Generalized Convex Disjunctive Programming: Nonlinear Convex Hull Relaxation

Ignacio E. Grossmann* and Sangbum Lee

Department of Chemical Engineering, Carnegie Mellon University
Pittsburgh, PA15213

February 11, 2002

Abstract

Generalized Disjunctive Programming (GDP) has been introduced recently as an alternative to mixed-integer programming for representing discrete/continuous optimization problems. The basic idea of GDP consists of representing these problems in terms of sets of disjunctions in the continuous space, and logic propositions in terms of Boolean variables. In this paper we consider GDP problems involving convex nonlinear inequalities in the disjunctions. Based on the work by Stubbs and Mehrotra [19] and Ceria and Soares [5], we propose a convex nonlinear relaxation of the nonlinear convex GDP problem that relies on the convex hull of each of the disjunctions that is obtained by variable disaggregation and reformulation of the inequalities. The proposed nonlinear relaxation is used to formulate the GDP problem as a Mixed-Integer Nonlinear Programming (MINLP) problem that is shown to be tighter than the conventional "big-M" formulation. A disjunctive branch and bound method is also presented, and numerical results are given for a set of test problems.

Keywords disjunctive programming, convex programming, mixed integer nonlinear programming, convex hull

1 Introduction

Mixed-integer programming models are used in many applications involving discrete/continuous optimization. Most of the research has been directed

*Author to whom correspondence should be addressed (email: grossmann@cmu.edu).

to the solution of Mixed-Integer Linear Programming (MILP) problems, and significant progress has been made in the recent past (see [2], [13], and [21]). In contrast, the solution of Mixed-Integer Nonlinear Programming (MINLP) problems has received relatively little attention despite their practical importance. MINLP problems arise, for instance, in process synthesis (heat exchanger networks and reactor networks), in engineering design (truss structures and feed location in distillation column), in planning of process networks, in the optimal positioning of products, in financial planning, and in the scheduling of batch and continuous multiproduct plants [9]. Algorithms for solving MINLP problems include the following: Branch and Bound (BB) (see [3], [10] and [15]), Outer-Approximation (OA) (see [6], [7], and [26]), Generalized Benders Decomposition (GBD) [8], Extended Cutting Plane (ECP) [24], LP/NLP based branch and bound [16], and branch-and-cut [19]. For a recent review of these methods, see Grossmann and Kravanja [9].

Generalized Disjunctive Programming (GDP), which can be regarded as a generalization of disjunctive programming [1] and as one of the approaches to logic-based optimization [12], has been introduced as an alternative model to MINLP by using disjunctions and logic propositions [18]. While the MINLP model is based entirely on algebraic equations and inequalities for the discrete/continuous optimization problem, the GDP model allows at the modeling stage the specification of a mixture of algebraic and logic equations, which can facilitate problem formulation. Türkay and Grossmann [20] have proposed a logic-based Outer-Approximation algorithm for solving nonlinear GDP problems for process networks. This algorithm is based on the idea of extending the Outer-Approximation algorithm by solving NLP subproblems in reduced space, and MILP master problems corresponding to the convex hull of the linearization of the nonlinear inequalities. Also, several NLP subproblems must be solved to initialize the master problem in order to cover all the terms in the disjunctions. This method has been implemented in LOGMIP, a computer code developed by Vecchietti and Grossmann [22].

In this paper, we address the solution of GDP problems that involve convex nonlinear inequalities. Based on the work by Stubbs and Mehrotra [19] and Ceria and Soares [5], we first describe the convex hull of a disjunction involving convex nonlinear inequalities. This is used as a basis to develop a convex nonlinear problem that can be used for formulating the GDP problem as an MINLP problem, which we prove has a tighter relaxation than the conventional big-M formulation. A disjunctive branch and bound method is

also described for solving the proposed GDP model, and its application is illustrated on a set of test problems.

2 Generalized Convex Disjunctive Programming

Consider the Generalized Convex Disjunctive Programming problem (P), which is an extension of the formulation presented by Balas [1],

$$\begin{aligned} \textbf{(P)} \qquad & \min \ Z = \sum_{k \in K} c_k + f(x) \\ & \text{s.t. } r(x) \leq 0 \\ & \bigvee_{j \in J_k} \begin{bmatrix} Y_{jk} \\ g_{jk}(x) \leq 0 \\ c_k = \gamma_{jk} \end{bmatrix}, \ k \in K \\ & \Omega(Y) = True \\ & 0 \leq x \leq U, \ c_k \in R^1, \ Y_{jk} \in \{true, false\}, \ j \in J_k, \ k \in K \end{aligned}$$

Here $x \in R^n$ is the vector of non-negative continuous variables, and Y_{jk} are Boolean variables. c_k are continuous variables and γ_{jk} are fixed charges. Parameter U is a valid upper bound to x. $f(x) : R^n \rightarrow R^1$ is a convex function in terms of continuous variables x in the objective function, and $r(x) : R^n \rightarrow R^q$ are constraints that hold regardless of the discrete decisions. A disjunction is composed of a number of terms $[\ \cdot\ ]$ that are joined by the OR operator ($\vee$) which is assumed to be exclusive. K is the index set for the disjunctions, and $J_k = \{j | j = 1, 2, ..., m_k\}$ is the index set of the corresponding terms for each disjunction $k \in K$. The jth term of the disjunction k contains the Boolean variable Y_{jk}, the nonlinear inequalities $g_{jk}(x) \leq 0$, where $g_{jk}(x) : R^n \rightarrow R^{t_{jk}}$, and a cost equation $c_k = \gamma_{jk}$. If Y_{jk} is $true$, then $g_{jk}(x) \leq 0$ and $c_k = \gamma_{jk}$ are enforced. Otherwise, these constraints are ignored. Finally, $\Omega(Y)$ corresponds to logic propositions in terms of the Boolean variables that are expressed in Conjunctive Normal Form (CNF):

$$\Omega(Y) = \bigwedge_{l=1,2,...,L} \left[\bigvee_{Y_{jk} \in \ P_l} (Y_{jk}) \bigvee_{Y_{jk} \in \ Q_l} (\neg Y_{jk}) \right] \tag{1}$$

For each clause $l, l = 1, 2, ..., L$, P_l is the subset of Boolean variables Y_{jk} that are non-negated, and Q_l is the subset of Boolean variables Y_{jk} that are negated.

In problem (P) and in the following problems, the functions $f(x)$, $r(x)$, and $g_{jk}(x)$ are assumed to be convex, bounded, and differentiable over the

convex set $0 \leq x \leq U$. Also, it is assumed that problem (P) has a non-empty compact feasible set, and that each term in the disjunctions gives rise to a non-empty feasible set.

3 Convex Hull Relaxation

Consider a disjunction $k \in K$, which arises in problem (P),

$$\begin{array}{c} \bigvee \\ j \in J_k \end{array} \left[\begin{array}{c} Y_{jk} \\ g_{jk}(x) \leq 0 \\ c_k = \gamma_{jk} \end{array} \right] \\ 0 \leq x \leq U,\ c_k \in R^1 \tag{2}$$

Figure 1 shows a simple example for the case of a disjunction with three terms that give rise to three subsets, S_1, S_2, and S_3.

We extend the work by Stubbs and Mehrotra [19] for obtaining the convex hull of the disjunctive feasible set in (2) in the (x, c_k) space. We first establish the following definition.

Definition *The convex hull of the disjunction in (2) is given by the projection* $P(F_k^D) = \{(x, c_k) | (x, c_k, u_{jk}, \lambda_{jk}) \in F_k^D\}$ *where* F_k^D *consists of all* $(x, c_k, u_{jk}, \lambda_{jk})$ *which satisfy*

$$x = \sum_{j \in J_k} \lambda_{jk} u_{jk} \tag{3a}$$

$$c_k = \sum_{j \in J_k} \lambda_{jk} \gamma_{jk} \tag{3b}$$

$$\sum_{j \in J_k} \lambda_{jk} = 1 \tag{3c}$$

$$g_{jk}(u_{jk}) \leq 0,\ j \in J_k \tag{3d}$$

$$0 \leq u_{jk} \leq U,\ j \in J_k \tag{3e}$$

$$0 \leq \lambda_{jk} \leq 1,\ j \in J_k \tag{3f}$$

The equations and inequalities in (3) correspond to a continuous relaxation of the disjunction in (2) and can in principle be used as a continuous

nonlinear relaxation of problem (P). The difficulty, however, is that the definition of x as a convex combination of the variables u_{jk} involves bilinearities. The convexification of (3) can be accomplished by first considering the following proposition.

Proposition 1 *Let $h(\lambda, \nu)$ be defined as*

$$h(\lambda, \nu) \equiv \begin{cases} \lambda g(\nu/\lambda) & if \ \lambda > 0 \\ 0 & if \ \lambda = 0 \end{cases} \tag{4}$$

over the set $0 \leq \nu \leq \lambda U$ and $0 \leq \lambda \leq 1$. If $g(x)$ is convex and bounded over the set $\{x \mid 0 \leq x \leq U\}$, then $h(\lambda, \nu)$ is a bounded convex function.

Proof. See Stubbs and Mehrotra [19].

Note that $h(0, \nu) = h(0, 0) = 0$ since $\nu = 0$ when $\lambda = 0$ from $0 \leq \nu \leq \lambda U$. In the following, we will always understand that $\lambda g(\nu/\lambda)$ is zero when $\lambda = 0$, which is the continuous extension of $h(\lambda, \nu)$. Also, note that the function $h(\lambda, \nu)$ is continuous, but not differentiable at $(\lambda, \nu) = (0, 0)$. The convexity of the function $\lambda g(\nu/\lambda)$ is discussed in detail in [11].

Proposition 2 *The following equations and inequalities define a convex set $\hat{F}_k^D = \{(x, c_k, \nu^{jk}, \lambda_{jk})\}$ where $P(\hat{F}_k^D) = P(F_k^D)$.*

$$x = \sum_{j \in J_k} \nu^{jk} \tag{5a}$$

$$c_k = \sum_{j \in J_k} \lambda_{jk} \gamma_{jk} \tag{5b}$$

$$\sum_{j \in J_k} \lambda_{jk} = 1 \tag{5c}$$

$$\lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) \leq 0, \ j \in J_k \tag{5d}$$

$$0 \leq \nu^{jk} \leq \lambda_{jk} U, \ j \in J_k \tag{5e}$$

$$0 \leq \lambda_{jk} \leq 1, \ j \in J_k \tag{5f}$$

Proof. Since the equation (3a) is bilinear, we linearize it by introducing the continuous variable $\nu^{jk} = \lambda_{jk} u_{jk}$, which leads to the equation

$$x = \sum_{j \in J_k} \nu^{jk} \tag{6}$$

Since $\nu^{jk} = \lambda_{jk} u_{jk}$, it follows that for $\lambda_{jk} \geq 0$

$$\lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) \leq 0, \ j \in J_k \tag{7}$$

which is a convex inequality from Proposition 1. Also, we define $\lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) = 0$ for $\lambda_{jk} = 0$ by (4). Hence, the equations and inequalities in (5) define a convex set $\hat{F}_k^D$.

Choose an arbitrary point $(x, c_k) \in P(F_k^D)$. Then there exist u_{jk} and λ_{jk} which satisfy (3). Defining $\nu^{jk} = \lambda_{jk} u_{jk}$, it is clear that (5a) and (5e) hold. In addition, for $\lambda_{jk} > 0$, (5d) follows from (3d). For $\lambda_{jk} = 0$, $\lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) = 0$ by definition. Hence, (5) holds and $(x, c_k) \in P(\hat{F}_k^D)$. It follows that $P(F_k^D) \subseteq P(\hat{F}_k^D)$.

Now let $(x, c_k) \in P(\hat{F}_k^D)$. Then there exist λ_{jk} and ν^{jk} which satisfy (5). For $\lambda_{jk} > 0$, define $u_{jk} = \nu^{jk}/\lambda_{jk}$, which then satisfies (3d) and (3e). For $\lambda_{jk} = 0$, choose arbitrary u_{jk} satisfying (3d) and (3e), which exist by assumption. Clearly, (3a) holds as well. Thus, $P(\hat{F}_k^D) \subseteq P(F_k^D)$. This implies $P(\hat{F}_k^D) = P(F_k^D)$. □

It should be noted that the equations and inequalites in (5) are a convex relaxation of the disjunction in (2).

4 Nonlinear Convex Relaxation Problem

We define a continuous relaxation for problem (P) using as a basis the equations (5) of the convex hull of each disjunction $k \in K$ in (2). Since this relaxation problem involves no Boolean variables, the logic propositions are represented in terms of the continuous variables λ_{jk} with the linear inequalities $A\lambda \leq a$ which can be systematically derived from the CNF form of $\Omega(Y)$ [25]. The Convex Relaxation Programming (CRP) problem is then given as follows:

$$
\begin{aligned}
\textbf{(CRP)} \qquad \min \;\; & Z^L = \sum_{k\in K}\sum_{j\in J_k} \gamma_{jk}\lambda_{jk} + f(x) \\
& \text{s.t.} \;\; r(x) \le 0 \\
& x = \sum_{j\in J_k} \nu^{jk}, \;\; k \in K \\
& \lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) \le 0, \;\; j \in J_k, \;\; k \in K \\
& \sum_{j\in J_k} \lambda_{jk} = 1, \;\; k \in K \\
& 0 \le \nu^{jk} \le \lambda_{jk} U, \;\; j \in J_k, \;\; k \in K \\
& A\lambda \le a \\
& 0 \le x, \;\; \nu^{jk} \le U, \;\; 0 \le \lambda_{jk} \le 1, \;\; j \in J_k, \;\; k \in K
\end{aligned}
$$

It should be noted that by assumption, the function $g_{jk}(x)$ is bounded over the feasible set. Therefore, if λ_{jk} converges to 0, ν^{jk} and the term $\lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk})$ also converges to zero. However, this function is not differentiable at $\lambda_{jk} = 0$.

Problem (CRP), which can be regarded as an extension of the work of Ceria and Soares [5] for disjunctive optimization, does not include a barrier function in the objective to avoid division by zero in the nonlinear inequalities. Instead, for the implementation, if the function $g_{jk}(x)$ is nonlinear we approximate the constraint $\lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) \le 0$ by,

$$
(\lambda_{jk} + \varepsilon) g_{jk}(\nu^{jk}/(\lambda_{jk} + \varepsilon)) \le 0 \tag{8}
$$

where ε is a small tolerance. The approximation problem given by (CRP) and with the constraint in (8) will be denoted by CRP(ε). The approximated constraint in (8) is also convex under the assumption that $g_{jk}(x) \le 0$ is convex as proved in Appendix A. Furthermore, the constraint function in (8) is continuous and differentiable. Hence, by setting ε to a sufficiently small value, problem (CRP) can be approximated arbitrarily close by the convex differentiable nonlinear problem CRP(ε). However, too small a value of ε can lead to ill-conditioning and round-off errors. Therefore, in practice a small finite value must be used (e.g. $\varepsilon = 10^{-4}$).

Proposition 3 *The optimal solution $(Z^L)^*$ of problem (CRP) is a lower bound to the optimal solution Z^* of problem (P).*

Proof. This follows trivially from the fact that the constraints in problem (CRP) correspond to a continuous relaxation of the disjunction in problem

(P). □

The above properties of problem (CRP) can be exploited for formulating problem (P) as an MINLP problem or for developing a branch and bound search procedure as will be shown later in the paper. It should be noted, however, that since problem CRP(ε) is the one that is actually solved in the implementation, this problem will yield an approximation to problem (CRP), and hence an approximation to the lower bound of problem (P).

5 MINLP Formulation

Having derived the nonlinear convex relaxation problem (CRP), there are two major solution approaches one can take for solving problem (P). The simplest and most direct way is to formulate the following MINLP problem,

$$
\begin{aligned}
\textbf{(PR)} \qquad \min \;\; & Z = \sum_{k \in K} \sum_{j \in J_k} \gamma_{jk} \lambda_{jk} + f(x) \\
\text{s.t.} \;\; & r(x) \leq 0 \\
& x = \sum_{j \in J_k} \nu^{jk}, \;\; k \in K \\
& \lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) \leq 0, \;\; j \in J_k, \;\; k \in K \\
& 0 \leq \nu^{jk} \leq \lambda_{jk} U, \;\; j \in J_k, \;\; k \in K \\
& \sum_{j \in J_k} \lambda_{jk} = 1, \;\; k \in K \\
& A\lambda \leq a \\
& 0 \leq x, \;\; \nu^{jk} \leq U, \;\; \lambda_{jk} \in \{0,1\}, \;\; j \in J_k, \;\; k \in K
\end{aligned}
$$

Note that in problem (PR) the variables λ_{jk} are binary variables. If we relax them as continuous variables between 0 and 1, problem (PR) is identical to problem (CRP). Similarly as in problem (CRP), the nonlinear inequalities $\lambda_{jk} g_{jk}(\nu^{jk}/\lambda_{jk}) \leq 0$ are approximated by (8) to avoid division by zero and to eliminate the non-differentiability. This approximation problem PR(ε) represents an approximation to problem (P) which can be set arbitrarily close with a sufficiently small value of ε.

The GDP problem (P) can also be formulated as the following MINLP problem (BM) by replacing the Boolean variables Y_{jk} by binary variables y_{jk}, and using big-M constraints. The logic constraints $\Omega(Y)$ are converted

into linear inequalities, which leads to the following problem,

$$\begin{aligned}
\textbf{(BM)} \qquad \min \;\; & Z = \sum_{k\in K}\sum_{j\in J_k} \gamma_{jk} y_{jk} + f(x) \\
& \text{s.t.} \;\; r(x) \leq 0 \\
& g_{jk}(x) \leq M_{jk}(1 - y_{jk}), \;\; j \in J_k, \;\; k \in K \\
& \sum_{j\in J_k} y_{jk} = 1, \;\; k \in K \\
& Ay \leq a \\
& 0 \leq x \leq U, \;\; y_{jk} \in \{0,1\}, \;\; j \in J_k, \;\; k \in K
\end{aligned}$$

In the above MINLP model, M_{jk} are the "big-M" parameters that render the inequalities $g_{jk}(x) \leq 0$ redundant when $y_{jk} = 0$. The inequalities $Ay \leq a$ are identical to those in problem (PR). Also, note that problem (BM) has considerably fewer variables and constraints than problem (PR).

The following proposition holds for problems (PR) and (BM).

Proposition 4 *Let Z_R^{PR} be the optimal solution of problem (PR) when the binary variables are relaxed as $0 \leq \lambda_{jk} \leq 1$. And let Z_R^{BM} be the optimal solution of problem (BM) when the binary variables are relaxed as $0 \leq y_{jk} \leq 1$. Then, $Z_R^{BM} \leq Z_R^{PR}$.*

Proof. Consider any feasible point $(x, \lambda_{jk}, \nu^{jk})$ in problem (CRP) which results from relaxing the variables $0 \leq \lambda_{jk} \leq 1$ in problem (PR). Since $x = \sum_{j\in J_k} \nu^{jk}, \;\; k \in K$, it implies that there exist u_{jk} such that $\lambda_{jk} u_{jk} = \nu^{jk}, \;\; g_{jk}(u_{jk}) \leq 0, \;\; j \in J_k, \;\; k \in K$. Since $g_{jk}(x)$ are convex functions, for any $l \in J_k$,

$$g_{lk}(x) = g_{lk}\Big(\sum_{j\in J_k} \lambda_{jk} u_{jk}\Big) \leq \sum_{j\in J_k} \lambda_{jk} g_{lk}(u_{jk}) \tag{9}$$

For $g_{lk}(u_{lk}) \leq 0$ and $g_{lk}(u_{jk})_{j\neq l} \leq M_{lk}$, it follows from (9) that for $0 \leq \lambda_{jk} \leq 1$, $\sum_{j\in J_k} \lambda_{jk} = 1$,

$$g_{lk}(x) \leq \sum_{j\in J_k, j\neq l} \lambda_{jk} M_{lk} = M_{lk}(1 - \lambda_{lk}) \tag{10}$$

But (10) corresponds to relaxing each constraints $l \in J_k$ in problem (BM) by setting $0 \leq y_{lk} \leq 1, y_{lk} = \lambda_{lk}$. Hence, since any feasible point $(x, \lambda_{jk}, \nu^{jk})$ in problem (CRP) corresponds to a feasible point (x, y_{jk}) in the relaxation of

problem (BM) with the same objective value , it follows that $Z_R^{BM} \leq Z_R^{PR}$. $\square$

Thus, the relaxation of problem (PR) is generally tighter than the relaxation of problem (BM). It should be noted that for implementation problem PR(ε) is the one that is actually solved. This problem PR(ε) can be solved with any standard method for MINLP problem discussed in the introduction section (e.g., Branch and Bound, Outer-Approximation, Generalized Benders Decomposition, and Extended Cutting Plane method). The other alternative is to develop a specific branch and bound method for solving problem (P).

6 A Branch and Bound Algorithm with Convex Relaxation

We consider problem CRP(ε) as a basis for a branch and bound algorithm for solving problem (P) with a sufficiently small value of ε. The variables λ_{jk} can be interpreted as weights that measure the "closeness" by which each term of the disjunction is satisfied ($x = \nu^{jk}$ for $\lambda_{jk} = 1$). Generally, solving problem CRP(ε) yields a solution λ_{jk} with non-integer values. Therefore, the branching rule that can be used in the branch and bound algorithm is to select the variable λ_{im} closest to 1 because this corresponds to the disjunctive term that is closest to being feasible. The corresponding NLP subproblem is then defined by the ith term of the mth disjunction, and the convex hull of all remaining disjunction $k \in K,\ k \neq m$. This NLP problem should potentially yield a good upper bound.

After branching on $\lambda_{im} = 1$, we propose for the complementary node to select the convex hull of the remaining terms $j \neq i$ of the disjunction m, and the convex hull of the disjunctions $k \in K,\ k \neq m$. This then corresponds to the following dichotomy at the root node:

- $either\ fix[S_i^m \bigcap_{\substack{k \in K \\ k \neq m}} conv(\bigcup_{j \in J_k} S_j^k)]\ or\ fix[\ conv(\bigcup_{\substack{j \in J_m \\ j \neq i}} S_j^m) \bigcap_{\substack{k \in K \\ k \neq m}} conv(\bigcup_{j \in J_k} S_j^k)]$

where S_j^k is the feasible set corresponding to term j of disjunction $k \in K$. As will be shown later with the results, this branching rule, which can be easily generalized for every node, is generally very effective. The main steps of the proposed branch and bound algorithm are then as follows (see Figure 4 for a specific example):

6.1 A Branch and Bound Algorithm for Generalized Disjunctive Programs

Step 0. Initialization
(a) Set the upper bound $Z^U = \infty$.
(b) Set the tolerances ε, δ.
Step 1. Root node
(a) For each disjunction $k \in K$, formulate the convex hull of $S_j^k,\ j \in J_k$.
(b) Solve problem CRP(ε). Let the optimal solution be x^L and Z^L.
(lower bound)
Step 2. Check feasibility
(a) If all λ_{jk} are 0 or 1, then the optimal solution is found.
Set $x^* = x^L, Z^* = Z^L$. Exit.
(b) Else go to step 3.
Step 3. While there are unexamined nodes in the branch and bound tree,
Step 3.1. Branch on a disjunction.
(a) Select the largest value of λ_{jk} $(\neq 0, 1)$, $j \in J_k,\ k \in K$ in the solution.
(b) Let the largest one be $\lambda_{im},\ i \in J_m,\ m \in K$. Fix λ_{im} as 1.
(fix term i in the disjunction m)
(c) Solve problem CRP(ε). Let the optimal solution be x^L and Z^L.
(d) If all λ_{jk} are 0 or 1, then a feasible solution is found.
If $Z^L \leq Z^U + \delta$, then set $Z^U = Z^L$ and $x^U = x^L$. Go to step 3.2.
(e) Else if the solution is infeasible, then go to step 3.2.
(f) Else if $Z^L > Z^U + \delta$, then go to step 3.2.
(g) Else go to step 3.1 (repeat branching).
Step 3.2. Branch on the convex hull of the complement.
(a) Fix λ_{im} as 0 (remove term i from disjunction m).
(b) Solve problem CRP(ε). Let the optimal solution be x^L and Z^L.
(c) If all λ_{jk} are 0 or 1, then a feasible solution is found.
If $Z^L \leq Z^U + \delta$, then set $Z^U = Z^L$ and $x^U = x^L$. Backtrack.
(d) Else if the solution is infeasible, then backtrack.
(e) Else if $Z^L > Z^U + \delta$, then backtrack.
(f) Else go to step 3.1.

6.2 Remarks

The above algorithm has finite convergence since the number of terms in the disjunction is finite. Also, since the nonlinear functions are convex, the problem CRP(ε) has a unique optimal objective function value. Hence, the rigorous validity of the bounds can be guaranteed within the tolerance δ,

with which the branch and bound method is in turn guaranteed to obtain the optimal solution within a tolerance δ.

An important point in the proposed branch and bound algorithm is the case when the proposed branching rule does not yield a partition of the relaxed feasible set. This may arise as follows. Consider for simplicity the case when $|K| = 1$, and denote by x^L the optimal solution for the convex hull of that disjunction in the x space. After searching one particular feasible subset S_i^k, the convex hull of the remaining feasible subsets ($S_{j,j\neq i}^k$) should generally yield an increase to the lower bound compared to the solution from the parent node. However, if this lower bound is unchanged, there is the need to verify whether partitioning has in fact taken place. This can be performed with the following test. If $x^L \notin conv(\vee S_{j,j\neq i}^k)$, then this is a 'partitionable set' (see Figure 2(a)). If $x^L \in conv(\vee S_{j,j\neq i}^k)$, then the set of subsets is a 'non-partitionable set' (see Figure 2(b)) because the point x^L remains feasible in the convex hull of the subsets ($S_{j,j\neq i}^k$) and hence the lower bound remains unchanged. When such a case arises, the branching rule can be modified to consider a further partition of the remaining subsets (e.g. consider $(S_1 \vee S_3)$ in Figure 2(b)).

7 Numerical Results

7.1 Example 1

This problem was originally formulated as a convex MINLP problem [9]. The corresponding GDP problem is given by (11) and its feasible set projected onto the (x_1, x_2) space is shown in Figure 3. The optimal solution is $Y^* = (false, true, false)$, $x^* = (1, 1)$, and $Z^* = 3.5$.

$$
\begin{gathered}
\min \;\; Z = c + x_1^2 + x_2^2 \\
\text{s.t.} \;\; (x_1 - 2)^2 - x_2 \leq 0 \\
\begin{bmatrix} Y_1 \\ x_1 - 2 \geq 0 \\ x_1 - x_2 \leq 4 \\ c = 1 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ x_1 - x_2 \leq 0 \\ x_1 - 1 \geq 0 \\ x_2 - 1 \geq 0 \\ c = 1.5 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ x_1 - x_2 \leq 4 \\ x_1 + x_2 \geq 3 \\ x_1 - 1 \geq 0 \\ c = 0.5 \end{bmatrix} \\
0 \leq x_1, x_2 \leq 4, \;\; c \in R^1 \\
Y_j \in \{true, false\}, \;\; j = 1, 2, 3
\end{gathered}
\tag{11}
$$

Table 1 shows the results of the standard branch and bound and outer-approximation as applied to the big-M formulation. These are compared with the proposed branch and bound, and the outer-approximation method [6]. Note that since the disjunction is linear, there is no need to use the approximation in (8). The following big-M MINLP (BM) was used for comparison:

$$\begin{aligned} \min \quad & Z = y_1 + 1.5y_2 + 0.5y_3 + x_1^2 + x_2^2 \\ & \text{s.t.} \quad (x_1 - 2)^2 - x_2 \leq 0 \\ & x_1 - 2y_1 \geq 0 \\ & x_1 - x_2 - 4(1 - y_2) \leq 0 \\ & x_1 - (1 - y_1) \geq 0 \\ & x_2 - y_2 \geq 0 \\ & x_1 + x_2 \geq 3y_3 \\ & y_1 + y_2 + y_3 = 1 \\ & 0 \leq x_1, x_2 \leq 4 \\ & y_j \in \{0, 1\}, \ \ j = 1, 2, 3 \end{aligned} \tag{12}$$

In Table 1 the lower bounds are obtained from the corresponding relaxation problems where the integrality of y_j is relaxed. Note that the relaxation problem (CRP) predicts tighter lower bound than problem (BM) (3.468 vs. 2.532). As can also be seen from Table 1, the proposed branch and bound algorithm requires three subproblems, and its corresponding tree is shown in Figure 4.

7.2 Numerical Comparisons with Big-M Formulation

In this section, a total of 9 convex GDP test problems are considered. The model equations of examples 2, 3, and 4 are given in the Appendix B. Examples 5, 6 and 7 are from [17] and [18], and are linear; all others are nonlinear GDP problems. Example 8 is the test problem 4 of Duran and Grossmann [6], which deals with the optimal positioning of new products. The GDP formulation for this example can be found in Lee and Grossmann [14]. Example 9 is a process network problem by Türkay and Grossmann [20] based on example 3 of Duran and Grossmann [6].

All the test problems were solved with GAMS [4] on a 300MHz Pentium II PC. The GAMS/CONOPT NLP solver was used in both BB algorithms and comparisons were also performed with the convex option of DICOPT++

[23], which is equivalent to the OA algorithm and starts by solving the relaxed MINLP. For the approximation in (8), the value of $\varepsilon = 10^{-4}$ was used in all cases. Values of ε ranging from 10^{-5} to 10^{-3} result in about 0.06 % $\sim$ 0.1 % of difference in the optimal solution of problem PR(ε) when compared with the exact solution of problem (P) as is shown in Table 2.

Table 3 shows a comparison between a standard branch and bound algorithm and the proposed algorithm. The first method converts the GDP problem into a big-M MINLP (BM)[1]. The second method solves problem CRP(ε) using the branch and bound algorithm proposed in this paper. The first five columns in Table 3 indicate the problem identity, number of binary variables, number of continuous variables, number of constraints, and number of nonlinear constraints. The optimal solutions are shown in the sixth column. The optimal solutions of the relaxed (BM) problems and CRP(ε) problems are the lower bounds obtained at the root node of the search tree, where the discrete variables are relaxed. As shown in the last four columns, and as predicted by Proposition 4, problem CRP(ε) yields tighter lower bounds than the relaxation of problem (BM), and hence the proposed branch and bound method requires fewer nodes. Note that the proposed algorithm finds the optimal solution of example 9 in only 5 nodes compared with 17 nodes of the standard branch and bound method. The branch and cut method by Stubbs and Mehrotra [19] required 12 nodes to solve this problem after adding 16 cuts.

Table 4 shows the comparison of the problems PR(ε) and (BM). For nonlinear problems, the OA algorithm as implemented in GAMS/DICOPT++ was used for solving both MINLP problems. For linear problems CPLEX was used as the MILP solver. Since the problem size of PR(ε) is larger than (BM) due to the presence of the additional variables and constraints, sometimes it requires more CPU time to find the optimal solution. This is particularly true for the linear examples 5, 6, and 7. However, problem PR(ε) yields tighter lower bounds (see Table 3) and enhances the efficiency of the search in several cases as is clearly shown in Examples 8 and 9, in which the CPU time and number of major iterations are considerably reduced.

[1] Interested readers can obtain these models from grossmann@cmu.edu

8 Conclusion

This paper has addressed the solution of generalized disjunctive programming problems, which correspond to discrete/continuous optimization problems that involve disjunctions with convex nonlinear inequalities and logic propositions. A convex nonlinear relaxation of the GDP problem has been proposed and its properties presented. The proposed CRP(ε) problem of the GDP problem is based on the convex hull of each nonlinear disjunction. It was proved that the solution of problem (CRP) yields a lower bound to the GDP problem, which is greater than or equal to the lower bound obtained from the relaxation of problem (BM). The proposed MINLP problem PR(ε), which is based on the convex hull relaxation of the GDP problem, can be solved with algorithms such as branch and bound or OA method. A special purpose branch and bound algorithm for the GDP problem was also proposed that is based on problem CRP(ε).

The numerical results of nine GDP test problems were presented and compared with standard MINLP algorithms. Both, the special purpose branch and bound method and the problem PR(ε), showed improved performance in terms of lower bounds and number of subproblems that were solved. Larger test problems will have to be solved to draw more definite conclusions.

Acknowledgment The authors would like to acknowledge financial support from the NSF Grant CTS-9710303 and the partial support from Eastman Chemical. The helpful comments by Andreas Wächter are also gratefully acknowledged.

References

[1] Balas E., Disjunctive Programming and a Hierarchy of Relaxations for Discrete Optimization Problems. SIAM J. Alg. Disc. Meth. **6,** (1985) 466-486.

[2] Balas E., S. Ceria and G. Cornuejols, A lift and project cutting plane algorithm for mixed 0-1 programs. Mathematical Programming **58,** (1993) 295-324.

[3] Borchers B. and J.E. Mitchell, An Improved Branch and Bound Algorithm for Mixed Integer Nonlinear Programming. Computers and Operations Research **21,** (1994) 395-367.

[4] Brooke A., D. Kendrick, A. Meeraus and R. Raman, *GAMS Language Guide, Release 2.25, Version 92.* (GAMS Development Corporation, 1997)

[5] Ceria S. and J. Soares, Convex Programming for Disjunctive Optimization. Mathematical Programming **86,** (1999) 595-614.

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

[7] Fletcher R. and S. Leyffer, Solving Mixed Nonlinear Programs by Outer Approximation. Mathematical Programming **66,** (1994) 327-349.

[8] Geoffrion A.M., Generalized Benders Decomposition. Journal of Optimization Theory and Application **10,** (1972) 237-260.

[9] Grossmann I.E. and Z. Kravanja, Mixed-Integer Nonlinear Programming: A Survey of Algorithms and Applications *Large-Scale Optimization with Applications, Part II: Optimal Design and Control (eds. L.T. Biegler et al.)* (Springer-Verlag, 1997) 73-100.

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

[11] Hiriart-Urruty J. and C. Lemaréchal, *Convex Analysis and minimization algorithms,* vol. 1. (Springer-Verlag, 1993).

[12] Hooker J.N., *Logic-based methods for Optimization: Combining Optimization and Constraints Satisfaction.* (Wiley, May 2000).

[13] Johnson E.L., G.L. Nemhauser and M.W.P. Savelsbergh, Progress in Linear Programming Based Branch-and-Bound Algorithms: An Exposition. INFORMS Journal on Computing **12,** (2000) 2-23.

[14] Lee S., I.E. Grossmann, New Algorithms for nonlinear generalized disjunctive programming. Computers Chem. Engng. **24,** (2000) 2125-2141.

[15] Leyffer S., Integrating SQP and branch-and-bound for Mixed Integer Nonlinear Programming. Computational Optimization and Applications **18,** (2001) 295-309.

[16] Quesada I. and I.E. Grossmann, An LP/NLP Based Branch and Bound Algorithm for Convex MINLP Optimization Problems. Computers Chem. Engng. **16,** (1992) 937-947.

[17] Raman R. and I.E. Grossmann, Symbolic Integration of Logic in MILP Branch and Bound Methods for the Synthesis of Process Networks. Annals of Operations Research **42,** (1993) 169-191.

[18] Raman R. and I.E. Grossmann, Modelling and Computational Techniques for Logic Based Integer Programming. Computers Chem. Engng. **18,** (1994) 563-578.

[19] Stubbs R. and S. Mehrotra, A Branch-and-Cut Method for 0-1 Mixed Convex Programming. Mathematical Programming **86,** (1999) 515-532.

[20] Türkay M. and I.E. Grossmann, Logic-based MINLP Algorithms for the Optimal Synthesis of Process Networks. Computers Chem. Engng. **20,** (1996) 959-978.

[21] Van Roy T.J. and L.A. Wolsey, Solving Mixed 0-1 Programs by Automatic Reformulation, Operations Research **35,** (1987) 45-57.

[22] Vecchietti A. and I.E. Grossmann, LOGMIP: A Disjunctive 0-1 Nonlinear Optimizer for Process Systems Models. Computers Chem. Engng. **23,** (1999) 555-565.

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

[24] Westerlund T. and F. Petterson, An Extended Cutting Plane Method for Solving Convex MINLP Problems. Computers Chem. Engng. **19,** (1995) S131-S136.

[25] Williams H.P., *Model building in mathematical programming* (John Wiley & Sons, Inc., 1985).

[26] Yuan X, S. Zhang, L. Piboleau and S. Domenech, Une Methode d′optimization Nonlineare en Variables Mixtes pour la Conception de Porcedes. Rairo Recherche Operationnele **22,** (1988) 331.

9 Appendix

A. Proof of Convexity of (8)

PROPERTY 1. Let $h'(\lambda,\nu) = (\lambda+\varepsilon)g(\nu/(\lambda+\varepsilon))$ be defined over $0 \le \lambda \le 1,\ 0 \le \nu \le \lambda U$. If $g(x)$ is convex and bounded over the feasible set $0 \le x \le U$, then $h'(\lambda,\nu)$ is a bounded convex function.

Proof. Let (λ_1,ν^1) and (λ_2,ν^2) be any two points in the feasible set, and $0 \le \alpha \le 1$. For $\lambda_1 \ge 0$ and $\lambda_2 \ge 0$,

$$
\begin{aligned}
&h'(\alpha\lambda_1+(1-\alpha)\lambda_2, \alpha\nu^1+(1-\alpha)\nu^2)\\
&= (\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon)g\left(\frac{\alpha\nu^1+(1-\alpha)\nu^2}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}\right)\\
&= (\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon)g\left(\frac{\alpha\nu^1}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}+\frac{(1-\alpha)\nu^2}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}\right)\\
&= (\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon)\times\\
&g\left(\frac{\alpha(\lambda_1+\varepsilon)}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}\times\frac{\nu^1}{(\lambda_1+\varepsilon)}+\frac{(1-\alpha)(\lambda_2+\varepsilon)}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}\times\frac{\nu^2}{(\lambda_2+\varepsilon)}\right)
\end{aligned}
$$

Since

$$\frac{\alpha(\lambda_1+\varepsilon)}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}+\frac{(1-\alpha)(\lambda_2+\varepsilon)}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}=\frac{\alpha(\lambda_1+\varepsilon)+(1-\alpha)(\lambda_2+\varepsilon)}{\alpha\lambda_1+(1-\alpha)\lambda_2+\varepsilon}=1$$

it follows that

$$
\begin{aligned}
&h'(\alpha\lambda_1+(1-\alpha)\lambda_2, \alpha\nu^1+(1-\alpha)\nu^2)\\
&\le\ \alpha(\lambda_1+\varepsilon)g(\nu^1/(\lambda_1+\varepsilon))+(1-\alpha)(\lambda_2+\varepsilon)g(\nu^2/(\lambda_2+\varepsilon))\\
&=\ \alpha h'(\lambda_1,\nu^1)+(1-\alpha)h'(\lambda_2,\nu^2)
\end{aligned}
$$

The inequality above follows from the convexity of $g(x)$. The boundedness of $h'(\lambda,\nu)$ follows from λ and $g(\nu/(\lambda+\varepsilon))$ being bounded over the feasible set. □

B. GDP Models of Example Problems

Example 2

$$\min \quad Z = c + (x_1 - 2)^2 + (x_2 - 1)^2$$
$$\text{s.t.}$$
$$\begin{bmatrix} Y_1 \\ (x_1-4)^2 - x_2 \le 0 \\ -(x_1-2) + x_2 \le 0 \\ c = 5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ 2x_1 + x_2 - 4 \le 0 \\ 2 - x_2 \le 0 \\ c = 7 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1-4)^2 - x_2 \le 0 \\ x_1 - x_2 \le 0 \\ c = 9 \end{bmatrix}$$
$$0 \le x_1, x_2 \le 5, \ 0 \le c$$
$$Y_j \in \{true, false\}, \ j = 1, 2, 3$$
$$Solution : Z^* = 6.0, \ Y^* = \{true, false, false\}, \ x^* = (3.0, 1.0)$$

Example 3 (adapted from Duran and Grossmann [6])

$$\min \quad Z = c + 10x_1 - 7x_6 - 18\ln(x_2 + 1) - 19.2\ln(x_1 - x_2 + 1) + 10$$
$$\text{s.t.} \quad 0.8\ln(x_2 + 1) + 0.96\ln(x_1 - x_2 + 1) - 0.8x_6 \ge 0$$
$$x_2 - x_1 \le 0$$
$$\begin{bmatrix} Y_1 \\ x_2 - 2 \le 0 \\ x_1 - x_2 \le 0 \\ c = 5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ x_1 - x_2 - 2 \le 0 \\ x_2 \le 0 \\ c = 6 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ \ln(x_2 + 1) + 1.2\ln(x_1 - x_2 + 1) - x_6 \ge 0 \\ x_1 - x_2 \le 0, x_2 \le 0 \\ c = 8 \end{bmatrix}$$
$$0 \le x_1, x_2 \le 2, \ 0 \le x_6 \le 1, \ 0 \le c$$
$$Y_j \in \{true, false\}, \ j = 1, 2, 3$$
$$Solution : Z^* = 6.009, \ Y^* = \{false, true, false\}, \ x^* = (1.301, 0, 1.0)$$

Example 4 (adapted from Duran and Grossmann [6])

$$\min \quad Z = \sum_{k=1}^{5} c_k - 10x_3 - 15x_5 - 15x_9 + 15x_{11} + 5x_{13} - 20x_{16}$$
$$+ \exp(x_3) + \exp(x_5/1.2) - 60\ln(x_{11} + x_{13} + 1) + 140$$
$$\text{s.t.} \quad -ln(x_{11} + x_{13} + 1) \leq 0,\ -x_3 - x_5 - 2x_9 + x_{11} + 2x_{16} \leq 0,$$
$$-x_3 - x_5 - 0.75x_9 + x_{11} + 2x_{16} \leq 0,\ x_9 - x_{16} \leq 0,$$
$$2x_9 - x_{11} - 2x_{16} \leq 0,\ -0.5x_{11} + x_{13} \leq 0,\ 0.2x_{11} - x_{13} \leq 0$$
$$\begin{bmatrix} Y_1 \\ \exp(x_3) - 11 \leq 0 \\ c_1 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_3 = 0 \\ c_1 = 0 \end{bmatrix}$$
$$\begin{bmatrix} Y_2 \\ \exp(x_5/1.2) - 11 \leq 0 \\ c_2 = 8 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_5 = 0 \\ c_2 = 0 \end{bmatrix}$$
$$\begin{bmatrix} Y_3 \\ 1.25x_9 - 10 \leq 0 \\ c_3 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_9 = 0 \\ c_3 = 0 \end{bmatrix}$$
$$\begin{bmatrix} Y_4 \\ x_{11} + x_{13} - 10 \leq 0 \\ c_4 = 10 \end{bmatrix} \vee \begin{bmatrix} \neg Y_4 \\ x_{11} = x_{13} = 0 \\ c_4 = 0 \end{bmatrix}$$
$$\begin{bmatrix} Y_5 \\ -2x_9 + 2x_{16} - 10 \leq 0 \\ c_5 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_5 \\ x_9 \geq x_{16} \\ c_5 = 0 \end{bmatrix}$$
$$[Y_1 \wedge \neg Y_2] \vee [\neg Y_1 \wedge Y_2]$$
$$\neg Y_4 \vee \neg Y_5$$
$$x \in R^6,\ a \leq x \leq b, a = (0,0,0,0,0,0),\ b = (2,2,2,\infty,\infty,3)$$
$$Y_k \in \{true, false\},\ 0 \leq c_k,\ k = 1,2,...,5$$
$$Solution: Z^* = 73.04,\ Y^* = \{false, true, true, true, false\}$$
$$x^* = (0, 2, 1.078, 0.652, 0.326, 1.078)$$

Table 1: Comparison of the results for example 1

Method (Problem)	No. of NLP Subproblems (lower bound)	No. of MIP Subroblems
Standard BB (BM)	5(2.532)	–
OA (BM)	3(2.532)	3
Proposed BB (CRP)	3(3.468)	–
OA (PR)	3(3.468)	3

Table 2: Solutions of problem PR(ε) with different ε

Problem Number	GDP exact Solution	$\varepsilon =$ 10^{-5}	$\varepsilon =$ 10^{-4}	$\varepsilon =$ 10^{-3}
1*	3.5000	3.5000	3.5000	3.5000
2	6.0000	6.0001	6.0006	6.0063
3	6.0097	6.0097	6.0097	6.0097
4*	73.0353	73.0353	73.0353	73.0353
8	-8.0641	-8.0689	-8.0665	-8.0641
9	68.0097	68.0097	68.0097	68.0097

* Linear disjunctions are not affected by ε.

Table 3: Comparison of Standard and Proposed BB for GDP examples*

Prob. No.	No. of Boolean var.	No. of Cont. var.	No.of Const.	No. of Nonlin. Const.	GDP Opt. Sol.	Relaxed (BM) Opt.	CRP(ε) Opt.	Standard BB(BM) Nodes	Proposed BB((CRP(ε)) Nodes
1	3	3	12	1	3.500	2.532	3.468	5	3
2	3	3	9	2	6.000	5.456	5.600	5	3
3	3	4	8	2	6.009	0.759	2.531	5	3
4	5	11	30	3	73.04	-0.554	31.63	9	9
5	3	4	9	0	11.00	8.000	8.162	13	5
6	3	10	21	0	0	-13.96	-12.83	9	5
7	6	10	26	0	-168.9	-193.7	-177.4	11	5
8	25	30	105	25	-8.064	-19.10	-8.685	89	11
9	8	33	67	5	68.01	15.08	62.48	17	5

* $\varepsilon = 10^{-4}$ for nonlinear inequalities

Table 4: Comparison of problem PR(ε) and (BM) for GDP examples

Problem Number (MINLP)	GDP Opt. Sol.	DICOPT++ (BM) Major It.(CPU sec)	DICOPT++ PR(ε) Major It.(CPU sec)
1	3.500	3(0.641)	3(1.060)
2	6.000	5(0.990)	4(1.156)
3	6.009	7(2.735)	4(1.719)
4	73.04	13(4.481)	5(3.078)
8	-8.064	9(5.817)	3(2.863)
9	68.01	11(3.044)	2(1.094)
Problem Number (MILP)	**GDP Opt. Sol.**	**CPLEX (BM) No. of Nodes(CPU sec)**	**CPLEX PR(ε) No. of Nodes(CPU sec)**
5	11.00	4(0.219)	6(0.227)
6	0	1(0.270)	2(0.710)
7	-168.9	2(0.400)	6(0.460)

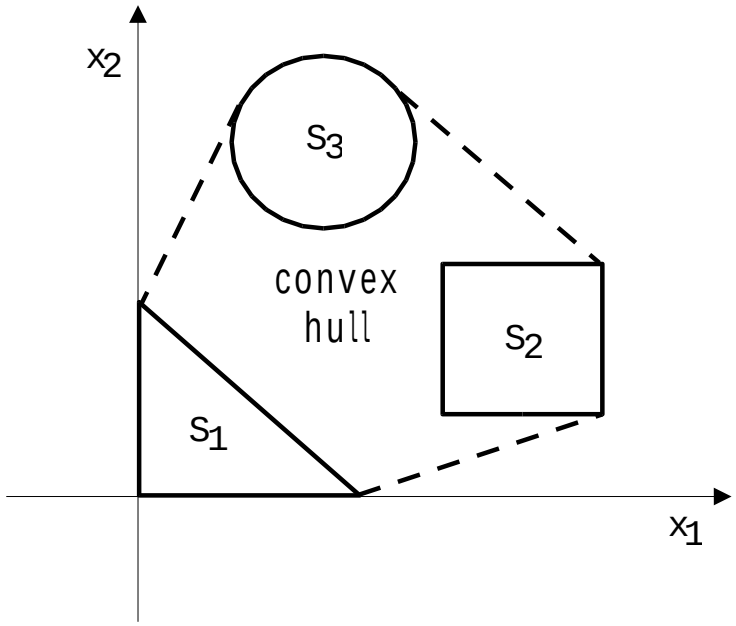
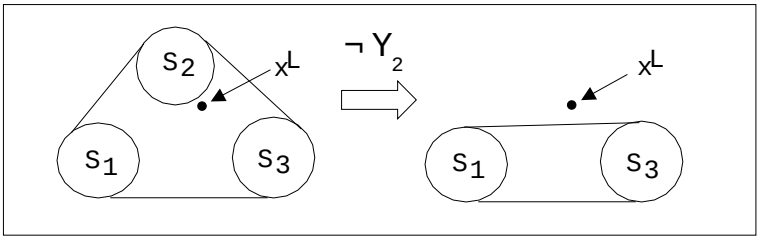
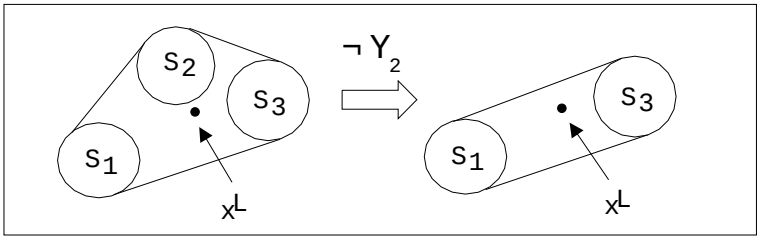


Figure 1: Example of disjunctive feasible set



a) Partitionable set



b) Non−partitionable set

Figure 2: Partitionable and Non-partitionable sets

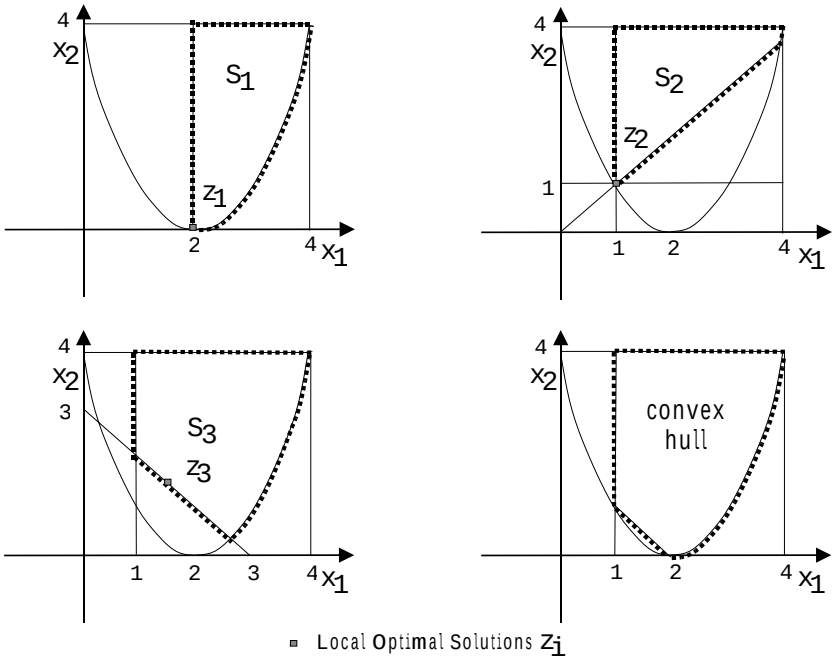


Figure 3: Feasible set of example 1

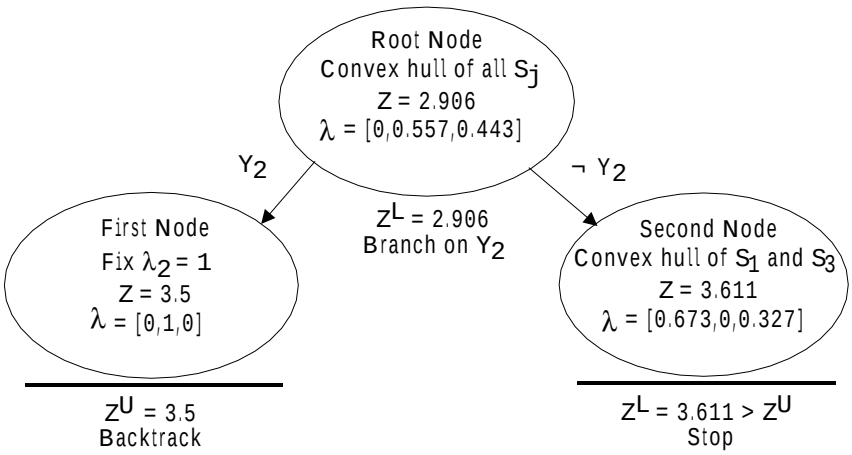


Figure 4: The proposed branch and bound tree: example 1