fw + Gv \leq b$$

$$c^{T}y + a^{T}w - a = 0$$

c) When t(v) =0 and r(v) =0, problem (6) reduces to an MILP and therefore the constraints in (11) become,

$$Ey + Fw + Gv \leq b \qquad (14)$$

$$c^{T}y + a^{T}w - a = 0$$

Also, when all the continuous variables appear in nonlinear terms (i.e. there are no w variables) and there are no linear equations, the constraints in (11) reduce to,

$$<x= c^{T}y + p$$

$$P \geq r(v^{k}) + (\wedge F l C y + Uv^{*}\}] \qquad (15)$$

which is equivalent to a Benders cut projected in the y space. In the case that the binary variables y and the linear variables w are treated as complicating variables, and the term G=0, the constraints in (11) reduce to ,

$$a= c^{T}y + a^{T}w + (5$$

$$P \geq r(v^{*}) -h (\wedge F l C y + Dw + Uv^{*})] \qquad (16)$$

$$Ey + Fw + Gv \leq b$$

which is also equivalent to a Benders cut projected in the (y,w) space.

The representation in (11) is a partial surrogate of the approximations to the nonlinear part of the objective and nonlinear constraints in the full space of the original problem. The main advantage of this type of surrogate is that the linear part of the objective and the linear constraints are not included in the cut, but instead they are considered in explicit form in the MILP formulation. In this way, the cuts in (11) are stronger than the GBD constraints in (15). This follows trivially from the proof in the Appendix. The partial surrogate in (11) can be also interpreted as a surrogate constraint of the linearizations (7a) and (7b) in the outer approximation algorithm. Here the weight factors for the surrogate are the Lagrange multipliers of the nonlinear constraints in the NLP subproblem.

The use of partial surrogates should be specially useful in cases where the solution of the nonlinear variables v in the successive NLP subproblems is basically the same. In this situation the linear outer approximations involving nonlinear functions would be essentially redundant constraints for the master problem. Through the partial surrogate this problem is avoided.

A major consideration in the algorithm is the number of NLP subproblems which need to be solved during the tree search. The original outer approximation algorithm normally requires a relatively small number of subproblems to obtain a good characterization of the feasible region. With the proposed approach the potential number of NLP subproblems that must be solved is likely to be larger. One possible criterion for deciding if an NLP subproblem needs to be solved is the difference between the solution of the LP at the integer node and the upper bound. This aspect, however, has not been thoroughly investigated.

Partial surrogate example

To provide some insight into the nature of the partial surrogate consider the following MINLP problem that involves two continuous variables and one binary variable.

$$Z = \min 10x_1^2 - x_2 + 5(y-1)$$

$$\text{st.} \quad x_2 - 5\log(x_1 + 1) - 3y \leq 0$$

$$-x2+x1^2-y\leq 1 \qquad (17)$$
$$x2+x1+20y\leq 24$$
$$2x2+3x1\leq 10$$
$$x1, x2\in R^+, y\in\{0,1\}$$

The continuous feasible regions for y=l and y =0 are ilustrated in Fig. 5 and Fig. 6 respectively. An initial NLP problem is solved for y=l and an optimal solution of Z=-3.5026, xl=0.179 and x2=3.821 is obtained. Using this solution the Benders, Outer Approximation and Partial Surrogate representations are as follows:

(a) Benders $a\geq -8.4466+4.944y$ (18)

(b) Outer Approximation
$$a\geq 3.58\,x1-x2-5.3204+5\,(y-1)$$
$$-4.2409\,x1+x2-3y\leq 0.0642$$
$$0.358\,x1-x2-y\leq -0.968 \qquad (19)$$
$$x1+x2+20y\leq 24$$
$$3x1+2x2\leq 10$$

(c) Partial Surrogate
$$a\geq 1.1696\,x1-0.128\,x2-5.6429+5\,(y-1)$$
$$x1+x2+20y\leq 24 \qquad (20)$$
$$3x1+2x2\leq 10$$

Note that the variable p in (11) has been eliminated for deriving the inequalities in (20). By minimizing a in each of the above representations the Outer Approximation problem gives the stronger lower bound (a= -5.9528) with y=0, xl=0.86, x2= 3.71; the approximation to the continuous feasible region for y=0 is shown in Fig. 7. Benders gives a lower bound of a = -8.4466 with y=0. The patial surrogate provides a stronger lower bound than Benders (a =-6.2829) with y=0, xl=0, x2=5; the approximation to the continuous feasible region is shown in Fig. 8.

Extension to nonconvcx problems

When the original MINLP problem is nonconvex the initial outer approximation and subsequent constraints no longer overestimate the feasible region, and the solution of the nodes cannot be used as a valid lower bound (see Kocis and Grossmann, 1988). This then means that some feasible solutions of the MINLP problem could be infeasible at the branch and bound level because of the undesirable effect of the linearizations in nonconvex terms.

A possible extension of the proposed algorithm to deal with this type of problems, is an adaptation of the OA/AP algorithm (Viswawathan and Grossmann, 1990). An exact penalty term is added to allow violations of the outer approximations of nonconvex terms, given the following linear representation:

$$\begin{aligned}
z=\min \quad & \alpha + \omega_0^T p^0 + \omega_1^T p \\
\text{st} \quad & \alpha \geq c^T y + f(x^0) + \nabla f(x^0)^T (x - x^0) - p^0 \\
& By + g(x^0) + \nabla g(x^0)^T (x - x^0) \leq p \qquad\qquad (21) \\
& y \in Y,\ x \in X, \quad p^0, p \geq 0, \ \alpha \in R^1
\end{aligned}$$

where the positive variables p^0 and p are slack variables. These variables are set to zero if the linearization corresponds to a convex constraint. Additional slacks are used as more constraints are generated in the tree search. The weights ω_0, ω_1 of the penalty term in the objective function are selected to be proportional to the value of the Kuhn-Tucker multipliers at the optimal solution of the NLP subproblems (typically by a factor of 1000).

Since in the nonconvex case the solution of the nodes is no longer a valid lower bound to the optimal solution of the MINLP, different criteria must be used to prune the tree. The heuristics that are used in this work include:

-If the solution of an NLP subproblem at an integer node is greater than the best current upper bound, close this node in the tree. This criteria is in accordance with the convergence criteria used in the OA/ER/AP algorithm.

-Bounding over the optimal solution of the nodes. The first criterion is not enough for pruning the tree in a significant form, since the penalty term also allows for solutions that are infeasible in the original problem to become feasible at the branch and bound level. In this way the number of potential NLP subproblems to be solved can be very high. A non-rigorous bounding can be applied if the best upper bound is compared with the objective function of the node without including the contribution of the penalty term. This criterion reduces to the normal pruning in the case of convex problems.

Both criteria are not rigorous, and the likelihood of finding the global solution or a good solution increases if all the nonconvexities are placed in the constraints and if the objective function is convex. This is because the second criterion for pruning nodes does not consider the influence of the penalty terms on the linearizations of the objective function.

The advantage of the proposed approach over the OA/AP algorithm is that since a larger number of NLP subproblems is analyzed and the termination criteria are similar, it is more likely to find a better solution. However, this implies that a greater expense is required for solving the MINLP problem.

Nonconvex example

A similar example to one presented for the convex case is considered (see Fig. 9). In this case three equipment can be used in parallel to produce w from x, and the objective function is a concave function. The formulation is given by:

$$\min Z = 4x + x_1^{0.83} + x_2^{0.83} + x_3^{0.83} + 3y + 5y_2 + 2y_3 - 10w$$

$$\text{st} \quad x_1 + x_2 + x_3 = x$$

$$w_1 + w_2 + w_3 = w$$

$$w_1 = 4\log(x_1 + 1)$$

$$w_2 = 1.5\log(1.2x_2 + 1)$$

$$*> = 105^3 - \qquad (22)$$

$$x_1 \le 20y_1$$

$$x_2 \le 15y_2$$

$$x_3 \le 15y_3$$

$$w \ge 5$$

$$x \le 10$$

$x_1, x_2, x_3, w, w_1, w_2, w_3 \geq 0 \qquad y_1, y_2, y_3 \in \{0,1\}$

The binary variables y1, y2 and y3 denote the existence or not of the units. The solution for the different possible integer combinations is given in Table 3

The global optimum is given by y=(1,0,1) with a value of -32.71. This problem was solved using DICOPT ++ (Viswanathan and Grossmann, 1990) and DICOPT (Kocis and Grossmann, 1989) with an initial guess of y=(1,0,0). DICOPT obtained the solution y(0,0,1) and DICOPT++ obtained y=(1,1,1) with and objective function of z=-31.64 which is slightly higher than the global optimum.

In this proposed approach, the same initial guess of y=(1,0,0) is used. The nodes that were analyzed before getting an integer solution, the integer solution and the objective function value with penalty and without penalty term are reported in Table 4.

A total of 13 nodes were necessary at the branch and bound level, and 4 NLP subproblems had to be solved. The global optimum of -32.71 is found at the third NLP subproblem. For that integer solution y=(0,1,0) the objective function with the penalty is above the upper bound, but when the objective function is considered without the penalty term it is still below the upper bound and that particular configuration is analyzed. With the OA algorithm in DICOPT a total of 7 nodes and 2 NLP subproblems are required to obtain a solution. When using the OA/AP algorithm in DICOPT++ 14 nodes and 4 NLP are analyzed. It is interesting to note that when an initial value of y=(1,0, 0) was used instead of the relaxed NLP for the OA/AP algorithm, a solution of -32.71 is obtained after 14 nodes were examined.

Computational Results

Several convex MINLP problems have been solved using the proposed LP/NLP based branch and bound algorithm. At this point the proposed method has not been automated and therefore the emphasis in the comparisons will be in the number of nodes and size of LP's, rather than on the CPU time. The size and characteristics of the test problems are given in Table 5. These problems have been reported in the literature, and an explanation of them is given in

Viswanathan and Grossmann (1990). These problems were solved using both the original OA algorithm by Duran and Grossmann (1986b) and the algorithm proposed in this paper.

The first set of computational results in which the linearizations as in (2) were used for both methods are given in Table 6. Note that the proposed method achieves reductions between 46% and 84% of the nodes that need to be enumerated in the branch and bound. Problems batch5 and batch8 were solved using special order sets for the binary variables. In these cases the branching strategy that was used is the standard one for SOS1 (Beale and Forrest, 1976). Only one of the LP problems generated by the branching is examined when the branch is expanding; the other LP is solved at the moment of backtracking. Also, note that the number of NLP subproblems is similar in both methods. As is to be expected, this number can be larger in the proposed method, like in the case of problems ex3 and ex4. This is not a major limitation, however, since for instance in problem ex4 80% of the total time in the OA algorithm is required to solve the MILP master problems.

The algorithm has also been tested on two test problems using the partial surrogate in (11) for storing the information of subsequent NLP subproblems. In this case the first approximation used is the same as in the OA algorithm (see (2)). The results are given in Table7. These results are relevant since in both cases the total number of NLP subproblems to be solved remains the same. For problem french a slightly larger number of nodes is examined due to the weaker bounds, but then the size of the problems to be solved at the nodes is smaller in the latter as seen in Table 8. It is worth noting that in problem ex4 the number of nodes is reduced from 103 down to 45 as seen in Table 7. This is due to the fact that the order of branching in tree was different and the optimal solution was found early. It is significant to note the difference in the size of the LP problems that are solved at subsequent nodes of the tree, since almost all the constraints are nonlinear in problem ex4. For instance, as seen in Table 9, the OA algorithm included 81 rows in the MILP of the last iteration. For the proposed algorithm with the full set of linearizations the size goes up to 131 rows since 5 NLP subproblems are solved instead of 3. However, when the partial surrogate constraints in (11) are used only 35 constraints had to be included in the final nodes.

The above results are clearly encouraging for solving MINLP problems in which the bottleneck lies in the optimization of 0-1 variables. Futhermore, the proposed method can be complemented with the symbolic integration logic scheme by Raman and Grossmann (1991) to further reduce the number of nodes to be enumerated in the branch and bound tree. Efforts are currently under way to automate the proposed method, which however is not entirely trivial to implement.

Conclusions

This paper has presented a branch and bound method for MINLP problems that is based on the solution of LP and NLP subproblems. As has been shown, this method avoids the sequential solution of NLP subproblems and MILP master problems that is required in the standard implementation of the GBD and OA algorithms. As was shown with several test problems, substantial reductions (up to 84 %) can be achieved in the number of nodes that need to be examined for the MILP master problem. Futhermore, through the use of partial surrogate constraints the size of the LP subproblems in the tree search can be kept small. The proposed method should prove to be most useful in cases where the MINLP problem involves large number of 0-1 variables and the NLP subproblems are relatively inexpensive to slove.

Acknowledgment

The authors are most grateful for the financial support provided for this project by the Engineering Design Research Center at Carnegie Mellon University.

References

Beale, E.M.L. (1977). Integer Programming. The State of the Art in Numerical Analysis (D. Jacobs, ed), Academic Press, London, pp 409-448.

Beale. E.M.L. and Forrest, J.J. H. (1976). Global Optimization Using Special Ordered Sets. *Mathematical Programming* 10, 52-69.

Duran M.A. and Grossmann I. E. (1986a). A Mixed -Integer Nonlinear Programming Approach for Process Systems Synthesis. *AIChE Journal* 32 (4) 592-606.

Duran M.A. and Grossmann I.E. (1986b). An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs. *Mathematical Programming* 36, 307-339.

Floudas, C.A. and Paules, G.E. (1988). A Mixed-Integer Nonlinear Programming Formulation for the Synthesis of Heat Integrated Distillation Sequences. *Computers and Chem. Eng.* 12(6), 531-546.

Floudas, C.A., Aggarwal, A. and Ciric, A.R. (1989). Global Optimum Search for Non-Convex NLP and MINLP Problems, *Computers and Chem. Eng.,* 13(10), 1117-1132.

Floudas, C.A. and Viswewaran, V. (1990) A Global Optimization Algorithm (GOP) for Certain Classes of Nonconvex NLPs-I. Theory *Computers and Chem. Eng.* 14(12), 1397-1419.

Geoffrion, A.M. (1972). Generalized Benders Decomposition. *Journal of Optimization Theory and Applications,* 10(4), 237-260.

Grossmann, I.E. (1989). MINLP Optimization Strategies and Algorithms for Process Synthesis, in Siirola J.J. , Grossmann I.E. and Stephanopoulos G. Foundations of Computer Aided Process Design, Proc. of 3rd Int. Conf. on FOCAPD Snowmass Village, Colorado, Elsevier pp 105-132.

Grossmann, I.E. (1990). Mixed- Integer Non-Linear Programming Techniques for the Synthesis of Engineering Systems, *Res. Eng. Design.* 1 205-228.

Gupta, O.K. and Ravindran, V. (1985). Branch and Bound Experiments in Convex Nonlinear Integer Programming. *Management Science,* 31(12), 1533-1546.

Kocis, G.R and Grossmann, I.E. (1987). Relaxation Strategy for the Structural Optimization of Process Flowsheets. *Industrial and Enginnering Chemistry Research,* 26(9), 1869-1880.

Kocis, G.R and Grossmann, I.E. (1988). Global Optimization of Nonconvex MINLP Problems in Process Synthesis. *Industrial and Enginnering Chemistry Research,* 27,1407-1421.

Kocis, G.R and Grossmann, I.E. (1989). Computational Experience with DICOPT Solving MINLP Problems in Process Synthesis Engineering. *Computers and Chem. Eng.* 13, 307-315.

Nabar, S.V. and Schrage (1990). Modeling and Solving Nonlinear Integer Programming Problems. Paper No. 22a, Annual AIChE Meeting, Chicago, IL.

Nielsen, S.S. and Zenios, S.A. (1990). Mixed-Integer Nonlinear Programming on Generalized Networks. Report 89-09-12 HERMES Laboratory for Financial Modeling.

Raman, R. and Grossmann, I.E. (1991). Symbolic Integration of Logic in Mixed-Integer Linear Programming Techniques for Process Synthesis. Paper No. 157a, Annual AIChE Meeting, Los Angeles, CA.

Sagli, B., Gundersen T. and Yee T.F. (1990). Topology Traps in Evolutionary Strategies for Heat Exchanger Network Synthesis. Comp. Appl. in Chem. Eng. (Bussemaker H.T. and Iedema P.D. ed), Elsevier, Amsterdam.

Sahinidis, N.V. and Grossmann, I.E. (1991). Convergence Properties of Generalized Benders Decomposition. *Computers and Chem. Eng.* 15, 481-491.

Viswanathan, J. and Grossmann, I.E. (1990). A Combined Penalty Function and Outer-Approximation Method for MINLP Optimization. *Computers and Chem. Eng.* **14(7),**769-782.

Yee, T.F. and Grossmann, I.E. (1991). A screening and optimization approach for the retrofit of heat exchanger networks. *Ind. Engng. Chem. Res.* 30, 146-162.

Appendix. On the relation between QA and GBD cuts.

This appendix will show that the cuts in the master problem of GBD are surrogate constraints of the linearizations of the master problem in OA. Convexity of the functions f(x) and g(x) is assumed. Consider the MINLP model:

Z smirix.y $c^T y$ + flx) + M u

st By+g|M<u.

u >0 (MINLP)

x eX={x|x €R,x^L<x<x^u>

y eY={y |y *e* {0, 1}^{m}), ueR^1

Here the new variable u is included to allow violation of the constraints in the case of infeasible NLP subproblems. A large weight factor M is added to the objective function. The Outer Approximation master problem is given by :

z = min (XQ^

st $CC_O A$^$c^T y$ + fIx^k)+ Vflxty (x- x^k) + Mu^k

By +$>£) + Vgf** (x- x^k) ≤ u^k k = 1,.... K (Al)

u≥0

ae R*,y *e* Y,x *e X*

where x^k is the value of the continuous variables in the optimal solution of the k NLP subproblem. From the Kuhn-Tucker conditions with respect to x of the k-NLP subproblem we have that,

VHxty + Vgtxty X^k = 0 (A2)

Multiplying by (x-x^k) yields,

VfIx^k)^T(x-x^k) + (X^k)^{T}Vg(x^k)^T(x-x^k)= 0 (A3)

A surrogate of the constraints in the OA master problem is obtained by multiplying the linearization of the constraints in (Al) by the Langrange multipliers (which are non-negative) and adding them to the linearized objective:

α_{OA}≥ c^Ty + f(x^k) + VfIx^k)^T (x - x ^ + **M u^k+**

(Jl^k)^T[By +g(x^k) + VgW'Mx- x^k) -u^k] k =1....K (A4)

Using (A3)

$$\alpha_{OA} \geq c^T y + f(x^k) + Mu^k + U^k)^T [By + g(x\wedge) - u^k] \qquad k = 1 \ldots K \qquad (A5)$$

When the NLP subproblem is feasible, u^k is zero. For infeasible subproblems, u^k is strictly positive and then any linear combination of the constraints using positive weights is a valid cut. So the constraints can be divided in two subsets

$$a_{Gm} \pounds c^r y + f\&) + (\backslash^k)^T [By + g(x^k)] \qquad k = 1 \ldots K_{fcas} \qquad (A6)$$

$$(X^k)^T [By + g(x^k)] \leq 0 \qquad k = 1 \ldots K\wedge$$

which are the constraints of the Generalized Benders Decomposition master problem. Futhennore, we have that $OLQ_A \geq OGBD$ at any given iteration.

Table 1. Progress of major Iterations In convex example.

Iteration	y	NLP subproblem	MILP master problem
1	(0.1.0)	1.0	-3.388
2	(1. 1. 0)	-1.72	-3.0
3	(1.0. 1)	-1.92	-1.92

Table 2. Number of rows in convex example.

		Number of LP's	
NLP	#rowsLP	Original	Proposed
1	7	5	2
2	9	3	3
3	11	5	2

Table 3. Integer solutions for nonconvex example.

y	NLP subproblem	y	NLP subproblem
(0.0.0)	infeasible	(1,0.0)	-15.73
(0,1.0)	infeasible	(0,0.1)	-16.58
(1.1.0)	-14.66	(1.0.1)	-32.71
(0.1.1)	-15.91	(1.1,1)	-31.64

Table 4. Computational results of nonconvex example.

		LP subproblems		NLP
nodes	y	obj with penalty	without penalty	upper bound
(1.2)	(0,0.1)	-57.55	-57.55	-15.73
(2.3)	(1.0.1)	-14.4	-20.53	-16.58
(3.4..5.6.7.8J	(0.1.0)	11.16	-33.659	-32.71
(8.9)	----	----	-----	-32.71

Table 5. Test problems.

problem	binary	continuous	constraints	(non linear)	Starting point
hw74	3	7	8	(2)	(0.1.0)
hw3	3	2	7	(2)	(1.1.1)
french	4	7	13	(5)	(0.1,1.0)
ex3	8	25	31	(5)	(1.0.1.1,0.0.1.1)
batch5	24	22	73	(2)	y..4=i
batch8	40	32	141	(2)	yi.3=i
ex4	25	5	31	(25)	yi=l;i=l,5.6.8.11.15. 16.1718,20.21.24.25

Table 6. Computational results of OA and proposed algorithm.

	OA		proposed	
problem	nodes	NLP	nodes	NLP
hw74	13	3	7	3
hw3	13	3	7	3
french	47	5	18	5
ex3	39	3	21	6
batch5	90	4	32	4
batch8	523	10	84	10
ex4	201	3	103	5

Table 7. Comparison with partial surrogate

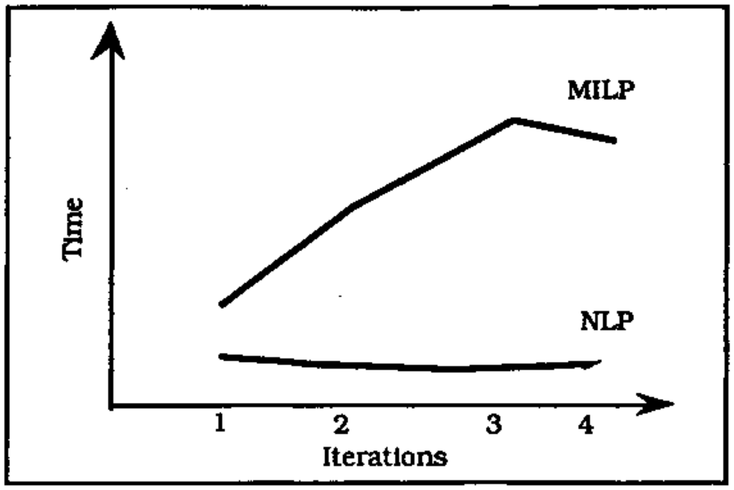
	With partial surrogate		Without	
problem	nodes	NLP	nodes	NLP
french	23	5	18	5
ex4	45	5	103	5

Table 8. Size of LP subproblems in problems french

french	Number of LP's		
LP rows	OA	proposed	partial surrogate
13	3	2	2
14	—	—	4
15	--	—	6
16	--	—	6
17	—	—	5
18	5	5	—
23	13	6	—
28	13	4	—
33	13	1	—

Table 9. Size of LP subproblems in problem ex4.

ex4	Number of LP's		
LP rows	OA	proposed	partial surrogate
31	21	9	9
32	—	—	11
33	—	—	2
34	--	--	10
35	—	—	13
56	91	13	—
81	89	42	—
106	«	21	—
131	~	18	—



Batch Design Problem: Batch 5

Iteration	NLP	MILP
1	1.52	2.44
2	1.02	5.90
3	0.96	10.16
4	1.20	8.80

CPU seconds in a Vax 6320

Figure 1. Computational cost of NLP and MILP problems.

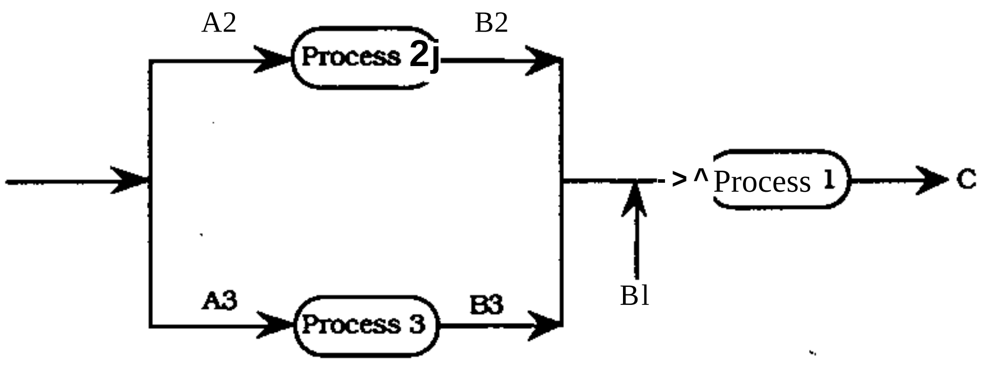
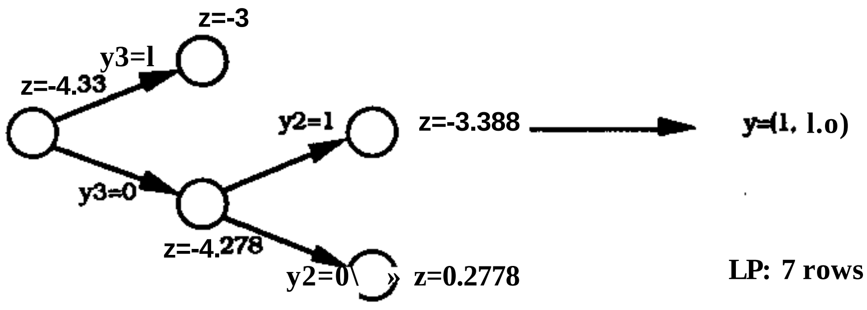
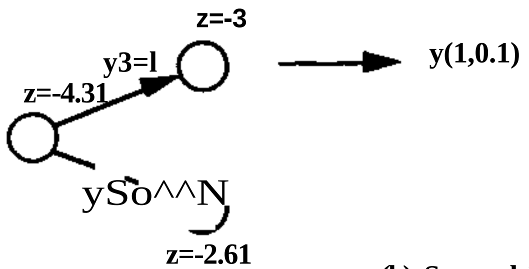


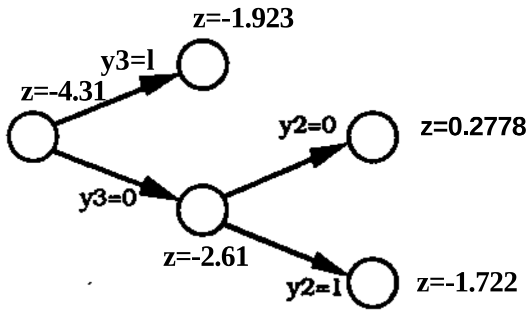
Figure 2. Superstructure for convex example.



(a) First iteration

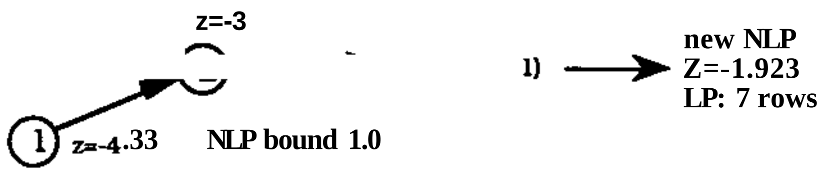


(b) Second iteration

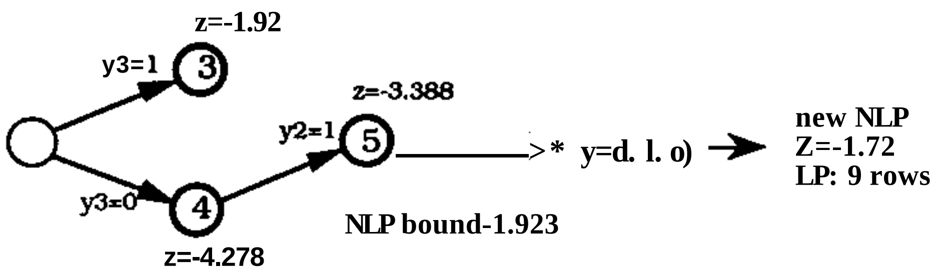


(c) Third iteration

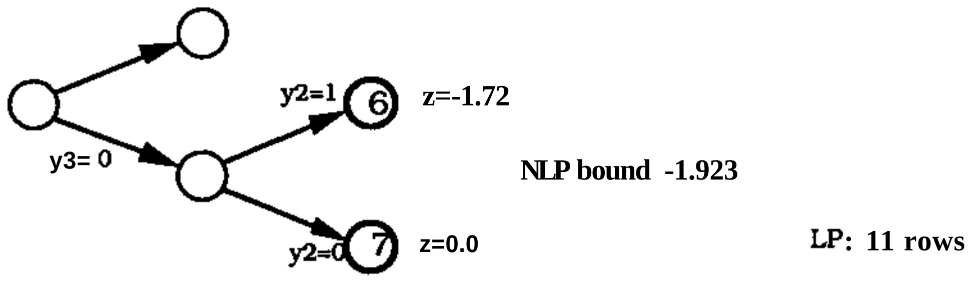
Figure 3. Branch and bound trees of MILP master in convex example.



(a) One linearization



(b) Two linearizations



(c) Three linearizations

Figure 4. Branch and bound tree in proposed method

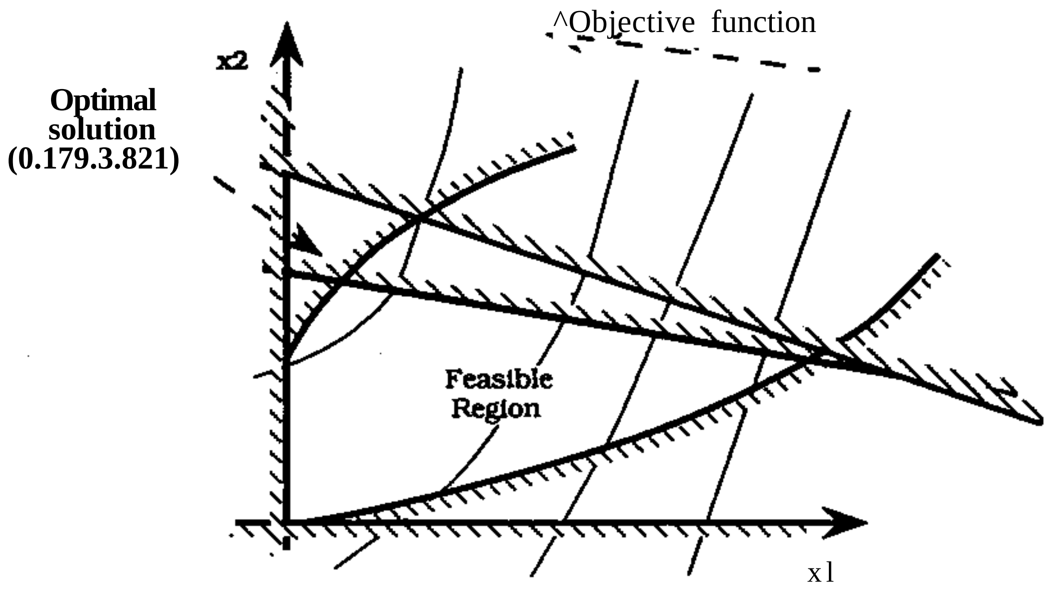


Figure 5. Continuous feasible region for y=l.

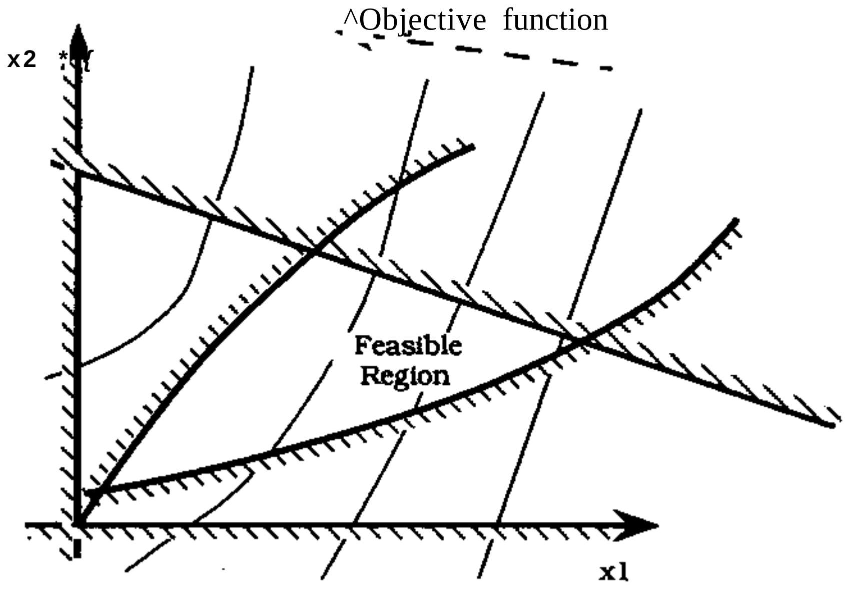


Figure 6. Continuous feasible region for y=0.

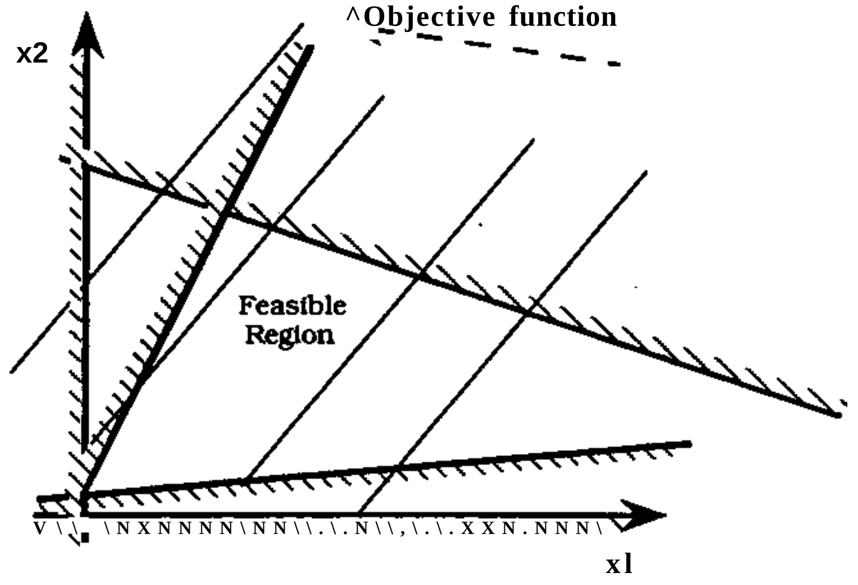


Figure 7. Continuous feasible region for OA with y=0

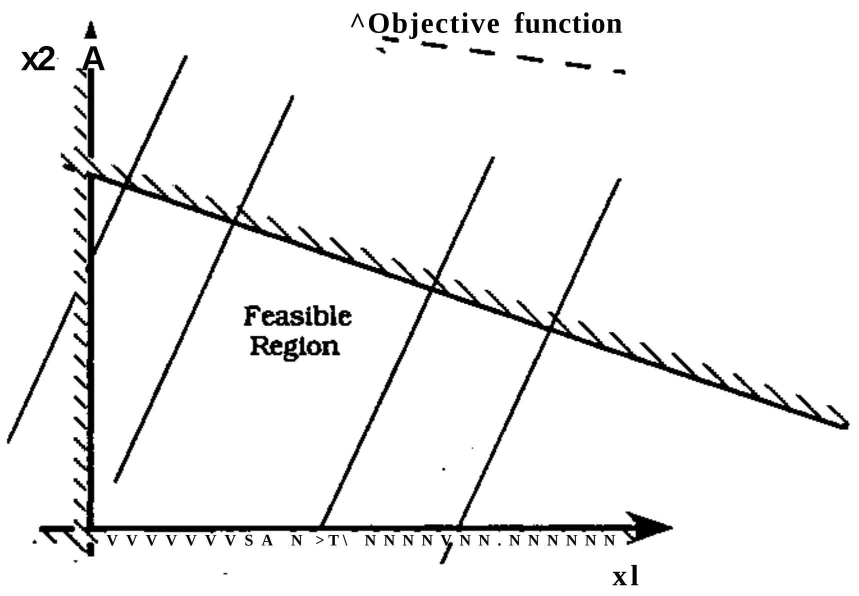


Figure 8. Continuous feasible region for partial surrogate with y=0.

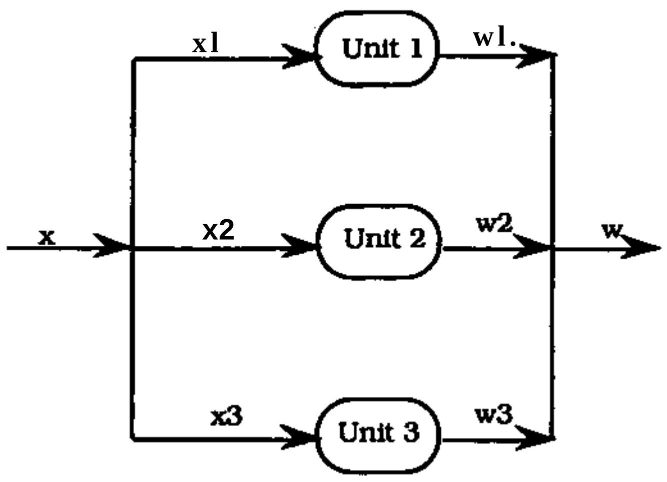


Figure 9. Superstructure for nonconvex example

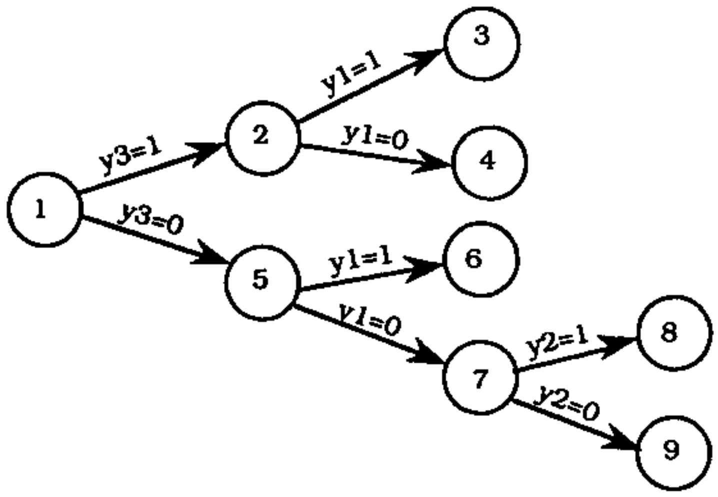


Figure 10. Branch and bound tree of nonconvex example.