Systematic Modeling of Discrete-Continuous Optimization Models through Generalized Disjunctive Programming

Ignacio E. Grossmann and Francisco Trespalacios
Center for Advanced Process Decision-making, Dept. of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213

DOI 10.1002/aic.14088
Published online May 31, 2013 in Wiley Online Library (wileyonlinelibrary.com)

Discrete-continuous optimization problems are commonly modeled in algebraic form as mixed-integer linear or nonlinear programming models. Since these models can be formulated in different ways, leading either to solvable or nonsolvable problems, there is a need for a systematic modeling framework that provides a fundamental understanding on the nature of these models. This work presents a modeling framework, generalized disjunctive programming (GDP), which represents problems in terms of Boolean and continuous variables, allowing the representation of constraints as algebraic equations, disjunctions and logic propositions. An overview is provided of major research results that have emerged in this area. Basic concepts are emphasized as well as the major classes of formulations that can be derived. These are illustrated with a number of examples in the area of process systems engineering. As will be shown, GDP provides a structured way for systematically deriving mixed-integer optimization models that exhibit strong continuous relaxations, which often translates into shorter computational times. © 2013 American Institute of Chemical Engineers *AIChE J*, 59: 3276-3295, 2013

Keywords: optimization, mixed-integer programming, logic-based optimization

Introduction

Mixed-integer programming provides a powerful framework for the mathematical modeling of many optimization problems that involve discrete and continuous variables. Over the last few years there has been a pronounced increase in the development of mixed-integer linear/nonlinear programming (MILP/MINLP) models in process systems engineering.[1–5] Synthesis models can be formulated as MILP for high-level targeting, and as MINLP for more detailed superstructure optimization.[6] For planning and scheduling, the majority tends to be MILP models[7–10] although gradually there is also an increasing trend to MINLP, especially as process models are incorporated.[11] MILP/MINLP models, however, are based on algebraic formulations which are not unique. Although there has been significant progress in software for solving mixed-integer problems, especially MILP, how one formulates a model can have a major impact in the performance and ability to find a solution. Therefore, deriving "good" formulations or finding potential improvements in existing models is commonly regarded as an art and strongly depends on the modeler skills.

Generalized disjunctive programming (GDP) originated with the goal of facilitating the modeling of discrete/continuous optimization problems through the use of higher level logic constructs.[12,13] This approach involves algebraic equations, disjunctions and logic propositions in the formulation of a model. Unlike direct MINLP formulations, this higher-level modeling framework makes the formulation process more intuitive and systematic, while retaining in the model the underlying logic structure of the problem. Although there are some special techniques to solve this type of problems, such as disjunctive branch and bound[14] and logic-based outer approximation,[15] GDPs are normally directly reformulated as MILP/MINLP[16,17] to exploit the developments in these solvers (a review on MINLP methods is provided by Grossmann[18]).

The focus of this article is on the systematic formulation of MILP/MINLP models using GDP as a theoretical framework. In the section Generalized Disjunctive Programming, we introduce the concept of GDP and provide some chemical engineering examples. We next examine the process for transforming GDP to MILP/MINLP models, describing alternative reformulations and application of logic tools to improve the formulations. Finally, we illustrate the application of these concepts on several problems.

Generalized Disjunctive Programming

An alternative approach for representing discrete/continuous optimization problems is by modeling them using algebraic equations, disjunctions and logic propositions.[12,13,15,16,19,20] Such a model is known as generalized disjunctive programming,[13,21] the main focus of this article, which can be regarded as a generalization of disjunctive programming developed by Balas.[22] Process design[2,15,23] and planning and

This article is dedicated to the memory of Professor Neil Amundson, who pioneered the application of mathematical modeling and analysis in chemical engineering.

Correspondence concerning this article should be addressed to I. E. Grossmann at grossmann@cmu.edu.

scheduling[7,11] are some of the areas where GDP formulations have shown to be successful.

Motivation

In order to illustrate that the way we formulate MILP/MINLP models can have a major impact in time and ability to find a solution, consider the following simple example. A company has to decide whether to produce either product A or product B, in order to maximize its profit. The profit of product A is 3, and the profit of product B is 2. The limit on production of A is 4, and the limit in production of B is 5. There are at least three different formulations for this problem for this problem P1a, P1b and P1c that are given below

(P1a):

$$\begin{aligned} &\max z=3A+2B \\ &s.t. \quad A*y_2=0 \\ &B*y_1=0 \\ &y_1*y_2=0 \\ &0\le A\le 4 \\ &0\le B\le 5 \\ &y_1+y_2=1 \\ &0\le y_1,y_2\le 1 \\ &A,B,y_1,y_2\in\mathbb{R} \end{aligned}$$

(P1b):

$$\begin{aligned} &\max z=3A+2B \\ &s.t. \quad A\le 10*(1-y_2) \\ &B\le 10*(1-y_1) \\ &0\le A\le 4 \\ &0\le B\le 5 \\ &y_1+y_2=1 \\ &A,B\in\mathbb{R} \\ &y_1,y_2\in\{0,1\} \end{aligned}$$

(P1c):

$$\begin{aligned} &\max z=3A+2B \\ &s.t. \quad 0\le A\le 4*y_1 \\ &0\le B\le 5*y_2 \\ &y_1+y_2=1 \\ &A,B\in\mathbb{R} \\ &y_1,y_2\in\{0,1\} \end{aligned}$$

The basic idea in P1a, P1b and P1c is to formulate the problem as an optimization problem in which the variables y_1, y_2 are needed to determine whether A or B are produced, respectively. The three formulations are valid representations of the problem. While P1a corresponds to a nonlinear program (NLP) due to the nonlinear functions involved, both P1b and P1c correspond to two alternative linear MILP models. Note that one can only set either y_1 or y_2 equal to 1. When $y_1=1$ and $y_2=0$ then B $=0$ and $0\le A\le 4$; when $y_1=0$ and $y_2=1$ then $A=0$ and $0\le B\le 5$. It is important to note that in P1b this holds true because 10 is a large enough number, so when $y_1=1$ or $y_2=1$, the constraints $A\le 10*(1-y_2)$ and $B\le 10*(1-y_1)$ become redundant, respectively. Since all three formulations are valid MILP/NLP, they have the same optimal solution $z_{NLP}=z_{MILP}=12$. However, the solution efficiency is strongly linked to the type of model. P1a is nonlinear and nonconvex, which requires a global optimization algorithm and is, therefore, expected to be considerably slower than the other two small MILP formulations. In the MILPs, the difference between z_{MILP} and the value of the optimal solution to the continuous LP relaxation z_{LP}, in which y_1 and y_2 are treated as continuous variables between 0 and 1, is known as the relaxation gap. The relaxed solution z_{LP} of P1b and P1c is 22 and 12, respectively. Since the relaxation gap in P1b is 10 while the gap in P1c is zero, the time to solve P1c is expected to be much faster than the other two formulations. Figure 1 illustrates the feasible region for A and B, in the continuous relaxation of the formulations.

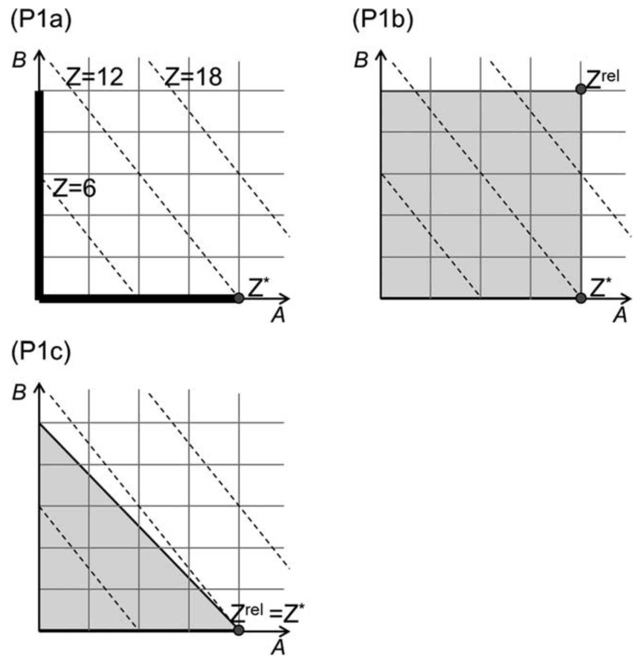


Figure 1. Feasible region of continuous relaxation of P1a, P1b and P1c projected in A and B.

Not only are there several ways in which to formulate MILP/MINLP problems, but also the transformation of a conditional argument into a constraint is not always obvious. Consider the following process synthesis requirement[24]: (P2) "If either the absorber to recover the product (y_a) and/or the membrane separator (y_m) is selected, then do not use cryogenic separation (y_c)". This logic condition can be modeled with any of the following three constraint(s), in which y_a, y_m and y_c are 0–1 variables

(P2a):

$$y_a+y_m+2y_c\le 2$$

(P2b):

$$\begin{aligned} y_a+y_c\le 1 \\ y_m+y_c\le 1 \end{aligned}$$

(P2c):

$$y_c+y_a+y_m-y_a*y_m\le 1$$

While the three aforementioned constraints are all valid representations of the logic condition, with the first two being linear and the last nonlinear, the modeling of logic constraints can be made more systematic by using logic propositions in which the 0–1 variables y are replaced by Boolean variables Y.[36] Following the previous example, the corresponding logic proposition is $If(Y_a\ or\ Y_m)\ then\ (not\ Y_c)$, where or is an inclusive or. In standard Boolean logic this statement can be expressed as follows

(P2d):

$$Y_a \vee Y_m \Rightarrow (\neg Y_c)$$

As will be seen later, P2d can be systematically transformed into linear inequalities with 0–1 variables, yielding constraints P2b, which in fact provide a "tighter" formulation than P2a when relaxing the integrality constraints (i.e., considering the 0–1 variables as continuous variables between 0 and 1).

The following notation (*logic operators*) will be used throughout the article:

$\vee$ or; means OR (inclusive, that is, A or B or both)
$\wedge$ and; means AND
$\neg$ means NOT (negation or complement)
$\Rightarrow$ means IF…THEN… (implication)
$\underline{\vee}$ means XOR (exclusive OR, that is, either A or B)
$\Longleftrightarrow$ means IF AND ONLY IF

As shown in the next section, GDP is a higher-level representation that allows the use of logic arguments in optimization problems as in the aforementioned example. This approach not only facilitates the development of models by making the formulation process more systematic and intuitive, but it also retains the underlying logic structure of the problem that can be exploited to find the solution more efficiently.

GDP Formulation

The general structure of a GDP can be represented as follows: (Note that this is a more general form than the one originally presented[13])

(GDP):

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

As shown in GDP, the objective is a function of the continuous variables x. It contains the global constraints $g(x) \le 0$ that must hold true regardless of the discrete decisions. The logic in the continuous space is represented by a set of disjunctions, $k \in K$, each of which contains $i \in D_k$ terms, linked by an OR operator ($\vee$). Each term of the disjunction has a Boolean variable Y_{ki} and an associated set of inequalities $r_{ki}(x) \le 0$. Exactly one of the Boolean variables can be selected in each disjunction $\left(\underline{\vee}_{i \in D_k} Y_{ki}\right)$. For an active term in a disjunction ($Y_{ki} = True$), the corresponding inequalities are enforced. When the term is not active ($Y_{ki} = False$), the corresponding constraints are ignored. The symbolic equation $\Omega(Y) = True$ represents the set of logic propositions that relates the Boolean variables. Note that GDP is a more structured alternative for modeling discrete-continuous optimization problems compared to mixed-integer programming, which is represented as:

(MINLP):

$$\begin{aligned} & \min z = f(x, y) \\ s.t.\ & g(x, y) \le 0 \\ & x \in \mathbb{R}^n,\ y \in \{0, 1\}^m \end{aligned}$$

where $f(x, y)$ is the objective function, $g(x, y) \le 0$ are inequality constraints, and y are the 0–1 variables, the counterpart to the Boolean variables Y. It should be noted that since (MINLP) is expressed through algebraic equations, the modeler has to directly express the logic in the format of $f(x, y)$ and $g(x, y)$, while in (GDP) the disjunctions capture the logic in continuous form and the logic propositions $\Omega(Y)$ capture the logic in the Boolean space.

While model (GDP) is quite general, in the Appendix we present the reformulation for the case of embedded disjunctions and inclusive or, in which the models are eventually transformed to the form of model (GDP). Note that in both general forms (GDP) and (MINLP), we only represent inequalities, since any equality can be considered as a set of inequalities (i.e., $h(x, y) = 0$ can be expressed as the two inequality constraints $h(x, y) \le 0$ and $-h(x, y) \le 0$).

GDP in process systems engineering

Decision making is a major element in process systems engineering (PSE), making GDP a very useful framework for modeling PSE optimization problems. Particularly in process synthesis, the decisions are normally associated as to whether certain equipment should be included or not in a process flow sheet. If the equipment is selected, then the mass and energy balance, physical and chemical equilibrium (if any), and cost constraints need to be satisfied. If it is not selected, then all the equations can be ignored. Therefore, in PSE GDP problems will normally take the following form[13]:

(GDP1):

$$\begin{aligned} & \min z = \hat{f}(x) + \sum_{k \in K} c_k \\ & s.t.\ g(x) \le 0 \\ & \vee_{i \in D_k} \begin{bmatrix} Y_{ki} \\ \hat{r}_{ki}(x) \le 0 \\ c_k = \gamma_{ki} \end{bmatrix} \qquad k \in K \\ & \underline{\vee}_{i \in D_k} Y_{ki} \qquad k \in K \\ & \Omega(Y) = True \\ & x \in \mathbb{R}^n,\ c_k \in \mathbb{R}^1 \\ & Y_{ki} \in \{True, False\},\ i \in D_k,\ k \in K \end{aligned}$$

Note that GDP1 is a particular case of (GDP), where $f(x) = \hat{f}(x) + \sum_{k \in K} c_k$ and $r_{ki}(x) \le 0$ is $\hat{r}_{ki}(x) \le 0$ and $c_k = \gamma_{ki}$ in (GDP1), where c_k represents the cost associated with enforcing a term in the disjunction. In process synthesis the continuous variables normally represent flows, temperatures and pressures of the stream in a process superstructure; while the Boolean variables represent the selection of equipment i for performing a process task k. The objective is a function of the continuous variables and the cost associated with each of the disjunctions k. The global constraints $g(x) \le 0$ normally represent mass and energy balances in the system. For an active term in a disjunction $Y_{ki} = True$, the cost $c_k = \gamma_{ki}$ and the constraints $\hat{r}_{ki}(x) \le 0$ are normally associated with the

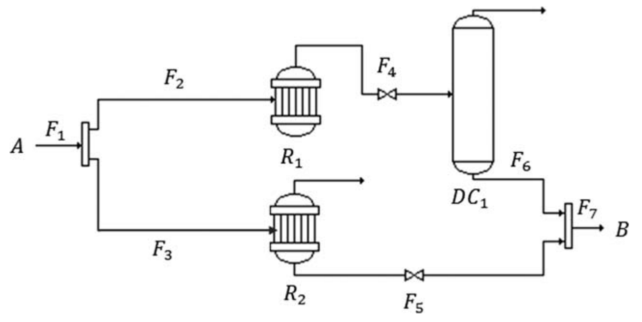


Figure 2. Process network example.

investment cost, the energy and mass balance, and physical and chemical equilibrium for the particular equipment i that is selected for the processing task k. The argument $\Omega(Y)=True$ generally represents the logic implications among the equipment in order to define a feasible topology for the process flow sheet. The logic constraints may be "hard" constraints like specification of choices, or redundant constraints in the sense that they may be implied by the mass balances at a process network. In the latter case the logic constraints can help to expedite the search for the optimum.

(GDP1) provides a general and systematic framework for modeling process networks. Although it is a particular case of (GDP), it has some MILP/MINLP reformulation advantages that will be discussed in later in this article.

Illustrative examples

We illustrate the concept of the model (GDP) with four PSE examples: Synthesis of a process flow sheet, determination of number of trays in a distillation column, job shop scheduling, and design of a batch process.

Process Synthesis. Consider the optimization of a simple process superstructure shown in Figure 2 that produces a product B by consuming raw material A. The variables F represent material flows. The problem is to determine the amount of product to produce (F_7) with a selling price P_7, the amount of raw material to buy (F_1) with a cost P_1 and the set of unit operations to use (i.e., R1, R2, DC1) with a cost c_k in order to maximize the profit.[25]

The GDP that represents the problem can be formulated as model (GDP1) as follows:

(P3):

$$\max z=P_7F_7-P_1F_1-\sum_{k\in K} c_k \tag{1}$$

$$\text{s.t.}$$

$$\begin{aligned} F_1&=F_2+F_3 \\ F_7&=F_5+F_6 \end{aligned} \tag{2}$$

$$\begin{bmatrix} Y_{R2} \\ F_5=\beta_{R2}F_3 \\ c_{R2}=\gamma_{R2} \end{bmatrix} \vee \begin{bmatrix} \neg Y_{R2} \\ F_3=F_5=0 \\ c_{R2}=0 \end{bmatrix}$$

$$\begin{bmatrix} Y_{R1} \\ F_4=\beta_{R1}F_2 \\ c_{R1}=\gamma_{R_1} \end{bmatrix} \vee \begin{bmatrix} \neg Y_{R1} \\ F_2=F_4=0 \\ c_{R1}=0 \end{bmatrix} \tag{3}$$

$$\begin{bmatrix} Y_{DC_1} \\ F_6=\beta_{DC_1}F_4 \\ c_{DC_1}=\gamma_{DC_1} \end{bmatrix} \vee \begin{bmatrix} \neg Y_{Dc_1} \\ F_4=F_6=0 \\ c_{DC_1}=0 \end{bmatrix}$$

$$Y_{R1} \iff Y_{DC_1} \tag{4}$$

$$F_j \geq 0 \qquad j=1,\ldots,7$$

$$F_j, c_k \in \mathbb{R}, Y_k \in \{True, False\} \qquad j=1,7 \qquad k\in\{DC1, R1, R2\}$$

In the aforementioned, (1) represents the objective function; (2) are the global constraints representing the mass balances around the splitter and mixer, respectively; (3) are the disjunctions that represent the selection or not of the unit k, with their respective characteristic constraints and fixed costs; and (4) the logic proposition that enforces the selection of DC1 if and only if R1 is chosen. For illustration purposes we have presented here a simple linear model. In the actual application to a process problem there would be hundreds or thousands of nonlinear equations in the GDP model.

Distillation Column Design. The optimal design of selecting the feed tray and the number of trays in a distillation column has remained a major challenge since the pioneering work by Sargent and Gaminibanadara[26] reported in 1976. Viswanathan and Grossmann[27] proposed an MINLP formulation involving a superstructure with variable reflux location as depicted in Figure 3. The idea is to consider a fixed feed tray and a fixed

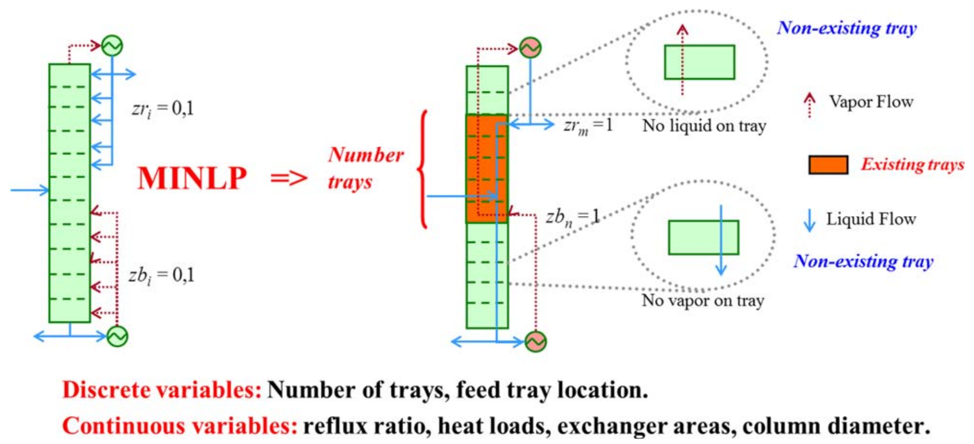


Figure 3. Variable reflux and reboil location with fixed feed tray.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

number of trays above and below the feed that represent an upper bound for the rectifying and stripping sections, respectively. The reflux is returned to any of the trays above the feed, and the reboil to any tray below the feed. 0–1 variables zr_i, zb_i are assigned to the reflux and reboil return to each tray i, respectively. The solution of the MINLP assigns only one binary variable zr_i to one, for the return of the reflux into the selected tray, so that all the trays above the selected tray are not required in the optimal design. Similarly, there is only one binary variable zb_i equal to one in the reboil return, and all the trays below this selected tray are not included in the optimal design. The main difficulty with this approach is that in the trays that are not selected (i.e., trays are inexistent and there is no mass transfer), vapor liquid equilibrium (VLE) equations still have to be satisfied.[27,28]

In contrast, Yeomans and Grossmann[23] proposed a GDP model for the same problem. In this model the selection of the trays is represented with the Boolean variables Y_n, where the tray numbering starts form the condenser ($n = 1$). If a certain tray is selected ($Y_n = True$), then the VLE equations have to be applied for that tray; if it is not selected ($Y_n = False$), then there is no heat and mass transfer in the tray (e.g., simply a "by-pass"). In this way, the formulation overcomes the difficulties of satisfying VLE in trays that are not selected. Figure 4 shows a column representation for this approach.

Given the complexity of the formulation, we present the GDP model as shown in P4, without explicitly including all the well-known equations

(P4):

$$\min Cost$$

$$s.t.$$

$$MESH\ equtions\ for\ permanet\ trays$$

$$\begin{bmatrix} Y_n \\ f^L_{n,i} = f\left(T^L_n, P_n, x_{n,i}\right) \\ f^V_{n,i} = f\left(T^V_n, P_n, y_{n,i}\right) \\ f^L_{n,i} = f^V_{n,i} \\ T^L_n = T^V_n \\ LIQ_{n,i} = L_n x_{n,i} \\ VAP_{n,i} = V_n y_{n,i} \\ stg_n = 1 \end{bmatrix} \vee \begin{bmatrix} \neg Y_n \\ f^L_{n,i} = 0 \\ f^V_{n,i} = 0 \\ T^L_n = T^L_{n-1} \\ T^V_n = T^V_{n+1} \\ L_n = L_{n-1} \\ V_n = V_{n+1} \\ x_{n,i} = x_{n-1,i} \\ y_{n,i} = y_{n+1,i} \\ stg_n = 0 \end{bmatrix}$$

$$ntray = \sum stg_n$$

$$\Omega(Y) = True$$

Note that if a tray is not selected ($\neg Yn$) the inlet and outlet flows $\left(L_{n-1}, L_n, V_n, V_{n+1}\right)$, (composition $\left(x_{n-1,i}, x_{n,i}, y_{n,i}, y_{n+1,i}\right)$ and temperatures of the liquid and vapor $\left(T^L_{n-1}, T^L_n, T^V_n, T^V_{n+1}\right)$ remain unchanged, while the fugacity $(f^L_{n,i}, f^V_{n,i})$ is set to zero.

Job shop scheduling. Consider a job shop scheduling problem where a set of jobs $i \in I$ must be processed sequentially on a set of consecutive stages $j \in J$, with given processing times for each stage. All jobs can be sequenced on a subset of stages. Furthermore, zero-wait transfer is assumed between stages, and the objective is to obtain a schedule that minimizes the makespan ms. A small example of such a problem is given in Table 1, where it can be seen that job A requires only stages 1 and 3, job B, stages 2 and 3, and C stages 1 and 2.

By defining the Boolean variables Y^1_{ik} and Y^2_{ik} to indicate whether job i is executed before job k or job k before job i, respectively, Raman and Grossmann[13] proposed the following model:

(P5):

$$\min ms \tag{P5.1}$$

s.t.

$$ms \geq t_i + \sum_{j \in J(i)} \tau_{ij} \quad \forall i \in I \tag{P5.2}$$

$$\begin{bmatrix} Y^1_{ik} \\ t_i + \sum_{\substack{m \in J(i) \\ m \leq j}} \tau_{im} \leq t_k + \sum_{\substack{m \in J(k) \\ m < j}} \tau_{km} \end{bmatrix} \vee \begin{bmatrix} Y^2_{ik} \\ t_k + \sum_{\substack{m \in J(k) \\ m \leq j}} \tau_{km} \leq t_i + \sum_{\substack{m \in J(i) \\ m < j}} \tau_{im} \end{bmatrix} \quad \forall j \in C_{ik} \forall i, k \in I, i < k \tag{P5.3}$$

$$Y^1_{ik} \underline{\vee} Y^2_{ik}$$

$$ms, t_i \in \mathbb{R}, Y^1_{ik}, Y^2_{ik} \in \{True, False\} \forall i, k \in I, i < k$$

in which t_i is the start time of job i. Equations P5.1 and P5.2 correspond to the objective function and aim to minimize the makespan ms. The disjunction in P5.3 ensures that no clash between jobs occurs at any stage at the same time, where for each pair of jobs i,k, the stages with potential clashes are $C_{ik} = \{J(i) \cap J(k)\}$. More specifically, the first term in each disjunction states that at a given stage j the completion time of job i is less or equal to the start of job k, and vice versa in the second term of each disjunction. Note that this requires the second job to have its processing times summed up to one stage less ($m < j$) than the first job ($m \leq j$), to represent start and completion times, respectively. The optimal schedule for the small example in Table 1, which has a minimum makespan of 11 h, is shown in Figure 5.

Synthesis of Multiproduct Batch Plant. The simultaneous synthesis, sizing, and scheduling of a flow shop multiproduct batch plants can be stated as follows.[29] It is desired to design a batch plant to produce different products ($i = 1,\ldots,N_p$) in a sequence of stages. The manufacturing of all of these products requires that they undergo a sequence of processing tasks ($t = 1,\ldots,T$) in exactly the same order to obtain the final products. There are different types of units available ($j = 1,\ldots,M$). Production requirements Q_i of the different products are specified over a given time horizon H. Processing times pt^T_{it} and size factors S_{it} (volume required per unit batch size) are given for each product i at the processing task t. To simplify the problem, cleanup times are assumed to be negligible. Each of the units j is capable of performing a corresponding task or a subset) of the T tasks. The problem then consists in determining the structure of the plant by deciding which tasks should be assigned to which unit, the number of parallel equipment for each of these units, and their corresponding sizes. Also, a production schedule must be determined that will ensure that the plant will be able to meet the production requirements over the given horizon time. Since the plant to be synthesized is a flow shop network, the tasks to be merged in any processing unit need to be adjacent. In addition, the cost of each unit is assumed to be given by an equation of the form $C_j=\gamma_j+\alpha_j V_j^{\beta}$, where γ is a fixed cost charge and α and β are cost parameters for the unit size V. For simplicity, the sizes are assumed to be continuous within specified lower and upper bounds V^L and V^U. Finally, the scheduling policies of unlimited intermediate storage (UIS) and zero wait (ZW) are considered for both types of campaigns, single and mixed product campaigns (SPCs and MPCs). The GDP formulation for the superstructure in Figure 6 is given by P6:

(P6):

$$\min COST=\sum_{j=1}^{M} N_j^{EQ} C_j+\sum_j CS_j$$

$$s.t.\ V_t^T \geq B_i S_{it} \quad i=1,\dots,N_P; t=1,\dots,T$$

$$pt_{ij}=\sum_{t\in T_j} pty_{itj} \quad i=1,\dots,N_P; j=1,\dots,M$$

$$n_i B_i \geq Q_i \quad i=1,\dots,N_P$$

$$\sum_{i=1}^{N_P} n_i T_{Li} \leq H$$

$$\bigvee_{j\in J_t} \begin{bmatrix} Y_{tj} \\ V_j \geq V_t^T \\ pty_{itj}=pt_{it}^T \\ pty_{itj'}=0, j'\neq j \end{bmatrix} \quad t\in T$$

$$\begin{bmatrix} YEX_j \\ C_j=\gamma_j+\alpha_j V_j^{\beta_j} \\ V_j^L \leq V_j \leq V_j^U \\ \begin{bmatrix} YC_{1,j} \\ N_j^{EQ}=1 \\ T_{Li} \geq pt_{ij} \end{bmatrix} \underline{\vee}\dots\underline{\vee} \begin{bmatrix} YC_{\max\left(N_j^{EQ}\right)j} \\ N_j^{EQ}=\max\left(N_j^{EQ}\right) \\ \left(\max\left(N_j^{EQ}\right)\right) T_{Li} \geq pt_{ij} \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \neg YEX_j \\ C_j=0 \\ V_j=0 \\ N_j^{EQ}=0 \\ pt_{ij}=0 \\ T_{Li} \geq 0 \end{bmatrix} \quad j\in J$$

$$\begin{bmatrix} YS_j \\ -\phi \leq B_{ij}-B_{ij'} \leq \phi \\ VST_j \geq S'_{ij} B_{ij} NEQ_j \\ VST_j \geq S'_{ij} B_{ij'} NEQ_{j'} \\ VST_j^{LB} \leq VST_j \leq VST_j^{UB} \\ CS_j=\gamma S_j+\alpha S_j VST_j^{\beta S_j} \end{bmatrix} \vee \begin{bmatrix} \neg YS_j \\ B_{ij}=B_{ij'} \\ VST_j=0 \\ CS_j=0 \end{bmatrix} \quad j\in J$$

$$\underline{\bigvee}_{j\in J_t} Y_{tj} \quad t\in T$$

$$YEX_j \Longleftrightarrow \bigvee_{t\in T_j} Y_{tj} \quad j\in J$$

$$W_{0j}\underline{\vee}\dots\underline{\vee}W_{kj} \quad j\in J, k=1,\dots,K_j$$

$$W_{0j} \Longleftrightarrow \bigwedge_{t\in T_j} \neg Y_{tj} \quad j\in J$$

$$W_{kj} \Longleftrightarrow \underline{\bigvee}_{\substack{t_i\in T_j \\ \sum i=k}} \left[\wedge Y_{t_i j} \bigwedge_{\substack{t'\in T_j \\ t'\neq t_i}} \neg Y_{t'j} \right] j\in J, k=1,\dots,K_j$$

$$0 \leq C_j, V_j, V_t^T, n_i, B_i, T_{Li}, pt_{ij}, N_j^{EQ}, pty_{itj}; YEX_{tj}, Y_{tj}, YC_{cj}, YS_j, W_{kj} \in \{True, False\}$$

In the aforementioned, the global constraints define equations for sizing, satisfying demands for each product and completing the production time within the specified time horizon H. pt_{ij} represents the processing time for product i in unit j, while pty_{itj} represents the processing time of a specific task t for product i in unit j. The first set of disjunctions refers to the assignment of a task t to an equipment j. The second disjunction corresponds to the selection of equipment j, which within its term has an embedded disjunction for selecting the number of parallel units (See Appendix 1 for embedded disjunctions). The third disjunction is for deciding whether to install a storage tank after unit j. The first logic constraint corresponds to the XOR associated with the Boolean variables of the first disjunction. The second logic constraints define if equipment (YEX_j) can perform task (Y_{tj}). Auxiliary Boolean variables W_{kj} are introduced to define the number of tasks k that equipment j will perform (where the maximum number of tasks is $K_j \leq |T_j|$). Only one of these variables W_{kj} can be selected

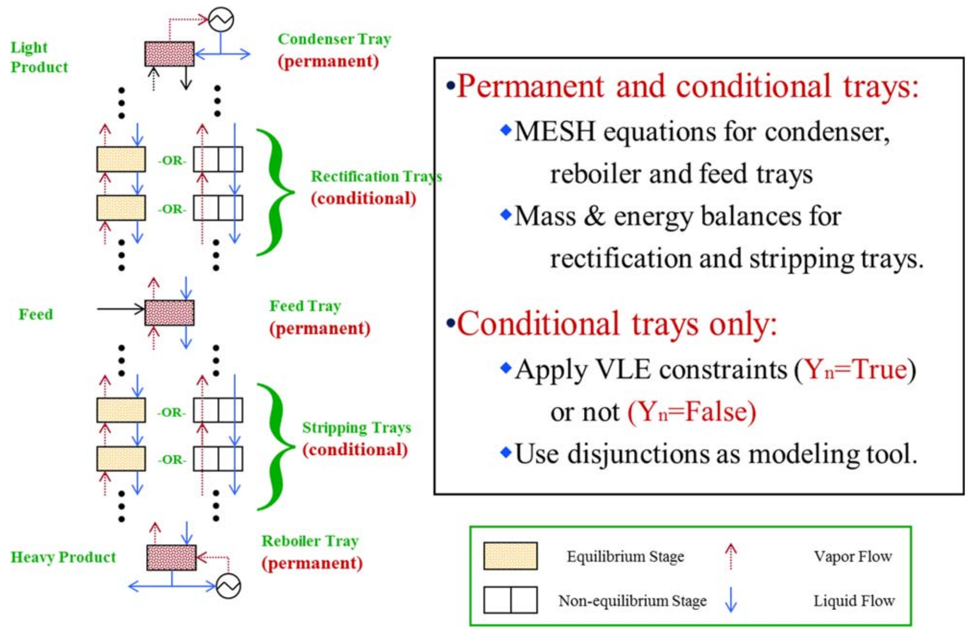

Light Product
Condenser Tray (permanent)
-OR-
-OR-
Rectification Trays (conditional)
Feed
Feed Tray (permanent)
-OR-
-OR-
Stripping Trays (conditional)
Heavy Product
Reboiler Tray (permanent)
•Permanent and conditional trays:
◆MESH equations for condenser, reboiler and feed trays
◆Mass & energy balances for rectification and stripping trays.
•Conditional trays only:
◆Apply VLE constraints (Yn=True) or not (Yn=False)
◆Use disjunctions as modeling tool.
Equilibrium Stage
Vapor Flow
Non-equilibrium Stage
Liquid Flow

Figure 4. Structure of disjunctive model with permanent and conditional trays.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

$W_{0j}\underline{\vee}\ldots\underline{\vee}W_{kj})$. If the unit will not perform any task, then $W_{0j}=True$. If it performs one task ($k=1$), then $W_{1j}=True$ and exactly one of the tasks t is assigned to equipment j ($Y_{tj}=True$), which requires that all other tasks are not assigned to that same unit $\left(Y_{t'j}{=}False\text{ for } t'\neq t\right)$. If it performs two tasks ($k=2$), then exactly two of the tasks $t_1, t_2 \in T_j$ are assigned to equipment $j\left(Y_{t_1j}, Y_{t_2j}=True\right)$, which requires that all other tasks are not assigned to that same unit $\left(Y_{t'j}{=}False\text{ for } t'\neq t_1,t_2\right)$. Same follows for $k{=}3,\ldots,K_j$, which is represented in the last logic constraints. These constraints define the topology of the network, and some examples can be found in Lee and Grossmann.[30]

GDP-to-MILP/MINLP Transformation

A number of methods have been proposed for solving directly GDP models as disjunctive problems (see Appendix 2). However, in order to take advantage of the existing MILP/MINLP solvers (see Grossmann[18] for a review on MINLP algorithms), GDPs are often reformulated as MILP/MINLP. This section addresses this procedure, which some software programs have partially automated.[31–33] To describe this transformation process, the GDP formulation is partitioned into three sections as shown in Figure 7.

For the first partition, the objective function and global constraints remain the same in both, the GDP and the MINLP formulations.

For the transformations of the second and third partition (disjunctions, logic propositions) it is required to convert them into algebraic form. Before we outline the detailed description of these transformations, the Boolean variables Y_{ki} are converted into binary variables. To do this, binary variables $y_{ki}\in\{0,1\}$ are introduced with a one-to-one correspondence with the Boolean variables: $Y_{ki}=True$ will be transformed to $y_{ki}=1$; $Y_{ki}=False$ will be transformed to $y_{ki}=0$.

Table 1. Processing times (τ_{ij}) for Small Illustrative Example

Jobs/ Stages	1	2	3
A	5	0	3
B	0	3	2
C	2	4	0

Logic proposition to MILP constraints transformation

Each of the *logic operators* can be transformed into a set of linear 0–1 MILP constraints as shown in Table 2[24,34–36]

In order to systematically derive the linear constraints with 0–1 variables using propositional logic, the following Boolean algebra rules are used (see Williams[36]):

$$(1)\ \text{Removing implication}\quad Y_1\Rightarrow Y_2 \iff \neg Y_1\vee Y_2$$

$$(2)\ \text{De Morgan's laws}:\quad \neg(Y_1\vee Y_2) \iff (\neg Y_1)\wedge(\neg Y_2)$$
$$\neg(Y_1\wedge Y_2) \iff (\neg Y_1)\vee(\neg Y_2)$$

$$(3)\ \text{Distributivity}:\quad (Y_1\wedge Y_2)\vee Y_3 \iff (Y_1\vee Y_3)\wedge(Y_2\vee Y_3)$$

There are many equivalent logic forms for representing a logic proposition. One form of particular interest is the *conjunctive normal form*. In this form, the logic statement is represented by a conjunction of clauses $Q_1\wedge Q_2\wedge\ldots\wedge Q_s$ (i.e., connected by AND operators $\wedge$), where the clauses correspond to logic propositions that only involve the OR operator. For the conjunctive normal form to be true, each clause must be true. With the Boolean rules described earlier, it is possible to systematically transform any logic proposition into its conjunctive normal form by applying recursively the following rules[24,35,37]:

1. Replace the implication by its equivalent disjunction.
2. Move the negation inward by applying DeMorgan's rules.
3. Recursively distribute the "OR" over the "AND".

Once the logic proposition is converted into conjunctive normal form, each of the clauses Q_i has to be satisfied, and

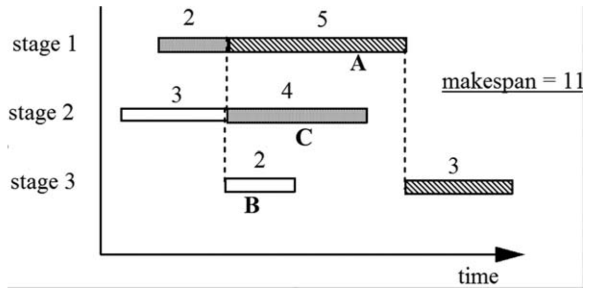


Figure 5. Optimal schedule for Illustrative example.

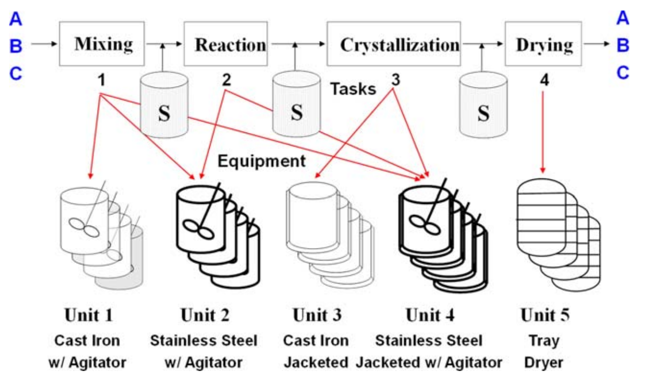


Figure 6. Illustrative superstructure for multiproduct batch plant synthesis.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

therefore each Q_i will translate into an inequality constraint according to the linear representation of Boolean expressions. This procedure is illustrated with the following example:

Consider the proposition[24] (P7):

$$(Y_1 \wedge Y_2) \vee Y_3 \Rightarrow (Y_4 \vee Y_5) \tag{P7.1}$$

Replace the implication by its equivalent disjunction

$$\neg[(Y_1 \wedge Y_2) \vee Y_3] \vee (Y_4 \vee Y_5) \tag{P7.2}$$

Move the negation inward by applying DeMorgan's rules

$$[\neg(Y_1 \wedge Y_2) \wedge \neg Y_3] \vee Y_4 \vee Y_5 \tag{P7.3}$$

$$[(\neg Y_1 \vee \neg Y_2) \wedge \neg Y_3] \vee Y_4 \vee Y_5 \tag{P7.4}$$

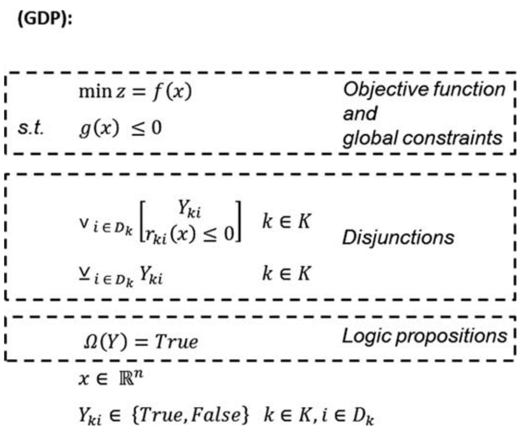
Recursively distribute the "OR" over the "AND"

$$[(\neg Y_1 \vee \neg Y_2 \vee Y_4 \vee Y_5) \wedge (\neg Y_3 \vee Y_4 \vee Y_5) \tag{P7.5}$$

Since P7.5 is in conjunctive normal form, and since it is required to satisfy each of the two clauses, the original logic proposition is equivalent to the two linear constraints in terms of the 0–1 variables y_1, y_2, y_3, y_4, y_5

$$y_1 + y_2 - y_4 - y_5 \leq 1$$
$$y_3 - y_4 - y_5 \leq 0$$

Note that if we apply this procedure to the example P2d $Y_a \vee Y_m \Rightarrow (\neg Y_c)$, we obtain the two constraints shown in P2b $y_a + y_c \leq 1$ and $y_m + y_c \leq 1$



(GDP):

$\min z = f(x)$		*Objective function and global constraints*
s.t. $g(x) \leq 0$		
$\vee_{i \in D_k} \begin{bmatrix} Y_{ki} \\ r_{ki}(x) \leq 0 \end{bmatrix}$	$k \in K$	*Disjunctions*
$\underline{\vee}_{i \in D_k} Y_{ki}$	$k \in K$	
$\Omega(Y) = True$		*Logic propositions*

$x \in \mathbb{R}^n$

$Y_{ki} \in \{True, False\}$ $k \in K, i \in D_k$

Figure 7. Partition of GDP formulation.

Table 2. MILP Representation of Boolean Connectives

Logic operator	Boolean Expression	MILP representation
$\vee$ *(OR)*	$Y_1 \vee Y_2 \vee \ldots \vee Y_n$	$y_1 + y_2 + \ldots + y_n \geq 1$
$\underline{\vee}$ *(XOR)*	$Y_1 \underline{\vee} Y_2 \underline{\vee} \ldots \underline{\vee} Y_n$	$y_1 + y_2 + \ldots + y_n = 1$
$\wedge$ *(AND)*	$Y_1 \wedge Y_2 \wedge \ldots \wedge Y_n$	$y_1 = 1$ $y_2 = 1$ $\ldots$ $y_n = 1$
$\neg$ *(NOT)*	$\neg Y_1$	$1 - y_1$
$\Longrightarrow$ *(IF)*	$Y_1 \Rightarrow Y_2$	$y_1 \leq y_2$
$\Longleftrightarrow$ *(IF AND ONLY IF)*	$Y_1 \Longleftrightarrow Y_2$	$y_1 = y_2$

Disjunctions to MILP/MINLP constraints transformation

Disjunctions in GDP problems are often reformulated using either Big-M (BM)[17] or Hull Reformulation (HR)[16] formulations:

Big-M (BM). In Table 3 the constraint $\sum_{i \in D_k} y_{ki} = 1$ guarantees that for each disjunction $k \in K$ only one term $i \in D_k$ is active (i.e., only one binary variable y_{ki} in each $k \in K$ will take a value of one). In this formulation when $y_{ki} = 1$ (Boolean variable is active), the disjunction constraints are enforced, where $r_{ki}(x)$ can be linear or nonlinear. When $y_{ki} = 0$ and the parameter M^{ki} is significantly large, the associated constraints become redundant.

To illustrate the big-M reformulation of disjunctions, consider a very small example, the modeling of the installation of equipment (a larger example is shown in Appendix 3). If equipment is installed, there is a capital cost γ, and an operating cost αF^{β}, where F is the amount produced. If the equipment is not installed, the cost (C) and the amount produced are zero. This can be modeled as shown in P8.GDP:

(P8.GDP):

$$\begin{bmatrix} Y \\ C = \gamma + \alpha F^{\beta} \end{bmatrix} \vee \begin{bmatrix} \neg Y \\ C, F = 0 \end{bmatrix}$$
$$0 \leq C \leq C^{up}$$
$$0 \leq F \leq F^{up}$$
$$Y \in \{True, False\}$$

The big-M formulation of this disjunction is as follows:

(P8.BM):

$$C \leq \gamma + \alpha F^{\beta} + M^1(1-y)$$
$$C \geq \gamma + \alpha F^{\beta} - M^2(1-y)$$
$$C \leq M^3 y$$
$$F \leq M^4 y$$
$$0 \leq C \leq C^{up}$$
$$0 \leq F \leq F^{up}$$
$$y \in \{0, 1\}.$$

where M^1,M^2, M^3, M^4 are large parameters.

Table 3. (BM) Reformulation of Disjunctions

(GDP) Disjunctions		(BM) Constraints	
$\vee_{i \in D_k} \begin{bmatrix} Y_{ki} \\ r_{ki}(x) \leq 0 \end{bmatrix}$	$k \in K$	$r_{ki}(x) \leq M^{ki}(1 - y_{ki})$	$k \in K, i \in D_k$
$\underline{\vee}_{i \in D_k} Y_{ki}$	$k \in K$	$\sum_{i \in D_k} y_{ki} = 1$	$k \in K$

Table 4. (HR) Reformulation of Disjunctions with Linear Constraints

(GDP) Disjunctions		(HR) Constraints	
		$x=\sum_{i\in D_k} v^{ki}$	$k\in K$
$\vee_{i\in D_k}\begin{bmatrix} Y_{ki} \\ A^{ki}x\le a^{ki}\end{bmatrix}$	$k\in K$	$A^{ki}v^{ki}\le a^{ki}y_{ki}$	$k\in K, i\in D_k$
$\underline{\vee}_{i\in D_k} Y_{ki}$	$k\in K$	$x^{lo}y_{ki}\le v^{ki}\le x^{up}y_{ki}$	$k\in K, i\in D_k$
		$\sum_{i\in D_k} y_{ki}=1$	$k\in K$

Hull-Reformulation (HR). For the Hull-Reformulation it is convenient to distinguish between the linear and nonlinear case, as shown in Tables 4 and 5.

In Tables 4 and 5, the constraint $\sum_{i\in D_k} y_{ki}=1$ will guarantee that for each disjunction $k\in K$ only one term $i\in D_k$ is active (same as in (BM)). The variables x are disaggregated into the variables v^{ki} for each disjunctive term. An upper and lower bound constraint of the disaggregated variables v^{ki} associated with the binary variables is added $x^{lo}y_{ki}\le v^{ki}\le x^{up}y_{ki}$, so when $y_{ki}=1$ the variable can take values between the upper and lower bound, and when $y_{ki}=0$ then $v^{ki}=0$. Also, when $y_{ki}=1$ the constraints $r_{ki}(v^{ki})\le 0$ are enforced; and when $y_{ki}=0$, they are trivially satisfied $(0\le 0)$. Also note that in Table 5 the second inequality is defined by the perspective function $y_{ki}r_{ki}(v^{ki}/y_{ki})$ which is convex if $r_{ki}(x)$ is convex. Furthermore, for the linear case where $r_{ki}(x)=A^{ki}x-a^{ki}$, this function reduces to the second inequality in Table 4.

To illustrate the (HR) formulation, consider again P8.GDP. If $\beta=1$ then the problem is linear, and the (HR) is given by P8.HR1.

(P8.HR1):

$$\begin{aligned} &C=C_1+C_2\\ &F=F_1+F_2\\ &C_1=\alpha F_1+\gamma * y\\ &C_2=0\\ &F_2=0\\ &0\le C_1\le C^{up}y\\ &0\le F_1\le F^{up}y\\ &y\in\{0,1\}\end{aligned}$$

If $\beta\neq 0,1$ then the problem is nonlinear, and the (HR) is given by P8.HR2.

Table 5. (HR) Reformulation of Disjunctions with Nonlinear Constraints

(GDP) Disjunctions		(HR) Constraints	
		$x=\sum_{i\in D_k} v^{ki}$	$k\in K$
$\vee_{i\in D_k}\begin{bmatrix} Y_{ki} \\ r_{ki}(x)\le 0\end{bmatrix}$	$k\in K$	$y_{ki}r_{ki}(v^{ki}/y_{ki})\le 0$	$k\in K, i\in D_k$
$\underline{\vee}_{i\in D_k} Y_{ki}$	$k\in K$	$x^{lo}y_{ki}\le v^{ki}\le x^{up}y_{ki}$	$k\in K, i\in D_k$
		$\sum_{i\in D_k} y_{ki}=1$	$k\in K$

(P8.HR2):

$$\begin{aligned} &C=C_1+C_2\\ &F=F_1+F_2\\ &C_1=y*\left[\alpha\left(\frac{F_1}{y}\right)^{\beta}+\gamma\right]\\ &C_2=0\\ &F_2=0\\ &0\le C_1\le C^{up}y\\ &0\le F_1\le F^{up}y\\ &y\in\{0,1\}\end{aligned}$$

To avoid singularities in the nonlinear case the following approximation can be used for the perspective function[38]

(APP):

$$y_{ki}r_{ki}(v^{ki}/y_{ki})\approx((1-\varepsilon)y_{ki}+\varepsilon)r_{ki}\left(\frac{v^{ki}}{(1-\varepsilon)y_{ki}+\varepsilon}\right)-\varepsilon r_{ki}(0)(1-y_{ki})$$

where ε is a small finite number (e.g., 10^{-5}). This approximation is convex if $r_{ki}(x)$ is convex, and it yields an exact value of the function at $y_{ki}=0, y_{ki}=1$. It should also be noted that for linear disjunctions it is sometimes possible to eliminate the disaggregated variables by algebraic manipulation, particularly when a variable is set to zero in one of the disjunctive terms.[13]

Disjunction Reformulation in Model (GDP1): For the particular form (GDP1) of (GDP), the (HR) and (BM) can be modified to provide improved reformulations. The inequality in the disjunction $\hat{r}_{ki}(x)\le 0$ is reformulated with either (BM) or (HR) as explained earlier; but the cost equality $c_k=\gamma_{ki}$, where γ_{ki} is a fixed cost, is directly transformed in the objective, so the objective function $\hat{f}(x)+\sum_{k\in K}c_k$ becomes $\hat{f}(x)+\sum_{k\in K}\sum_{i\in D_k}\gamma_{ki}y_{ki}$. It is not difficult to show that this is simply the (HR) of $c_k=\gamma_{ki}$ after algebraic substitution. The advantage of this reformulation is that it eliminates the need of introducing disaggregated variables for c_k, while producing the (HR) of this equality. Sawaya and Grossmann[39] provide further insight in the difference between both formulations for the linear case.

Comparing MINLP reformulations

In summary, by denoting $Hx\ge h$ the MILP transformation of the logic propositions (obtained through the process described in Disjunctions to MILP/MINLP constraints transformation section), and applying the (BM) and (HR) for the disjunctions (Tables 3, 4 and 5), the reformulation of the linear and nonlinear GDP is shown in Tables 6 and 7.[16,17]

Note that because of the disaggregated variables v^{ki}, (HR) has more variables than BM. Additionally, in (HR) two constraints are included for each disaggregated variables (to set upper and lower limits multiplied by the binary variables), as well as the constraint that relates a variable to the disaggregated variables. Therefore, (HR) has more constraints than (BM). On the other hand, as proved in Grossmann and Lee[40] and discussed by Vecchietti et al.[41] the continuous relaxation of the (HR) formulation is at least as tight and generally tighter than the (BM) when the discrete domain is relaxed (i.e.,$0\le y_{ki}\le 1; k\in K, i\in D_k$). This is of great importance considering that the efficiency of the MILP/MINLP solvers heavily relies on the quality of these relaxations. Figure 8 illustrates the (BM) and (HR) relaxations projected in the space of the continuous variables of a linear GDP that involves two disjunctions, each with two terms. Note that the feasible region is disjoint as it consists of the two rectangles with bold dashes. While this feasible region is the same for both formulations, the continuous relaxations are different, with the case of the (HR) clearly being tighter as it corresponds to the intersection of the convex hulls of each of the two disjunctions.

GDP-to-MINLP Illustration

We illustrate the process of transforming a GDP into an MILP/MINLP with the optimization of a quadratic function over three circles. A larger example of a linear GDP for a process synthesis problem is presented in Appendix 3.

Consider the analytical problem given in Lee and Grossmann[16] which consists in finding the minimum of a quadratic objective function, subject to a disjoint feasible region defined by circles as shown in Figure 9. The problem formulation is given by P9.GDP.

(P9.GDP):

$$\begin{aligned}
&\min Z=(x_1-5)^2+(x_2-5)^2\\
&s.t.\\
&\begin{bmatrix} Y_1 \\ x_1^2+x_2^2 \le 1 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1-4)^2+(x_2-1)^2 \le 1 \end{bmatrix}\\
&\vee \begin{bmatrix} Y_3 \\ (x_1-2)^2+(x_2-4)^2 \le 1 \end{bmatrix}\\
&Y_1 \underline{\vee} Y_2 \underline{\vee} Y_3\\
&-5 \le x_1, x_2 \le 5;\, Y_1, Y_2, Y_3 \in \{True, False\}
\end{aligned}$$

P9.BM and P9.HR show the corresponding MINLP transformation. For the case of the (HR) transformation we use the approximation APP[38]

(P9.BM):

$$\begin{aligned}
&\min Z=(x_1-5)^2+(x_2-5)^2\\
&s.t.\\
&x_1^2+x_2^2 \le 1+M(1-y_1)\\
&(x_1-4)^2+(x_2-1)^2 \le 1+M(1-y_2)\\
&(x_1-2)^2+(x_2-4)^2 \le 1+M(1-y_3)\\
&y_1+y_2+y_3=1\\
&-5 \le x_1, x_2 \le 5\\
&y_{1,2,3} \in \{0,1\}
\end{aligned}$$

$$\begin{aligned}
&(\mathbf{P9.HR}):\\
&\min Z=(x_1-5)^2+(x_2-5)^2\\
&st.\\
&x_1=x_{1_1}+x_{1_2}+x_{1_3}\\
&x_2=x_{2_1}+x_{2_2}+x_{2_3}\\
&((1-\varepsilon)y_1+\varepsilon)\left[\left(\frac{x_{1_1}}{(1-\varepsilon)y_1+\varepsilon}\right)^2+\left(\frac{x_{2_1}}{(1-\varepsilon)y_1+\varepsilon}\right)^2-1\right]+\varepsilon(1-y_1) \le 0\\
&((1-\varepsilon)y_2+\varepsilon)\left[\left(\frac{x_{1_2}}{(1-\varepsilon)y_2+\varepsilon}-4\right)^2+\left(\frac{x_{2_2}}{(1-\varepsilon)y_2+\varepsilon}-1\right)^2-1\right]-16\varepsilon(1-y_2) \le 0\\
&((1-\varepsilon)y_3+\varepsilon)\left[\left(\frac{x_{1_3}}{(1-\varepsilon)y_3+\varepsilon}-2\right)^2+\left(\frac{x_{2_3}}{(1-\varepsilon)y_3+\varepsilon}-4\right)^2-1\right]-19\varepsilon(1-y_3) \le 0\\
&y_1+y_2+y_3=1\\
&-5y_i \le x_{j_i} \le 5y_i \quad i=1,2,3; \quad j=1,2\\
&y_{1,2,3} \in \{0,1\}
\end{aligned}$$

The optimal solution to P9.GDP is 4.68. If the continuous relaxations of P9.BM and P9.HR are considered by solving the corresponding NLPs, the respective lower bounds 0.45 and 4.20 are obtained, showing that the (HR) provides a much stronger lower bound since it has a tighter relaxation.

Improving convex GDP reformulations through Basic Steps

While the HR formulation provides tighter formulation than (BM), it is possible to develop formulations that are even tighter than HR using some basic concepts of disjunctive programming.[22,42] GDP can be regarded as an extension of the well-known disjunctive programming model developed by Balas,[22,42] and which consists of a linear program with disjunctions, but no Boolean variables. Sawaya and Grossmann[38,39] proved that any linear (GDP) can be equivalently formulated as a disjunctive program, and, therefore, the rich theory behind disjunctive programming can be used in order to solve linear GDPs more efficiently. Ruiz and Grossmann[25,43] extended this theory to nonlinear convex GDP. A convex GDP is a particular case of (GDP) in which the functions $f(x)$, $g(x)$ and $r_{ki}(x)$ are convex.

Table 6. Linear GDP Reformulation

(GDP)	(BM)	(HR)
$\min z=f(x)$	$\min z=f(x)$	$\min z=f(x)$
$s.t. \quad g(x) \le 0$	$s.t. \quad g(x) \le 0$	$s.t. \quad g(x) \le 0$
		$x=\sum_{i \in D_k} v^{ki} \quad k \in K$
$\vee_{i \in D_k}\begin{bmatrix} Y_{ki} \\ A^{ki}x \le a^{ki} \end{bmatrix} \quad k \in K$	$A^{ki}x \le a^{ki}+M^{ki}(1-y_{ki}) \quad k \in K, i \in D_k$	$A^{ki}v^{ki} \le a^{ki}y_{ki} \quad k \in K, i \in D_k$
$\underline{\vee}_{i \in D_k} Y_{ki} \quad k \in K$	$\sum_{i \in D_k} y_{ki}=1 \quad k \in K$	$x^{lo}y_{ki} \le v^{ki} \le x^{up}y_{ki} \quad k \in K, i \in D_k$
		$\sum_{i \in D_k} y_{ki}=1 \quad k \in K$
$\Omega(Y)=True$	$Hx \ge h$	$Hx \ge h$
$x^{lo} \le x \le x^{up}, \quad x \in \mathbb{R}^n$	$x^{lo} \le x \le x^{up}, \quad x \in \mathbb{R}^n$	$x \in \mathbb{R}^n$
$Y_{ki} \in \{True, \ False\} \ k \in K, \ i \in D_k$	$y_{ki} \in \{0,1\} \quad k \in K, \ i \in D_k$	$y_{ki} \in \{0,1\} \quad k \in K, \ i \in D_k$

A particular logic operation in disjunctive programming that helps to generate a tighter formulation than the (HR) of the problem is the so-called Basic Step.[42]

Basic Step in Disjunctive Programming[22,42,43]. To describe this operation it is important to consider the following definitions[42,43]:

Convex inequality: (Halfspace in linear case) $H=\{x \in \mathbb{R}^n] \phi(x) \le 0\}$ where $\phi(x):\mathbb{R}^n \to \mathbb{R}^1$ is a convex function

Convex set: (Polyhedron in linear case)

$$P=\cap_{i \in M} H_i$$

Elementary disjunctive set:

$$D=\cup_{i \in M} H_i$$

A disjunctive set can be expressed in many different logically equivalent forms. Three of them are of particular interest:

$$\text{Conjunctive Normal Form (CNF)} : F=\cap_{i \in T} D_i=\cap_{i \in T}\cup_{i \in M} H_i$$
$$\text{Disjunctive Normal Form (DNF)} : F=\cup_{i \in Q} P_i=\cup_{i \in Q}\cap_{i \in M} H_i$$

Regular Form (RF) (intersection of DNFs) : $F=\cap_{j \in T} S_j$, where $S_j=\cup_{i \in Q} P_i$

Note that the CNF and DNF are two extremes of equivalent forms, while RF is between them.

As stated and proved in Theorem 2.1 of Balas,[42] any disjunctive set in Regular Form can be converted into DNF by a recursive application of an operation called Basic Step. In particular, given a disjunctive set in RF, it can be brought to DNF by $|T|-1$ recursive applications of the following basic step, that involves intersecting pairs of disjunctions. That is, for some $k, l \in T, k \neq l, S_k \cap S_1$ is brought to DNF by replacing it with $S_{kl}=\cup_{i \in Q_k, j \in Q_1}(P_i \cap P_j)$.

Additionally, as stated and proved in Theorem 4.3 of Balas,[42] the hull relaxation of a disjunctive set $hrel(F):=\cap_{j \in T} cl\, conv(S_j)$, where *cl conv* is the closed convex set, after the application of a basic step is as tight, if not tighter, than the previous hull relaxation. It follows that the hull relaxation of a disjunctive set in DNF is equal to its convex hull (or perfect formulation, since it can be solved as a continuous optimization problem).

It is important to note that when a basic step is applied, the hull relaxation of the new disjunctive set has more continuous variables and constraints. There are different rules that have been proposed for deciding when to apply or not a basic step.[38,42,43]

Illustrative example of Basic Step in GDP. A Basic Step is illustrated with the following example: The objective is to maximize profit by selling a product P at a price 10. To produce it, it is required to buy either reactor R1 with cost C = 5 * (inlet flow) and conversion of 90% for raw material A and 70% for B, or reactor R2 with C = 4.6 * (inlet flow) and 85% conversion for raw material A and 80% for B. The cost of raw material A is 1.1 and of B is 1. The maximum

Table 7. Nonlinear GDP Reformulation

(GDP)	(BM)	(HR)
$\min z=f(x)$	$\min z=f(x)$	$\min z=f(x)$
$s.t. \quad g(x) \le 0$	$s.t. \quad g(x) \le 0$	$s.t. \quad g(x) \le 0$
		$x=\sum_{i \in D_k} v^{ki} \quad k \in K$
$\vee_{i \in D_k}\begin{bmatrix} Y_{ki} \\ r_{ki}(x) \le 0 \end{bmatrix} \quad k \in K$	$r_{ki}(x) \le M^{ki}(1-y_{ki}) \quad k \in K, i \in D_k$	$y_{ki}r_{ki}(v^{ki}/y_{ki}) \le 0 \quad k \in K, i \in D_k$
$\underline{\vee}_{i \in D_k} Y_{ki} \quad k \in K$	$\sum_{i \in D_k} y_{ki}=1 \quad k \in K$	$x^{lo}y_{ki} \le v^{ki} \le x^{up}y_{ki} \quad k \in K, i \in D_k$
		$\sum_{i \in D_k} y_{ki}=1 \quad k \in K$
$\Omega(Y)=True$	$Hx \ge h$	$Hx \ge h$
$x^{lo} \le x \le x^{up}, \quad x \in \mathbb{R}^n$	$x^{lo} \le x \le x^{up}, \quad x \in \mathbb{R}^n$	$x \in \mathbb{R}^n$
$Y_{ki} \in \{True, \ False\} \ k \in K, \ i \in D_k$	$y_{ki} \in \{0,1\} \quad k \in K, \ i \in D_k$	$y_{ki} \in \{0,1\} \quad k \in K, \ i \in D_k$

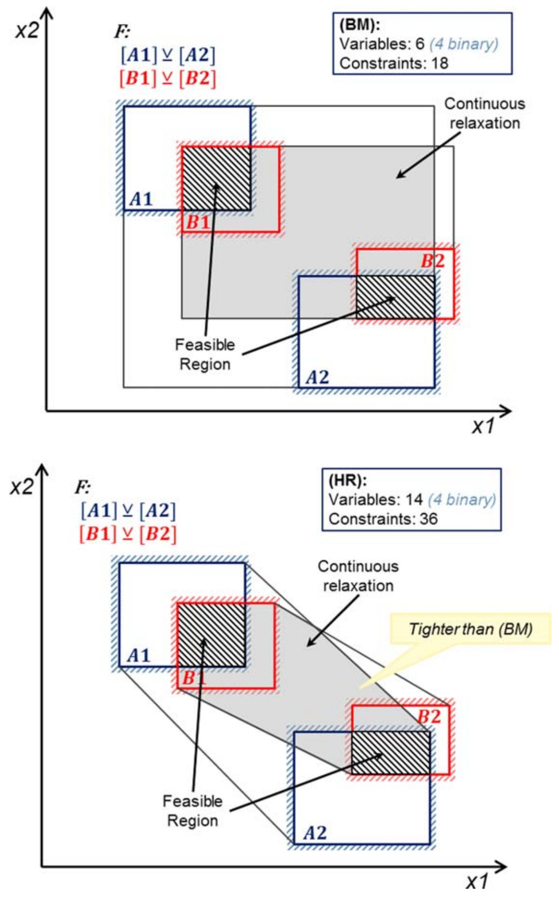


Figure 8. Illustration of big-M and Hull reformulation relaxation.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

allowed cost for any selected equipment is 30, and it is possible to select only one of the raw materials. There is a limit on the availability, for A = 5, and B = 7. This process network is illustrated in Figure 10, and the GDP can be formulated as follows:

(P10.1):

$$\max \text{Profit} = 10F_P - C_{eq} - C_{raw}$$

$$\text{s.t.}$$

$$\begin{bmatrix} Y_{R1} \\ F_P = 0.9F_A + 0.7F_B \\ C_{eq} = 5.0(F_A + F_B) \end{bmatrix} \vee \begin{bmatrix} Y_{R2} \\ F_P = 0.85F_A + 0.8F_B \\ C_{eq} = 4.6(F_A + F_B) \end{bmatrix}$$

$$\begin{bmatrix} Y_{RawA} \\ C_{raw} = 1.1F_A \\ F_B = 0 \end{bmatrix} \vee \begin{bmatrix} Y_{RawB} \\ C_{raw} = 1F_B \\ F_A = 0 \end{bmatrix}$$

$$0 \le F_A \le 5$$
$$0 \le F_B \le 7$$
$$0 \le C_{eq} \le 30$$
$$Y_{R1} \veebar Y_{R2}$$
$$T_{RawA} \veebar Y_{RawB}$$
$$Y_{R1}, Y_{R2}, Y_{RawA}, Y_{RawB} \in \{True, False\}$$

Note that the problem has two *proper* disjunctions (those that contain the OR operator), with two terms each disjunction, and two equations each term. It also contains six *improper* disjunctions (the upper and lower bounds of *FA, FB* and C_{eq}, i.e., the global constraints). The solution to this problem is to select reactor 2 and purchase raw material A, with **profit = 15.65**. For the relaxation of (BM) **profit = 50,096**, and for the relaxation of (HR), **profit = 16.13**, which is clearly much tighter.

Intersecting the two disjunctions, which represents performing a basic step and distributing the OR over the AND operator, leads to the following problem:

(P10.2):

$$\max \text{Profit} = 10F_P - C_{eq} - C_{raw}$$

$$\text{s.t.}$$

$$\begin{bmatrix} Y_{R1}\ AND\ Y_{RawA} \\ F_P = 0.9F_A + 0.7F_B \\ C_{eq} = 5.0(F_A + F_B) \\ C_{raw} = 1.1F_A \\ F_B = 0 \end{bmatrix} \vee \begin{bmatrix} Y_{R1}\ AND\ Y_{RawB} \\ F_P = 0.9F_A + 0.7F_B \\ C_{eq} = 5.0(F_A + F_B) \\ C_{raw} = 1F_B \\ F_A = 0 \end{bmatrix} \vee \begin{bmatrix} Y_{R2}\ AND\ Y_{RawA} \\ F_P = 0.85F_A + 0.8F_B \\ C_{eq} = 4.6(F_A + F_B) \\ C_{raw} = 1.1F_A \\ F_B = 0 \end{bmatrix} \vee \begin{bmatrix} Y_{R2}\ AND\ Y_{RawB} \\ F_P = 0.85F_A + 0.8F_B \\ C_{eq} = 4.6(F_A + F_B) \\ C_{raw} = 1F_B \\ F_A = 0 \end{bmatrix}$$

$$0 \le F_A \le 5$$
$$0 \le F_B \le 7$$
$$0 \le C_{eq} \le 30$$
$$Y_{R1} \veebar Y_{R2}$$
$$Y_{RawA} \veebar Y_{RawB}$$
$$Y_{R1}, Y_{R2}, Y_{RawA}, Y_{RawB} \in \{True, False\}$$

It is easy to see that P10.2 and P10.2 are logically equivalent, since P10.2 is only enumerating the potential combinations between R1 and R2 with raw materials A and B. Also note that the problem now contains a single *proper* disjunction, with four disjunctive terms, and five equations in each term. This "distribution" operation is called *proper* Basic

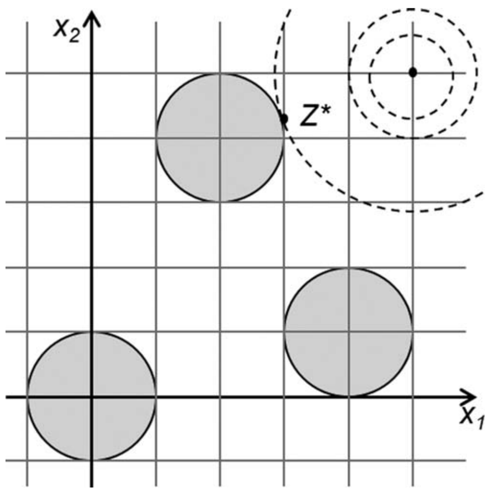


Figure 9. Illustration of P9.GDP.

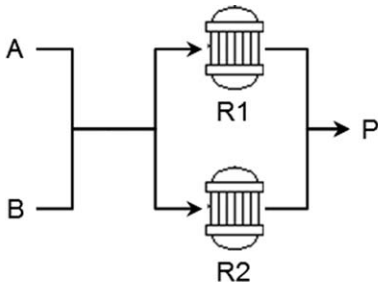


Figure 10. Structure of reactor and raw material selection in P10.1.

Step. The application of this operation to the improper disjunctions (global constraints), is called an *improper* Basic Step. The extreme case in which the problem is defined over a single disjunction is called Disjunctive Normal Form. The DNF of P10.1 is then as follows:

$$
\begin{aligned}
&(\mathbf{P10.3 = DNF}):\\
&\max \text{Profit} = 10F_P - C_{eq} - C_{raw}\\
&\text{s.t.}\\
&\begin{bmatrix} Y_{R1}\ AND\ Y_{RawA} \\ F_P = 0.9F_A + 0.7F_B \\ C_{eq} = 5.0(F_A+F_B) \\ C_{raw} = 1.1F_A \\ F_B = 0 \\ 0 \le F_A \le 5 \\ 0 \le C_{eq} \le 30 \end{bmatrix} \vee
\begin{bmatrix} Y_{R1}\ AND\ Y_{RawB} \\ F_P = 0.9F_A + 0.7F_B \\ C_{eq} = 5.0(F_A+F_B) \\ C_{raw} = 1F_B \\ F_A = 0 \\ 0 \le F_B \le 7 \\ 0 \le C_{eq} \le 30 \end{bmatrix} \vee
\begin{bmatrix} Y_{R2}\ AND\ Y_{RawA} \\ F_P = 0.85F_A + 0.8F_B \\ C_{eq} = 4.6(F_A+F_B) \\ C_{raw} = 1.1F_A \\ F_B = 0 \\ 0 \le F_A \le 5 \\ 0 \le C_{eq} \le 30 \end{bmatrix} \vee
\begin{bmatrix} Y_{R2}\ AND\ Y_{RawB} \\ F_P = 0.85F_A + 0.8F_B \\ C_{eq} = 4.6(F_A+F_B) \\ C_{raw} = 1F_B \\ F_A = 0 \\ 0 \le F_B \le 7 \\ 0 \le C_{eq} \le 30 \end{bmatrix}\\
&Y_{R1} \underline{\vee} Y_{R2}\\
&Y_{RawA} \underline{\vee} Y_{RawB}\\
&Y_{R1}, Y_{R2}, Y_{RawA}, Y_{RawB} \in \{True, False\}
\end{aligned}
$$

It is also clear to see the logic equivalence of P10.2 and P10.3, only that in P10.3 the global constraints are inside the disjunctive terms. The relaxation of the (HR) of P10.3 yields a **profit = 15.65**, which is the same as the optimal MILP solution. Note that with the application of |K|-1 Basic Steps a problem achieves DNF[39,43] form.

There are three main consequences of this operation[42]: (1) The size of the problem increases as we apply BS (exponentially in terms when applying *proper* Basic Steps, and increase in the number of constraints when applying *improper* Basic Steps); (2) the problem generated after applying a BS is at least as tight as before, and possibly tighter, and (3) the HR of the DNF is a perfect formulation. There has been recent algorithmic implementation of these concepts.[44]

Extension of Basic Step to Global Optimization of Nonconvex GDP. A discussion on global optimization of nonconvex GDP problems is out of the scope of this article. However, to provide some context we include the following brief description. In order to apply the theory of Disjunctive Programming into nonconvex GDP for global optimization, Ruiz and Grossmann[45] proposed a two-stage approach. The first stage relaxes the nonconvexities by replacing them with over and under estimating functions, or convex envelopes, such as the ones by McCormick[46] for bilinear terms. The second stage involves strengthening the resulting convex GDP using basic steps. Figure 11 shows the general framework of this approach, which has the net result of providing stronger lower bounds for the global optimum solution of the nonconvex GDP. This is important because the effectiveness of the global optimization method relies heavily on the quality of this bound.

Numerical Examples

In this section, we compare the different formulations and improvement after the application of Basic Steps (according the corresponding reference) for three examples of modest size taken from the literature. The first example P11 is linear; the next is

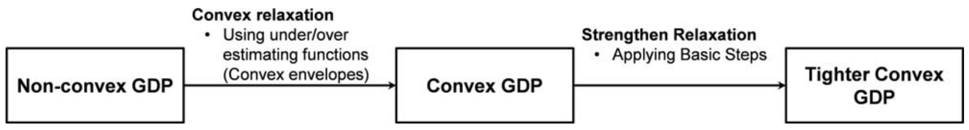


Figure 11. Two phase approach for nonconvex GDP.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

nonlinear but convex P12, and the final example is nonconvex P13. For the nonconvex problem, problem size refers to the MINLP reformulation of the convex GDP relaxation.

Linear GDP: Strip-packing problem

A given set of rectangles is to be packed into a strip of fixed width W but unknown length L. The objective is to minimize the length of the strip while fitting all rectangles without any overlap and without rotation. The GDP formulation is as follows[39]:

$$
\begin{array}{lll}
 & (\mathbf{P11}): & \\
 & \min lt & \\
s.t. & It \geq x_i + L_i & i \in N \\
 & \begin{bmatrix} Y_{ij}^1 \\ x_i + L_i \leq x_j \end{bmatrix} \vee \begin{bmatrix} Y_{ij}^2 \\ x_j + L_j \leq x_i \end{bmatrix} \vee \begin{bmatrix} Y_{ij}^3 \\ y_i - H_i \geq y_j \end{bmatrix} \vee \begin{bmatrix} Y_{ij}^4 \\ y_j - H_j \geq y_i \end{bmatrix} & i,j \in, i<j \\
 & 0 \leq x_i \leq UB - L_i & i \in N \\
 & H_i \leq y_i \leq W & i \in N \\
 & Y_{ij}^1 \underline{\vee} Y_{ij}^2 \underline{\vee} Y_{ij}^3 \underline{\vee} Y_{ij}^4 & i,j \in N, i<j \\
 & It, x_i, y_i \in \mathbb{R}^1, Y_{ij} \in \{True, False\} & i,j \in N, i<j
\end{array}
$$

where x_i and y_i represent the coordinates of the upper-left corner of the rectangles $i \in N$. There is one disjunction for each pair of rectangles (e.g., if there are 4 rectangles there will be 6 disjunctions representing the combinations of rectangles (1,2), (1,3), (1,4), (2,3), (2,4), (3,4)). Each term in a disjunction represents the possible relative placement of the rectangles to ensure that they will not overlap. Specifically, the four terms represent that rectangle i is either to the right or to the left, or above or below rectangle j, respectively.

The example presented in Sawaya and Grossmann[39] consists of 8 rectangles with the dimensions shown in Table 8 and with a width of 10. This example is illustrated in Figure 12.

The solution, size of the problem and tightness of the relaxation as reported in Sawaya and Grossmann[39] is shown in Table 9. The MILP solver used was CPLEX.

It is clear from Table 9 that the size of the problem is considerably larger in (HR) than in (BM). It is also possible to observe that the application of some Basic Steps leads to a major improvement in the lower bound predicted by the continuous relaxation, from **6** in (HR) to **11** after basic steps, which is actually the same value as the optimal solution. For this strong improvement in relaxation, there is only a slight increase in problem size, namely 862 to 932 constraints and 535 to 563 variables. This example also shows the improvement in the solution time after the application of basic steps, due to the better continuous relaxation. It is important to mention that the authors performed a preprocessing to select in which disjunctions to apply the basic steps, and such preprocessing would require some execution time. Finally, although the problem in explicit DNF form represents the convex-hull of the problem[42] (i.e., its continuous relaxed solution will always provide an integral value for the discrete variables), the size of the problem makes it impossible to solve it in less than 3,600 s using the MILP solver CPLEX.

Table 8. Dimensions for 8-Rectangle Strip-Packing Problem

Rectangle	Length	Height
1	4	3
2	3	3
3	2	2
4	2	2
5	3	3
6	3	5
7	4	7
8	4	7

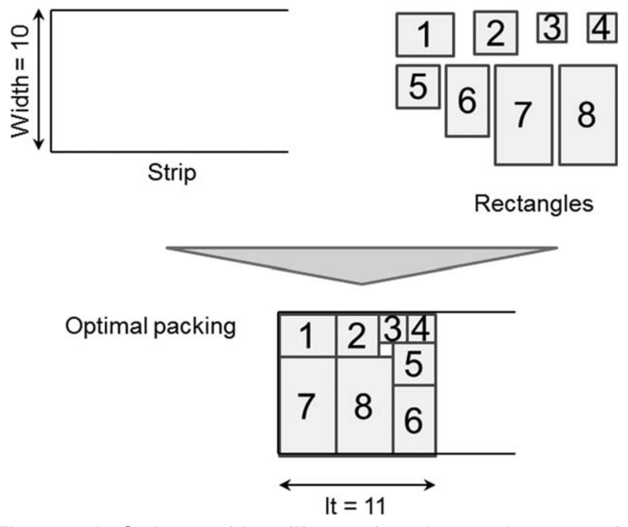


Figure 12. Strip packing illustration for a 13 rectangle problem.

Table 9. Results for 8-Rectangle Strip-Packing Problem

	(BM)	(HR)	(HR) after Basic Steps[a]	(HR) in DNF
Optimal solution	11	11	11	11
Continuous relaxation	**4**	**6**	**11**	**11**
Number of constraints	168	862	932	344,081
Number of variables	123	535	563	147,473
Number of binaries	106	106	106	-
Number of nodes	212,464	2,596	1,464	0
Solution time (s)	13.7	2.5	1.6	>3,600

[a]Selection of disjunctions to apply the basic steps is discussed in source

Table 10. Results for 12-Unit Process Network Problem

	(BM)	(HR)	(HR) after Basic Steps[a]
Optimal solution	−69.5	−69.5	−69.5
Continuous relaxation	**−1,108.9**	**−74.8**	**−69.5**
Number of constraints	114	184	1,462
Number of variables	69	149	807
Number of binaries	12	12	12
Number of nodes	234	8	2
Solution time (s)	27.7	1.0	2.9

[a]Selection of disjunctions to apply the basic steps is discussed in source

Nonlinear convex GDP: Process Network

In general, the underlying goal of a classical process synthesis problem consists in selecting the process that maximizes the profit when selling a product or set of products considering the cost of raw materials and the cost of the process units. The model in the form of the GDP problem involves disjunctions for the selection of units, and propositional logic for the relationship of these units. Each disjunction contains the equation for each unit (these relax as convex inequalities). This example is a particular instance with 12 unit operations as illustrated in Figure 13. A general GDP model is shown in (P12)[43]:

(P12):

$$\min \sum_{i=1}^{12} c_i + \sum_{j=1}^{32} p_j x_j + \gamma$$

st.

$$\sum_{j=1}^{32} r_{jn} x_j \leq 0 \quad \forall n \in N$$

$$\begin{bmatrix} Y_i \\ \sum_{j=1}^{32} d_{ij}\left(e^{x_j/t_{ij}}-1\right) - \sum_{j=1}^{32} S_{ij} x_j \leq 0 \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ x_j = 0; j \in J^i \\ c_i = 0 \end{bmatrix} i=1,\ldots,12$$

$$\Omega(Y) = True$$

$$x_j, c_i \geq 0 \quad i=1,\ldots,12$$

$$Y_i \in \{True,\ False\}$$

The solution, size of the problem and tightness of the relaxation as reported[43] is shown in Table 10. The MINLP solver used was SBB.

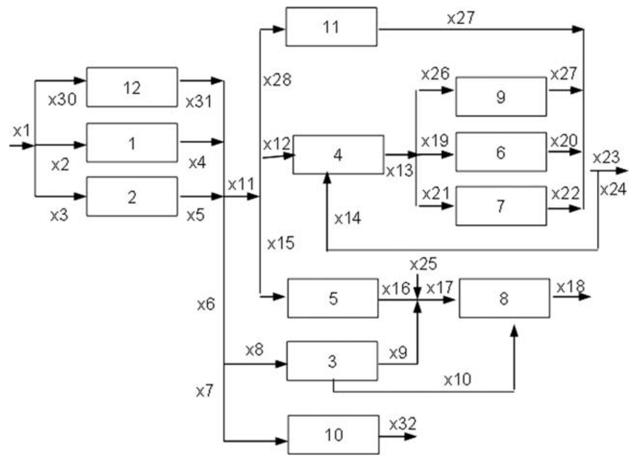


Figure 13. Process network illustration for a 12-unit instance.

As shown in Table 10, it is clear that (HR) has much better relaxation than (BM). In this example, the application of Basic Steps slightly improves the relaxation of (HR), yielding in fact the optimal solution. Furthermore, it also reduces the number of nodes in the MINLP branch and bound method (SBB was used). However, the problem size grows considerably, although the computational time is quite reasonable.

Nonconvex GDP: Heat exchange network (HEN)

This problem consists of finding the heat loads of utilities, the intermediate temperatures and the area of each exchanger that minimizes the investment and operation cost. In this case the investment cost is given by a discontinuous cost function defined over three intervals in terms of the area. Note that the structure of the network is fixed and that we use the arithmetic mean temperature as the driving force. Figure 14 illustrates the HEN structure.

The problem can be formulated as follows[45]:

(P13):

$$MinZ = \sum_i CP_i + FCP_h\left(T_1 - T_{omt,h}\right)C_{cu} + FCP_c\left(T_{out,c} - T_2\right)C_{hu}$$

$s.t.$

$$FCP_h\left(T_{in,h} - T_1\right) = A_1 U_1 \frac{\left(T_{in,h} - T_2\right) + \left(T_1 - T_{in,c}\right)}{2}$$

$$FCP_h\left(T_1 - T_{out,c}\right) = A_2 U_2 \frac{\left(-T_{in,cw} + T_{out,h}\right) + \left(T_1 - T_{out,cw}\right)}{2}$$

$$FCP_c\left(T_{out,c} - T_2\right) = A_3 U_3 \frac{\left(T_{out,s} - T_2\right) + \left(T_{in,s} - T_{out,c}\right)}{2}$$

$$FCP_h\left(T_{in,h} - T_1\right) = FCP_c\left(T_2 - T_{in,c}\right)$$

$$\begin{bmatrix} Y_{1i} \\ CP_i = 2750 A_i^{0.6} + 3000 \\ 0 \leq A_i \leq 10 \end{bmatrix} \vee \begin{bmatrix} Y_{2i} \\ CP_i = 1500 A_i^{0.6} + 15000 \\ 10 \leq A_i \leq 25 \end{bmatrix} \vee \begin{bmatrix} Y_{3i} \\ CP_i = 600 A_i^{0.6} + 46500 \\ 25 \leq A_i \leq 50 \end{bmatrix} \quad i=1,2,3$$

$$T_1^{lo} \leq T_1 \leq T_1^{up}$$

$$T_2^{lo} \leq T_2 \leq T_2^{up}$$

$$Y_{1i} \veebar Y_{2i} \veebar Y_{3i} \qquad i=1,2,3$$

$$Y_{ij} \in \{True,\ False\} \quad i=1,2,3 j=1,2,3$$

$$T_i \in R^1 \quad i=1,2$$

$$A_i \in R^1 \quad i=1,2,3$$

where,

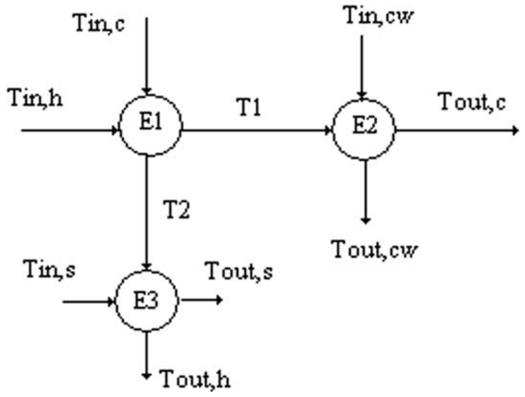


Figure 14. Heat exchange network (HEN) structure.

FCP_i : Heat Capacity of stream i, where i ∈{h,c}

$T_{in,i}$: Inlet temperature conditions of the stream i, where i ∈{h,c}

$T_{out,i}$: Outlet temperature conditions of the stream i, where i ∈{h,c}

U_i : Overall Heat Transfer Coefficient for exchanger E_i, where i ∈{1,2,3}

C_{hu} : Cost of hot utility

C_{cw} : Cost of cooling water

$T_{1,2}$: Intermediate temperatures

A_i : Area in m^2 of heat exchanger E_i, where i ∈{1,2,3}

The parameters for this problem are shown in Table 11.

After replacing in (P13) the bilinear and the concave terms by convex envelopes, Table 12 shows the solution, size of the problem and tightness of the relaxation.[45] Note that the relaxation in this case represents the lower bound of the global minimum (144,385). In this case, the HR after basic steps yields a continuous relaxation of **94,926** vs. **91,671** before basic steps (an increase of 3.5% in the lower bound). This improvement in the lower bound is reflected in the fact that solving the problem after basic steps requires less time than the HR without basic steps.

Table 11. Data for Heat Exchange Example

Heat Exchanger	Overall Heat Transfer (kW/m^2K)
1	1.5
2	0.5
3	1

Stream	FCP(kW/K)	T_{in}(K)	T_{out}(K)	Cost($/kW yr)
Hot	10.0	500	340	
Cold	7.5	350	560	
Cooling Water		300	320	20
Steam		600	600	80

Table 12. Results for HEN Problem

	(HR)	(HR) after Basic Steps[a]
Optimal solution	114,385	114,385
Continuous relaxation	**91,671**	**94,926**
Number of constraints	87	206
Number of variables	52	106
Number of binaries	9	9
Number of nodes	10	1
Solution time (s)	9.0	5.0

[a]Selection of disjunctions to apply the basic steps is discussed in source

Concluding Remarks

This article has given a general overview of generalized disjunctive programming (GDP), a systematic modeling framework that represents discrete-continuous optimization problems in terms of algebraic equations and high-level symbolic logic expressions. Several examples in process systems engineering including optimization of flowsheet superstructures, design of distillation columns, design of batch processes and to scheduling problems have been presented to illustrate how GDP greatly facilitates the modeling of these problems. Furthermore, big-M and hull reformulations have been presented as two major mixed-integer algebraic models that can be systematically derived from a GDP model. The former is smaller in size, but the latter exhibits stronger continuous relaxations, and, hence, stronger lower bounds, which may potentially translate into a more efficient solution times, although this is not always the case due to the increased size of the reformulation.

The concept of basic steps for reformulation in GDP models has also been presented in order to further improve the continuous relaxations of these problems. In the limit where the disjunctions are converted into disjunctive normal form through these basic steps, a perfect mixed-integer formulation can be obtained that corresponds to the convex hull, meaning that the mixed-integer problem can be solved as a continuous optimization problem. The application of basic steps following some basic rules have been illustrated with a number of computational results from several examples in the area of process systems engineering showing that the predicted lower bounds can be significantly improved, including the case of global optimization problems. Finally, it is hoped that this article will help to clarify the scope of GDP as a fundamental theoretical framework for modeling discrete-continuous optimization problems.

Acknowledgment

The authors would like to acknowledge financial support from the National Science Foundation under Grant OCI-0750826. Ignacio Grossmann would also like to acknowledge the work of his former Ph.D. students, Ramesh Raman, Metin Turkay, Sangbum Lee, Nicolas Sawaya, Juan Pablo Ruiz and Hector Yeomans, who have made possible the exciting developments in the area of Generalized Disjunctive Programming. Ignacio Grossmann also would like to acknowledge Professor Egon Balas from Carnegie Mellon for his inspiring and pioneering work in the area of disjunctive programming.

Literature Cited

1. Furman KC, Sahinidis NV. A critical review and annotated bibliography for heat exchanger network synthesis in the 20th century. *Ind Eng Chem Res*. 2002;41(10):2335–2370.
2. Grossmann IE, Caballero JA, Yeomans H. Advances in mathematical programming for automated design, integration and operation of chemical processes. *Kor J Chem Eng*. 1999;16:407–426.
3. Kallrath J. Mixed integer optimization in the chemical process industry: experience, potential and future. *Trans I Chem E*. 2000;78:809–822.
4. Karuppiah R, Grossmann IE. Global optimization for the synthesis of integrated water systems in chemical processes. *Comput Chem Eng*. 2006;30(4):650–673.

5. Méndez CA, Cerdá J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng.* 2006;30:913–946.
6. Grossmann IE, Kravanja Z. Mixed-integer nonlinear programming techniques for process systems engineering. *Comput Chem Eng.* 1998;19(Supplement 1):189–204.
7. Castro P, Grossmann IE. Generalized disjunctive programming as a systematic modeling framework to derive scheduling formulations. *Ind Eng Chem Res.* 2012;51:5781–5792.
8. Ierapetritou MG, Floudas CA. Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Ind Eng Chem Res.* 1998;37(11):4341–4359.
9. Kondili E, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations - I. MILP formulation. *Comput Chem Eng.* 1993;17:211–227.
10. Meńdez CA, Henning GP, Cerdá J. An MILP continuous time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. *Comput Chem Eng.* 2001;25:701–711.
11. Grossmann IE. Advances in mathematical programming models for enterprise-wide optimization. *Comput Chem Eng.* 2012;47:2–18.
12. Hooker JN, Osorio MA. Mixed logical/linear programming. *Discrete Appl Math.* 1999;96-97:395–442.
13. Raman R, Grossmann IE. Modelling and computational techniques for logic-based integer programming. *Comput Chem Eng.* 1994;18:563–578.
14. Lee S, Grossmann IE. Logic-based modeling and solution of nonlinear discrete/continuous optimization problems. *Annals Operat Res.* 2005;139(1):267–288.
15. Türkay M, Grossmann IE. A logic based outer-approximation algorithm for minlp optimization of process flowsheets. *Comput Chem Eng.* 1996;20:959–978.
16. Lee S, Grossmann IE. New algorithms for nonlinear generalized disjunctive programming. *Comput Chem Eng.* 2000;24:2125–2141.
17. Nemhauser GL, Wolsey LA. Integer and Combinatorial Optimization. New York: John Wiley & Sons, Inc; 1988.
18. Grossmann IE. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optimiz Eng.* 2002;3:227–252.
19. Beaumont N. An algorithm for disjunctive programs. *Euro J Operat Res.* 1991;48:362–371.
20. Hooker JN. Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction. New York: John Wiley & Sons, Inc; 2000.
21. Grossmann IE, Ruiz JP. Generalized disjunctive programming: a framework for formulation and alternative algorithms for MINLP optimization. *IMA Vol Math Applications.* 2011;154:93–115.
22. Balas E. Disjunctive programming. *Annals Discrete Math.* 1979;5:3–51.
23. Yeomans H, Grossmann I E. Disjunctive programming models for the optimal design of distillation columns and separation sequences. *Ind Eng Chem Res.* 2000;39(6):1637–1648.
24. Biegler LT, Grossmann IE, Westerberg AW. Systematic Methods of Chemical Process Design. New York: Prentice Hall; 1997.
25. Ruiz JP. Global Optimization of Nonconvex Generalized Disjunctive Programming [Thesis]. Pittsburgh, PA: Carnegie Mellon University; 2006.
26. Sargent RWH, Gaminibandara K. Optimal Design of Plate Distillation Columns. Optimization in Action. Dixon LCW, ed. New York: Academic Press; 1976.
27. Viswanathan J, Grossmann IE. An Alternative MINLP model for finding the number of trays required for a specified separation objective. *Comput Chem Eng.* 1993;17(9):949–955.
28. Grossmann IE, Aguirre PA, Barttfeld M. Optimal synthesis of complex distillation columns using rigorous models. *Comput Chem Eng.* 2005;29:1203–1215.
29. Birewar DB, Grossmann IE. Simultaneous synthesis, sizing, and scheduling of multiproduct batch plants. *Ind Eng Chem Res.* 1990;29(11):2242–2251.
30. Lee S, Grossmann IE. A Global optimization algorithm for nonconvex generalized disjunctive programming and applications to process systems. *Comput Chem Eng.* 2001;25(11):1675–1697.
31. Hart WE, Laird C, Watson JP, Woodruff DL. Pyomo - Optimization Modeling in Python. New York: Springer; 2012.
32. Kallrath J. Algebraic Modeling Systems: Modeling and Solving Real World Optimization Problems. New York: Springer; 2012.
33. Vecchietti A, Grossmann IE. Modeling issues and implementation of language for disjunctive programming. *Comput Chem Eng.* 2000;24:2143–2155.
34. Cavalier TM, Pardalos PM, Soyster AL. Modeling and integer programming techniques applied to propositional calculus. *Comput Operat Res.* 1990;17(6):561–570.
35. Raman R, Grossmann IE. Relation between MILP modeling and logical inference for chemical process synthesis. *Comput Chem Eng.* 1991;15(2):73–84.
36. Williams HP. Model Building in Mathematical Programming. Wiley, 1985.
37. Clocksin WF, Mellish CS. Programming in Prolog. New York: Springer-Verlag; 1981.
38. Sawaya N. Reformulations, Relaxations and Cutting Planes for Generalized Disjunctive Programming [Thesis]. Pittsburg, PA: Carnegie Mellon University; 2006.
39. Sawaya N, Grossmann IE. A hierarchy of relaxations for linear generalized disjunctive programming. *Euro J Operat Res.* 2012;216:70–82.
40. Grossmann IE, Lee S. Generalized convex disjunctive programming: nonlinear convex hull relaxation. *Comput Optimiz Applications.* 2003;26:83–100.
41. Vecchietti A, Lee S, Grossmann IE. Modeling of discrete/continuous optimization problems: characterization and formulation of disjunctions and their relaxations. *Comput Chem Eng.* 2003;27:433–448.
42. Balas E. Disjunctive programming and a hierarchy of relaxations for discrete continuous optimization problems. *SIAM J Algebraic Discrete Methods.* 1985;6(3):466–486.
43. Ruiz JP, Grossmann IE. A hierarchy of relaxations for nonlinear convex generalized disjunctive programming. *Euro J Operat Res.* 2012;218(1):38–47.
44. Trespalacios F, Grossmann IE. An algorithmic approach for improving convex generalized disjunctive programming models through the application of basic steps. Manuscript in preparation. 2013.
45. Ruiz JP, Grossmann IE. Strengthening of lower bounds in the global optimization of bilinear and concave generalized disjunctive programs. *Comput Chem Eng.* 2010;34:914–930.
46. McCormick GP. Computability of global solutions to factorable nonconvex programs. Part I. Convex underestimating problems. *Math Program.* 1976;10:146–175.

APPENDIX: 1

Embedded disjunctions and inclusive OR formulations

(GDP) represents the general form of generalized disjunctive programming. However, there are logic models that are formulated in different forms than (GDP). Two particular cases are embedded disjunctions[33] and inclusive OR arguments. We show here that these can be converted into the form of (GDP).

Embedded Disjunctions.

Embedded disjunctions may arise in PSE problems (see for example P6 for discontinuous cost functions). An embedded disjunction is illustrated in P14.1:

(P14.1):

$$\begin{bmatrix} Y_1 \\ r_1(x) \le 0 \\ \begin{bmatrix} Y_{11} \\ r_{11}(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} Y_{12} \\ r_{12}(x) \le 0 \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ r_2(x) \le 0 \end{bmatrix}$$
$$Y_1 \underline{\vee} Y_2$$
$$Y_1 \iff Y_{11} \underline{\vee} Y_{12}$$

which implies that if $Y_1 = TRUE$, then $r_1(x) \le 0$ must be satisfied, and there is an additional disjunction Y_{11} XOR Y_{21}. If then must be satisfied. To achieve (GDP) form, P14.1 can be transformed as follows:

(P14.2):

$$\begin{bmatrix} Y_1 \\ r_1(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ r_2(x) \le 0 \end{bmatrix}$$
$$\begin{bmatrix} Y_{11} \\ r_{11}(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} Y_{12} \\ r_{12}(x) \le 0 \end{bmatrix} \vee [Y_{13}]$$
$$Y_1 \underline{\vee} Y_2$$
$$Y_{11} \underline{\vee} Y_{12} \underline{\vee} Y_{13}$$
$$Y_1 \iff Y_{11} \underline{\vee} Y_{12}$$

It is easy to see that P14.1 and P14.2 are logically equivalent.

Inclusive OR

Another logic proposition that can arise is when the disjunctions are not necessarily exclusive (i.e., the logic constraint $\veebar_{i\in D_k} Y_{ki}$ is not present in the GDP formulation). P15.1 illustrates an inclusive OR disjunction:

(P15.1):

$$\begin{bmatrix} Y_1 \\ r_1(x)\leq 0 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ r_2(x)\leq 0 \end{bmatrix}$$

Note that Y_1 and Y_2 cannot both be false (i.e., Y_1 can be TRUE and Y_2 FALSE; or Y_1 FALSE and Y_2 TRUE, or Y_1 TRUE and Y_2 TRUE). This means that it is possible to enforce both and To achieve (GDP) form, P15.1 can be formulated as follows:

(P15.2):

$$\begin{gathered}\begin{bmatrix} Y_1 \\ r_1(x)\leq 0 \end{bmatrix} \vee [\neg Y_1] \\ \begin{bmatrix} Y_2 \\ r_2(x)\leq 0 \end{bmatrix} \vee [\neg Y_2] \\ Y_1 \vee Y_2 \end{gathered}$$

P15.1 and P15.2 are logically equivalent and do not require the XOR operator, since they are exclusive in nature.

APPENDIX: 2

Review of GDP methods

Two solution methods have been proposed for direct solution convex nonlinear GDP, namely, the **Branch and Bound** method by Lee and Grossmann,[16] which builds on the concept of disjunctive Branch and Bound method by Beaumont,[19] and the **Logic Based Outer-Approximation** method by Turkay and Grossmann.[15]

The basic idea in the **B&B method** is to directly branch on the constraints corresponding to particular terms in the disjunctions, while considering the convex hull of the remaining disjunctions. Although the tightness of the relaxation at each node is comparable with the one obtained when solving the HR reformulation with a MINLP solver, the size of the problems solved are smaller and the numerical robustness is improved.

For the case of **Logic Based Outer-Approximation** methods, similar to the case of OA for MINLP, the main idea is to solve iteratively a master problem given by a linear GDP, which will provide a lower bound of the solution, and an NLP subproblem that will yield an upper bound. As described in Turkay and Grossmann[15], for fixed values of the Boolean Variables $Y_{k\hat{i}}=true, Y_{ki}=false$ with $\hat{i}\neq i$, the corresponding NLP subproblem SNLP is as follows:

(SNLP):

$$\begin{aligned} & Min\, Z=f(x) \\ & s.t.\;\; g(x)\leq 0 \\ & r_{ki}(x)\leq 0\, for\; Y_{ki}=true\; k\in K, i\in D_k \\ & x^{lo}\leq x\leq x^{up} \\ & x\in R^n, c_k\in R^1, Y_{ki}\in\{True, False\} \end{aligned}$$

It is important to note that only the constraints that belong to the active terms in the disjunction (i.e., associated Boolean variable $Y_{ki}=True$) are imposed. Constraints involved in the inactive disjunctive terms are disregarded. This leads to a substantial reduction in the size of the NLP subproblem compared to the direct application of the traditional OA method on the MINLP reformulation. Assuming that L subproblems are solved in which sets of linearizations $\ell=1,2\ldots L$ are generated for subsets of disjunction terms $L_{ki}=\{\ell | Y_{ki}^{\ell}=True\}$, one can define the following disjunctive OA master problem MLGDP:

(MLGDP):

$$\begin{aligned} & MinZ=\alpha \\ & s.t. \\ & \left.\begin{aligned} & \alpha\geq f(x^\ell)+\nabla f(x^\ell)^T(x-x^\ell) \\ & g(x^\ell)+\nabla g(x^\ell)^T(x-x^\ell)\leq 0 \end{aligned}\right\}\ell=1,2\ldots,L \\ & \underset{i\in D_k}{\vee}\begin{bmatrix} Y_{ki} \\ r_{ki}(x^\ell)+\nabla r_{ki}(x^\ell)(x-x^\ell)\leq 0 \quad \ell\in L_{ki} \end{bmatrix} \quad k\in K \\ & \Omega(Y)=True \\ & x^{lo}\leq x\leq x^{up} \\ & \alpha\in R^1, x\in R^n, c_k\in R^1, Y_{ki}\in\{True, False\} \end{aligned}$$

It should be noted that before applying the aforementioned master problem it is necessary to solve various subproblems (SNLP) for different values of the Boolean variables Y_{ik} so as to produce one linear approximation of each of the terms $i\in D_k$ in the disjunctions $k\in K$. As shown by Turkay and Grossmann[15] selecting the smallest number of subproblems amounts to solving a set covering problem, which is of small size and easy to solve. It is important to note that the number of subproblems solved in the initialization is often small since the combinatorial explosion that one might expect is in general limited by the propositional logic. Moreover, terms in the disjunctions that contain only linear functions need not be considered for generating the subproblems. This frequently arises in process networks since they are often modeled by using two terms disjunctions where one of the terms is always linear. Also, it should be noted that the master problem (MLGDP) can be reformulated as an MILP by using the big-M or Convex Hull reformulation, or else solved directly with a disjunctive branch and bound method.

APPENDIX: 3

Linear GDP-to-MILP: process synthesis network

Consider the process network illustrated in Figure A1, in which only one reactor and at most one separation unit can be selected. The objective is to maximize the profit of selling F_{11} at a price P_{11}. The cost of purchasing is. Each equipment has an associated cost, but unlike (GDP1), it is not constant. It can be described by a linear function representing the investment cost γ_k and the operating cost as a function of the flow coming out of the equipment $C_k=\gamma_k+\alpha_k F_k^{out}$

The problem can be formulated as follows:

(P16.1):

$$\begin{aligned} & \max Profit=P_{11}F_{11}-P_1F_1-C_R-C_S \\ & F_1=F_2+F_4 \\ & F_5=F_6+F_7 \\ & F_{10}=F_8+F_9 \\ & F_{11}=F_3+F_{10} \end{aligned}$$

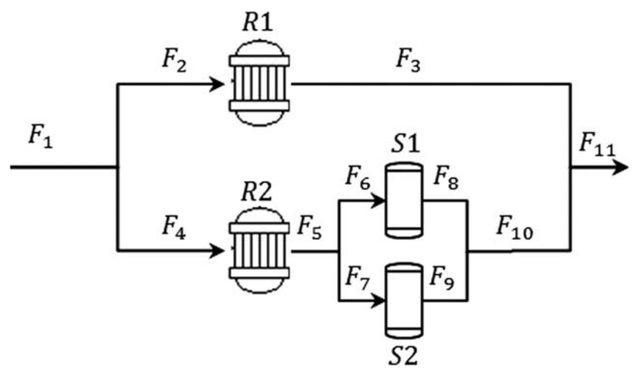


Figure A1. Illustration of a process network problem.

$$\begin{bmatrix} YR_1 \\ F_3=\beta_{R1}F_2 \\ C_S,F_{4,5,6,7,8,9,10}=0 \\ C_R=\gamma_{R1}+\alpha_{R1}F_3 \end{bmatrix} \vee \begin{bmatrix} Y_{R2} \\ F_5=\beta_{R2}F_4 \\ F_{2,3}=0 \\ C_R=\gamma_{R2}+\alpha_{R2}F_5 \\ \begin{bmatrix} Y_{S1} \\ F_8=\beta_{S1}F_6 \\ F_{7,9}=0 \\ C_S=\gamma_{S1}+\alpha_{S1}F_8 \end{bmatrix} \vee \begin{bmatrix} Y_{S2} \\ F_9=\beta_{S2}F_7 \\ F_{6,8}=0 \\ C_S=\gamma_{S2}+\alpha_{S2}F_9 \end{bmatrix} \end{bmatrix}$$

$Y_{R1} \veebar Y_{R2}$

$Y_{R2} \iff Y_{S1} \veebar Y_{S2}$

$0 \le F_i \le F_i^{UP}; 0 \le C_R \le C_R^{UP}; 0 \le C_S \le C_S^{UP};$

$Y_{R1}, Y_{R2}, Y_{S1}, Y_{S2} \in \{True, False\}$

The first step is to reformulate the problem in (GDP) form (see Appendix 1 for embedded disjunctions). Additionally, in order to obtain a smaller reformulation, in 5 we relax the equality constraints into inequalities. In this way, instead of adding two inequalities for each inequality, we only consider one of them. It is easy to show that the problem with inequality relaxations is equivalent to the original problem.

(P16.GDP):

$$\max Profit=P_{11}F_{11}-P_1F_1-C_R-C_S$$

s.t.

$F_1=F_2+F_4$

$F_5=F_6+F_7$

$F_{10}=F_8+F_9$

$F_{11}=F_3+F_{10}$

$$\begin{bmatrix} Y_{R1} \\ F_3 \le \beta_{R1}F_2 \\ F_{4,5} \le 0 \\ C_R \ge \gamma_{R1}+\alpha_{R1}F_3 \end{bmatrix} \vee \begin{bmatrix} Y_{R2} \\ F_5 \le \beta_{R2}F_4 \\ F_{2,3} \le 0 \\ C_R \ge \gamma_{R2}+\alpha_{R2}F_5 \end{bmatrix}$$

$$\begin{bmatrix} Y_{S1} \\ F_8 \le \beta_{S1}F_6 \\ F_{7,9} \le 0 \\ C_S \ge \gamma_{S1}+\alpha_{S1}F_8 \end{bmatrix} \vee \begin{bmatrix} Y_{S2} \\ F_9 \le \beta_{S2}F_7 \\ F_{6,8} \le 0 \\ C_S \ge \gamma_{S2}+\alpha_{S2}F_9 \end{bmatrix} \vee \begin{bmatrix} Y_{not_{R2}} \\ C_S,F_{6,7,8,9,10} \le 0 \end{bmatrix}$$

$Y_{R1} \veebar Y_{R2}$

$Y_{S1} \veebar Y_{S2} \veebar Y_{not_{R2}}$

$Y_{R2} \iff Y_{S1} \veebar Y_{S2}$

$0 \le F_i \le F_i^{UP}; 0 \le C_R \le C_R^{UP}; 0 \le C_S \le C_S^{UP}; Y_{R1}, Y_{R2},$

$Y_{not_{R2}}, Y_{S1}, Y_{S2} \in \{True, False\}$

With the procedure described for the logic propositions and the (BM) or (HR) transformation to the disjunctions, the two MILP reformulations are as follows:

(P16.BM):

$$\max Profit=P_{11}F_{11}-P_1F_1-C_R-C_S$$

$s.t.$

$F_1=F_2+F_4$

$F_5=F_6+F_7$

$F_{10}=F_8+F_9$

$F_{11}=F_3+F_{10}$

$F_3 \le \beta_{R1}F_2+M^{R1}(1-y_{R1})$

$F_{4,5} \le M^{R1}(1-y_{R1})$

$C_R \ge \gamma_{R1}+\alpha_{R_1}F_3-M^{R1}(1-y_{R1})$

$F_5 \le \beta_{R2}F_4+M^{R2}(1-y_{R2})$

$F_{2,3} \le M^{R2}(1-y_{R2})$

$C_R \ge \gamma_{R2}+\alpha_{R2}F_5-M^{R2}(1-y_{R2})$

$F_8 \le \beta_{S1}F_6+M^{S1}(1-y_{S1})$

$F_{7,9} \le M^{S1}(1-y_{S1})$

$C_S \ge \gamma_{S1}+\alpha_{S1}F_8-M^{S1}(1-y_{S1})$

$F_9 \le \beta_{S2}F_7+M^{S2}(1-y_{S2})$

$F_{6,8} \le M^{S2}(1-y_{S2})$

$C_S \ge \gamma_{S2}+\alpha_{S2}F_9-M^{S2}(1-y_{S2})$

$C_S,F_{6,7,8,9,10} \le M^{not_{R2}}(1-y_{not_{R2}})$

$y_{R1}+y_{R2}=1$

$y_{S1}+y_{S2}+y_{not_{R2}}=1$

$y_{R1}=y_{not_{R2}}$

$y_{R2}=y_{S1}+y_{S2}$

$0 \le F_i \le F_i^{UP} \quad i=1,\dots 11$

$0 \le C_R \le C_{RUP}$

$0 \le C_S \le C_S^{UP}$

$y_k \in \{0,1\}; k \in \{R1,R2,S1,S2,not_{R2}\}$

(P16.HR):

$$\max Profit=P_{11}F_{11}-P_1F_1-C_R-C_S$$

$s.t.$

$F_1=F_2+F_4$

$F_5=F_6+F_7$

$F_{10}=F_8+F_9$

$F_{11}=F_3+F_{10}$

$F_i=F_i^{R1}+F_i^{R2} \quad i=1,\dots,11$

$F_i=F_i^{S1}+F_i^{S2}+F_i^{not_{R2}} \quad i=1,\dots,11$

$$C_R = C_R^{R1} + C_R^{R2}$$
$$C_S = C_S^{S1} + C_S^{S2} + C_S^{not_{R2}}$$
$$F_3^{R1} \leq \beta_{R1} F_2^{R1}$$
$$F_{4,5}^{R1} \leq 0$$
$$C_R^{R1} \geq \gamma_{R1} y_{R1} + \alpha_{R1} F_3^{R1}$$
$$F_5^{R2} \leq \beta_{R2} F_4^{R2}$$
$$F_{2,3}^{R2} \leq 0$$
$$C_R^{R2} \geq \gamma_{R2} y_{R2} + \alpha_{R2} F_5^{R2}$$
$$F_8^{S1} \leq \beta_{S1} F_6^{S1}$$
$$F_{7,9}^{S1} \leq 0$$
$$C_S^{S1} \geq \gamma_{S1} y_{S1} + \alpha_{S1} F_8^{S1}$$
$$F_9^{S2} \leq \beta_{S2} F_7^{S2}$$
$$F_{6,8}^{S2} \leq 0$$
$$C_S^{S2} \geq \gamma_{S2} y_{S2} + \alpha_{S2} F_9^{S2}$$
$$C_S^{not_{R2}}, F_{6,7,8,9,10}^{not_{R2}} \leq 0$$
$$y_{R1} + y_{R2} = 1$$
$$y_{S1} + y_{S2} + y_{not_{R2}} = 1$$
$$y_{R1} = y_{not_{R2}}$$
$$y_{R2} = y_{S1} + y_{S2}$$
$$0 \leq F_i^k \leq F_i^{UP} y_k \qquad i = 1, \ldots, 11;$$
$$0 \leq C_R^k \leq C_R^{UP} y_k \quad k \in \{R1.R2, S1, S2, not_{R2}\}$$
$$0 \leq C_S^k \leq C_S^{UP} y_k$$
$$y_k \in \{0, 1\}; k \in \{R1, R2, S1, S2, not_{R2}\}$$

As can be seen, these two alternative formulations have been derived systematically from the P16.GDP. Thus, we have avoided a direct *ad hoc* formulation of this problem in algebraic form as an MILP which may or may not correspond to a "good" formulation. Having the two models P16.BM and P16.HR we have a better understanding and insight of what we can expect from each formulation, although clearly the numerical performance will depend on the actual parameter of the given instance.

Manuscript received Oct. 24, 2012, and revision received Feb. 6, 2013.