Improved Big-M Reformulation for Generalized Disjunctive Programs

Francisco Trespalacios and Ignacio E. Grossmann*
Department of Chemical Engineering
Carnegie Mellon University, Pittsburgh, PA 15213
* *Author to whom correspondence should be addressed: grossmann@cmu.edu*

Abstract

In this work, we present a new Big-M reformulation for Generalized Disjunctive Programs. The proposed MINLP reformulation is stronger than the traditional Big-M, and it does not require additional variables or constraints. We present the new Big-M, and analyze the strength in its continuous relaxation compared to that of the traditional Big-M. The new formulation is tested by solving several instances of process networks and muli-product batch plant problems with an NLP-based branch and bound method. The results show that, in most cases, the new reformulation requires fewer nodes and less time to find the optimal solution.

1 Introduction

Many Process Systems Engineering (PSE) applications are modelled using MINLP. Furthermore, many developments in MINLP and global optimization have been motivated by applications in PSE[1]. A comprehensive review of MINLP methods in process synthesis is provided by Grossmann et al.[2]. MINLP has been applied in the design of distillation columns and distillation sequencing[3], heat exchange networks[4], and water networks[5]. Other areas such as planning and scheduling[6, 7], process control[8] and molecular design[9, 10], have also applied MINLP methods. We refer the reader to the online library of problems www.minlp.org which contains PSE problems, as well as a recent review presents MINLP applications in PSE[11].

MILP/MINLP models can be formulated in different ways, and therefore efficiency of the algorithms to solve these problems strongly depends on the size of the corresponding formulation and tightness of its continuous relaxation.

Generalized Disjunctive Programming (GDP) is an alternative higher-level representation of problems proposed by Raman and Grossmann[12]. GDP involves not only algebraic equations, but also disjunctions and logic propositions. Although there are some special techniques to solve this type of problems, such as Disjunctive Branch and Bound[13] and Logic Based Outer Approximation[14], GDPs are normally reformulated as MILP/MINLP[15][16] to exploit the developments in these solvers. A comprehensive review of modeling through is given by a previous work[17].

The reformulation of GDP models to MILP/MINLP problems is typically performed by using either the Big-M (BM) or the Hull-Reformulation (HR), where the former generates a smaller MILP/MINLP, while the latter generates a tighter one[18][19].

In this work, we present an alternative reformulation to GDPs, which is an improved version of the Big-M. The multiple-parameter Big-M (MBM) has a continuous relaxation that is as strong, and usually stronger, than the traditional (BM). This is achieved by assigning more than one big-M term in the constraints involved in each disjunction.

This paper is organized as follows. Section 2 provides an overview of GDP and the Big-M reformulation. The section also provides a method for obtaining the strongest (BM) reformulation of a GDP. Section 3 first presents the alternative reformulation (MBM), and a method for obtaining the tightest possible (MBM) for a GDP. Second, it shows that this reformulation is strong, and normally stronger, than the (BM). Finally, it provides a simple example to illustrate the (MBM) reformulation, comparing it to the traditional (BM). The new reformulation is tested with multi-product batch plant problems, and the results are presented in Section 4

2 Background: Generalized Disjunctive Programming (GDP)

Generalized Disjunctive Programming[20][12] allows the systematic modeling of optimization problems by using algebraic equations, disjunctions and logic propositions. GDP can be considered an extension to the well-known theory of disjunctive programming developed by Balas[21], and has been applied to a number of process systems engineering areas [7, 11, 14].

2.1 GDP formulation

The general GDP formulation can be represented as follows:

$$
\begin{aligned}
& min\ z = f(x) & & \\
s.t.\quad & g(x) \le 0 & & \\
& \bigvee_{i \in D_k} \begin{bmatrix} Y_{ki} \\ r_{ki}(x) \le 0 \end{bmatrix} & & k \in K \\
& \underline{\vee}_{i \in D_k}\ Y_{ki} & & k \in K \\
& \Omega(Y) = True & & \\
& x^{lo} \le x \le x^{up} & & \\
& x \in \mathbb{R}^n & & \\
& Y_{ki} \in \{True, False\} \quad & & k \in K, i \in D_k
\end{aligned}
\tag{GDP}
$$

The objective in (GDP) is function of the continuous variables x. The global constraints $g(x)$ must hold true regardless of the discrete decisions. Each of the disjunctions $k \in K$ contains disjunctive terms $i \in D_k$, linked together by an OR operator ($\vee$). For each disjunctive term in each disjunction, a Boolean variable Y_{ki} is assigned with a corresponding set of inequalities $r_{ki}(x) \le 0$. Only one term in each disjunction must be enforced from the condition $\underline{\vee}_{i \in D_k} Y_{ki}$. When enforced, its corresponding Boolean variable takes a value of $True$. Therefore, when a disjunctive term is active ($Y_{ki} = True$), its corresponding constraints are also enforced. When it is not active ($Y_{ki} = False$), the constraints are ignored. $\Omega(Y) = True$ represents the logic relations between the Boolean variables.

2.2 Big-M reformulation of GDP

In order to take full advantage of existing solvers[22], GDP problems are normally reformulated as MILP/MINLP by using either the Big-M[16] (BM) or Hull Reformulation[15] (HR). (BM) generates a smaller MILP/MINLP, while (HR) generates a tighter one[18][19]. The interest of this work is a modified version of the (BM). We refer the reader to the MINLP and GDP review work[11] for details on the (HR).

The (BM) reformulation is as follows:

$$
\begin{array}{lll}
 & min\ z = f(x) & \\
s.t. & g(x) \leq 0 & \\
 & r_{ki}(x) \leq M^{ki}(1 - y_{ki}) & k \in K, i \in D_k \\
 & \sum\limits_{i \in D_k} y_{ki} = 1 & k \in K \\
 & Hy \geq h & \\
 & x^{lo} \leq x \leq x^{up} & \\
 & x \in \mathbb{R}^n & \\
 & y_{ki} \in \{0,1\} & k \in K, i \in D_k
\end{array}
\tag{BM}
$$

In (BM) the Boolean variables Y_{ki} are transformed into binary variables y_{ki} with a one-to-one correspondence (i.e. $Y_{ki} = True$ is equivalent to $y_{ki} = 1$, while $Y_{ki} = False$ is equivalent to $y_{ki} = 0$). The transformation of logic relations ($\Omega(Y) = True$) to integer linear constraints ($Hy \geq h$) can be easily obtained[23][24][25]. The equation $\sum\limits_{i \in D_k} y_{ki} = 1$ guarantees that only one disjunctive term is selected per disjunction. For a selected term ($y_{ki} = 1$) the corresponding constraints $r_{ki}(x) \leq 0$ are enforced. For a term not selected ($y_{ki} = 0$) and a large enough M^{ki}, the corresponding constraint $r_{ki}(x) \leq M^{ki}$ becomes redundant. Note that the smaller the M-parameters are (M^{ki}), the tighter the (BM) reformulation is (the right hand side in the constraints is smaller). For this reason, the ideal M-parameter of a constraint is the smallest number that makes such a constraint redundant when required (i.e. when a disjunctive term that does not correspond to such constraint is selected). In many problems, the M-parameter can be obtained from knowledge of the meaning of the constraints. In general, the optimal value of the M-parameter can be calculated as described in the next section.

2.3 Finding the optimal M-parameter

Consider E_{ki} to be the set of equations corresponding to a disjunctive term ki. In order to find the optimal M-parameter for a constraint in a disjunctive term ($r_{kie}(x), k \in K, i \in D_k, e \in E_{ki}$), with respect to another term of the same disjunction $i', i' \in D_k, i' \neq i$, it is possible to solve the following GDP:

$$
\begin{array}{lll}
 & max\ M^{kiei'} & \\
s.t. & r_{kie}(x) = M^{kiei'} & \\
 & r_{ki'}(x) \leq 0 & \\
 & g(x) \leq 0 & \\
 & \bigvee\limits_{i \in D_{\hat{k}}} \left[\begin{array}{c} Y_{\hat{k}i} \\ r_{\hat{k}i}(x) \leq 0 \end{array} \right] & \hat{k} \in K, \hat{k} \neq k \\
 & \underline{\vee}_{i \in D_{\hat{k}}} Y_{\hat{k}i} & \hat{k} \in K, \hat{k} \neq k \\
 & \Omega(Y) = True & \\
 & x^{lo} \leq x \leq x^{up} & \\
 & x \in \mathbb{R}^n & \\
 & Y_{\hat{k}i} \in \{True, False\} & \hat{k} \in K, \hat{k} \neq k, i \in D_{\hat{k}}
\end{array}
\tag{1}
$$

Problem (1) seeks to maximize the value of a constraint ($r_{kie}(x)$), over the feasible region of the complete problem. It considers that a disjunctive term from the same disjunction different from i ($i' \in D_k, i' \neq i$) was selected (i.e. $Y_{ki'} = True$, so $r_{ki'}(x) \leq 0$ is enforced).

(1) has to be solved for a given constraint ($r_{kie}(x), k \in K, i \in D_k, e \in E_{ki}$), for all the terms in it's corresponding disjunction ($\forall i^{'} \in D_k, i^{'} \neq i$). Then, the optimal M-parameter is $M^{kie} = \max\limits_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} \{M^{kiei^{'}}\}$.

Note that (1) has almost the same difficulty as solving the original problem, and more so in some cases. Furthermore, it has to be solved several times for every single constraint inside a disjunction. For this reason, a more practical approach is to find an M-parameter that is optimal for the feasible region of its corresponding disjunction (instead of the feasible region of the complete problem). This can be achieved by solving the following NLP (LP for linear constraints):

$$\begin{aligned} & max\ M^{kiei^{'}} \\ s.t. \quad & r_{kie}(x) = M^{kiei^{'}} \\ & r_{ki'}(x) \leq 0 \\ & x^{lo} \leq x \leq x^{up} \end{aligned} \tag{2}$$

Problem (2) will provide a $M^{kiei^{'}}$ that is optimal for a particular disjunction, although it is weaker than $M^{kiei^{'}}$ obtained through (1). In a similar manner as with (1), the optimal M^{kie} for a constraint ($r_{kie}(x), k \in K, i \in D_k, e \in E_{ki}$) is $M^{kie} = \max\limits_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} \{M^{kiei^{'}}\}$. There are a few things to note about NLP (2):

1. If the problem is linear, then the optimal M-parameters are obtained through the solution of LPs.

2. If the original problem has convex constraints, and $r_{kie}(x)$ is nonlinear, then (2) is non-convex. However, there is no need for a global optimal solution. Problem (2) will normally be a small problem with few constraints, so it is expected to provide a good upper bound (valid M-parameter) fast.

3. It is important to include the bounds of the variables to avoid an unbounded problem. However, (2) is not restricted to only include these constrains. It is possible to include additional constraints that correspond to the continuous relaxation of (1). The addition of these constraints to (2) will provide better M-parameters, but the NLP will become larger. Considering that in general the NLP is non-convex, finding a good trade-off between good M-parameter and the speed to obtain it is an important consideration.

4. Some conclusions might be drawn from (2). At least two of them provide valuable information. First, if for a given constraint $r_{kie}(x)$, $M^{kiei^{i'}} \leq 0$ for all $i' \neq i$, then that constraint can be removed from the disjunction and be regarded as a global constraints. Second, if (2) is infeasible, then the term ki' is infeasible and can be removed from the original problem.

3 New Big-M reformulation

In disjunctions that contain more than two disjunctive terms, it is possible to have stronger (BM) formulations by assigning more than one big-M term in the constraints (i.e. one big-M term for each other disjunctive term in the disjunction). This formulation is as follows:

$$
\begin{aligned}
& min\ z = f(x) \\
s.t. \quad & g(x) \leq 0 \\
& r_{ki}(x) \leq \sum_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} M^{kii^{'}} y_{ki'} \quad && k \in K, i \in D_k \\
& \sum_{i \in D_k} y_{ki} = 1 && k \in K \\
& Hy \geq h \\
& x^{lo} \leq x \leq x^{up} \\
& x \in \mathbb{R}^n \\
& y_{ki} \in \{0,1\} && k \in K, i \in D_k
\end{aligned}
\tag{MBM}
$$

The idea behind (MBM) is similar to that of (BM). When a disjunctive term is selected ($y_{ki} = 1,\ i \in D_k$), the other terms in the corresponding disjunction are not ($y_{ki'} = 0,\ \forall i^{'} \in D_k, i^{'} \neq i$). Therefore, $\sum_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} M^{kii^{'}} y_{ki'} = 0$, and the corresponding constraints $r_{ki}(x) \leq 0$ are enforced. If it is not selected ($y_{ki} = 0,\ k \in D_k$), then another term must be selected ($y_{ki'} = 1,\ i^{'} \in D_k, i^{'} \neq i$). If $M^{kii^{'}}$ is large enough, then $r_{ki}(x) \leq M^{kii^{'}}$ becomes redundant.

The optimal value of $M^{kii^{'}}$ can be directly obtained as with (BM). $M^{kii^{'}}$ is then the vector of M-parameters directly obtained from (1) or (2), with the advantages of using (1) or (2) aforementioned: $M^{kii^{'}} = [..., M^{kiei^{'}}, ...]^T$, where $e \in E_{ki}$ are equations corresponding to a disjunctive term ki.

We note that the general concept of this reformulation was presented before[26]. More recently, a similar reformulation was presented for linear problems in the context of MILP formulation techniques[27].

3.1 Tightness of continuous relaxation of (MBM)

In this section we show that (MBM) is at least as tight as (BM), and can be tighter.

Theorem 3.1 *Let M^{ki} and $M^{kii^{'}}$ be the optimal values of the M-parameters for (BM) and (MBM) respectively, obtained through (2). Using such parameters in the reformulation, let (F-BM) and (F-MBM) be the feasible region of the continuous relaxation (BM) and (MBM), respectively. Then (F-MBM) $\subseteq$ (F-BM).*

Proof. The only difference between (BM) and (MBM) are the two following constraints:

$$
r_{ki}(x) \leq M^{ki}(1 - y_{ki}) \quad k \in K, i \in D_k \tag{3}
$$

$$
r_{ki}(x) \leq \sum_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} M^{kii^{'}} y_{ki'} \quad k \in K, i \in D_k \tag{4}
$$

In (3), $(1 - y_{ki})$ can be substituted by $\sum_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} y_{ki'}$, since $\sum_{i \in D_k} y_{ki} = 1$. We then obtain for (BM) the following equation:

$$
r_{ki}(x) \leq M^{ki} \sum_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} y_{ki'} \quad k \in K, i \in D_k \tag{5}
$$

or:

$$r_{ki}(x) \leq \sum_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} M^{ki} y_{ki'} \quad k \in K, i \in D_k \tag{6}$$

(6) represents the same feasible region than (3) in (BM). By definition $M^{ki} = \max_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} \{M^{kii^{'}}\}$. Therefore, the right hand side of (4) dominates the right hand side of (6) ■.

Remark 3.2 *(MBM) = (BM) when, for all $k \in K, i \in D_k$:*

$$M^{kii^{'}} = M^{kij} \; \forall j \in D_k, j \neq i, j \neq i, j \neq i^{'}$$

From 3.1 it is easy to se that (MBM) and (BM) are the same only when the $M^{kii^{'}}$ is exactly the same for all $i^{'}$ as stated in Remark 3.2. In such a case $M^{ki} = \max_{\substack{i^{'} \in D_k \\ i^{'} \neq i}} \{M^{kii^{'}}\} = M^{kii^{'}}$. Note that when a disjunction has only two disjunctive terms, there is only one M-parameter, and (3) = (4) for that disjunction. If all disjunctions have only two terms, then (MBM) = (BM).

Note that if M^{ki} and $M^{kii^{'}}$ are obtained through (1) instead of through (2), Theorem 3.1 still holds true.

3.2 Illustration of (MBM) reformulation

The improved Big-M reformulation is illustrated with the following example:

$$\begin{aligned}
& min\ z = -2x_1 + x_2 \\
s.t. \quad & \begin{bmatrix} Y_1 \\ (x_1)^2 + (x_2)^2 \leq 1 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1 - 1)^2 + (x_2 - 5)^2 \leq 2 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1 - 4)^2 + (x_2 - 3)^2 \leq 4 \end{bmatrix} \\
& Y_1 \veebar Y_2 \veebar Y_3 \\
& -1 \leq x_1 \leq 6 \\
& -1 \leq x_2 \leq 7 \\
& Y_1, Y_2, Y_3 \in \{True, False\}
\end{aligned} \tag{7}$$

The traditional Big-M reformulation of (7), using the optimal M-parameter obtained from (2), is the following:

$$\begin{aligned}
& min\ z = -2x_1 + x_2 \\
s.t. \quad & (x_1)^2 + (x_2)^2 \leq 1 + 48(1 - y_1) \\
& (x_1 - 1)^2 + (x_2 - 5)^2 \leq 2 + 35.1981(1 - y_2) \\
& (x_1 - 4)^2 + (x_2 - 3)^2 \leq 4 + 32(1 - y_3) \\
& y_1 + y_2 + y_3 = 1 \\
& -1 \leq x_1 \leq 6 \\
& -1 \leq x_2 \leq 7 \\
& y_1, y_2, y_3 \in \{0, 1\}
\end{aligned} \tag{8}$$

The (MBM) reformulation, using the optimal M-parameter obtained from (2), is the following:

$$
\begin{aligned}
& min\ z = -2x_1 + x_2 \\
s.t. \quad & (x_1)^2 + (x_2)^2 \le 1 + 41.42221 y_2 + 48 y_3 \\
& (x_1 - 1)^2 + (x_2 - 5)^2 \le 2 + 35.1981 y_1 + 29.4223 y_3 \\
& (x_1 - 4)^2 + (x_2 - 3)^2 \le 4 + 32 y_1 + 21.1981 y_2 \\
& y_1 + y_2 + y_3 = 1 \\
& -1 \le x_1 \le 6 \\
& -1 \le x_2 \le 7 \\
& y_1, y_2, y_3 \in \{0, 1\}
\end{aligned} \tag{9}
$$

In the traditional Big-M reformulation (8), the “optimal” M-parameter of the first constraint is $M_1 = max\{41.42221, 48\} = 48$. The RHS of the first constraint in (8) is $1 + 48(1 - y_1)$. Therefore, the first constraint in (9) is tighter than its corresponding constraint in (8). This can be easily seen if $(1 - y_1)$ is substituted by $y_2 + y_3$ in (9). This would yield $1 + 48y_2 + 48y_3$ as RHS of the first constraint of (9). Clearly, the RHS of the constraint in (9) dominates the RHS of the one in (9). The same holds true for the other two Big-M constraints. In this example, the optimal solution of this problem is $\mathbf{z^* = -9.472}$. The relaxation of (8) is $\mathbf{z^{BM} = -10.493}$, and the relaxation of (9) is $\mathbf{z^{MBM} = -9.735}$. It is clear that (9) provides a tighter relaxation.

4 Examples and results

The new reformulation was tested with several instances of two problems: the process network problem and the design of multi-product batch plant problem.

4.1 Process Network (Process)

The process network problem ”Process” is a classic optimization problem in process design. The model seeks to maximize the profit of selling a set of products taking into account the cost of raw materials and equipment. The model that describes the performance of each unit is normally large and quite complex. In this example, however, the process is simplified to single input-output relations that give rise to a convex GDP[28]. We refer the reader to the original source for the details of the formulation. The presented problem is a modified version of the original one, for which each disjunction represents the selection of alternative units[29]. The GDP problem formulation is as follows:

$$
\begin{aligned}
& min\ Z = \sum_{k \in I} c_k + \sum_{j \in J} p_j x_j + \alpha \\
s.t. \quad & \sum_{j \in J} r_{jn} x_j \le 0 & \forall n \in N \\
\bigvee_{i \in D_k} & \begin{bmatrix} Y_{ki} \\ \sum\limits_{j \in J^{ki}} d_{kij} (e^{x_j / t_{kij}} - 1) - \sum\limits_{j \in J^i} s_{kij} x_j \le 0 \\ c_k = \gamma_{ki} \end{bmatrix} & i \in I \\
& \Omega(Y) = True \\
& c_k, x_j \ge 0 \\
& Y_{ki} \in \{True, False\}
\end{aligned} \tag{10}
$$

4.2 Design of multi-product batch plant problem formulation

This problem was presented by Ravemark[30] and a convexified version was provided by Vecchietti and Grossmann[31]. The problem seeks to minimize the investment cost in the design of a plant with multiple units in parallel and intermediate storage tanks . The design involves selecting the number of parallel units, volume of the equipment, and volume and location of the intermediate storage tanks. For a detailed explanation of the problem, we refer the reader to the original sources and a recent review[17]. The problem is as follows:

$$
\begin{array}{lll}
 & min\ Z = \alpha_1 \sum_j \exp(n_j + m_j + \beta_1 v_j) + \alpha_2 \sum_{Tj} \exp(\beta_2 v_{Tj}) & \\
s.t. & v_j \geq \ln(S_{ij}) + b_{ij} - n_j & \forall i, j \\
 & e_i \geq \ln(T_{ij}) - b_{ij} - m_j & \forall i, j \\
 & H \geq \sum_i (Q_i exp(e_i)) & \\
 & \begin{bmatrix} YS_j & \\ v_{Tj} \geq \ln(S_j^*) + b_{ij+1} & \forall i \\ v_{Tj} \geq \ln(S_j^*) + b_{ij} & \forall i \\ b_{ij} - b_{ij+1} \leq \ln(S_{ij}^*) & \forall i \\ b_{ij} - b_{ij+1} \geq -\ln(S_{ij}^*) & \forall i \end{bmatrix} \vee \begin{bmatrix} \neg YS_j \\ v_{Tj} = 0 \\ b_{ij} - b_{ij+1} = 0 \quad \forall i \end{bmatrix} & \forall j < |J| \\
 & \begin{bmatrix} YM_{j,1} \\ m_j = \ln(1) \end{bmatrix} \vee ... \vee \begin{bmatrix} YM_{j,maxp} \\ m_j = \ln(maxp) \end{bmatrix} & \forall j \\
 & \begin{bmatrix} YN_{j,1} \\ n_j = \ln(1) \end{bmatrix} \vee ... \vee \begin{bmatrix} YN_{j,maxp} \\ n_j = \ln(maxp) \end{bmatrix} & \forall j \\
 & YM_{j,1} \veebar ... \veebar YM_{j,maxp} & \forall j \\
 & YN_{j,1} \veebar ... \veebar YN_{j,maxp} & \forall j \\
 & YS_j, YM_{j,p}, YN_{j,p} \in \{True, False\} & \forall j, p = 1, ..., maxp
\end{array}
\tag{11}
$$

4.2.1 Optimal M-parameters in the multiproduct batch reactor problem

Problem (11) contains three sets of disjunctions. The first set has only two terms, so the optimal M-parameters are the same for the (BM) and the (MBM). The other two sets of disjunctions have to be written as inequalities to perform the reformulation:

$$
\begin{bmatrix} YM_{j,1} \\ m_j \leq \ln(1) \\ m_j \geq \ln(1) \end{bmatrix} \vee ... \vee \begin{bmatrix} YM_{j,maxp} \\ m_j \leq \ln(maxp) \\ m_j \geq \ln(maxp) \end{bmatrix} \quad \forall j \tag{12}
$$

$$
\begin{bmatrix} YN_{j,1} \\ n_j \leq \ln(1) \\ n_j \geq \ln(1) \end{bmatrix} \vee ... \vee \begin{bmatrix} YN_{j,maxp} \\ n_j \leq \ln(maxp) \\ n_j \geq \ln(maxp) \end{bmatrix} \quad \forall j \tag{13}
$$

In disjunction (12) and (13) the optimal M-parameters of the (BM) and (MBM) are different.

For $p = 1, ..., maxp; \quad q = 1, ..., maxp; \quad q \neq p$, the optimal M-parameters of (MBM) are:

Table 1: Solution of multi-product batch plant instances

Instance	Solutions	Continuous relaxation		Time (s)		Nodes	
		(BM)	(MBM)	(BM)	(MBM)	(BM)	(MBM)
Proc-1-21	17.2	1.7	**10.0**	8.4	**4.0**	906	**404**
Proc-1-31	12.2	5.9	**7.9**	6.5	**1.2**	639	**86**
Proc-1-36	12.1	5.6	**7.8**	14.2	**1.5**	1,408	**112**
Proc-1-48	12.1	5.5	**7.4**	63.7	**5.7**	6,102	**405**
BatchS101006	769,440	734,943	734,943	237	**40**	10,894	**1,595**
BatchS121208	1,241,125	1,202,365	1,202,365	657	**365**	23,890	**10,587**
BatchS141208	1,487,664	1,440,995	1,440,995	1,148	**1,018**	38,643	**29,083**
BatchS151208	1,543,472	1,499,913	1,499,913	**121**	1,872	**3,729**	47,958
BatchS181210	2,042,327	2,006,860	2,006,860	145	145	3,719	**3,088**
BatchS201210	2,295,349	2,255,304	2,255,304	525	**502**	13,774	**10,158**

$M^{jpq1} = \ln(q) - \ln(p)$

$M^{jpq2} = -\ln(q) + \ln(p)$

The optimal M-parameters of the (BM) are:

$M^{jp1} = \underset{\substack{q=1,...,maxp \\ q \neq p}}{max} \{\ln(q) - \ln(p)\} = \ln(maxp) - \ln(p)$

$M^{jp1} = \underset{\substack{q=1,...,maxp \\ q \neq p}}{max} \{-\ln(q) + \ln(p)\} = -\ln(1) + \ln(p)$

The (BM) of constraints (12) are:

$$
\begin{aligned}
&m_j \leq \ln(p) + (\ln(maxp) - \ln(p))(1 - ym_{j,p}) \quad && p = 1, ..., maxp \\
&m_j \geq \ln(p) - (\ln(p) - \ln(1))(1 - ym_{j,p}) && p = 1, ..., maxp
\end{aligned}
\tag{14}
$$

The (MBM) of constraints (12) are:

$$
\begin{aligned}
&m_j \leq \ln(p) + \sum_{\substack{p=1,...,maxp \\ p \neq q}} (\ln(q) - \ln(p)) ym_{j,q} = 1 \quad && p = 1, ..., maxp \\
&m_j \geq \ln(p) - \sum_{\substack{p=1,...,maxp \\ p \neq q}} (-\ln(q) + \ln(p)) ym_{j,q} = 1 && p = 1, ..., maxp
\end{aligned}
\tag{15}
$$

4.3 Results

The new reformulation was tested with 10 instances. 4 of these benchmark examples for the multi-product batch plant problem[32]. Two additional multi-product batch problems were generated using data of the benchmark problems. The 4 instances for the process network problems were used as test cases in recent work[29]. The instances were solved with SBB (NLP-based branch and bound) from GAMS 24.3.3[33], using in an Intel(R) Core(TM) i7 CPU 2.93 GHz and 4 GB of RAM. The M-parameters for the multi-product batch problems were presented in 4.2.1. The M-parameters for the process network problems were obtained by solving (2).

Table 1 shows the relaxation, number of nodes and time required to solve each of the instances. It is clear that for the process problems (MBM) provides a much better continuous relaxation. The continuous relaxation of both the (BM) and (MBM) is the same for the batch problems (although (MBM) is tighter than (BM) as can bee seen from the constraints described in Section 4.2.1). Fewer number of nodes are required for the (MBM) in 9 of the 10 instances, which generally yields reductions in the solution times. There is only one instance in which the (BM) performs better than the (MBM).

5 Conclusions

In this work we presented an enhanced Big-M reformulation for GDP problems. The proposed reformulation is at least as tight as the traditional Big-M, and it does not require any additional variables or constraints. We reviewed the method for obtaining optimal M-parameters in the traditional Big-M, and adapted it for the new proposed reformulation. The reformulation was tested with 10 instances, and the new Big-M formulation performed better than the traditional Big-M in 9 out of 10 instances.

References

[1] Floudas, C. A.; Gounaris, C. E. A review of recent advances in global optimization. *Journal of Global Optimization* **2009**, *45*, 3–38.

[2] Grossmann, I. E.; Caballero, J. A.; Yeomans, H. Mathematical programming approaches to the synthesis of chemical process systems. *Korean Journal of Chemical Engineering* **1999**, *16*, 407–426.

[3] Caballero, J. A.; Grossmann, I. E. *Chapter 11 of Distillation: Fundamentals and Principles*; Academic Press, 2014.

[4] Furman, K. C.; Sahinidis, N. V. A critical review and annotated bibliography for heat exchanger network synthesis in the 20th century. *Industrial and Engineering Chemistry Research* **2002**, *41*, 2335–2370.

[5] Jezowski, J. Review of water network design methods with literature annotations. *Industrial and Engineering Chemistry Research* **2010**, *49*, 4475–4516.

[6] Harjunkoski, I.; Maravelias, C. T.; Bongers, P.; Castro, P. M.; Engell, S.; Grossmann, I. E.; Hooker, J.; Méndez, C.; Sand, G.; Wassick, J. Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering* **2014**, *62*, 161–193.

[7] Castro, P. M.; Grossmann, I. E. Generalized disjunctive programming as a systematic modeling framework to derive scheduling formulations. *Industrial & Engineering Chemistry Research* **2012**, *51*, 5781–5792.

[8] Bemporad, A.; Morari, M. Control of systems integrating logic, dynamics, and constraints. *Automatica* **1999**, *35*, 407–427.

[9] Achenie, L.; Venkatasubramanian, V.; Gani, R. *Computer aided molecular design: theory and practice*; Elsevier, 2002; Vol. 12.

[10] Sahinidis, N. V.; Tawarmalani, M.; Yu, M. Design of alternative refrigerants via global optimization. *AIChE Journal* **2003**, *49*, 1761–1775.

[11] Trespalacios, F.; Grossmann, I. E. Review of Mixed-Integer Nonlinear and Generalized Disjunctive Programming Methods. *Chemie Ingenieur Technik* **2014**, *86*, 991–1012.

[12] Raman, R.; Grossmann, I. E. Modeling and computational techniques for logic-based integer programming. *Computers and Chemical Engineering* **1994**, *18*, 563–578.

[13] Lee, S.; Grossmann, I. E. Logic-based modeling and solution of nonlinear discrete/continuous optimization problems. *Annals of Operations Research* **2005**, *139*, 267–288.

[14] Turkay, M.; Grossmann, I. E. A logic-based outer-approximation algorithm for minlp optimization of process flowsheets. *Computers and Chemical Engineering* **1996**, *20*, 959–978.

[15] Lee, S.; Grossmann, I. E. New algorithms for nonlinear generalized disjunctive programming. *Computers and Chemical Engineering* **2000**, *24*, 2125–2141.

[16] Nemhauser, G. L.; Wolsey, L. A. *Integer and Combinatorial Optimization, Wiley-Interscience*; Wiley, 1988.

[17] Grossmann, I. E.; Trespalacios, F. Systematic modeling of discrete-continuous optimization models through generalized disjunctive programming. *AIChE Journal* **2013**, *59*, 3276–3295.

[18] Grossmann, I. E.; Lee, S. Generalized convex disjunctive programming: nonlinear convex hull relaxation. *Computational Optimization and Applications* **2003**, *26*, 83–100.

[19] Vecchietti, A.; Lee, S.; Grossmann, I. E. Modeling of discrete/continuous optimization problems: characterization and formulation of disjunctions and their relaxations. *Computers and Chemical Engineering* **2003**, *27*, 433–448.

[20] Grossmann, I. E.; Ruiz, J. P. Generalized Disjunctive Programming: A Framework for Formulation and Alternative Algorithms for MINLP Optimization. *The IMA Volumes in Mathematics and its Applications* **2012**, *154*, 93–115.

[21] Balas, E. Disjunctive programming. *Annals of Discrete Mathematics* **1979**, *5*, 3–51.

[22] Grossmann, I. E. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optimization and Engineering* **2002**, *3*, 227–252.

[23] Clocksin, W. F.; Mellish, C. S. *Programming in Prolog*; Springer, 1981.

[24] Williams, H. P. *Model Building in Mathematical Programming*; Wiley, 1985.

[25] Biegler, L. T.; Grossmann, I. E.; Westerberg, A. W. *Systematic methods of chemical process design*; Prentice-Hall international series in the physical and chemical engineering sciences; Prentice Hall PTR, 1997.

[26] Trespalacios, F. PhD Thesis Proposal: Algorithmic approach for improved mixed-integer reformulations of Generalized Disjunctive Programs. Ph.D. Thesis, Carnegie Mellon University, 2013.

[27] Vielma, J. P. Mixed integer linear programming formulation techniques.

[28] Ruiz, J. P.; Grossmann, I. E. A hierarchy of relaxations for nonlinear convex generalized disjunctive programming. *European Journal of Operational Research* **2012**, *218*, 38–47.

[29] Trespalacios, F.; Grossmann, I. E. Cutting plane algorithm for convex Generalized Disjunctive Programs. *Informs JOC. Submitted for publication.*

[30] Ravemark, E. Optimization models for design and operation of chemical batch processes. Ph.D. Thesis, ETH Zurich, 1995.

[31] Vecchietti, A.; Grossmann, I. E. LOGMIP: a disjunctive 0-1 non-linear optimizer for process system models. *Computers and Chemical Engineering* **1994**, *23*, 555–565.

[32] CMU; IBM CMU-IBM Open Source MINLP Project. http://egon.cheme.cmu.edu/ibm/page.htm.

[33] Brooke, A.; Kendrick, D.; Meeraus, A.; Raman GAMS, a Users Guide. *The Scientific Press* **1998**.