Pergamon

0098-1354(95)00219-7

Computers chem. Engng Vol. 20, No. 8, pp. 959–978, 1996

0098-1354/96 $15.00 + 0.00

LOGIC-BASED MINLP ALGORITHMS FOR THE OPTIMAL SYNTHESIS OF PROCESS NETWORKS

METIN TÜRKAY and IGNACIO E. GROSSMANN†
Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.

(Received 1 December 1994; final revision received 22 June 1995)

Abstract—In this paper, the MINLP problem for the optimal synthesis of process networks is modeled as a discrete optimization problem involving logic disjunctions with nonlinear equations and pure logic relations. The logic disjunctions allow the conditional modeling of equations (e.g. if a unit is selected, apply mass/heat balances; otherwise, set the flow variables to zero). It is first shown that this framework for representing discrete optimization problems greatly simplifies the step of modeling. The outer approximation algorithm is then used as a basis to derive a new logic-based OA solution method which naturally gives rise to NLP sub-problems that avoid zero flows and a disjunctive LP master problem. The initial NLP sub-problems, that provide linearizations for all the terms in the disjunctions, are selected through a set-covering problem for which we consider both the cases of disjunctive and conjunctive normal form logic. The master problem, on the other hand, is converted to mixed-integer form using a convex-hull representation. Furthermore, based on some interesting relations of outer approximation with generalized Benders decomposition, it is also shown that it is possible to derive a logic-based method for the latter algorithm. The proposed algorithm has been tested on several structural optimization problems, including a flowsheet example showing distinct advantages in robustness and computational efficiency when compared to standard MINLP models and algorithms.

INTRODUCTION

Optimization methods for the synthesis and design of process networks involve mathematical models that deal with discrete and continuous variables (Grossmann, 1990). Discrete variables play an important role in the synthesis of process systems. These are generally restricted to binary variables with "1" indicating the selection of a unit or assignment of a task, "0" indicating the absence of a unit or a task. Simultaneous optimization of discrete and continuous variables is necessary for a process network problem which is nonlinear in nature. Therefore, the problem is formulated as a mixed-integer nonlinear programming (MINLP) problem.

During the last few years, there has been a significant increase in the development of MINLP models for synthesizing complete processes or sub-systems (see Grossmann and Daichendt, 1994; Grossmann and Kravanja, 1995) and solving scheduling problems for batch and continuous processes (Reklaitis, 1991). The efficiency in solving these problems highly depends on the problem structure. While certain models with a large number of variables and constraints can be solved to optimality with modest computational effort, some models with a small number of variables and constraints might be very hard to get even a feasible solution (Nemhauser and Wolsey, 1988). Modeling discrete/continuous optimization problems is also very user dependent; the same problem might be formulated quite differently from one designer to another. With current MINLP techniques, there is also the computational difficulty that each and every constraint must be solved even if units "disappear" from a superstructure. It would be clearly desirable to eliminate constraints of non-existing tasks (or process units), not only to reduce the dimensionality of the problem, but also to eliminate the effect of singularities and non-convex constraints.

One of the most common methods for solving MINLP problems is the NLP-based branch and bound algorithm (Beale, 1977; Nabar and Schrage, 1990), which is very similar to the LP-based branch and bound for MILP problems. Another method is the generalized Benders decomposition (GBD) (Geoffrion, 1972) which involves the projection of continuous variables onto the space of binary variables. The algorithm consists of solving pseudo-integer MILP master problems and NLP sub-problems for fixed 0–1 variables. The outer-approximation (OA) method (Duran and Grossmann, 1986; Fletcher and Leyffer, 1994) is similar to GBD. However, the MILP master problem is defined in the full space of discrete and continuous variables, and thus predicts stronger bounds than Benders decomposition. A heuristic extension of the OA algorithm to deal with non-

† To whom all correspondence should be addressed.

convexities in MINLP problems was proposed by Viswanathan and Grossmann (1990). This algorithm includes an exact penalty function to allow violation of the linearizations of non-convex constraints through slack variables. The approach has proved successful for finding near-global optimal solutions, but it may also fail because the method is not rigorous for finding global optima. Other solution methods for MINLP problems include the extended cutting plane method by Westerlund and Pettersson (1992) and the feasibility search method by Mawekwang and Murtagh (1986).

The OA algorithm can be used to solve MINLP problems with linear 0–1 variables and nonlinear continuous variables. However, as the size of the optimization problem increases, the increase in the size of master and sub-problems that must be solved becomes a major computational bottleneck. Furthermore, singularities due to linearizations at zero flows and non-convexities often cut off the global optimal solution. Therefore, it is necessary to improve the solution strategy for large-scale flowsheet optimization problems. The difficulty of large numbers of 0–1 variables has been addressed by Quesada and Grossmann (1992) who proposed an LP/NLP based branch and bound method. However, convexity was assumed in this work. Knowledge of the topology of the process network and the logic relations among the units provides important information about the set of alternative feasible solutions, which can be used to reduce the combinatorial complexity of the problem. Raman and Grossmann (1993) showed that it is possible to use logic relationships between the process network units to achieve order of magnitude reductions in the computational effort for linear problems. A modeling/decomposition strategy was proposed by Kocis and Grossmann (1989) to address the difficulty of singularities arising due to zero flows in nonlinear flowsheet synthesis problems. These authors showed that the modeling strategy could eliminate some non-convexities in the interconnection nodes through linear constraints and outer approximations, and the decomposition strategy required solution of NLP sub-problems with existing units only. The modeling/decomposition strategy has been implemented in a computer program, PROSYN–MINLP, for the structural optimization of process flowsheets (Kravanja and Grossmann, 1990; 1994).

Motivated by the potential of using logic to improve the modeling and solution of network synthesis problems, two new logic-based MINLP algorithms are proposed in this paper. These are based on modeling synthesis problems with the generalized disjunctive modeling framework outlined in Raman and Grossmann (1994). These authors addressed the solution of the linear case, while this paper is aimed at the solution of nonlinear problems. The proposed logic-based MINLP algorithms correspond to the outer approximation method by Duran and Grossmann (1986) and Benders decomposition (Benders, 1962) as applied to the generalized disjunctive programming model. The proposed algorithms have the important property of avoiding solution of the NLP sub-problems with zero flows (see Kocis and Grossmann, 1989). Furthermore, the proposed methods provide a formalization of the ideas behind the modeling/decomposition strategy. Several interesting theoretical properties of the methods are presented, as well as numerical results of several example problems.

BACKGROUND

Raman and Grossmann (1994) recently proposed a new modeling framework for the discrete optimization of superstructures of linear process network problems. A logic-based representation is used, in which the mixed-integer logic is represented through disjunctions. The generalized disjunctive programming model is given as follows:

$$\min Z = \sum_i c_i + f(x)$$

s.t.

$$g(x) \leq 0$$

$$\begin{bmatrix} Y_i \\ h_i(x) \leq 0 \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ B^i x = 0 \\ c_i = 0 \end{bmatrix} \quad i \in D \qquad \text{(P)}$$

$$\Omega(Y) = \text{True}$$

$$x \in R^n, \quad c \geq 0, \quad Y \in \{\text{True, False}\}^m.$$

The nonlinear model (P) involves the following three types of variables: x and c_i are the continuous variables (the former correspond to flows, pressures, temperatures, alike the latter are used to exclusively represent fixed charges); the Boolean variables, Y_i, that are associated with the existence of units and are used to indicate whether a given disjunction i is true or false. It should be noted that there are no binary variables involved in the model (P) (see Appendix A for an example of a logic-based MINLP model). In the generalized disjunctive nonlinear optimization model (P), the objective function and the first set of constraints may involve linear or nonlinear functions. The first set of constraints represent global inequalities that hold irrespective of the discrete choices. The set of disjunctions, D, apply for the processing units. If a process

unit exists (Y_i = True), then the equations and constraints describing that unit are enforced and a fixed charge is applied; otherwise ($\neg Y_i$ = False) a subset of continuous variables and the fixed charge are set to zero. We define $B^i = [b_j^T]$, such that $b_j^T = e_j^T$ if $x_j = 0$, and $b_j^T = 0^T$ if $x_j \neq 0$. In this way, only a subset of the variables x is forced to zero (typically flows). It should be noted that in certain cases the disjunctions may involve more than two terms. In this case, the set of disjunctions will have the following form:

$$\bigvee_{j \in SD_i} \begin{bmatrix} Y_{ij} \\ d_{ij}(x) \leq 0 \\ c_{ij} = \gamma_{ij} \end{bmatrix} \quad i \in \hat{D}. \tag{1}$$

For the sake of simplicity in the presentation, however, we do not consider explicitly the above disjunctions in problem (P). The logical relationships among the Boolean variables describing connections and interactions between the units in the superstructure are given in the form of propositional logic, $\Omega(Y)$, which can be given in either of two forms (see Raman and Grossmann, 1993).

(a) Conjunctive normal form (CNF):

$$\Omega_C = \left[\bigvee_{i \in P_1} Y_i \bigvee_{i \in \bar{P}_1} \neg Y_i\right] \wedge \left[\bigvee_{i \in P_2} Y_i \bigvee_{i \in \bar{P}_2} \neg Y_i\right] \wedge \ldots \wedge \left[\bigvee_{i \in P_S} Y_i \bigvee_{i \in \bar{P}_S} \neg Y_i\right]. \tag{2}$$

(b) Disjunctive normal form (DNF):

$$\Omega_D = \left[\bigwedge_{i \in Q_1} Y_i \bigwedge_{i \in \bar{Q}_1} \neg Y_i\right] \vee \left[\bigwedge_{i \in Q_2} Y_i \bigwedge_{i \in \bar{Q}_2} \neg Y_i\right] \vee \ldots \vee \left[\bigwedge_{i \in Q_R} Y_i \bigwedge_{i \in \bar{Q}_R} \neg Y_i\right]. \tag{3}$$

The CNF logic describes the relationships between the process units (or tasks) in the superstructure and every clause in (2) must be satisfied for the problem. On the other hand, each term in DNF logic describes a feasible configuration within the superstructure and only one of the terms has to be satisfied. The total number of terms in DNF logic is equal to the total number of alternative configurations for the superstructure and therefore corresponds to the maximum number of structurally feasible sub-problems. The DNF formulation can be used for problems requiring relatively few feasible configurations, while the CNF formulation is used for problems for which the total number of feasible configurations can be large.

Problem (P) can be transformed into an algebraic MINLP problem by replacing the Boolean variables Y_i by 0–1 variables y_i and by representing the disjunctions using the "big-M" constraints:

$$h_i(x) \leq M_i(1 - y_i)$$
$$x \leq x^U y_i, \; c_i = \gamma_i y_i \tag{4}$$

where x^U is an upper bound and M_i is a sufficiently large constant. Constraint (4), however, often yields poor relaxations. For the case of linear models, it was shown by Raman and Grossman (1994) that it is possible to formally characterize those disjunctions for which their mixed-integer equations yield the same convex hull and logic information (these type of disjunctions are called "well-behaved" or "w-MIP representable"). The authors used these ideas to develop a branch and bound method for mixed-integer linear programming problems in which disjunctions are explicitly treated and propositional logic is also integrated symbolically. In this paper we concentrate instead on the development of algorithms that can explicitly solve nonlinear versions of problem (P). The following section provides a specific example.

Example 1

Consider the optimization of a process network with fixed charges in Fig. 1 (Kocis and Grossmann, 1989). The raw material A can be processed in either processes 2 or 3 to produce B which is required for the production of C. Alternatively, B can be purchased from the market eliminating processes 2 and 3 if it is profitable. If any unit is selected, a fixed cost has to be paid for that selection. The objective function includes the operating costs and the revenue from the sales of product C. The superstructure involves three units and the Boolean variables, Y_1, Y_2 and Y_3 represent the existence of units in the flowsheet. Design specification of the problem requires that processes 2 and 3 cannot appear together in a feasible flowsheet. The project is canceled when the production of C from any feasible flowsheet is not profitable. The continuous variables x_1–x_8 represent the amount of material flow. The relationships between the units in the superstructure are established through logic propositions of the Boolean variables.

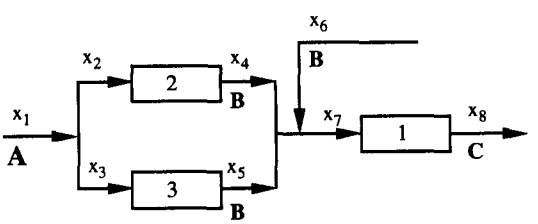

Fig. 1. Superstructure for Example 1.

The optimization problem can be expressed in the generalized disjunctive modeling framework (P) as follows:

(i) objective function:

$$\min Z = c_1 + c_2 + c_3 + x_4 + 1.8x_1 + 1.2x_5 + 7x_6 - 11x_8 \quad (5)$$

subject to

(ii) mass balances for mixer/splitters:

$$x_1 - x_2 - x_3 = 0$$
$$x_7 - x_4 - x_5 - x_6 = 0 \quad (6)$$

(iii) upper bound on the material flows and production:

$$x_5 \leq 5$$
$$x_8 \leq 1 \quad (7)$$

(iv) disjunction for each unit:

$$\text{Unit 1:} \quad \begin{bmatrix} Y_1 \\ x_8 = 0.9x_7 \\ c_1 = 3.5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_7 = x_8 = 0 \\ c_1 = 0 \end{bmatrix} \quad (8)$$

$$\text{Unit 2:} \quad \begin{bmatrix} Y_2 \\ x_4 = \ln(1 + x_2) \\ c_2 = 1 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_2 = x_4 = 0 \\ c_2 = 0 \end{bmatrix} \quad (9)$$

$$\text{Unit 3:} \quad \begin{bmatrix} Y_3 \\ x_5 = 1.2\ln(1 + x_3) \\ c_3 = 1.5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_3 = x_5 = 0 \\ c_3 = 0 \end{bmatrix} \quad (10)$$

(v) design specification and logic propositions:

$$Y_1 \Rightarrow Y_2 \vee Y_3 \vee (\neg Y_2 \wedge \neg Y_3)$$
$$Y_2 \Rightarrow Y_1 \quad (11)$$
$$Y_3 \Rightarrow Y_1$$
$$\neg Y_2 \vee \neg Y_3$$

(vi) variables:

$$c_1, c_2, c_3 \geq 0, x_i \geq 0 \quad i = 1, \ldots, 8$$
$$Y_j \in \{\text{True, False}\} \quad j = 1, 2, 3. \quad (12)$$

LOGIC-BASED OUTER APPROXIMATION

The original outer approximation algorithm by Duran and Grossmann (1986) requires the solution of NLP sub-problems, which are obtained by fixing the binary variables, and of MILP master problems that provide lower bounds and new values for the integer variables. The NLP sub-problems and the resulting MILP master problems are solved until the bounds for those two problems converge. From problem (P), the NLP sub-problem for the fixed choice of the Boolean variables, Y_i^l, is as follows:

$$\min Z_U = \sum_i c_i + f(x)$$

s.t.

$$g(x) \leq 0$$
$$\left.\begin{matrix} h_i(x) \leq 0 \\ c_i = \gamma_i \end{matrix}\right\} \text{for } Y_i^l = \text{True}$$
$$\left.\begin{matrix} B^i x = 0 \\ c_i = 0 \end{matrix}\right\} \text{for } Y_i^l = \text{False} \quad (S^l)$$
$$x \in R^n, c \geq 0$$

where Z_U corresponds to an upper bound for the objective function of problem (P). Note that problem (S^l) requires only solution of the constraints that belong to those disjunctions in which their corresponding Boolean variable is True. The sub-problem (S^l) avoids the solution of the NLP for the entire superstructure since a subset of continuous variables for the non-existing units are set equal to zero and removed from the model. This has the advantage of not only reducing the dimensionality, but also of avoiding zero flows which can lead to singularities and eliminating non-convexities of the non-existing units (Kocis and Grossmann, 1989). The master problem corresponding to problem (P) can be derived by using a similar reasoning as in Duran and Grossmann (1986). Given L major iterations, the linearization set $K_L^i = \{l \mid Y_i^l = \text{True}, l = 1, \ldots, L\}$ can be defined for linearizations generated for disjunctions i in which their corresponding Boolean variable is True. Nonlinear terms in the objective and in the global inequalities are linearized at the solution points determined from the corresponding subproblems (S^l), $l = 1, \ldots, L$. The master problem is given by the following disjunctive linear program:

$$\min Z_L = \alpha_{oa} + \sum_i c_i$$

s.t.

$$\left.\begin{matrix} \alpha_{oa} \geq f(x^l) + \nabla f(x^l)^T (x - x^l) \\ g(x^l) + \nabla g(x^l)^T (x - x^l) \leq 0 \end{matrix}\right\} l = 1, \ldots, L$$

$$\begin{bmatrix} Y_i \\ h_i(x^l) + \nabla h_i(x^l)^T (x - x^l) \leq 0 \quad l \in K_L^i \\ c_i = \gamma_i \end{bmatrix}$$

$$\bigvee \begin{bmatrix} \neg Y_i \\ B^i x = 0 \\ c_i = 0 \end{bmatrix} i \in D \qquad (M_{OA}^L)$$

$$\Omega(Y) = \text{True}$$

$$x \in R^n, c \geq 0, Y \in \{\text{True}, \text{False}\}^m$$

where Z_L corresponds to the lower bound for the objective function for the problem (P) if $f(x)$, $g(x)$ and $h(x)$ are convex functions. The important feature of the master problem (M_{OA}^L) is that it preserves the structure of the original problem (P).

Rather than solving problem (M_{OA}^L) as a disjunctive problem (e.g. see Beaumont, 1990) we will consider its solution as a mixed-integer linear program. To avoid using "big-M" constraints and in order to obtain a tight or "sharp" formulation, the convex-hull representation of the linear disjunction by Balas (1985) is used as shown in Appendix B. The advantage of the convex-hull representation of the above disjunctive program is that the formulation is generally much tighter than the formulation with the "big-M" constraints in (4). A detailed discussion on the treatment of convex-hull representation of disjunctions can be found in Raman and Grossmann (1994).

The treatment of disjunctions with more than two terms [see equation (1)] within the framework of convex-hull representation is illustrates with a single choice splitter in Appendix C. Also, we convert the logic relations into integer inequalities, $Ay \leq a$, which are generally redundant and do not alter the optimal solution. However, they are useful in expediting the search for the optimum. An example for converting the logic propositions into integer inequalities is illustrated in Table 1 for example 1. By replacing the Boolean variables by 0–1 variables and applying the equations for the convex-hull representation (see Appendix B), the proposed MILP formulation of the disjunctive master problem (M_{OA}^L) is as follows:

$$\min Z_L = \alpha_{oa} + \sum_i \gamma_i y_i$$

s.t.

Table 1. CNF logic expressions for Example 1

Logic proposition	Clauses	Constraint
$Y_1 \Rightarrow Y_2 \vee Y_3 \vee (\neg Y_2 \wedge \neg Y_3)$	$(\neg Y_1 \vee Y_2 \vee \neg Y_2 \vee Y_3)$	$y_1 - y_3 \leq 1$
	$(\neg Y_1 \vee Y_3 \vee \neg Y_3 \vee Y_2)$	$y_1 - y_2 \leq 1$
$Y_2 \Rightarrow Y_1$	$\neg Y_2 \vee Y_1$	$y_1 - y_2 \geq 0$
$Y_3 \Rightarrow Y_1$	$\neg Y_3 \vee Y_1$	$y_1 - y_3 \geq 0$
Specification	$\neg Y_2 \vee \neg Y_3$	$y_2 + y_3 \leq 1$

$$\left.\begin{aligned} \alpha_{oa} &\geq f(x^l) + \nabla f(x^l)^T (x - x^l) \\ g(x^l) &+ \nabla g(x^l)^T (x - x^l) \leq 0 \end{aligned}\right\} l = 1, \ldots, L$$

$$\nabla h_i(x^l)^T x \leq [-h_i(x^l) + \nabla h_i(x^l)^T x^l] y_i \quad l \in K_L^i, \quad i \in D \qquad (\hat{M}_{OA}^L)$$

$$Ay \leq a$$

$$\alpha_{oa} \in R^1, x \in R^n, y \in \{0, 1\}^m.$$

It is interesting to note that the linearization of functions of the disjunctions indicates the use of the binary variables y_i to force the variables to zero if a unit does not exist. Kocis and Grossmann (1989) used a similar scheme but gave no theoretical justification for it. The master problem ($\hat{M}_{OA}^L$) can be solved directly as an MILP or with the branch and bound method proposed by Raman and Grossmann (1993) if the logic relations are handled symbolically. For non-convex NLP problems, the augmented penalty MILP master problem proposed by Viswanathan and Grossmann (1990) can be used. This method involves the addition of slacks into the constraints and the introduction of a penalty function in the master problem to reduce the problem of cutting off the continuous feasible region.

ALGORITHM

Application of the logic-based OA algorithm requires successive solution of NLP sub-problems (S^l) and disjunctive master problems (M_{OA}^L) which in this work will be solved with the MILP problem ($\hat{M}_{OA}^L$). A major feature of the proposed method is that the NLP sub-problems (S^l) are defined and solved for the existing units only.

An interesting question that arises is how many NLP subproblems (S^l) should be solved in order to define linearizations for all the terms in the master problem ($\hat{M}_{OA}^L$). This task can be accomplished with a set-covering problem which determines the minimum number of substructures providing outer approximations for all the units. This method provides an alternative procedure to the modeling/decomposition strategy by Kocis and Grossmann (1989) which consists of decomposing the flowsheet into an initial flowsheet and disjoint sub-systems which are optimized by a Lagrangian method. It can generally be expected that the quality of the outer approximations with the proposed method will be better, since they are evaluated at the optimal conditions of a feasible structure.

The logic propositions describing the topology of the superstructure of the problem can be used to determine the initial and structurally feasible NLP problems for generating all the appropriate outer approximations. The superstructure of a network

can be described in either CNF or DNF logic [see (2) and (3) respectively]. If the topology of the flowsheet is described in DNF logic, the potential flowsheet structures are specified as follows:

$$\Omega_D = \bigvee_{r=1}^{R} \left[\bigwedge_{i \in I_p^r} Y_i \bigwedge_{i \in I_n^r} \neg Y_i \right] \tag{13}$$

where R is the number of feasible designs and I_p^r and I_n^r are the corresponding subsets of non-negated and negated Boolean variables ($I = I_p^r \cup I_n^r$) that define the existing and non-existing units, respectively.

The problem of identifying the fewest number of flowsheet structures that are to be optimized in order to generate at least one linearization for each disjunction can be formulated as the following set-covering problem:

$$\min NF = \sum_{r=1}^{R} w_r \tag{14}$$

s.t.

$$\sum_{r=1}^{R} a_{ri} w_r \geq 1 \quad \forall i \in I$$

where

$$a_{ri} = \begin{cases} 1 \text{ if unit } i \text{ is present in alternative } r\ (Y_i) \\ 0 \text{ else } (\neg Y_i) \end{cases}$$

$$w_r = \begin{cases} 1 \text{ if alternative } r \text{ is selected among DNF} \\ 0 \text{ else} \end{cases}$$

The NF sub-problems (S^r) are then defined by the following combinations of Boolean variables:

$$Y_i^r = \text{True } i \in I_p^r,$$

$$Y_i^r = \text{False } i \in I_n^r \quad \text{for } w_r = 1, r = 1, \ldots, R. \tag{15}$$

The NF sub-problems include every unit at least once, but typically a number of them will appear more than once. It should be noted that generally the number NF of initial NLP sub-problems that must be solved is considerably smaller than the total number of alternative flowsheets (e.g. see Example 3).

Appendix D describes a modified version of the set covering algorithm which can handle the propositional logic given in CNF form as in (2). This form is especially useful for large problems because it does not involve specifying all the feasible configurations as is the case in the DNF form (3). Instead, only the logic relations between the units need to be postulated.

Assuming feasible NLP sub-problems, the steps of the proposed logic-based outer-approximation method are as follows.

Step 1: Model the MINLP problem in generalized disjunctive form as in (P).

Step 2: Identify the NF sub-problems to be solved from the set covering problems (for DNF expression of propositional logic, solve the problem (14) and for CNF expression of propositional logic, apply the procedure in Appendix D).

Step 3: Solve NLP sub-problems (S^l) for the NF sub-structures determined in Step 2. The lowest cost solution of these NLPs gives an upper bound, Z_U, for the problem.

Step 4: Linearize the objective function and constraints of the current NLP sub-problem(s) and set up the MILP master problem ($\hat{M}_{OA}^L$). The solution of this problem gives the lower bound, Z_L, for the problem.

Step 5: Compare the current upper bound Z_U with the lower bound Z_L. If $|Z_U - Z_L| \leq \varepsilon$, where ε is a tolerance, then stop. The solution with the current Z_U is the optimal solution. Otherwise go to Step 6.

Step 6: Solve NLP sub-problem (S^l) by fixing the Boolean variables predicted by the master problem. The objective function value of the solution is Z_{NLP}. If $Z_{NLP} < Z_U$, set $Z_U = Z_{NLP}$.

Step 7: Compare the upper bound Z_U with the lower bound Z_L. If $|Z_U - Z_L| \leq \varepsilon$ then stop, the solution with the current Z_U is the optimal solution. Otherwise go to Step 4.

LOGIC-BASED BENDERS DECOMPOSITION

The original generalized Benders decomposition by Geoffrion (1972) involves the solution of NLP sub-problems, and pseudo-integer master problems which correspond to the projection of the mixed-integer space of OA onto the space of binary variables only (see Quesada and Grossmann, 1992). Since the logic-based modeling framework in (P) involves Boolean variables and not binary variables in the problem formulation, it is not possible to directly derive a logic-based version of GBD. More specifically, since there are no binary variables involved in model (P), it is not possible to project the continuous space of the disjunctions onto the space of binary variables to derive Lagrangian cuts. Alternatively, the projection of disjunctions to the space of Boolean variables is unclear. What we can do, however, is to use the MILP master problem ($\hat{M}_{OA}^L$) for the logic-based outer-approximation method as a basis for deriving a logic-based generalized Benders decomposition. In particular, the idea

is to exploit the following equivalence between solving the MILP master problem of the outer-approximation method by using one Benders iteration and solving the master problem of the GBD method.

Theorem 1. Given is the following MINLP problem (MIP):

$$\begin{aligned} &\min Z = c^T y + \phi(x) \\ &\text{s.t.} \\ &\psi_j(x) + b_j^T y \leq 0 \quad j \in J \\ &Ay \leq a \\ &x \in X, y \in \{0,1\}^n \end{aligned} \tag{MIP}$$

in which the functions $\phi(x)$ and $\psi_j(x)$ are convex and differentiable.

Solving the MILP master problem of the outer-approximation method by one Benders iteration yields a solution that is equivalent to the solution of the master problem of generalized Benders decomposition.

Proof. See Appendix E.

Following the proof of Theorem 1, we can perform one Benders iteration on the MILP problem ($\hat{M}_{OA}^{L}$) to obtain a "Benders-like" prediction for the 0–1 variables. The LP sub-problem for fixed 0–1 variables, y_i^k, which are the same as the Boolean variables, Y_i^k, that were fixed for the corresponding NLP sub-problem $k = 1, \ldots, L$, is given as follows:

$$\min Z_B^k = \alpha_{oa} + \sum_i \gamma_i y_i^k$$

s.t.

$$\left.\begin{aligned} &\alpha_{oa} \geq f(x^l) + \nabla f(x^l)^T (x - x^l) \\ &g(x^l) + \nabla g(x^l)^T (x - x^l) \leq 0 \end{aligned}\right\} l = 1, \ldots, L \tag{LM^k}$$

$$\nabla h_i(x^l)^T x \leq [-h_i(x^l) + \nabla h_i(x^l)^T x^l] y_i^k \quad l \in K_L^i, i \in D$$

$$\alpha_{oa} \in R^1, x \in R^n.$$

Note that at the initial phase of the algorithm in which NF NLP sub-problems (S^l) are first solved, the above LP sub-problem can be solved for each of the initial 0–1 variables, y^k, $k = 1, \ldots, NF$, and with the NF linearizations accumulated at that point, NF Benders cuts can be generated. Assuming feasible NLP sub-problems $k = 1, \ldots, L$, the logic-based Benders master problem is then as follows:

$$\min Z_B = \alpha_{bd} + \sum_i \gamma_i y_i$$

s.t.

$$\begin{aligned} \alpha_{bd} \geq \alpha_{oa} &+ \sum_l \mu_l^k [-\alpha_{oa} + f(x^l) + \nabla f(x^l)^T (x^k - x^l)] \\ &+ \sum_i \sum_l \lambda_l^k [\nabla h_i(x^l)^T x^k \\ &[-h_i(x^l) + \nabla h_i(x^l)^T x^l] y_i] \quad k = 1, \ldots, L \\ &Ay \leq a \\ &\alpha_{bd} \in R^1, y \in \{0,1\}^m \end{aligned} \tag{MB^k}$$

where x^k and μ_l^k, λ_l^k are the primal and dual variables of the problem (LM^k) respectively. The first set of constraints are Benders cuts obtained from the objective function and the mixed-integer linear constraints. The second set of constraints are the propositional logic relationships converted into integer constraints. Having completed the solution of the initial NF sub-problems, only one new cut is generated from the LP problem (LM^k) at iteration $k \geq NF + 1$.

The steps of the logic-based Benders Decomposition algorithm are then as follows.

Step 1: Model the MINLP problem in generalized disjunctive form as in (P).

Step 2: Identify the NF sub-problems to be solved from the set covering problems [for DNF expression of propositional logic, solve problem (14) and for CNF expression of propositional logic, apply the algorithm in Appendix D].

Step 3: Solve NLP sub-problems (S^l) for the NF sub-structures determined in Step 2. The lowest cost solution of these NLPs gives an upper bound, Z_U, for the problem.

Step 4: Linearize the objective function and constraints of the current NLP sub-problem(s) and set up the master problem ($\hat{M}_{OA}^{L}$).

Step 5: Set the 0–1 variables to the same values of the Boolean variables as in the previous NLP sub-problem(s) and construct linear sub-problem(s) (LM^k). Solve these sub-problem(s) and obtain Lagrangian multipliers of the constraints and optimal values for the continuous variables to define the Lagrangian cuts in (MB^k).

Step 6: Set up and solve integer programming problem (MB^k). The solution of (MB^k) gives the lower bound on the objective function value (Z_L).

Step 7: Compare the current upper bound Z_U with the lower bound Z_L. If $|Z_U - Z_L| \leq \varepsilon$ then stop, the solution with the current Z_U is the optimal solution. Otherwise go to Step 8.

Step 8: Solve NLP sub-problem (S^l) by fixing the Boolean variables predicted by the master problem. The objective function

value of the solution is Z_{NLP}. If $|Z_U - Z_L| < \varepsilon$, set $Z_U = Z_{NLP}$.

Step 9: Compare the upper bound Z_U with the lower bound Z_L. If $|Z_U - Z_L| \leq \varepsilon$ then stop. The solution with the current Z_U is the optimal solution. Otherwise go to Step 4.

It should be noted that in the initial phase of the problem, after Step 5 is activated, *NF* different cuts are generated from the *NF* initial NLP sub-problems. Also, as shown in Appendix F, the interesting feature of the logic-based GBD algorithm is that it predicts lower bounds that are generally tighter than the conventional GBD method.

Theorem 2. The master problem of the logic-based generalized Benders decomposition (MB^k) yields tighter lower bounds than the master problem of GBD applied to problem (MIP).

Proof: See Appendix F. □

Qualitatively, the above property follows from the fact that with the convex-hull representation of the master problem (M_{OA}^L), the Lagrangian cut in the Benders master problem (MB^k) includes the effect of the equations with 0–1 variables. In contrast, in the standard GBD method, this term is not present (see Example 4 in Appendix F).

Example 1 revisited

The modeling example described earlier in the paper will be solved to illustrate the logic-based MINLP algorithms proposed in this paper. The first step is to convert to integer constraints the logical relations between the process units of the superstructure (11) as shown in Table 1. Note that, for completeness, we include the first logic proposition although it is a redundant condition. The CNF logic for the superstructure in (2) contains five clauses, all of which have to be satisfied simultaneously for a feasible flowsheet:

$$\Omega_C = [\neg Y_2 \vee \neg Y_3] \wedge [\neg Y_1 \vee Y_2 \vee Y_3 \vee \neg Y_2] \wedge [\neg Y_1 \vee Y_2 \vee Y_3 \vee \neg Y_3] \wedge [\neg Y_2 \vee Y_1] \wedge [\neg Y_3 \vee Y_1]. \quad (16)$$

If the above is converted to DNF (see Raman and Grossmann, 1993) form it yields:

$$\Omega_D = [Y_1 \wedge Y_2 \wedge \neg Y_3] \vee [Y_1 \wedge \neg Y_2 \wedge Y_3] \vee [Y_1 \wedge \neg Y_2 \wedge \neg Y_3][\neg Y_1 \wedge \neg Y_2 \wedge \neg Y_3]. \quad (17)$$

Applying the set-covering problem given in (14) indicates that two sub-problems have to be solved to obtain outer approximations for three units in the superstructure which are the first and second terms in (17). In this first sub-problem, unit 3 does not exist as shown in Fig. 2, therefore the constraint involving this unit can be discarded from the model.

The NLP sub-problem for the first sub-problem in Fig. 2. is:

$$\begin{aligned}
&\min Z_{NLP} = c_1 + c_2 + c_3 + x_4 + 1.8x_1 \\
&\qquad + 1.2x_5 + 7x_6 - 11x_8 \\
&\text{subject to} \\
&x_1 - x_2 - x_3 = 0 \\
&x_7 - x_4 - x_5 - x_6 = 0 \\
&x_5 \leq 5 \\
&x_8 \leq 1 \\
&x_8 = 0.9x_7 \\
&c_1 = 3.5 \\
&x_4 = \ln(1 + x_2) \qquad (18) \\
&c_2 = 1 \\
&x_3 = 0 \\
&x_5 = 0 \\
&c_3 = 0 \\
&c_1, c_2, c_3 \geq 0, x_i \geq 0 \quad i = 1, \ldots, 8.
\end{aligned}$$

The proposed modeling framework eliminates three out of 12 variables and one out of two nonlinear constraints which are written for the units, besides the fixed costs for the two existing units are fixed. The solution of (18) gives an objective value of -1.7210, which corresponds to a profit of $\$1.7210 \times 10^3$/h. The proposed algorithm differs from the original outer-approximation algorithm by requiring the solution of another NLP sub-problem as determined by the set covering problem to get enough outer approximations for the superstructure. The flowsheet for the second sub-problem consists of units 1 and 3 with the optimal objective function of -1.9231. After comparing the objective function values of these two sub-problems, the current upper on the problem is set to -1.9231. Then,

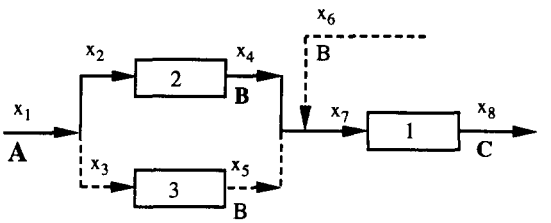

Fig. 2. Flowsheet of the first sub-problem for Example 1.

Table 2. Summary of major iterations for OA in Example 1

Iteration	Logic-based OA Z_{NLP}	Logic-based OA Z_L	OA Z_{NLP}	OA Z_L	DICOPT++ Z_{NLP}	DICOPT++ Z_L
NLP sub-problem 1	−1.7210					
NLP sub-problem 2	−1.9231	−1.9231				
1			−1.7210	−3.0000	−6.2999	−2.4671
2			−1.9321	−1.9321	−1.7210	−2.1902
3					−1.9231	0.0000

the master problem for the example is constructed by including the outer approximations from the first and second sub-problems with the convex-hull representation as follows:

$$\min Z_L = 3.5y_1 + y_2 + 1.5y_3 + x_4 + 1.8x_1 + 1.2x_5 + 7x_6 - 11x_8$$

subject to

$$\begin{aligned} x_1 - x_2 - x_3 &= 0 \\ x_7 - x_4 - x_5 - x_6 &= 0 \\ x_5 &\leq 5 \\ x_8 &\leq 1 \\ x_8 &= 0.9x_7 \qquad (19) \\ -0.329193x_2 + x_4 &\leq 0.440304y_2 \\ -0.475397x_3 + x_5 &\leq 0.386508y_3 \\ y_1 - y_2 &\geq 0 \\ y_1 - y_3 &\geq 0 \\ y_2 + y_3 &\leq 1 \\ y_1 - y_2 &\leq 1 \\ y_1 - y_3 &\leq 1 \\ y_1, y_2, y_3 &\in \{0,1\}, x_i \geq 0 \quad i = 1, \ldots, 8. \end{aligned}$$

Note that the master problem does not include the linearizations of nonlinear constraints at zero flows and the outer approximation of them are made at the optimal values of continuous variables in that particular sub-problem.

When the first MILP master problem (19) is solved, the solution has an objective function value of −1.9231 which corresponds to the upper bound on the problem. The upper and lower bounds are equal to each other and the algorithm terminates with the optimal configuration of units 1 and 3, and a profit of \$1923.1/h. When the problem is solved by OA algorithm with the same initial configuration, the same optimal configuration is obtained in two major iterations. It is interesting to note that both the proposed method and original OA algorithm require solution of 2 NLP sub-problems as shown in Table 2. The main difference between the two methods is the number of MILP master problems solved; one for the logic-based OA and two for the OA. In addition, the sizes of the NLP sub-problems with the proposed algorithm are smaller compared to that of the OA. DICOPT++ (with convex termination criterion) takes three major iterations to solve the same problem. The first iteration starts with a relaxed MINLP, which in this case provides a rigorous lower bound on the problem. The second and the third iterations correspond to similar iterations of OA respectively except that they include integer cuts.

The logic-based Benders decomposition algorithm requires the solution of two initial sub-problems for this problem too. Then, the MILP master problem given in (19) is solved as an LP sub-problem by fixing the binary variables to the corresponding Boolean variable values with the NLP sub-problems respectively. Therefore, two cuts are generated for the first integer master problem:

$$\begin{aligned} &\min Z_L = \alpha_{LB} \\ &\text{subject to} \\ &\alpha_{LB} \geq 3.5y_1 - 1.4076y_2 - 4.0561y_3 - 3.8130 \\ &\alpha_{LB} \geq 3.5y_1 - 0.7551y_2 + 0.0367y_3 - 5.4603 \qquad (20) \\ &y_1 - y_2 \geq 0 \\ &y_1 - y_3 \geq 0 \\ &y_2 + y_3 \leq 1 \\ &y_1 - y_2 \leq 1 \\ &y_1 - y_3 \leq 1 \\ &y_1, y_2, y_3 \in \{0, 1\}. \end{aligned}$$

The first master problem predicts the configuration in which none of the units are used with a lower bound of −3.8130. At this point, the NLP sub-problem does not have to be solved, since the proposed modeling framework eliminates the equations and variables for non-existing units. The algorithm continues with the LP sub-problem obtained by fixing the binary variables to 0. Then, the third Benders' cut is obtained as follows:

$$\alpha_{LB} \geq -46y_1 + y_2 + 1.5y_3. \qquad (21)$$

This cut is added to the integer master problem (20) and solved again with an optimal structure

Table 3. Summary of major iterations for Benders in Example 1

Iteration	Logic-based Benders Z_{NLP}	Logic-based Benders Z_L	GBD Z_{NLP}	GBD Z_L
NLP sub-problem 1	−1.7210			
NLP sub-problem 2	−1.9231	−3.8130		
1	0.0000	−1.9231	−1.7210	−23.8278
2			−1.9231	−6.2209
3			0.0000	−2.7210
4			0.2780	−1.9231

containing units 1 and 3 and the objective function of −1.9231. The solution with the generalized Benders decomposition, however requires four iterations, and finds the optimum solution only after visiting all feasible structures, as shown in Table 3. The master problem of the proposed algorithm is able to predict tighter lower bounds compared to GBD.

Example 2

The example problem proposed by Duran and Grossmann (1986) in Fig. 3 will be used to illustrate the application of the proposed methods in the optimization of a larger superstructure that includes convex constraints. The logic-based MINLP model formulation of the problem is given in Appendix A. It includes eight Boolean variables, 33 continuous variables and eight disjunctions.

The Boolean variables Y_i denote the existence or non-existence of processes 1–8. The propositional logic expressions indicate that exactly one of the processes 1 and 2 must be selected and processes 4 and 5 cannot appear together in a structure. If process 4 is not selected, then process 5 can be selected and selection of process 4 requires the selection of process 6 or 7. If process 3 or 5 is selected, then process 8 should also be selected. The CNF expression in (2) and the corresponding integer constraints for the flowsheet are given in Table 4.

Solution of the set covering problem indicates that optimization of the NLPs for sub-structures:

$$[Y_1 \wedge Y_3 \wedge Y_4 \wedge Y_7 \wedge Y_8 \wedge \neg Y_2 \wedge \neg Y_5 \neg Y_6],$$

$$[Y_2 \wedge Y_3 \wedge Y_4 \wedge Y_6 \wedge Y_8 \wedge \neg Y_1 \wedge \neg Y_5 \wedge \neg Y_7]$$

and

$$[Y_1 \wedge Y_3 \wedge Y_5 \wedge Y_8 \wedge \neg Y_2 \wedge \neg Y_4 \wedge \neg Y_6 \wedge \neg Y_7] \quad (22)$$

provides linearizations at non-zero flows for all units in the superstructure. Note that the solution of these three sub-problems may give more than one linearization for some units.

Solution of the three NLP sub-problems determined by the set covering problem gives the upper bound $Z_U = 73.278$. Solving the NLPs for the three sub-structures, the master problem at iteration 1 can be defined which, for the logic-based outer-approximation algorithm, predicts the optimal structure for the flowsheet: processes 2, 4, 6, 8 with an objective function value of $Z = 68.0097$. When solved with DICOPT++ (Viswanathan and Grossmann, 1990) the problem required four major iterations and the CPU time was higher (3.8 s vs 0.6 s, IBM/RS6000) since full-dimensional NLPs had to be solved as opposed to solving the reduced NLP sub-problems in (S^l).

The logic-based Benders decomposition algorithm requires one more iteration to find the optimal solution than the logic-based outer approximation algorithm. The first master problem predicts the flowsheet with processes 1, 4, 6, 8. The next master problem predicts the optimal flowsheet by replacing process 1 from the previous problem with the process 2. The lower bounds determined from the integer programming problem (MB) of logic-based Benders decomposition are substantially higher than the original generalized Benders decomposition as shown in Table 5. The logic-based algorithm takes advantage of the convex-hull representation in the

Table 4. CNF logic expressions for Example 2

Logic proposition	Clauses	Constraint
$Y_1 \Rightarrow Y_3 \vee Y_4 \vee Y_5$	$\neg Y_1 \vee Y_3 \vee Y_4 \vee Y_5$	$-y_1 + y_3 + y_4 + y_5 \geq 0$
$Y_2 \Rightarrow Y_3 \vee Y_4 \vee Y_5$	$\neg Y_2 \vee Y_3 \vee Y_4 \vee Y_5$	$-y_2 + y_3 + y_4 + y_5 \geq 0$
$Y_3 \Rightarrow Y_8$	$\neg Y_3 \vee Y_8$	$-y_3 + y_8 \geq 0$
$Y_3 \Rightarrow Y_1 \vee Y_2$	$\neg Y_3 \vee Y_1 \vee Y_2$	$y_1 + y_2 - y_3 \geq 0$
$Y_4 \Rightarrow Y_1 \vee Y_2$	$\neg Y_4 \vee Y_1 \vee Y_2$	$y_1 + y_2 - y_4 \geq 0$
$Y_4 \Rightarrow Y_6 \vee Y_7$	$\neg Y_4 \vee Y_6 \vee Y_7$	$-y_4 + y_6 + y_7 \geq 0$
$Y_5 \Rightarrow Y_1 \vee Y_2$	$\neg Y_5 \vee Y_1 \vee Y_2$	$y_1 + y_2 - y_5 \geq 0$
$Y_5 \Rightarrow Y_8$	$\neg Y_5 \vee Y_8$	$-y_5 + y_8 \geq 0$
$Y_6 \Rightarrow Y_4$	$\neg Y_6 \vee Y_4$	$y_4 - y_6 \geq 0$
$Y_7 \Rightarrow Y_4$	$\neg Y_7 \vee Y_4$	$y_4 - y_7 \geq 0$
$Y_8 \Rightarrow Y_3 \vee Y_5 \vee (\neg Y_3 \wedge \neg Y_5)$	$(\neg Y_8 \vee Y_3 \vee \neg Y_3 \vee Y_5) \wedge$	$-y_5 + y_8 \leq 1$
	$(\neg Y_8 \vee Y_5 \vee \neg Y_5 \vee Y_3)$	$-y_3 + y_8 \leq 1$
Specifications	$\neg Y_1 \vee Y_2$	$y_1 + y_2 \leq 1$
	$\neg Y_4 \vee \neg Y_5$	$y_4 + y_5 \leq 1$
	$\neg Y_6 \vee \neg Y_7$	$y_6 + y_7 \leq 1$

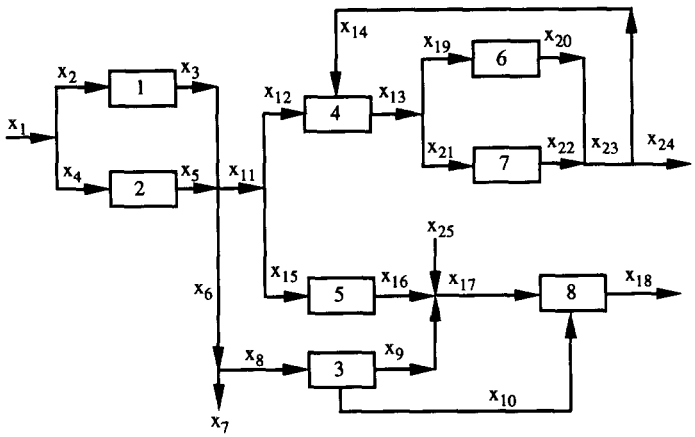

Fig. 3. Superstructure for Example 3.

Table 5. Comparison of the results for Example 2

Iteration	Logic-based OA Z_{NLP}	Logic-based OA Z_L	Logic-based Benders Z_{NLP}	Logic-based Benders Z_L	DICOPT++ Z_{NLP}	DICOPT++ Z_L	Generalized Benders decomposition Z_{NLP}	Generalized Benders decomposition Z_L
Sub-problem 1	103.6		103.6					
Sub-problem 2	73.3		73.3					
Sub-problem 3	113.8	67.9	113.8	66.3				
1	68.0	68.0	77.1	67.3	15.1	25.3	103.6	−715.7
2			68.0	68.0	104.3	60.4	76.4	−372.7
3					77.1	68.0	77.1	−133.9
4					68.0	72.6	73.3	−106.7
5							99.6	−101.7
6							91.2	33.1
7							82.4	47.2
8							94.5	67.3
9							68.0	68.0
CPU time*	0.59		1.13		3.79		3.23	

* In, IBM RS6000/530.

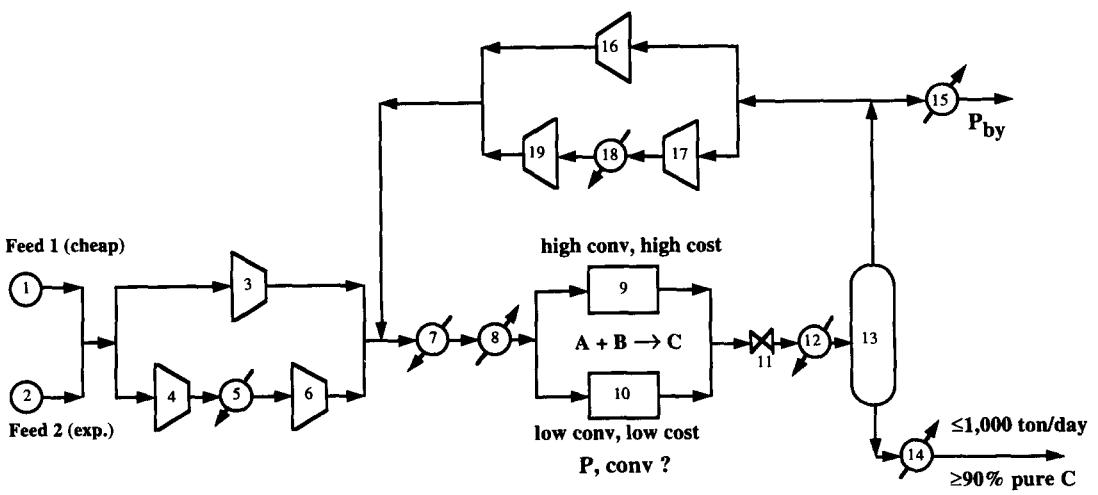

Fig. 4. Superstructure for Example 3.

master problem and eliminates the possibility of selecting infeasible structures, therefore generating tighter lower bounds. It should also be noted that generalized Benders decomposition in this problem took nine major iterations to converge.

Example 3

The proposed logic-based MINLP algorithms have been applied to the flowsheet problem in Fig. 4. It is desired to produce C with a minimum purity of 90% and an amount of 1000 tons/day. The superstructure contains 19 units; two different feedstocks, two compressor types in the reactor feed stream and the recycle streams, two reactors, four coolers, three heaters and a flash separator. There are two alternative feedstocks: the first one is cheaper and the second one is the more expensive, containing smaller concentration of inert D which has to be purged from the recycle stream. Different fixed cost charge applies for each feed and only one of them can be selected. The purge stream has a market value as fuel. There are two reactors in the superstructure: one of them is lower conversion, lower cost and the other is higher conversion and higher cost. The key optimization variables in the reactors are the operating pressure and the conversion per pass. It is anticipated that the reactors operate at high pressures; therefore, reactor feed stream and the recycle stream are compressed. Two choices of the compressor include a single-stage compressor and a two-stage compressor with intercooling. If the DNF representation is derived it can be shown that there are 16 structural alternatives for the flowsheets in the superstructure. The problem data for the example are given in Table 6.

Table 6. Problem data for Example 3

Feedstock	Composition		Cost ($/kg-mol)
Feed 1	A	60%	0.026
	B	25%	
	D	15%	
Feed 2	A	65%	0.033
	B	30%	
	D	5%	
Product	*Specification*		*Cost* ($/kg mol)
P	$\geq$90% C $\leq$1,000 tons/day		0.25
P_{by}			0.021
Utilities			*Cost*
Electricity			$0.03/kWh
Steam			$8.0/10^6 kJ
Water			$0.7/10^6 kJ

In order to satisfy safety and construction material specifications, the operating pressure in the reactors is restricted between 2.5 and 15 MPa and the outlet temperature cannot exceed 873K. The flash also operates at most of 15 MPa, and it is known that the separation in the flash is best achieved between 300 and 500K. These restrictions are also included in the model.

When this problem is modeled in the conventional way as an MINLP, it contains 19 binary variables, 293 continuous variables and 353 constraints. The model contains highly nonlinear and non-convex constraints and, therefore, the NLP sub-problems fail even to find a feasible solution due to singularities in the Jacobian. The zero flows involved in the nonconvex equations lead to singularities which are avoided in the logic-based model. The same problem can be modeled as a generalized disjunctive programming problem with 19 Boolean variables, 312 continuous variables, 19 disjunctions and 25 propositional logic relationships. The disjunctions

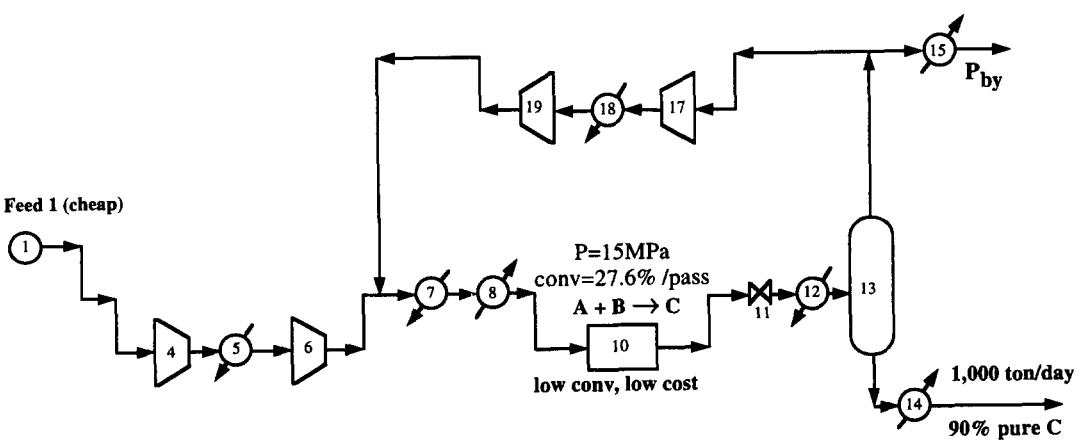

Fig. 5. First sub-problem for example 3.

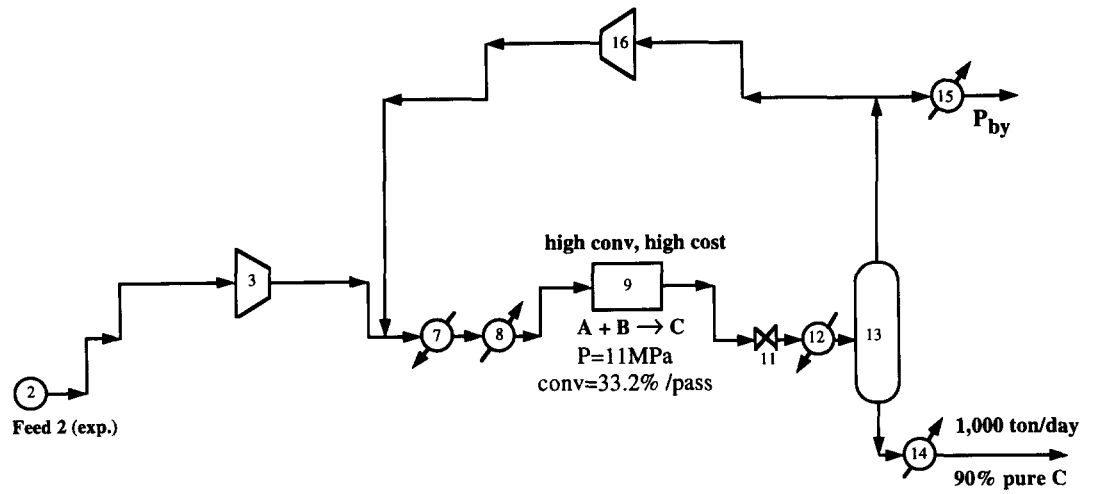

Fig. 6. Second sub-problem for Example 3.

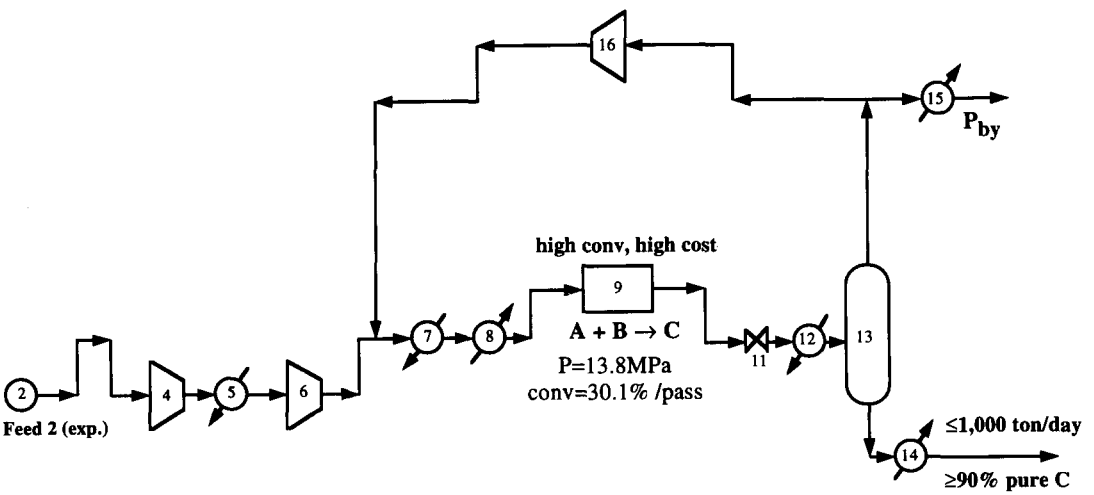

Fig. 7. Optimal flowsheet configuration in Example 3.

include the model equations and constraints for the units. The values of the Boolean variables indicate which side of the disjunction is applied. The propositional logic relationships describe the topology of the superstructure.

The logic-based OA algorithm starts with the solution of two initial NLP sub-problems which include all the units at least once. These sub-problems are determined from the set covering algorithm (see Appendix D). The flowsheet in the first sub-problem (as shown in Fig. 5) contains the cheaper feed (Feed 1), two-stage compressor with intercooler in the feed and recycle streams, low cost, low conversion reactor, flasher and the product and by-product heaters. The operating pressure of the reactor is 15 MPa and the reactor conversion per pass is 27.6%, while the overall conversion is 90.41%. The flowsheet meets the purity and amount specifications on the product stream. However, this flowsheet incurs in a loss of $859,000/yr.

Table 7. Summary of iterations with logic-based OA for Example 3

Iteration	Objective	CPU time (s*)
NLP sub-problem 1	−859,000	1.74
NLP sub-problem 2	1,575,000	1.68
Master problem 1	1,868,000	1.48
NLP subproblem 3	1,794,000	3.62
Master problem 2	1,741,000	2.95
Total time		11.47

* IBM RS6000/530.

The flowsheet in the second NLP sub-problem includes the more expensive feed, single stage compressor in the feed and recycle streams, high conversion high cost reactor, flash separator and the

heaters for the product and by-product streams as shown in Fig. 6. The reactor operates at 11 MPa with a conversion per pass of 33.2% and an overall conversion of 94.26%. The flowsheet meets the purity and amount specifications on the product stream. The objective function value indicates that this flowsheet makes profit with $1,575,000/yr.

The solution of two initial NLP sub-problems provides outer approximations for every unit in the superstructure with which the first master problem is constructed. The disjunctions in the master problem are replaced with their convex hull representations and, in order to handle non-convexities, slacks were added to the linearizations as discussed in Viswanathan and Grossmann (1990). The first master problem predicts a potential profit of $1,868,000/yr with the flowsheet in Fig. 7.

In the third NLP sub-problem (Fig. 7) only, the feed compressor is changed from the previous sub-problem. The two-stage feed compressor is able to attain higher pressures and the reactor pressure inreases to 13.8 MPa with a conversion per pass of 30.1% and an overall conversion of 99.96%. Since the feed compressor operates at a higher pressure, it is possible to use less feed with higher recycle ratio. This saves some money from the raw material cost and the profit increases to $1,794,000/yr. Since the upper bound ($1,794,000/yr) and the lower bound ($1,868,000) predicted from the master problem did not converge, another master problem is derived with the outer approximations from the three NLP sub-problems. The second master problem predicts a potential profit of $1,741,000/yr which is lower than the upper bound on the problem. The algorithm terminates after successfully finding the optimal configuration which includes the second feed, two-stage compressor in the feed stream, high conversion reactor, expansion valve, flash, single-stage compressor in the recycle stream and the product and by-product heaters.

The CPU time required for the solution of NLP subproblems and the master problems is given in Table 7. It is interesting to see that the NLP sub-problem takes less time compared to the NLP that arises from fixing the 0–1 variables in the conventional MINLP model.

The main reasons for the savings in the computation time is because the problem size with the logic-based algorithm is reduced almost by one half of the original MINLP problem size, as seen in Fig. 8.

When the problem is solved with the logic-based Benders decomposition, the optimal solution is found after solving five NLP sub-problems taking 20.28 s as shown in Table 8.

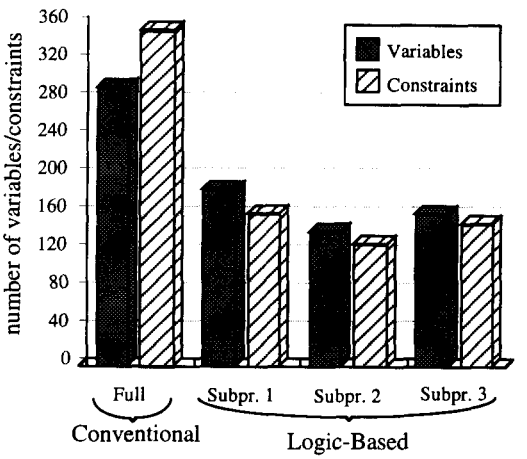

Fig. 8. Comparison of the problem sizes in Example 3.

CONCLUSIONS

A novel logic-based MINLP modeling and solution framework has been proposed for the optimal synthesis of process networks. The generalized disjunctive programming model is based on expressing constraints through logical disjunctions and propositions with which the NLP sub-problems only require the relevant equations for existing units. It was shown that the disjunctive models allow elimination of redundant constraints and avoid solution of nonlinear constraints at zero flows. Another advantage of the disjunctive model is that, by eliminating the constraints for non-existing units, difficulties with non-convexities are reduced. Therefore, the robustness of NLP sub-problems is increased with the present approach. Furthermore, due to the reduction in the problem size, solution times for NLP subproblems are also reduced.

The proposed approach uses the logic propositions for the determination of feasible flowsheet structures. At the initial stage of the proposed algorithms, it is required to solve a number of NLP subproblems so that enough linearizations for the first

Table 8. Summary of iterations with logic-based GBD for Example 3

Iteration	Objective	CPU time (s*)
NLP subproblem 1	−859,000	1.74
NLP subproblem 2	1,575,000	1.68
Master problem 1	21,211,000	1.21
NLP subproblem 3	1,158,000	3.26
Master problem 2	14,924,000	1.09
NLP subproblem 4	1,794,000	3.62
Master problem 3	6,314,000	1.60
NLP subproblem 5	−192,000	3.96
Master problem 4	114,000	2.12
Total time		20.28

* IBM RS6000/530.

master problem can be obtained. Two different formulations were presented for the determination of these initial sub-problems: CNF formulation and DNF formulation. The DNF formulation can be used for problems requiring relatively few feasible configurations. The CNF formulation is used for problems for which the total number of feasible configurations can be large. Both of the formulations are automated with the present approach. The derivation of an outer-approximation method for the logic-based MINLP model has been presented as well as for a logic-based generalized Benders decomposition method. It was also shown that the logic-based GBD algorithm predicts stronger lower bounds than the original GBD algorithm. Results of several examples were given and compared with the original OA and generalized Benders decomposition methods. A large-scale flowsheet example has shown that the generalized disjunctive modeling framework can be used effectively to model realistic process synthesis problems. Major advantages are that the robustness of the NLP sub-problems is greatly increased while solution times are decreased. Future work will include the handling of complex discontinuous cost functions and process equations.

Acknowledgements—The authors would like to acknowledge financial support from the Engineering Design Research Center at Carnegie Mellon University and from Eastman Chemical. Also, Ignacio E. Grossmann would like to acknowledge support for his sabbatical leave by the Centre of Process Systems Engineering at Imperial College in London.

REFERENCES

Balas E., Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM J. Alg. Disc. Meth.* **6**, 466–486 (1985).

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

Beaumont N., An algorithm for disjunctive programs. *Eur. J. Op. Res.* **48**, 362–271 (1990).

Benders J. F., Partitioning procedures for solving mixed-variables programming problems. *Num. Math.* **4**, 238–252 (1962).

Duran M. A. and I. E. Grossmann, An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math. Progr.* **36**, 307–339 (1986).

Fletcher R. and S. Leyffer, Solving mixed integer nonlinear programs by outer approximation. *Math. Progr.* **66**, 327–350 (1994).

Floudas C. A. and I. E. Grossmann, Algorithmic approaches to process synthesis: logic and global optimization. *FOCAPD'94 Meeting*, Snowmass, CO (1994).

Friedler F., K. Tarjan, Y. W. Huang and L. T. Fan, Graph-theoretic approach to process synthesis: axioms and theorems. *Chem. Engng Sci.* **47**, 1973–1988 (1992).

Geoffrion A. M., Generalized benders decomposition. *J. Opt. Theory Appl.* **10**, 237–260 (1972).

Grossmann, I. E., MINLP optimization strategies and algorithms for process synthesis. *Proc. FOCAPD Meeting*, Siirola *et al.* (Eds), pp. 105–132 (1990).

Grossmann I. E. and M. M. Daichendt, New trends in optimization-based approaches to process synthesis. *Proc. PSE'94*, Yoon E. S. (ed.), pp. 95–109 (1994).

Grossmann I. E. and Z. Kravanja, Mixed-integer nonlinear programming techniques for process systems engineering. *Computers chem. Engng* **19**, S189–S204 (1995).

Kocis G. R. and I. E. Grossmann, A modeling and decomposition strategy for the MINLP optimization of process flowsheets. *Computers chem. Engng* **13**, 797–819 (1989).

Kravanja Z. and I. E. Grossmann, PROSYN-A MINLP process synthesizer. *Computers chem. Engng* **14**, 1363–1378 (1990).

Kravanja Z. and I. E. Grossmann, New developments and capabilities in PROSYN—an automated topology and parameter process synthesizer. *Computers chem. Engng* **18**, 1097–1114 (1994).

Mawekwang H. and B. A. Murtagh, Solving nonlinear integer programs with large scale optimization software. *Ann. Op. Res.* **5**, 427–437 (1986).

Nabar S. V. and L. Schrage, Modeling and solving nonlinear integer programming problems. Paper No. 22a, *Annual A. I. Ch. E. Meeting*, Chicago, IL (1990).

Nemhauser G. L. and L. Wolsey, *Integer and Combinatorial Optimization*. Wiley, New York (1988).

Quesada I. and I. E. Grossmann, An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Computers chem. Engng* **16**, 937–947 (1992).

Raman R. and I. E. Grossmann, Symbolic integration of logic in mixed integer linear programming techniques for process synthesis. *Computers chem. Engng* **17**, 909–927 (1993).

Raman R. and I. E. Grossmann, Modeling and computational techniques for logic based integer programming. *Computers chem. Engng* **18**, 563–578 (1994).

Reklaitis G. V., Perspectives on scheduling and planning of process operations. *Proc. PSE'91*, Vol. **IV**, Montebello, Canada (1991).

Viswanathan J. and I. E. Grossmann, A combined penalty function and outer-approximation method for MINLP optimization. *Computers chem. Engng* **14**, 769–782 (1990).

Westerlund T. and F. Pettersson, An extended cutting plane (ECP) method for the solution on MINLP problems. *Report 92-124-A, Process Design Laboratory*, Åbo Akademi University (1992).

APPENDIX A

Logic-based Formulation for Example 2

Objective function:

$$\min Z = c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8 + x_2 - 10x_3 + x_4 - 15x_5 - 40x_9 + 15x_{10} + 15x_{14} + 80x_{17} - 65x_{18} + 25x_{19} - 60x_{20} + 35x_{21} - 80x_{22} - 35x_{25} + 122.$$

Material balances at mixing/splitting points:

$$x_3 + x_5 - x_6 - x_{11} = 0$$

$$x_{13} - x_{19} - x_{21} = 0$$

$$x_{17} - x_9 - x_{16} - x_{25} = 0$$

$$x_{11} - x_{12} - x_{15} = 0$$

$$x_6 - x_7 - x_8 = 0$$

$$x_{23} - x_{20} - x_{22} = 0$$

$$x_{23} - x_{14} - x_{24} = 0.$$

Specifications on the flows:

$$x_{10}-0.8x_{17}\leq 0$$
$$x_{10}-0.4x_{17}\geq 0$$
$$x_{12}-5x_{14}\leq 0$$
$$x_{12}-2x_{14}\geq 0$$

Unit 1: $\begin{bmatrix} Y_1 \\ \exp(x_3)-1-x_2=0 \\ c_1=5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_2=x_3=0 \\ c_1=0 \end{bmatrix}$

Unit 2: $\begin{bmatrix} Y_2 \\ \exp(x_5/1.2)-1-x_4=0 \\ c_2=8 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_4=x_5=0 \\ c_2=0 \end{bmatrix}$

Unit 3: $\begin{bmatrix} Y_3 \\ 1.5x_9-x_8+x_{10}=0 \\ c_3=6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_8=x_9=x_{10}=0 \\ c_3=0 \end{bmatrix}$

Unit 4: $\begin{bmatrix} Y_4 \\ 1.5(x_{12}+x_{14})-x_{13}=0 \\ c_4=10 \end{bmatrix} \vee \begin{bmatrix} \neg Y_4 \\ x_{12}=x_{13}=x_{14}=0 \\ c_4=0 \end{bmatrix}$

Unit 5: $\begin{bmatrix} Y_5 \\ x_{15}-2x_{16}=0 \\ c_5=6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_5 \\ x_{15}=x_{16}=0 \\ c_5=0 \end{bmatrix}$

Unit 6: $\begin{bmatrix} Y_6 \\ \exp(x_{20}/1.5)-1-x_{19}=0 \\ c_6=7 \end{bmatrix} \vee \begin{bmatrix} \neg Y_6 \\ x_{19}=x_{20}=0 \\ c_6=0 \end{bmatrix}$

Unit 7: $\begin{bmatrix} Y_7 \\ \exp(x_{22})-1-x_{21}=0 \\ c_7=4 \end{bmatrix} \vee \begin{bmatrix} \neg Y_7 \\ x_{21}=x_{22}=0 \\ c_7=0 \end{bmatrix}$

Unit 8: $\begin{bmatrix} Y_8 \\ \exp(x_{18})-1-x_{10}-x_{17}=0 \\ c_8=5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_8 \\ x_{10}=x_{17}=x_{18}=0 \\ c_8=0 \end{bmatrix}.$

Propositional logic $[\Omega=(Y_i)]$:

$$Y_1 \Rightarrow Y_3 \vee Y_4 \vee Y_5$$
$$Y_2 \Rightarrow Y_3 \vee Y_4 \vee Y_5$$
$$Y_3 \Rightarrow Y_1 \vee Y_2$$
$$Y_3 \Rightarrow Y_8$$
$$Y_4 \Rightarrow Y_1 \vee Y_2$$
$$Y_4 \Rightarrow Y_6 \vee Y_7$$
$$Y_5 \Rightarrow Y_1 \vee Y_2$$
$$Y_5 \Rightarrow Y_8$$
$$Y_6 \Rightarrow Y_4$$
$$Y_7 \Rightarrow Y_4$$
$$Y_8 \Rightarrow Y_3 \vee Y_5 \vee (\neg Y_3 \wedge \neg Y_5).$$

Specifications:

$$Y_1 \veebar Y_2$$
$$Y_4 \veebar Y_5$$
$$Y_6 \veebar Y_7.$$

Variables:

$$x_j, \quad c_i \geq 0, \quad Y_i = \{\text{True, False}\}$$
$$i = 1,2,\ldots,8; \quad j = 1,2,\ldots,25.$$

APPENDIX B

MILP Transformation of Disjunctive Master Problem

The master problem (M_{OA}^L) of problem (P) for process flowsheet synthesis problems is given by the disjunctive LP:

$$\min Z_L = \alpha_{oa} + \sum_i c_i$$

s.t.

$$\left.\begin{aligned} \alpha_{oa} &\geq f(x^l) + \nabla f(x^l)^T (x - x^l) \\ g(x^l) &+ \nabla g(x^l)^T (x - x^l) \leq 0 \end{aligned}\right\} l = 1, \ldots, L$$

$$\begin{bmatrix} Y_i \\ h_i(x^l) + \nabla h_i(x^l)^T (x - x^l) \leq 0 \; l \in K_L^i \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ B_x^i = 0 \\ c_i = 0 \end{bmatrix} i \in D \qquad (M_{OA}^L)$$

$$\Omega(Y) = \text{True}$$

$$x \in R^n, \; c \geq 0, \; Y \in \{\text{True, False}\}^m.$$

The first two constraints are linear. The continuous variables, x, are partitioned into two sets such that:

$$X = X_z \cup X_{nz}, \; X_z = \{x_z\}, \; X_{nz} = \{x_{nz}\}$$
$$\text{where } x_z = x_j \text{ when } b_j^T = e_j^T, \text{ and} \qquad \text{(B1)}$$
$$x_{nz} = x_j \text{ when } b_j^T = 0^T.$$

The disjunction then becomes:

$$\begin{bmatrix} Y_i \\ h_i(x^l) + \nabla_{x_z} h_i(x_z^l)^T (x_z - x_z^l) + \nabla_{x_{nz}} h_i(x_{nz}^l)^T (x_{nz} - x_{nz}^l) \leq 0 \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ x_z = 0 \\ x_{nz} = 0 \\ c_i = 0 \end{bmatrix}. \qquad \text{(B2)}$$

The above expression means that a linear constraint applies if Y_i is true. The subset of continuous variables in X_z (e.g. flows, sizes) are equal to zero if Y_i is false.

The disjunction (B2) can be converted into mixed integer from using the inequalities in (4). However, aside from the fact that the determination of tight values of x^u for each constraint is not an easy task, these constraints are well known to yield poor relaxations. Another form of converting the disjunctions into mixed-integer constraints that yield tighter relaxations is by using the procedure given in Balas (1985) (see also Raman and Grossmann, 1993) that involves obtaining the convex-hull representation. Replacing the Boolean variables Y_i by the corresponding binary variable y_i, and by disaggregating the continuous variables x and c, the convex hull of the disjunction (B2) is given by,

$$\nabla_{x_z} h_i(x_z^l)^T x_z^1 + \nabla_{x_{nz}} h_i(x_{nz}^l)^T x_{nz}^1 \leq [-h_i(x^l) + \nabla h_i(x^l)^T x^l] y_i^l$$
$$c_i^1 = \gamma_i y_i^1$$

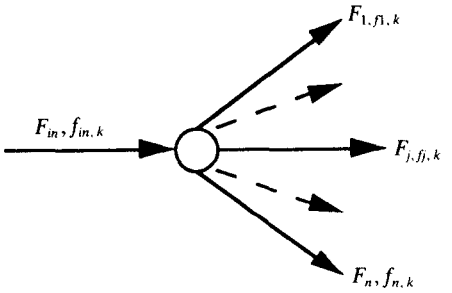

Fig. C1. A single choice splitter.

$$
\begin{aligned}
&x_z^2 = 0y_i^2\\
&x_{nz}^2 \geqslant 0y_i^2\\
&c_i^2 = 0y_i^2\\
&c_i = c_i^1 + c_i^2\\
&x_z = x_z^1 + x_z^2\\
&x_{nz} = x_{nz}^1 + x_{nz}^2\\
&y_i^1 + y_i^2 = 1.
\end{aligned}
\tag{B3}
$$

It is clear from (B3) that x_z^2 and c_i^2 are equal to zero and only one of the terms in disjunction (B2) applies, which is y_i^1. Hence, the convex-hull representation of the linear disjunction (B2) with the introduction of the binary variable y_i can be reduced to:

$$\nabla_{x_z} h_i(x_z^l)^T x_z + \nabla_{x_{nz}} h_i(x_{nz}^l)^T x_{nz}^l \leqslant [-h_i(x^l) + \nabla h_i) x^l)^T x^l] y_i \tag{B4}$$

$$c_i = \gamma_i y_i.$$

For the sake of simplicity in the presentation of $(\hat{M}_{OA}^L)$, it will be assumed that there are no non-zero variables (i.e. $X_{nz} = \emptyset$). Therefore, equation (B4) becomes:

$$
\begin{aligned}
&\nabla h_i(x^l)^T x \leqslant [-h_i(x^l) + \nabla h_i(x^l)^T x^l] y_i\\
&c_i = \gamma_i y_i.
\end{aligned}
\tag{B5}
$$

APPENDIX C

Treatment of Single Choice Interconnection Units

The treatment of disjunctions involving more than two terms within the disjunctive modeling framework for the convex-hull representation is illustrated with a single choice stream splitter. The treatment of other single choice interconnection units is similar and the procedure described in Balas (1985) can be applied to them as well.

The system of equations for the single choice splitter is as follows:

$$F_{in} = \sum_k f_{in,k}$$

$$F_j = \sum_k f_{j,k} \quad j = 1, \ldots, n$$

$$f_{in,k} = \sum_j f_{j,k}$$

$$\bigvee_{j=1,\ldots,n} \begin{bmatrix} Y_j \\ F_{in} = F_j \\ F_i = 0 \quad \forall i \neq j \\ F_j \leqslant U \end{bmatrix} \tag{C1}$$

The convex-hull representation for the disjunction can be derived by disaggregating the variables F_{in}, F_i and introducing binary variables y_j as follows:

$$
\begin{aligned}
&F_{in} = \sum_{j=1}^{n} F_{in}^j\\
&F_i = \sum_{j=1}^{n} F_i^j
\end{aligned}
\tag{C2}
$$

$$
\begin{aligned}
&F_{in}^j - F_j^j = 0y_j\\
&F_i^j = 0y_j \; \forall i \neq j\\
&F_j^j \leqslant U\\
&\sum_{j=1}^{n} y_j = 1.
\end{aligned}
\tag{C3}
$$

Substituting (C3) into (C2) yields the following for (C1):

$$
\begin{aligned}
&F_{in} = \sum_k f_{in,k}\\
&F_j = \sum_k f_{j,k}\\
&f_{in,k} = \sum_j f_{j,k}\\
&F_{in} = \sum_j F_j\\
&F_j \leqslant U y_j\\
&\sum_{j=1}^{n} y_j = 1
\end{aligned}
\tag{C4}
$$

which corresponds precisely to the single choice splitter equations by Kocis and Grossmann (1989).

APPENDIX D

Modified Set-covering Algorithm for CNF Propositional Logic

When the topology of the flowsheet is described in CNF, an iterative set covering problem has to be formulated in order to identify the sub-problems in Ω_C that are to be optimized,

$$\Omega_C = \bigwedge_{s=1}^{S} \left[\bigvee_{i \in I_p^s} Y_i \bigvee_{i \in I_p^s} \neg Y_i \right] \tag{D1}$$

where S is the number of propositional logic relationships between the unit of the flowsheet, and I_p^s and I_n^s are the corresponding subsets of non-negated and negated Boolean variables ($I_p^s \cup I_n^s = I$). The following procedure is based on the idea of successively finding feasible flowsheet structures that maximize the occurrence of new units, and gives the minimum number of subproblems to be solved to cover all the units in a given superstructure:

Step 1: Convert Ω_C into a set of linear inequalities: $Ay_i \geqslant a$

Step 2: Set $k = 1$, $CB^k = \emptyset$, $CN^k = I$, $(I_p^s \cup I_n^s = I)$.

Step 3: (a) Determine the weights of the binary variables as follows:

$$\beta_i = 1 \text{ for } i \in CB^k$$

$$\beta_i = |CB^k| + 1 \text{ for } i \in CN^k.$$

(b) Solve problem (D2) to determine the solution y^k corresponding to an NLP sub-problem:

$$\max Z = \sum_i \beta_i y_i$$

s.t.

$$Ay_i \geq a \qquad \text{(D2)}$$

$$\sum_{i \in B_r} y_i^k - \sum_{i \in N_r} y_i^k \leq |CB^k| - 1 \; (r < k)$$

$$y_i = 0,1.$$

(c) Define $B^k = \{i \text{ such that } y_i = 1\}$, $N^k = \{i \text{ such that } y_i = 0\}$.
(d) Set $CB^{k+1} = CB^k \cap B^k$, $CN^{k+1} = CN^k \; B^k$.
(e) If $CN^{k+1} = \emptyset$, set $K = k$, go to Step 4. Otherwise set $k = k + 1$, return to Step 3(a).

Step 4: The NF sub-problems are defined by the following combinations of Boolean variables:

$$Y_i^k = \text{True if } y_i^k = 1,$$
$$Y_i^k = \text{False if } y_i^k = 0 \; k = 1, \ldots, K. \qquad \text{(D3)}$$

APPENDIX E

Proof of the Theorem

Theorem 1. Given is the following MINLP problem (MIP):

$$\min Z = c^T y + \phi(x)$$

s.t.

$$\psi_j(x) + b_j^T y \leq 0 \quad j \in J \qquad \text{(MIP)}$$
$$Ay \leq a$$
$$x \in X, y \in \{0, 1\}^n$$

in which the functions $\phi(x)$ and $\psi_j(x)$ are convex and differentiable.

Solving the MILP master problem of the outer-approximation method by one Benders iteration yields a solution that is equivalent to the solution of the master problem of generalized Benders decomposition.

Proof. For a given set of linearizations K_{iter}, the MILP master problem of the outer approximation (MOA) for MINLP problem (MIP) is given as follows:

$$\min Z = c^T y + \alpha$$

s.t.

$$\left.\begin{array}{l} \phi(x^k) + \nabla\phi_j(x^k)^T(x - x^k) - \alpha \leq 0 \\ \psi_j(x^k) + \nabla\psi_j(x^k)^T(x - x^k) + b_j^T y \leq 0 \quad j \in J \end{array}\right\} k \in K_{\text{iter}} \quad \text{(MOA)}$$

$$Ay \leq a$$

$$\alpha \in R^1, x \in X, y \in \{0, 1\}^n$$

where x^k is the solution of feasible NLP sub-problems K_f, or else the solution of the feasibility problem:

$$\min Z = u$$

s.t.

$$\psi_j(x) + b_j^T y^k \leq u \qquad \text{(E1)}$$
$$x \in X$$

for infeasible NLP sub-problems, K_{inf}.

The pseudo IP master problem for the GBD is given as follows:

$$\min Z_{\text{GBD}}^k = c^T y + \alpha$$

s.t.

$$\phi(x^k) + \sum_{j \in J} \lambda_j^k [\psi_j(x^k) + b_j^T y] - \alpha \leq 0 \quad k \in K_f$$

$$\sum_{j \in J} \pi_j^k [\psi_j(x^k) + b_j^T y] \leq 0 \quad k \in K_{\text{inf}} \qquad \text{(MB)}$$

$$Ay \leq a$$

$$\alpha \in R^1, x \in X, y \in \{0, 1\}^n.$$

Assume that MOA is solved by one Benders iteration. By fixing the binary variables, y, to y^K where $K = |K_{\text{iter}}|$, the following LP is obtained:

$$\min Z_{OA}^k = c^T y^K + \alpha$$

s.t.

$$\left.\begin{array}{l} \phi(x^k) + \nabla\phi(x^k)^T(x - x^k) - \alpha \leq 0 \\ \psi_j(x^k) + \nabla\psi_j(x^k)^T(x - x^k) + b_j^T y^K \leq 0 \quad j \in J \end{array}\right\} k \in K_{\text{iter}} \text{ (LMB)}$$

$$\alpha \in R^1, x \in X.$$

When (LMB) is feasible, its Kuhn–Tucker conditions are as follows:

$$1 - \sum_{k=1}^{K} \mu_k = 0$$

$$\sum_{k=1}^{K} \mu_k \nabla\phi(x^k) + \sum_{k=1}^{K} \sum_j \lambda_j^k \nabla\psi_j(x^k) = 0 \qquad \text{(E2)}$$

where μ_k and λ_j^k are non-negative multipliers. Since at least one of the α constraints are active at the optimal solution and:

$$\phi(x^K) + \nabla\phi(x^K)^T(x - x^K) - \alpha = 0, \qquad \text{(E3)}$$

we can choose $\mu_K = 1$ and $\mu_k = 0$ for $k \neq K$. Also at y^K, the Kuhn–Tucker conditions of (MIP) are the following:

$$\nabla\phi(x^K) + \sum_j \hat{\lambda}_j^k \nabla\psi_j(x^K) = 0 \qquad \text{(E4)}$$

Hence, we can set in (E2) $\lambda_j^K = \hat{\lambda}_j^K$ and $\lambda_j^k = 0$ for $k \neq K$.

The Benders cut from the LP in (LMB) is given by:

$$L^K = \alpha + \mu_k[\phi(x^K) + \nabla\phi(x^K)^T(x - x^K) - \alpha]$$
$$+ \sum_{j \in J} \lambda_j^K [\psi_j(x^K) + \nabla\psi_j(x^K)^T(x - x^K) + b_j^T y]. \qquad \text{(E5)}$$

Since $\mu_K = 1$, and from (E4):

$$L^K = \phi(x^K) + \sum_{j \in J} \lambda_j^K [\psi_j(x^K) + b_j^T y] \qquad \text{(E6)}$$

which is identical to a cut in GBD for feasible sub-problems in the master (MB).

The cut for infeasible sub-problems follows a very similar reasoning with the only difference being that (LMB) is solved as a feasibility problem in order to generate the multipliers π_j^k. □

APPENDIX F

Proof on Lower Bounds of Logic-based Benders Decomposition vs GBD

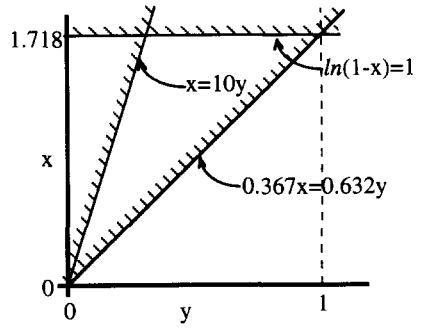

Fig. F1. Plot of the constraints for Example 4.

Lemma. Given K sub-problems of problem (MIP) for fixed y^k, $k=1,\ldots,K$:

$$\begin{aligned} &\min Z = c^T y + \phi(x) \\ &\text{s.t.} \\ &\psi_j(x) \le 0 \\ &x \le My \\ &x \in R^n, y \in \{0,1\}^m \end{aligned} \tag{MIP}$$

where $\phi(x)$ and $\psi_j(x)$ are convex and x has upper bound M. The Benders cut obtained from the linearizations of the active constraints is equivalent to the cut obtained from GBD.

Proof of Lemma. Consider K sub-problems in (MIP) for which linearization of the active constraints are determined. For y^k, define the set, J_A^k ($J_A^k \subseteq J$), containing the active constraints such that $J_A^k = \{j \mid \psi_j = 0\}$ in sub-problem k. Then, the linearization of (MIP) is given by:

$$\begin{aligned} &\min Z = \alpha \\ &\text{s.t.} \\ &c^T y + \phi(x^k) + \nabla\phi(x^k)^T(x - x^k) - \alpha \le 0 \quad k = 1,\ldots,K \\ &\psi_j(x^k) + \nabla\psi_j(x^k)^T(x - x^k) \le 0 \quad j \in J_A^k, \quad k = 1,\ldots,K \\ &x \le My \\ &\alpha \in R^1, x \in R^n, y \in \{0,1\}^m. \end{aligned} \tag{F1}$$

For $y = y^k$, the Lagrangian for the problem in (F1) is given by;

$$\begin{aligned} L = \alpha &+ \sum_{k=1}^{K} \mu_k [c^T y + \phi(x^k) + \nabla\phi(x^k)^T(x - x^k) - \alpha] \\ &+ \sum_{k=1}^{K} \sum_{j \in J_A^k} \lambda_j^k [\psi_j(x^k) + \nabla\psi_j(x^k)^T(x - x^k)] \\ &+ \sum_i \pi_i [x_i - M_i y_i] \end{aligned} \tag{F2}$$

where μ_k, λ_j^k and π_i are non-negative multipliers.

The Kuhn–Tucker conditions for problem (F1) at fixed y^k are:

$$\begin{aligned} &1 - \sum_{k=1}^{K} \mu_k = 0 \\ &\sum_{k=1}^{K} \mu_k \nabla\phi(x^k) + \sum_{k=1}^{K} \sum_{j \in J_A^k} \lambda_j^k \nabla\psi_j(x^k) + \sum_i \pi_i = 0. \end{aligned} \tag{F3}$$

Assume that at the optimal solution exactly one of the α-constraints is active for $k = K$:

$$\phi(x^K) + \nabla\phi(x^K)^T(x - x^K) - \alpha = 0. \tag{F4}$$

We can then choose $\mu_K = 1$ and $\mu_k = 0$ for $k \neq K$. Also at y^K, the Kuhn–Tucker conditions of (MIP) are the following:

$$\nabla\phi(x^K) + \sum_{j \in J_A^K} \hat{\lambda}_j^K \nabla\psi_j(x^K) = 0. \tag{F5}$$

Therefore, we can set $\lambda_j^K = \hat{\lambda}_j^K$ and $\lambda_j^k = 0$ for $k \neq K$ in (F2). The Benders cut from the LP in (F1) for y^K is then given by:

$$\begin{aligned} L^K = \alpha &+ \mu_K [c^T y + \phi(x^K) + \nabla\phi(x^K)^T(x - x^K) - \alpha] \\ &+ \sum_{j \in J_A^K} \lambda_j^K [\psi_j(x^K) + \nabla\psi_j(x^K)^T(x - x^K)] \\ &+ \sum_i \pi_i [x_i - M_i y_i]. \end{aligned} \tag{F6}$$

Since $\mu_K = 1$, $\psi_j(x^K)$, and from (F5):

$$L^K = c^T y + \phi(x^K) + \sum_i \pi_i [x_i - M_i y_i]. \tag{F7}$$

The Benders' cut in (F7) is identical to the cut generated by GBD for $y = y^K$ in problem (MIP). □

Theorem 2. The master problem of the logic-based generalized Benders decomposition (MB^k) yields tighter lower bounds than the master problem of GBD applied to problem (MIP).

Proof. Problem (MIP) can be formulated as the following disjunctive problem:

$$\begin{aligned} &\min Z = \sum_i \gamma_i + \phi(x) \\ &\text{s.t.} \\ &\begin{bmatrix} Y_i \\ \psi_j(x) \le 0 \quad j \in J_i \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ x = 0 \\ c_i = 0 \end{bmatrix} \quad i \in D \\ &x \in X, \quad Y_i \in \{\text{True}, \text{False}\}^m. \end{aligned} \tag{DP}$$

The MILP master problem of (DP) (see Appendix B) is given by:

$$\begin{aligned} &\min \hat{Z} = \alpha \\ &\text{s.t.} \\ &c^T y + \phi(x^k) + \nabla\phi(x^k)^T(x - x^k) - \alpha \le 0 \quad k = 1,\ldots,K \\ &\nabla\psi_j(x^k)^T x \le [-\psi_j(x^k) + \nabla\psi_j(x^k)^T x^k] y_i \quad j \in J_i, \\ &\qquad k = 1,\ldots,K, i \in D \\ &\alpha \in R^1, \quad x \in R^n, \quad y \in \{0,1\}^m. \end{aligned} \tag{F8}$$

for $y = y^K$, the Lagrangian for the problem in (F8) is given by:

$$\begin{aligned} \hat{L} = \alpha &+ \sum_{k=1}^{K} \mu_k [c^T y + \phi(x^k) + \nabla\phi(x^k)^T(x - x^k) - \alpha] \\ &+ \sum_{k=1}^{K} \sum_{j \in J_i} \lambda_j^k [\nabla\psi_j(x^k)^T x \\ &- [-\psi_j(x^k) + \nabla\psi_j(x^k)^T x^k] y_i]. \end{aligned} \tag{F9}$$

The Kuhn–Tucker conditions for the problem are:

$$1-\sum_{k=1}^{K}\mu_k=0$$

$$\sum_{k=1}^{K}\mu_k\nabla\phi(x^k)+\sum_{k=1}^{K}\sum_{j\in J_i}\lambda_j^k\nabla\psi_j(x^k)=0. \tag{F10}$$

The Benders cut for problem (F8) is determined by using the same idea in the proof of lemma:

$$\hat{L}^K=c^Ty+\phi(x^K)+\sum_{j\in J_i}\lambda_j^K[\nabla\psi_j(x^K)^Tx^K$$
$$-[-\psi_j(x^K)+\nabla\psi_j(x^K)^Tx^K]y_i]. \tag{F11}$$

To establish a relationship between the cut generated by the logic-based Benders, $\hat{L}^K$ in (F11) and the cut generated from GBD L^K in (F7), we have that $\hat{L}^K=\hat{Z}(y)$ and $L^K=Z(y)$. From (F1) and (F8), it follows that for sufficiently large value of M_i, $Z(y)\leq\hat{Z}(y)$. Hence, $L^K\leq\hat{L}^K$. Therefore, logic-based Benders produces tighter lower bounds than GBD. □

The above theorem can be illustrated with the following example.

Example 4.

Consider the problem given in the algebraic modeling framework as:

$$\min Z=3y-2x$$
s.t.
$$\ln(1+x)\leq 1 \tag{F12}$$
$$x\leq 10y$$
$$x\geq 0,\ y=0,1.$$

When the binary variable, y, is fixed to 1, the NLP sub-problem has an optimal objective function of $Z^U=-0.436564$ and the continuous variable is $x=1.718282$. Since the upper bound constraint is not active at the optimal solution its Lagrange multiplier, λ, is zero; the Lagrange multiplier for the nonlinear constraint is $\lambda=5.437$. The only term in the GBD cut is the objective function, and the following IP master problem is formulated:

$$\min Z^L=\alpha_{\mathrm{GBD}}$$
s.t.
$$\alpha_{\mathrm{GBD}}\geq 3y-3.436564 \tag{F13}$$
$$\alpha_{\mathrm{GBD}}\in R^1,\ y=0,1.$$

The master problem predicts a lower bound of $Z^L=-3.436564$ with binary variable equal to 0. The optimal solution of the second NLP sub-problem has an upper bound of $Z^U=0.0$ with $x=0$. In this sub-problem, the upper bound constraint is active with $\lambda=2$. The second IP master problem is formulated as follows:

$$\min Z^L=\alpha_{\mathrm{GBD}}$$
s.t.
$$\alpha_{\mathrm{GBD}}\geq 3y-3.436564 \tag{F14}$$
$$\alpha_{\mathrm{GBD}}\geq -17y$$
$$\alpha_{\mathrm{GBD}}\in R^1,\ y=0,1.$$

The second master problem finds the optimal solution with $Z^L=-0.436564$ and the binary variable equal to one. There are two integer points for this problem, and GBD had to visit both of them before confirming the optimal solution.

The problem in (F12) can be modeled in disjunctive form as follows:

$$\min Z=c-2x$$
s.t
$$\begin{bmatrix} Y \\ \ln(1+x)\leq 1 \\ c=3 \end{bmatrix}\vee\begin{bmatrix} \neg Y \\ x=0 \\ c=0 \end{bmatrix} \tag{F15}$$
$$c,x\geq 0,\ Y=\mathrm{True, False}.$$

The logic-based Benders algorithm starts with the same initial NLP sub-problem as in GBD obtained by fixing the Boolean variable to True. The solution to the first NLP sub-problem has an objective function value $Z^U=-0.436564$ and the continuous variable $x=1.718282$. Linearizing the nonlinear constraint as in (F8) yields:

$$\min Z=\alpha_{OA}$$
s.t.
$$\alpha_{OA}\geq 3y-2x$$
$$0.367879x\leq 0.632121y \tag{F16}$$
$$\alpha_{OA}\in R^1,\ x\geq 0,\ y=0,1.$$

Solving the LP in (F16) for $y=1$ yields the following Benders master problem:

$$\min Z^L=\alpha_{LB}$$
s.t.
$$\alpha_{LB}\geq -0.436839y+0.000275 \tag{F17}$$
$$\alpha_{LB}\in R^1,\ y=0,1$$

which has an optimal solution at $y=1$ with $Z^L=-0.436564$, and hence convergence is achieved in one iteration. The geometrical interpretation is shown in Fig. F1. Note that the upper bound constraint $x\leq 10y$ yields a very loose representation of the feasible region. In contrast, (F16) involves the tighter constraint $0.367879x\leq 0.632121y$, which in addition is active at $y=1$. Thus, the logic-based Benders method can predict a stronger lower bound requiring only one iteration.