ELSEVIER

Computers and Chemical Engineering 23 (1999) 555–565

Computers & Chemical Engineering

LOGMIP: a disjunctive 0–1 non-linear optimizer for process system models

Aldo Vecchietti, Ignacio E. Grossmann *

Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Received 11 February 1998; received in revised form 3 July 1998

Abstract

Discrete-continuous non-linear optimization models are frequently used to formulate problems in process system engineering. Major modeling alternatives and solution algorithms include generalized disjunctive programming and mixed integer non-linear programming (MINLP). Both have advantages and drawbacks depending on the problem they are dealing with. In this work, we describe the theory behind LOGMIP, a new computer code for disjunctive programming and MINLP. We discuss a hybrid modeling framework that combines both approaches, allowing binary variables and disjunctions for expressing discrete choices. An extension of the logic-based outer approximation (OA) algorithm has been implemented to solve the proposed hybrid model. Computational experience is reported on several examples, which are solved using disjunctive, MINLP and hybrid formulations.

1. Introduction

Mathematical programming models for addressing problems in process systems engineering have been extensively used over the last decade. Many applications in the synthesis and design of process networks have been formulated as mixed integer non-linear program (MINLP) models. These models assume an algebraic representation of the equations, and discrete variables are mainly restricted to 0–1 values. The current methods to solve this type of optimization problems are: branch and bound, generalized benders decomposition (GBD) and outer approximation (OA). An overview of these methods, the relationships between them, and references in process engineering applications can be found in Grossmann and Kravanja (1995). Recently Turkay and Grossmann (1996) have presented logic-based algorithms in which the discrete-continuous problem is modeled as a generalized disjunctive program. This model involves logic disjunctions with non-linear equations and pure logic relations. The main advantages of generalized disjunctive programs in structural flow-sheet optimization are its robustness and computational efficiency when compared to algebraic MINLP models and algorithms. This approach for modeling discrete-continuous non-linear problems is based on the work by Raman and Grossmann (1994) who investigated linear disjunctive problems. Starting with the disjunctive programming representation, a subset of the disjunctions is converted into algebraic mixed integer equations using the "w-MIP" representability criterion. This is a theoretical characterization that establishes conditions of equivalence between the disjunctions and mixed integer algebraic form. A solution algorithm restricted to the linear case was also presented. The above modeling schemes provide several alternatives and solution methods for the same problem. Depending on the representation that is selected, the computational efficiency and robustness to achieve the solution can be greatly affected.

In this work a hybrid modeling formulation for discrete-continuous non-linear problems for process system engineering is proposed. The model can involve disjunctions, binary variables and integer or mixed-integer constraints. It will be shown that from this formulation, the algebraic and the disjunctive formulation can be derived as particular cases. For the case of the hybrid formulation, we also introduce a new solution algorithm. Through the solution of several examples, we show that using disjunctions in the problem formu-

* Corresponding author. Fax: +1-412-268-7139.

0098-1354/99/$ - see front matter
PII: S0098-1354(98)00293-2

lation is often a better alternative for problems where avoiding zero flows is an important issue, or where big-M constraints yield poor relaxations. The hybrid representation is convenient when the w-MIP criteria applies to some disjunctions but not to all, or when it is not natural to express the entire model in terms of disjunctions and logic relations.

The above ideas we have implemented in LOGMIP, a new computer code for solving discrete-continuous non-linear optimization problems in which the problems can be modeled with either the algebraic, disjunctive or hybrid formulation. The program has a model recognition routine to check the model type, such that: if it is in the MINLP algebraic form, the OA/ER/AP algorithm by Viswanathan and Grossmann (1990) is applied; for the disjunctive case the logic-based OA by Turkay and Grossmann (1996); and the proposed algorithm for the hybrid problems. LOGMIP has been developed to provide a rather general modeling framework and solution tool for solving disjunctive, algebraic MINLP or hybrid non-linear optimization problems. It represents a new tool that is currently not available.

2. Motivation

In mixed integer linear program (MILP) optimization virtually all existing codes assume that the problem will be specified through 0–1 and continuous variables, and linear algebraic equations and inequalities. However, modeling of a MILP problem in this form has been recognized as a major bottleneck for the successful application of these techniques (see Nemhauser & Wolsey, 1998). The major reason is that alternate models can often be specified, leading usually to very different computational requirements for the LP-based branch and bound methods, that are largely due to the different tightness of the various formulations. In the case of MINLP this problem becomes even more acute given that these models require the convergence of non-linear equations even if the discrete choices render these equations to be redundant. The best example of this is perhaps when one tries to apply existing MINLP codes that require an algebraic description with 0–1 variables (e.g. DICOPT++ by Viswanathan & Grossmann, 1990) in the structural optimization of process flow-sheets. In this case units that are not selected require nevertheless that their corresponding mass and energy equations be satisfied, which often leads to singularities.

The significance of logic-based disjunctive optimization techniques that were reviewed in the introduction section, is that they offer the potential of overcoming the above cited difficulties, particularly for MINLP problems, as has been demonstrated by Turkay and Grossmann (1996). On the other hand it is not clear that it is always advantageous to solve a non-linear discrete optimization problem through disjunctions and Boolean variables. Therefore, ideally one would like to develop a modeling system that has the flexibility of accommodating a pure algebraic description as in conventional MINLP, a logic-based representation with disjunctions and Boolean variables, or a hybrid representation that combines both representations. Since there is no system to our knowledge that can accomplish these objectives, it is the main purpose of this paper to describe the development of the code LOGMIP that offers the capability of modeling and solving discrete/continuous optimization problems in the three forms. While LOGMIP at this point still does not cover all possible modeling cases (e.g. disjunctions with multiple terms, logic expressed in symbolic form), it is a useful tool in its own right that can also shed insights on the performance of the algebraic, disjunctive and hybrid models.

2.1. Hybrid model formulation

The purpose of this section is to provide a general representation for the hybrid model. In the hybrid formulation the discrete decisions are modeled by disjunctions and binary variables. Boolean variables are introduced in the model to establish if a term in a disjunction is true or false. The binary variables appear in linear form and where no equations are involved. The hybrid model has the following form:

$$\min Z = \sum c_i + f(x) + d^T y$$

$$\text{s.t.} \quad g(x) \le 0$$

$$r(x) + Dy \le 0$$

$$Ay \ge a$$

$$\begin{bmatrix} Y_i \\ h_i(x) \le 0 \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ B^i x = 0 \\ c_i = 0 \end{bmatrix}$$

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

$$x \in \mathbb{R}^n,\ y \in \{0,1\}^q,\ Y \in \{\text{True, False}\}^m,\ c_i \ge 0 \qquad \text{(PH)}$$

Where x and c_i are continuous variables, y represents the 0–1 variable, Y_i the Boolean variable to indicate if a disjunction is true or false, $\Omega(Y)$ is the logical relationship between Boolean variables, $g(x)$ represents the linear/nonlinear inequality that holds independent of the discrete choices, $f(x)$ is the linear/nonlinear objective function, $r(x) - Dy \le 0$ corresponds to the general mixed integer algebraic equation or to original disjunctions that were transformed into algebraic equations through the "w-MIP" criterion, $Ay \ge a$ is a set of integer inequalities and $d^T y$ are linear cost terms.

Problem PH is different to the one presented by Raman and Grossmann (1994) in the disjunctive terms. In our case they are restricted to two terms: if the Boolean variable used to handle the disjunction is true, then all the equations in the disjunction and a fixed charge apply; otherwise, if it is false, a subset of the continuous variables and the fixed charge is set to zero. The form of this disjunction is the one proposed by Turkay and Grossmann for modeling the optimal synthesis of process networks. It will be shown through the solution of some other examples that the form of that disjunction is applied to problems other than the synthesis of process networks.

From model (PH) it is possible to obtain the algebraic representation for a MINLP problem:

$$\min Z = f(x) + d^T y$$

$$\text{s.t.} \quad g(x) \leq 0$$

$$r(x) + Dy \leq 0$$

$$Ay \geq a$$

$$x \in \mathbb{R}^n,\ y \in \{0, 1\}^q \qquad \text{(PA)}$$

The algebraic model (PA) has only binary variables to express the discrete choices. The Boolean variables, the disjunctions and the fixed charges in the objective function corresponding to the disjunctions are eliminated from the model (PH).

On the other hand, if the model is expressed only through disjunctions, the representation is equivalent to the one presented by Turkay and Grossmann (1996), who proposed the logic-based outer approximation (OA) and the generalized benders decomposition (GBD) methods to solve a problem represented in this way. The disjunctive model has the following form:

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

$$\text{s.t.} \quad g(x) \leq 0$$

$$\begin{bmatrix} Y_i \\ h_i(x) \leq 0 \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ B^i x = 0 \\ c_i = 0 \end{bmatrix}$$

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

$$x \in \mathbb{R}^n,\ Y \in \{\text{True, False}\}^m,\ c_i \geq 0 \qquad \text{(PD)}$$

In the disjunctive representation, all the discrete choices are expressed with disjunctions and Boolean variables. The terms involving binary variables and mixed-integer constraints have been eliminated from model (PH). In structural flow-sheet optimization the disjunctive representation (PD) offers a more natural way to represent the problem.

3. Solution algorithm

To solve the problem in the hybrid representation as in (PH) an extension of the logic-based OA algorithm is proposed, to handle both disjunctions and mixed-integer algebraic equations. The algorithm, like its predecessors, decomposes the original problem into two sub-problems, the non-linear program (NLP) and the master MILP sub-problems. First, fixing the binary (y^k) and Boolean variables (Y_i) we obtain the following NLP problem:

3.1. Non-linear program sub-problem

$$\min Z = \sum c_i + f(x) + d^T y$$

$$\text{s.t.} \quad g(x) \leq 0$$

$$r(x) \leq -Dy^k$$

$$\left.\begin{matrix} h_i(x) \leq 0 \\ c_i = \gamma_i \end{matrix}\right\} \quad \text{if } Y_i = \text{True}$$

$$\left.\begin{matrix} B^i x = 0 \\ c_i = 0 \end{matrix}\right\} \quad \text{if } Y_i = \text{False} \qquad \text{(NLP)}$$

It should be noted that in the solution of the NLP sub-problem, the dimensionality is reduced because only the equations whose Boolean variables are true apply. Therefore, non-linear equations with zero value variables are avoided reducing difficulties with numerical singularities. On the other hand, depending on the problem, more than one initial NLP sub-problem should be solved, in order to set up the first MILP master problem. This MILP must contain linearizations of all non-linear equations in the disjunctions, to predict new binary and Boolean variable values for the next NLP. An interesting issue arises at the initialization step. While the Boolean variables should be fixed for each initial NLP sub-problem to be solved, the binary variables can be fixed or relaxed for each set of fixed Booleans, and remain constant or variable through the different initial NLPs. For some problems, e.g. synthesis of process networks, the minimum number of initial NLP sub-problems and the Boolean values, can be determined solving a modified set-covering algorithm for CNF propositional logic (Turkay & Grossmann, 1996). In other cases these values should be provided by the user.

The hybrid linear master sub-problem is obtained by the linearizations of the disjunctions and non-linear constraints at the solution point of the NLP sub-problem. Instead of working with a master problem involving linear disjunctions, it is transformed into algebraic form by using the convex-hull representation for the linear disjunctions proposed by Balas (1985). The original set of binary variables is increased by m, the

number necessary to replace the Boolean variables by 0–1 variables. In this way the master MILP problem can be solved with the conventional branch and bound method. This algebraic master MILP sub-problem has the following form.

3.2. Mixed integer linear program master sub-problem

$$\min Z_L = \alpha + \sum c_i y + d^T y$$

$$\text{s.t.} \quad f(x^l) + \nabla f(x^l)^T (x - x^l) \le \alpha$$

$$g(x^l) + \nabla g(x^l)^T (x - x^l) \le 0$$

$$r(x^l) + \nabla r(x^l)^T (x - x^l) + Dy \le 0$$

$$\nabla h(x^l)^T x \le (-h(x^l) + \nabla h(x^l)^T x^l) y, \quad l = 1, \ldots, L$$

$$Ay \ge a$$

$$Ey \ge e$$

$$\alpha \in \mathbb{R}^1, \; x \in \mathbb{R}^n, \; y \in \{0, 1\}^{q+m} \qquad \text{(MILP)}$$

where for simplicity we assume that all variables for a given disjunction are set to zero if it is false (see Turkay & Grossmann, 1996). The constraint $Ey \ge e$ are integer inequalities corresponding to the logic relationship between the Boolean variables ($\Omega(Y)$). After solving the master MILP new values for the binary and Boolean variables are obtained for the next iteration.

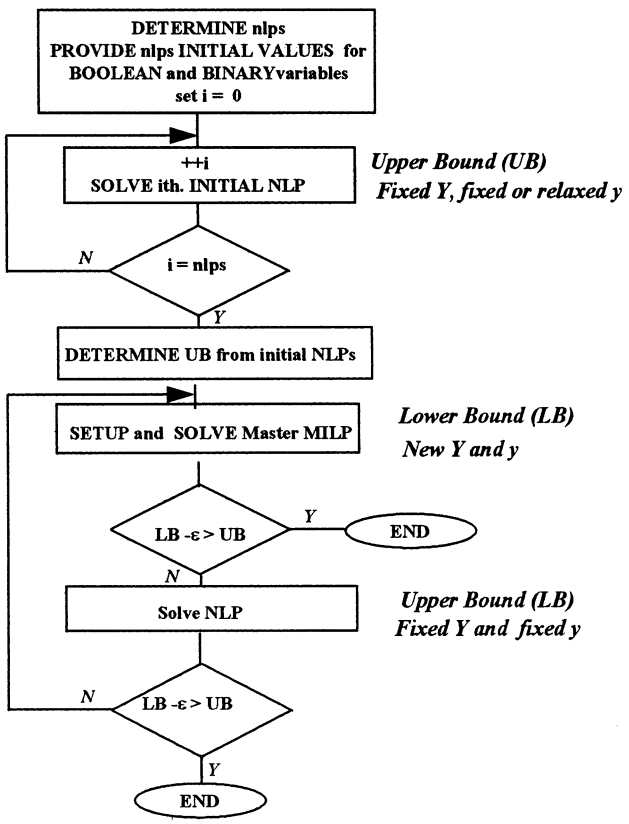


Fig. 1. Algorithm flow chart.

As can be seen in the algorithm flowchart, Fig. 1, the number of initial NLP problems to be solved (nlps) has to be determined first. This is needed in order to provide linearizations for all non-linear equations involved in the problem. This is an important issue to set up the first master MILP sub-problem. Depending on the problem, the number of initial NLP sub-problems can be determined in a systematic way or simply specified by the user. For the synthesis of a process network the modified set-covering algorithm for CNF propositional logic proposed by Turkay and Grossmann (1996) is used. Having identified the initial NLP sub-problems, the Boolean variables have to be fixed for each problem, and the binary variables can be fixed or relaxed, according to which strategy is specified. After solving the initial NLP sub-problems, we can obtain a valid upper bound from this set. Then we set-up the MILP master problem to predict the Boolean and binary variables values for the next NLP, and a lower bound. If the upper and the lower bound lie within a tolerance we stop; otherwise the iterations continue until convergence is achieved.

4. Overview of LOGMIP

LOGMIP is a computer code written in C allowing the specifications of disjunctions in the problem formulation. LOGMIP can be seen as a layer over the GAMS modeling language. The GAMS input/output library has been used for linking LOGMIP to GAMS. The GAMS modeling language is used to write the model in terms of disjunctions and algebraic equations. For specifying the disjunctions, the capability of the GAMS language that controls the domain of definition of the equation is used. The domain of definition is controlled by a condition for which the dollar sign is used. The following example shows the expressions for the first disjunction in the example 1 (see Appendix A: process superstructure) in the GAMS input file for LOGMIP:

$$\text{UNIT\_1BAL\$(BOOLE('1') EQ 1)}. . \exp(X(\text{'3'}) - 1 - X(\text{'2'}) = E = 0;$$

$$\text{UNIT\_1COS\$(BOOLE('1') EQ 1)}. . \; C(\text{'1'}) = E = 5;$$

$$\text{UNIT\_1NX2\$(BOOLE('1') EQ 0)}. . \; X(\text{'2'}) = E = 0;$$

$$\text{UNIT\_1NX3\$(BOOLE('1') EQ 0)}. . \; X(\text{'3'}) = E = 0;$$

$$\text{UNIT\_1NCO\$(BOOLE('1') EQ 0)}. . \; C(\text{'1'}) = E = 0;$$

As can be seen the expression for the disjunction is strongly typed due to using a language created to define

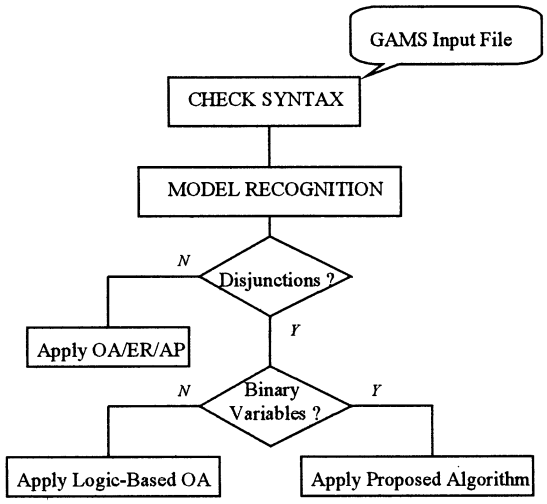


Fig. 2. LOGMIP flow chart.

a model in algebraic form. A language to express disjunctions in a natural way for the representation of optimization models is a subject of our current research.

The logic relations between the Boolean variables are presently handled in LOGMIP as inequalities. A PROLOG program developed to transform the logical expression into inequalities is used (Tourn, 1995). The program input consists of a file with propositions as shown below:

$$(P1 \wedge P2) \vee P3 \Rightarrow P4 \vee P5$$

the output is a file with equations and inequalities expressed as equations in the form of GAMS language. With the PROLOG program each proposition clause is translated to the equivalent mathematical linear form. The output file is included in the GAMS input file of LOGMIP. For example, for the proposition shown above the equivalent output looks like:

PUREINT_1 . .

$$Y('1') + Y('2') - Y('4') - Y('5') = L = 1;$$

PUREINT_2 . . $Y('3') - Y('4') - Y('5') = L = 0;$

where $Y('1')$ to $Y('5')$ are 0–1 variables. Future implementations of LOGMIP will have an automatic link with this PROLOG program.

In LOGMIP the model can be specified in the form of models (PA), (PD) and (PH). LOGMIP has a model recognition routine that works as follows: if no disjunctions are detected in the model, that means we are in the presence of a MINLP model and the OA/ER/AP algorithm is applied. If disjunctions are detected, LOGMIP finds if the model contains binary variables or not. In that way it can decide which algorithm has to be applied, the logic-based OA for the disjunctive model (PD) or the proposed algorithm for the hybrid model (PH). Fig. 2 shows the major steps involved in LOGMIP. Since the solution algorithms require the solution of NLP and MILP master problems, the I/O GAMS library is used to set up and solve the sub-problems. One aspect that needs attention in the implementation of the algorithms involving disjunctions is that the size of the matrix for the NLP sub-problem may change every major iteration. A flexible and efficient approach to handle the changes in the matrix size has to be implemented.

The program has been written in C to assure portability to other platforms. After supplying a GAMS input file with the discrete-continuous model, the input file syntax is checked through the GAMS language compiler. If it is correct, the control is transferred to LOGMIP.

5. Examples

A set of synthesis and design problems of different size have been solved with LOGMIP to illustrate the impact of formulating problems in the three different forms discussed in this paper. The detailed models and data can be found in Appendix A.

5.1. Example 1: process superstructure

The first example corresponds to a superstructure of eight chemical processes (Turkay & Grossmann, 1996). The model is described by non-linear equations that relax as convex constraints. The objective of this example is to find the configuration with the minimum cost. Fig. 3 shows the superstructure for this example.

We have modeled this example in three different ways: algebraic, disjunctive and hybrid. The disjunctive model is given in Appendix A. The transformation from the disjunctive model into the hybrid model has taken place for the linear models representing the process 3, 4 and 5. These linear models are "w-MIP" representable, and therefore these models have been written in algebraic equation form. The rest of process models that are non-linear remain as disjunctions. The

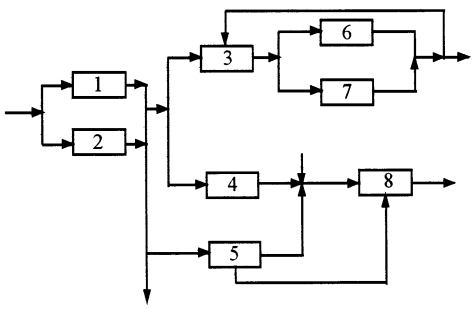


Fig. 3. Process superstructure

Table 1
Results process superstructure

	PA	PD	PH
Initialization	Relaxed	Fixed	Fixed
Constraints	32	52	52
Variables	33	42	42
Discrete variables	8	8	8
Objective value	68	68	68
Execution time (s)	2.7	2	0.74
Iterations	4 major	3 nlp, 1 major[a]	2 nlp, 1 nlp[a]

[a] One major iteration ⇒ 1 nlp + 1 master milp.

algebraic MINLP model can be found in Turkay and Grossmann (1996). The results obtained with this example are shown in Table 1.

In this example the constraints were increased in the disjunctive and the hybrid models because the propositional logic equations (relationships between the Boolean variables) were added to those models. The variables added correspond to the fixed charge costs. Even with this increase in constraints and variables the results obtained with these models are encouraging. The number of iterations have been reduced in the disjunctive and hybrid models compared to the algebraic MINLP. Therefore, the execution time is also reduced. Due to the small execution times the performance is better analyzed through the number of major iterations. In the disjunctive case the difference with respect to the algebraic model lies in the master MILP number of problems executed. Only one master problem is required in the disjunctive model. The hybrid model has allowed to reduce by one the initial number of NLPs to be executed to set up the first master problem.

5.2. Example 2: FTIR-spectroscopy example

The next example corresponds to a simultaneous model structure determination and parameter estimation for a FTIR-spectroscopic example by Brink and Westerlund (1995). It has a non-linear objective function subject to linear constraints. For the PA and the PD models the non-linear objective function has been transformed into a constraint. For the PD model the linear constraints have been transformed into disjunctions. The results obtained are shown in Table 2.

Table 2
Results FTIR-spectroscopy

	PA	PA	PD
Initialization	Relaxed	Fixed	Fixed
Constraints	39	39	102
Variables	69	69	99
Discrete variables	30	30	30
Objective value	13.98	13.98	13.98
Execution time (s)	55	100	7
Iterations	6 major[a]	11 major[a]	4 major*

[a] One major iteration ⇒ 1 nlp + 1 master milp.

For the FTIR-spectroscopic example the results show an impressive performance of the disjunctive model compared to the algebraic MINLP models. The disjunctive model is superior not only in the number of iterations, but also in the execution time, which has been reduced by a factor of eight compared to the algebraic model with relaxed initialization. The explanation for this behavior is that the convex-hull representation for the linear disjunctions in the master MILP gives a tighter relaxation, improving the prediction of lower bounds.

5.3. Example 3: synthesis HDA process

The third example is a structural flow-sheet optimization problem (Kocis & Grossmann, 1989). For this example, the HDA process was modeled in the algebraic and the disjunctive forms. There is a significant number of non-convex non-linear equations in this problem. The augmented penalty strategy (Viswanathan & Grossmann, 1990) was applied to solve it. No w-MIP representable disjunctions were found in this model, therefore, the hybrid model does not apply for this example. The objective for this optimization problem is to obtain the HDA flow-sheet with maximum profit. The superstructure for the process can be seen in Fig. 4. The results obtained with this problem are shown in the Table 3.

Two initial configurations were given to solve this example in the disjunctive representation. Therefore, two initial NLPs have been solved to obtain the linearizations for all non-linear constraints, in order to set up the first MILP master problem. For this case the disjunctive model (PD) obtained a solution with higher profit than the algebraic models (PA), presumably because zero flows are avoided in (PD). The profit was increased with the disjunctive model by approximately 9.5% compared to the algebraic model with relaxed MINLP as an initial point, and 2.4% compared to the algebraic with a fixed initial point.

5.4. Example 4: design of a multi-product batch plant

The last example is a batch plant design with multiple units in parallel and intermediate storage tanks (Ravemark, 1995). The problem consists of determining the volume of the equipment, the number of units in parallel, and the volume and location of the intermediate storage tanks. The objective is to minimize the plant investment cost. To ensure rigorous lower bounds the equations and constraints were convexified. The problem was modeled in both algebraic and hybrid repre-

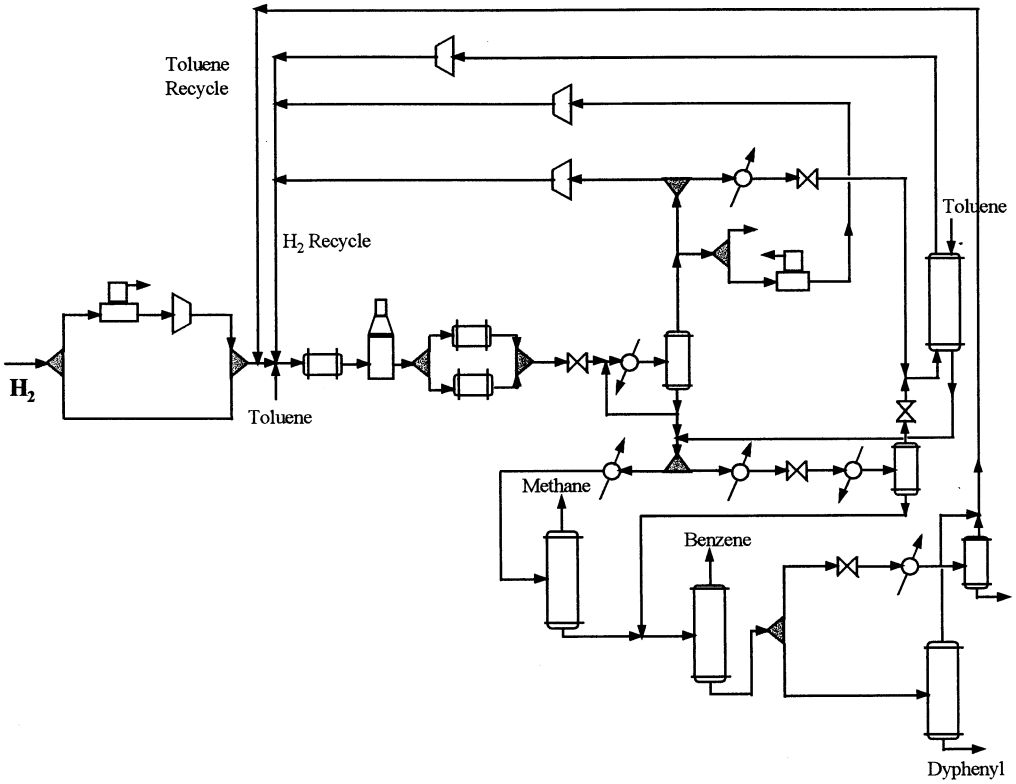


Fig. 4. HDA superstructure

sentation. In the hybrid representation, the disjunctions correspond to the storage tank volume equations and the batch size equations. The example corresponds to a batch plant with five products and six stages. The results are shown in Table 4.

For this example, which is difficult to solve in the algebraic mode, the results obtained for the hybrid model are very encouraging. The reduction in time and number of iterations for this example are very significant. The number of sub-problems solved with respect to the better algebraic model (with relaxed MINLP initialization) has been reduced by more than one half. The execution time has been reduced more than 3 times compared to the same model.

6. Remarks

The results obtained from the solution of the four examples show that for some cases, the use of disjunctions is a better modeling alternative compared to the algebraic MINLP. The model and algorithm to be applied for a particular problem depends on the type of equations and constraints the problem has. For convex problems, the disjunctive and hybrid models have shown a very significant improvement compared to the algebraic case. For non-convex problems the main advantage seems to lie in the fact that the disjunctive and hybrid models are less likely to get trapped into poor suboptimal solutions. In contrast, the algebraic model may be trapped more easily due to the difficulties with non-convex models where flows are set to zero. The reason for this behavior has largely to do with the fact that the quality of the linearizations of vanishing units is often poor, thereby decreasing the quality of the global approximation of the master problem.

7. Conclusions

In this work, the solution of hybrid models, with disjunctions and binary variables, for discrete-continu-

Table 3
Results HDA plant

	PA	PA	PD
Initialization	Relaxed	Fixed	Fixed
Constraints	719	719	737
Variables	722	722	717
Discrete variables	13	13	14
Objective value	5304.8	5671.4	5810.8
Execution time (s)	348	293	280
Iterations	1 nlp, 3 major[a]	1 nlp, 2 major[a]	2 nlp, 1 major[a]

[a] One major iteration ⇒ 1 nlp + 1 master milp.

Table 4
Results batch plant

	PA	PA	PH
Initialization	Relaxed	Fixed	Fixed
Constraints	186	186	187
Variables	112	112	113
Discrete variables	53	53	53
Objective value	261883	261883	261883
Execution time (s)	287	616	80
Iterations	1 nlp, 10 major[a]	1 nlp, 20 major[a]	1 nlp, 4 major[a]

[a] One major iteration ⇒ 1 nlp + 1 master milp.

ous non-linear optimization problem has been investigated. From this model the pure disjunctive or algebraic model can be derived. The computer code LOGMIP (acronym of logic mixed integer program) has been developed to deal with this representation. The problem input for LOGMIP can be written in three different forms: hybrid, disjunctive or algebraic form. For the hybrid model an extension of the logic based OA algorithm has been presented. LOGMIP has been written in C and linked to GAMS. The possibility to define different starting points has been added. All these capabilities make LOGMIP an important tool for the solution of non-linear discrete-continuous optimization problems. The novelty of this program is the capability to handle disjunctions. No other non-linear computer code has been reported in the open literature that can solve problems of this type. The results obtained in the solution of several examples have shown that the disjunctive and hybrid representation outperform the algebraic MINLP in terms of computational time and quality of solutions.

Acknowledgements

The authors would like to acknowledge financial support from NSF for US–Argentina Cooperative Research under Grant INT-9724823.

Appendix A

Example 1: process superstructure

Objective function

$$\min Z = \sum c_i + x_2 - 10x_3 + x_4 - 10x_5 - 40x_9 + 15x_{10} + 15x_{14} + 80x_{17} + 65x_{18} + 25x_{19} - 60x_{20} + 35x_{21} - 80x_{22} - 35x_{26} + 122$$

Material balances:

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

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

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

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

$$x_6 - x_7 - x_8 = 0$$

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

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

Flow specifications:

$$x_{10} - 0.8x_{17} \le 0$$

$$x_{10} - 0.4x_{17} \ge 0$$

$$x_{12} - 5x_{14} \le 0$$

$$x_{10} - 2x_{14} \ge 0$$

Disjunctions:

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

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

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

$$\begin{bmatrix} Y_7 \\ \exp(x_{22}/1.2) - 1 - x_{21} = 0 \\ c_7 = 4 \end{bmatrix} \vee \begin{bmatrix} \neg Y_7 \\ x_{21} = x_{22} = 0 \\ c_7 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_8 \\ \exp(x_5/1.2) - 1 - x_4 = 0 \\ c_8 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_8 \\ x_{10} = x_{17} = x_{18} = 0 \\ c_8 = 0 \end{bmatrix}$$

Original disjunctions transformed to equations*(*w-MIP *criteria):*

$$1.5x_9 - x_8 = x_{10} = 0$$

$$c_3 = 6y_3$$

$$1.5(x_{12} + x_{14}) - x_{13} = 0$$

$$c_4 = 10y_4$$

$$x_{15} - 2x_{16} = 0$$

$$c_5 = 6y_5$$

Propositional logic	⇒	Integer equations
$y_1 \Rightarrow y_3 \vee y_4 \vee y_5$		$-y_1+y_3+y_4+y_5 \geq 0$
$y_2 \Rightarrow y_3 \vee y_4 \vee y_5$		$-y_2+y_3+y_4+y_5 \geq 0$
$y_3 \Rightarrow y_1 \vee y_2$		$y_1+y_2-y_3 \geq 0$
$y_4 \Rightarrow y_1 \vee y_2$		$y_1+y_2-y_4 \geq 0$
$y_5 \Rightarrow y_1 \vee y_2$		$y_1+y_2-y_5 \geq 0$
$y_4 \Rightarrow y_8$		$-y_4+y_8 \geq 0$
$y_5 \Rightarrow y_8$		$-y_5+y_8 \geq 0$
$y_3 \Rightarrow y_6 \vee y_7$		$-y_3+y_6+y_7 \geq 0$
$y_6 \Rightarrow y_3$		$y_3-y_6 \geq 0$
$y_7 \Rightarrow y_3$		$y_3-y_7 \geq 0$
Specifications		
$y_1 \underline{\vee} y_2$		$y_1+y_2=1$
$y_3 \underline{\vee} y_4$		$y_3+y_4=1$
$y_6 \underline{\vee} y_7$		$y_6+y_7=1$

Appendix B

Example 2: FTIR-spectroscopy example

n = wave number, m = spectra number
Matrix A (n,m)

1	2	3	4	5	6	7	8
0.0003	0.0764	0.0318	0.0007	0.0534	0.0773	0.0536	0.0320
0.0007	0.0003	0.0004	0.0009	0.0005	0.0009	0.0005	0.0003
0.0066	0.0789	0.0275	0.0043	0.0704	0.0683	0.0842	0.0309
0.0044	0.0186	0.0180	0.0179	0.0351	0.0024	0.0108	0.0052
0.0208	0.0605	0.0601	0.0601	0.0981	0.0025	0.0394	0.0221
0.0518	0.1656	0.1491	0.1385	0.2389	0.0248	0.1122	0.0633
0.0036	0.0035	0.0032	0.0051	0.0015	0.0094	0.0015	0.0024
0.0507	0.0361	0.0433	0.0635	0.0048	0.0891	0.0213	0.0310
0.0905	0.0600	0.0754	0.1091	0.0038	0.1443	0.0420	0.0574
0.0016	0.0209	0.0063	0.0010	0.0132	0.0203	0.0139	0.0057

c = component
Matrix C(c, m)

502	204	353	702	0	1016	104	204
97	351	351	351	351	700	201	97
0	22	8	0	14	22	14	8

MINLP formulation
Objective function

$$\min \left\{ w_j + 2\sum y_{i,j} \right\}$$

s.t.

$$w_j = \sum e_k^T R^{-1} e_k$$

$$P_{i,j} - P_{i,j,\max} y_{ij} \leq 0$$

$$P_{i,j,\min} y_{i,j} - P_{i,j} \leq 0 y_{i,j} \in \{0, 1\}$$

where:

$$e_k = c_k - Pa_k$$

R = covariance matrix = identity matrix.
R is assumed to be known, it is equal to the identity matrix at first problem iteration.

Disjunctive model
Objective function

$$\min \left\{ w_j + 2\sum c_{i,j} \right\}$$

s.t.

$$w_j = \sum e_k^T R^{-1} e_k$$

$$\begin{bmatrix} Y_{i,j} \\ P_{i,j,\min} \leq P_{i,j} \leq P_{i,j,\max} \\ c_{i,j} \end{bmatrix} \vee \begin{bmatrix} \neg Y_{i,j} \\ P_{i,j} = 0 \\ c_{i,j} = 0 \end{bmatrix}$$

$$Y_{i,j} \in \{\text{True, False}\}$$

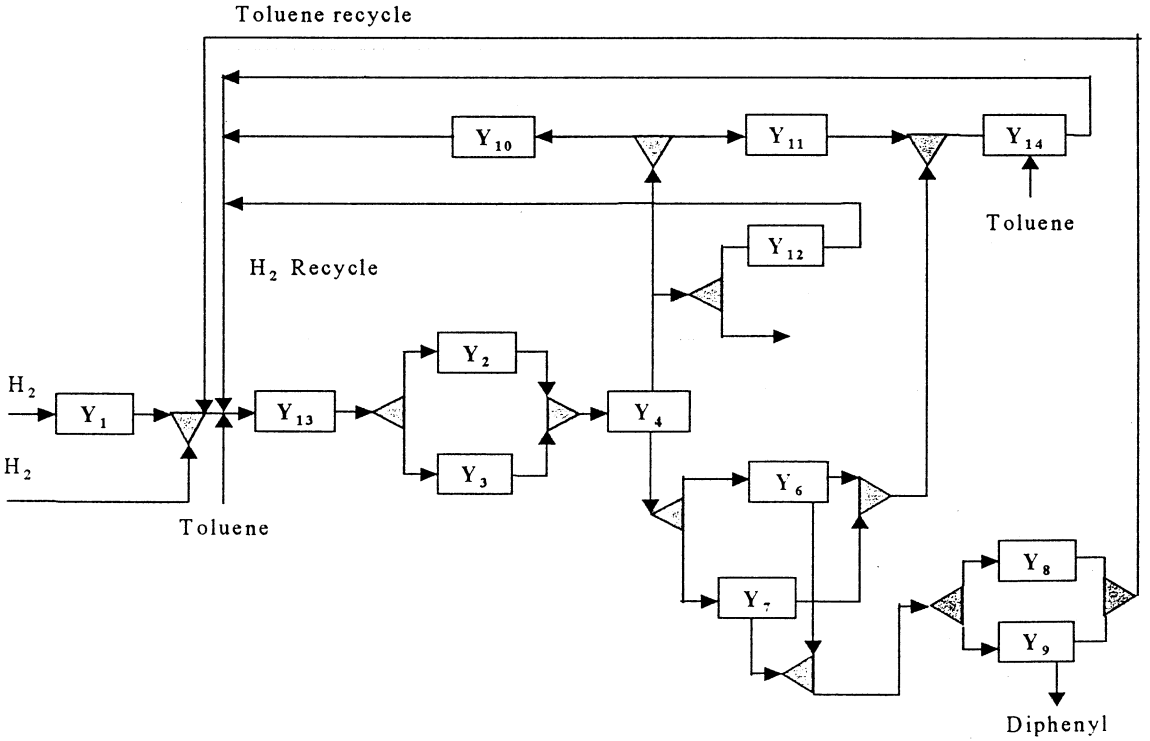
where:

$$e_k = c_k - Pa_k$$

Appendix C

Example 3: synthesis HDA process

Due to the size of the HDA optimization problem it is not possible to include here the equations and constraints corresponding to the model. In the figure below about the HDA process we are showing the way the pieces of equipment have been grouped.



Each group is handled by a particular Boolean variable. The pieces of equipment belonging to a particular set have to be in sequence such that if one is selected the others are also selected. Not all sets represent an alternative for the process. Some of them, for example set 4 and 13 have their Boolean variables fixed to True. The set of units has the advantage that the amount of Boolean variables necessary for the problem diminishes. In this way a generic disjunction for a particular set i has the following representation:

$$\begin{bmatrix} Y_i \\ \text{Apply equations and constraint belonging to every unit in set } i \\ \text{Apply a fixed charge every unit in set } i \end{bmatrix}$$

$$\vee \begin{bmatrix} \neg Y_i \\ \text{Some variables of the units in group } i \text{ are set to zero} \\ \text{Fixed charges are set to zero} \end{bmatrix}$$

Appendix D

Example 4: design of a multi-product batch plant

In this model parallel units operating in-phase and out-of-phase and the sizing of intermediate storage tanks are considered where:

$i = 5$ = products: A, B, C, D, E;
$j = 6$ stages;
H = horizon time = 6000 h
Q_i = production rate of product i: $A = 250000; B = 150000; C = 180000; D = 160000; E = 120000$

$S_{i,j}$ = size factor for product i at stage j

	1	2	3	4	5	6
A	7.9	2.	5.2	4.9	6.1	4.2
B	0.7	0.8	0.9	3.4	2.1	2.5
C	0.7	2.6	1.6	3.6	3.2	2.9
D	4.7	2.3	1.6	2.7	1.2	2.5
E	1.2	3.6	2.4	4.5	1.6	2.1

$T_{i,j}$ = processing time of product i at stage j

	1	2	3	4	5	6
A	6.4	4.7	8.3	3.9	2.1	1.2
B	6.8	6.4	6.5	4.4	2.3	3.2
C	1.0	6.3	5.4	11.9	5.7	6.2
D	3.2	3.0	3.5	3.3	2.8	3.4
E	2.1	2.5	4.2	3.6	3.7	2.2

To convexify the equations the following transformations have been made:

$$b_{i,j} = \log B_{i,j} \quad , \quad n_j = \log N_j$$
$$m_j = \log M_j \quad , \quad E_i = \exp e_i$$
$$v_j = \log V_j \quad , \quad v_{Tj} = \log V_{Tj}$$

Objective function

$$\min C = 250 \sum \exp(n_j + m_j + 0.6 v_j) + 150 \sum \exp(0.5 v_{Tj})$$

s.t.

$$\begin{bmatrix} Y_j \\ v_{Tj} \geq \log S_j^* + B_{i,j+1} \\ v_{Tj} \geq \log S_j^* + B_{i,j} \\ b_{i,j} - b_{i,j+1} \leq \log S_{i,j}^* \\ b_{i,j} - b_{i,j+1} \leq -\log S_{i,j}^* \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ v_{Tj} = 0 \\ b_{i,j} - b_{i,j+1} = 0 \end{bmatrix}$$

$$v_j = \log S_{ij} + b_{i,j} - n \quad \forall_i, \forall_j$$

$$e_i \geq \log (T_{i,j}) - b_{i,j} - m_j \quad \forall_i, \forall_j$$

$$H \geq \sum Q_i \exp(e_i)$$

$$n_j = \sum \text{coef}_k yn_{k,j} \quad \forall_j$$

$$\sum yn_{k,j} = 1$$

$$m_j = \sum \text{coef}_k ym_{k,j} \quad \forall_j$$

$$\sum ym_{k,j} = 1$$

$$y_n, y_m \in \{0, 1\}, \quad Y \in \{\text{True, False}\}$$

N_j: number of units in parallel in phase at stage j
M_j: number of units in parallel out-of-phase at stage j
V_j: unit size of stage j
V_{Tj}: size of intermediate storage tank between stage j and $j+1$
S_j^*: size factor for intermediate storage tank (constant = 5)
$S_{i,j}^*$: size factor for stages (constant = 3)
coef_k: constant coefficients for parallel units
B_{ij}: batch size product i at stage j
E_i: inverse production rate for product i
Q_i: production requirement of product i
Y_j: Boolean variables equal True if the intermediate storage tank is allocated at position j for product i; 0 otherwise
ynk, *ymk* binary variables for selecting number of units operating in and out of phase

References

Balas, E. (1985). Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM Journal on Algorithms for Discrete Mathematics*, *6*, 466–486.

Brink, A., & Westerlund, T. (1995). The joint problem of model structure determination and parameter estimation on quantitative IR-spectroscopy. *Chemomet. and Intellig. Lab. Sys.*, *29*(1), 29–36.

Grossmann, I. E. & Kravanja, Z. (1995). Mixed-integer non-linear programming techniques for process synthesis engineering. *Computers and Chemical Engineering*, 19(Suppl.), S189–204.

Kocis, G., & Grossmann, I. E. (1989). A modeling and decomposition strategy for the MINLP optimization of process flow-sheets. *Computers and Chemical Engineering*, *13*(7), 797–819.

Nemhauser, G. L., & Wolsey, L. A. (1998). *Integer and combinatorial optimization*. New York: Wiley.

Ravemark, E. (1995). *Optimization models for design and operation of chemical batch processes*. Ph.D. thesis.

Raman, R., & Grossmann, I. E. (1994). Modelling and computational techniques for logic based integer programming. *Computers and Chemical Engineering*, *18*(7), 563–578.

Turkay, M., & Grossmann, I. E. (1996). Logic-based algorithms for the optimal synthesis of process networks. *Computers and Chemical Engineering*, *20*(8), 959–978.

Tourn, M. (1995). A PROLOG program for transforming logic CNF statements into inequalities for GAMS. Users manual, Department of Chemical Engineering, Carnegie Mellon University. Pittsburgh, PA.

Viswanathan, J. V., & Grossmann, I. E. (1990). A combined penalty function and outer-approximation method for MINLP optimization. *Computers and Chemical Engineering*, *14*(7), 769–778.