# Computational Experience With DICOPT Solving MINLP Problems in Process Systems Engineering

by

G.R. Kocis and I.E. Grossmann

EDRC-06-43-88

# COMPUTATIONAL EXPERIENCE WITH DICOPT

# SOLVING MINLP PROBLEMS

# IN PROCESS SYSTEMS ENGINEERING*

**Gary R. Kocis and Ignacio E. Grossmann** **

**Department of Chemical Engineering**

**Carnegie Mellon University**

**Pittsburgh, PA 15213**

**U.S.A.**

**January, 1988**

*** This work was performed at Cornell University in 1986-1987 while Ignacio Grossmann was the Mary Upson Visiting Professor and Gary Kocis was a Visiting Scholar in the School of Chemical Engineering.**

**** Author to whom correspondence should be addressed.**

## ABSTRACT

**This paper discusses an implementation of the Outer-Approximation/Equality-Relaxation algorithm for solving MINLP problems that arise in process systems engineering. The computer code DICOPT has been developed utilizing state-of-the-art optimization tools and a powerful modelling language. Computational experience in solving fifteen MINLP problems with DICOPT is reported. Applications include design of batch processes, structural flowsheet optimization, planning, flexibility, and reliability problems. Results show that DICOPT provides a very efficient tool for solving MINLP problems.**

## INTRODUCTION

Many optimization models for design and operation of process systems require the treatment of both continuous and discrete (mainly 0-1) variables. The presence of nonlinearities in the models gives rise to mixed-integer nonlinear programming (MINLP) problems, which in the past have proved to be very expensive and difficult to solve. It is shown in this paper, however, that this situation has undergone a considerable change and that reliable and efficient algorithms now exist

The development of the Outer-Approximation (OA) algorithm by Duran and Grossmann (1986a), its extension with the Equality-Relaxation (OA/ER) strategy by Kocis and Grossmann (1987a), coupled with advances in nonlinear program (NLP) solvers (e.g. MINOS, Murtagh and Saunders, 1985) and mixed-integer linear program (MILP) solvers (e.g. MPSX, IBM, 1979), has led to the capability of solving MINLP problems quite efficiently. These algorithms and techniques have been implemented in the computer program DICOPT (Discrete Continuous OPTimizer) that features as interface the modelling language GAMS (General Algebraic Modelling System, Kendrick and Meeraus, 1985) which greatly facilitates the algebraic formulation of MINLP problems.

Computational results are reported on fifteen MINLP problems that have been solved with DICOPT. These include applications to design of batch processes, structural flowsheet optimization, planning, flexibility, and reliability design. Computer times on an IBM-3090/600 are typically of the order of only 2-100 seconds for problems with up to 60 0-1 variables and 410 variables and constraints.

This paper also briefly discusses a number of points that arise from this study and that seem to be rather unique to MINLP problems. The points include the effect that different modelling schemes have not only on the algorithm's efficiency but also on the global optimality of the solution, the role of convexity in the quality of the predicted bounds, and aspects of computer implementation that have to do with the use of different optimizers. Future directions are also briefly discussed.

## OUTER-APPROXIMATION/EQUALITY-RELAXATION ALGORITHM

The OA/ER algorithm addresses MINLP problems of the general form:

$$
\begin{aligned}
& z = \min \; c^T y + f(x) \\
s.t. \quad & h(x) = 0 \\
& g(x) \le 0 \\
& Ax = a \qquad\qquad \text{(MINLP)} \\
& By + Cx \le d \\
& x \in X = \{x \mid x \in R^n, x^L \le x \le x^u\} \\
& y \in Y = \{y \mid y \in \{0,1\}^m, Ey \le e\}
\end{aligned}
$$

In the above formulation the binary variables y appear linearly, while nonlinearities are involved in the continuous variables **x** in the functions f(x), h(x), and g(x). The OA/ER algorithm, however, is not necessarily restricted to problems with this structure. MINLP problems which contain nonlinear functions in **y** can also be solved with the OA/ER algorithm as will be discussed later in the paper.

The OA/ER algorithm can be classified as a decomposition scheme in which the continuous optimization and the discrete optimization are performed separately. The continuous optimization is performed through NLP subproblems that arise for fixed choices of y in problem **(MINLP).** The discrete optimization is performed via an MILP master problem which is a linear approximation to **(MINLP),** The iterative bounding procedure in the OA/ER algorithm is described below and **the** steps are formally stated in Appendix A.

Initially, an iteration counter K is set to 1 and the binary variables are selected and temporarily fixed at $y^K$ yielding the following NLP subproblem:

$$z(y^K)=\min\ c^T y^K+f(x)$$

$$\text{s.t.}\quad h(x)=O$$

$$g(x)\leq 0 \qquad (NLP^K)$$

$$Ax=a$$

$$Cx \wedge d-By^*$$

$$x \in X$$

The solution to this NLP subproblem provides an upper bound on the optimum objective function of **(MINLP)** and the optimal values for the continuous variables $x^K$ in $(NLP^K)$.

Linear approximations to the nonlinear functions in **(MINLP)** are then derived at the solution point $x^K$ and the binary variables are freed yielding the MILP master problem $(M^K)$ (see Appendix B). Note that this master problem includes linearizations for the previous and current NLP solution points $x^k$, k=l,2,...K. Also, the linearizations for nonlinear equations are relaxed as inequalities through the sign of the lagrange multipliers. The discrete optimization is performed through this master problem to select values for the binary variables $y^{K+1}$ which minimize the objective function in the linearized space. The optimal objective function value of the master problem provides a lower bound on the solution to problem (MINLP). If this lower bound is less that the upper bound from the NLP subproblem then the binary variables are fixed at $y^{K+1}$ and the resulting NLP subproblem is solved. The solution of NLP subproblems and MILP master problems is continued until the lower bound exceeds the best upper bound. At this point the optimal objective function value of **(MINLP)** is the minimum objective function of all NLP subproblems solved and the optimal solution point is the corresponding point x* , y*.

Since the NLP subproblem may not have a feasible solution for the particular choice of $y^K$, it is often convenient to introduce a non-negative slack variable u for constraint violations and a large scalar p. The objective function and inequality constraints in $(NLP^K)$ are replaced in a modified formulation **$(SNLP^K)$** as follows:

***a) Objective function-*** $z(y^K)=\min_x\ c^T y^K+f(x) + pw$

***b) Nonlinear inequalities-*** $g(x) \leq u$

***c) Linear mixed inequalities-*** $Cx \leq d-By^{\wedge} + u$

In this way, if a feasible solution to problem ($NLP^K$) exists then the slack variable u will be zero and formulation ($SNLP^K$) reduces to that of ($NLP^K$). If a feasible solution does not exist, then the objective function in ($SNLP^K$) will produce continuous variables $x^K$ that minimize the violation of the inequality constraints.

The assumption that the binary variables y appear linearly in problem (MINLP) can be relaxed in the OA/ER algorithm. MINLP formulations containing nonlinear terms in y, \|/(y), or nonlinear functions of both x and y, <I>(x,y), can be treated through reformulation to yield a problem which is linear in y. This can be done simply by defining new continuous variables $x^1$ = y, so that \|/(y) and <(>(x,y) can be formulated as nonlinear functions of continuous variables $V(x^x)$ and (J^XjX[1]). The additional equations $x^1$ = y are then linear in y. This, however, is equivalent to treating y(y) and <|>(x,y) directly as nonlinear functions which are linearized in x,y at the NLP subproblem solution points $x^k$,$y^k$, k=l,2,...K, to define the master problem.

It should be noted that the OA/ER is not the only method available for solving MINLP problems. Branch and bound techniques can be applied to MINLP problems as well as the Generalized Benders Decomposition (GBD) (Benders, 1962, Geoffrion, 1972). The latter is an iterative bounding algorithm which is similar to OA/ER except that its master problem is based on a dual representation of the NLP subproblems, and generally predicts weak lower bounds as compared to the OA/ER algorithm. The disadvantage of both branch and bound and GBD is that these methods typically require the solution of many NLP subproblems during the search for the optimal solution (see Duran and Grossmann, 1986a). The OA/ER algorithm was developed to minimize the number of NLP subproblems to be solved since this is often the bottleneck in MINLP problems. Among other methods that are currently under development for solving MINLP problems is the feasibility technique by Mawengkang and Murtagh (1986).

## DICOPT

As mentioned in the previous section, the OA/ER algorithm involves the alternating solution of NLP subproblems and MILP master problems. This algorithm has been implemented in the computer code DICOPT. MINOS (version 5.0 or 5.1) is used to solve the NLP subproblems and MPSX or ZOOM/XMP (Marsten, 1986) is used to solve the MILP master problems. A simplified flowchart of DICOPT is shown in Figure 1.

First the algebraic formulation of the MINLP problem is supplied to GAMS which creates the

formulation in a format recognizable to the NLP code (e.g. MINOS) and generates analytical gradients. The user must supply the initial values for the binary variables $y^1$ and also a starting point for the continuous variables x which MINOS uses in solving the subproblem ($NLP^1$).

The NLP solver is then called and returns a solution having one of three status conditions, optimal, infeasible, or unbounded* If the NLP subproblem is unbounded then the original MINLP problem is also unbounded and the OA/ER algorithm terminates. An infeasible status means that either the NLP subproblem is infeasible or that due to the starting point supplied to the solver, the problem appeared to be infeasible. Recall that the formulation ($SNLP^K$) can be used to avoid infeasibility due to inequality constraints and thus produce a point which satisfies the equality constraints while minimizing the violation of the inequalities. This is important when applying the equality-relaxation procedure because lagrange multipliers are required at a point which satisfies the nonlinear equalities in order to relax these equations to inequalities (see Kocis and Grossmann, 1987a).

The third possibility is that the optimal (or at least locally optimal) solution to subproblem **($NLP^K$)** was found. In this case, the upper bound on the solution to **(MINLP)** is updated as the minimum of the current upper bound and the objective function value from the NLP subproblem.

Following the NLP subproblem a call is made to a program which formulates the MILP master problem. The master problem contains all linear constraints from **(MINLP),** an accumulation of linear approximations at previous iterations of the nonlinear constraints in **(MINLP),** and a set of integer cuts which eliminate from consideration the previously analyzed points $y^k$, k=l,2,..K. If the NLP subproblem solution is optimal or feasible with respect to the nonlinear equality constraints then first-order linearizations derived at $x^K$,$y^K$ are included in the master problem. (Since gradient information is required by the NLP solver, this information is available for deriving the linearizations.) However, if for a given $y^K$ the resulting NLP subproblem has no point $x^K$ which satisfies the nonlinear equations, then the linear approximations are not added to the MILP formulation. The introduction of the integer cut will insure that the solution to the next master problem $y^{K+1}$ is different from $y^K$.

Having formulated the master problem, the next task in DICOPT is to call the appropriate MILP package. The MILP solver returns either the optimal integer solution (or an integer solution satisfying the specified tolerances) or an infeasible solution. If the status returned by the

MILP solver is the optimal integer solution, the objective function value then provides a lower bound on the solution to (MINLP). If the master problem is infeasible or the predicted lower bound is greater than the current upper bound, then the OA/ER algorithm terminates and the optimal solution to (MINLP) has the objective function value of the current upper bound and the optimal solution point given by the corresponding NLP subproblem solution x* , y*.

For the case when the lower bound from the master problem is less than the current upper bound, the binary variables y are temporarily fixed at the optimal value $y^{K+1}$ from the master problem defining the next NLP subproblem. The optimal values of the continuous variables x from the master problem are used as the starting point for the next NLP subproblem meaning that the user is required to supply a starting point for only the first NLP subproblem. Since the linearizations are accumulated as iterations proceed, the MILP master problem provides an increasingly good approximation to the original MINLP. Hence, the quality of the starting points supplied through the master problem increases making the subsequent NLP subproblems easier to solve. Thus, the MILP master problem in the OA/ER algorithm not only predicts strong lower bounds, but it also provides excellent initial guesses for the NLP subproblems.

The main components in DICOPT include GAMS, an NLP solver, an MILP solver, the code which formulates the master problem, and a control program which determines the flow between these four components. Efficiency and portability have been built into the computer implementation of DICOPT. Sparse matrix techniques as well as dynamic memory capabilities in GAMS are used to generate the linearizations in the MILP master problem.

Currently, versions of DICOPT exist for IBM mainframes (CMS), VAX (VMS), and IBM-PC (DOS). Since MINOS and ZOOM/XMP are written in FORTRAN, they are available on these three systems. However, MPSX is available only for IBM mainframes. The code which formulates the master problem and the code which performs the necessary logic were written in FORTRAN. The command files, which are very small, are written in REXX for IBM/CMS, DCL for VAX/VMS, and BAT for IBM-PC/DOS. It should be noted that due to the robustness and efficiency of MPSX, the IBM version is the most suitable for large scale MINLP problems.

To illustrate the use of the modelling language GAMS in DICOPT the algebraic formulation of a batch process design problem is given in Appendix C. Note the use of indexed sets which allows for a very compact problem formulation. In the solve statement given in the last line of

the Appendix C listing, the command MIDNLP stands for the use of the OA/ER algorithm for solving the MINLP problem. Also, note that in DICOPT the user need not be concerned with the details of the MINLP algorithm (e.g. supplying the master problem formulation) as is the case of APROS developed by Paules and Floudas (1987). Here all the steps of the particular MINLP algorithm (OA/ER or GBD) must be supplied in the GAMS input file.

## COMPUTATIONAL RESULTS

DICOPT, as implemented on the IBM/CMS system, has been tested on fifteen MINLP problems. A summary of problem size and computational results on an DBM-3090/600 (Cornell Theory Center) is given in Table L

Problem REAC corresponds to the selection of a two reactor configuration, while problems PLAN and EX3 are small planning examples where discrete choices are made pertaining to candidate processes for producing a set of desired final products, CAP1 and CAP2 are examples of the pure-integer quadratic programming formulation of the capital budgeting problem (Kettani and Oral, 1987). Problem EX4 addresses the optimal positioning of a new product in a multiattribute space (Duran and Grossmann, 1986a). REL is a reliability optimization problem from Henley and Kumamoto (1985) and FLEX is an MINLP formulation of the flexibility problem of a heat exchanger network through the active set strategy of Grossmann and Floudas (1987). BATCH5 through BATCH12 correspond to optimal design problems for multiproduct batch plants (see Kocis and Grossmann, 1987b) and problem TFY involves the retrofit design of heat exchanger networks. Finally, the problem FLOW is an example of a structural flowsheet optimization problem (see Kocis and Grossmann, 1987b). The detailed formulations are given in Kocis (1988) and are available upon request from the authors.

As shown in Table I, the size of the MINLP problems ranges from 2 to 60 0-1 variables, 0 to 410 continuous variables, and 1 to 421 constraints. Note that problem FLOW involves 140 nonlinear equations and problem EX4 25 nonlinear inequalities. The remaining problems exhibit fewer nonlinearities in the objective function or constraints. As can be seen, all the problems, except for the last one, required less than 30 seconds of CPU-time for solving both the NLP subproblems and MILP master problems. The time for the total overhead was always substantially less that the CPU-time for optimization. Also, the results in this table indicate that between 2 and 4 iterations are typically required.

From the results in Table I, it is clear that the performance of the OA/ER in DICOPT algorithm is quite impressive considering both the size and complexity of the different test problems. Note that although the MILP master problems often require more CPU-time than the NLP subproblems (especially when few nonlinearities are present), they have the important role of not only reducing the number of iterations, but also to make the solution of subsequent NLP subproblems easier to solve by supplying good initial points. In fact, in all these problems, once the first NLP problem converged there was no difficulty in subsequent iterations. Finally, the results show that in problem FLOW, which involves complex NLP subproblems due to the large number of nonlinearities, the MILP master problems require much less time that the NLP subproblems.

Another important point related to computational experience is the numerical solution of the MILP master problems. Since the constraints in the master problem are derived from linearizations, there is a tendency for the magnitude of the coefficients to cover a wide range of values making the numerical stability of the MILP solver important. Computational experience has indicated that MPSX outperforms ZOOM/XMP in solving the master problems both in efficiency and reliability. Results in Table II indicate that MPSX is more suitable for solving the MILP master problems as the size of the MINLP problem increases.

In addition to the problems reported in Table I, the OA/ER algorithm has been applied to the design of gas pipelines (Duran and Grossmann, 1986b), retrofit design of batch processes (Vaselenak et al, 1987), and the structural flowsheet optimization with heat integration (Kocis and Grossmann, 1987a). In all these problems the OA/ER algorithm required between 2 and 5 iterations. Effective use of the OA/ER algorithm in chemical engineering applications has also been recently confirmed by other researchers. Floudas and Paules (1987) developed an MINLP formulation for the synthesis of heat integrated distillation sequences. With the OA/ER algorithm they were able to solve the resulting convex MINLP problem in 3 to 6 iterations. Harsh (1987) developed a mixed-integer programming approach to the retrofit design of existing plants. He successfully implemented the OA algorithm in the commercial process simulator FLOWTRAN and performed a retrofit design study of an ammonia plant in only 2 iterations.

## EXTENSIONS

The OA/ER algorithm is guaranteed to find the global optimum of MINLP problems that involve convex objective function and inequalities, and quasiconvex relaxed equations. A difficulty, however, that may be encountered in MINLP problems is the presence of nonconvexities. In the OA/ER algorithm, nonconvexities can cause problems in two ways. First, since the procedure involves the solution of NLP subproblems, the presence of nonconvexities leads to the potential for local solutions in the subproblem ($NLP^K$). The global optimization of nonconvex NLP problems is an difficult task and rigorous algorithms currently do not exist

The second difficulty that nonconvexities can lead to occurs in the MILP master problem of the OA/ER algorithm. The rigorous guarantee of the global optimum to **(MINLP)** is based on the master problem providing a valid lower bound on the optimal objective function value. However, when nonconvexities are present, the linearizations in the master problem may not necessarily provide valid outer approximations to the nonlinear feasible region. Therefore, the lower bounds predicted by the MILP master may not be valid lower bounds and the global optimum can be cut off (see Kocis and Grossmann, 1987b).

In some special problems, nonconvexities can eliminated through convexifying transformations (e.g. log transformations as in Duran and Grossmann, 1986b). However, the identification of nonconvexities is often nontrivial for complex models and even if the nonconvexities are detected it is not always possible to convexity the problem. Thus, there is a need for a procedure which addresses the difficulty that nonconvexities cause in the MILP master problem.

The important point to take note of is that convexity in the nonlinear functions is a sufficient condition to guarantee the global optimum. When these conditions are not satisfied the OA/ER algorithm may or may not obtain the global optimum solution. However, computational experience by the authors has shown that in a good number of cases the OA/ER algorithm will find the global optimum solution for nonconvex MINLP problems. This would then suggest that a suitable strategy to tackle these problems is to solve them in two phases as has been suggested recently by Kocis and Grossmann (1987b). In the first phase the OA/ER algorithm is applied in its original form but with special local and global tests for the identification of nonconvex functions that may cause the global solution to be missed. If nonconvexities are not detected the

search is terminated. Otherwise, one proceeds to a second phase where linearizations are systematically modified in a new master problem so as to try to yield valid outer-approximations. Here the search is terminated at the point when no further improvements are found in the NLP subproblems.

The main advantage of the two-phase algorithm is that it can systematically and intelligently make use of all information from phase I to automatically identify nonconvexities in the MINLP problem that could prevent the OA/ER algorithm from finding the global optimum solution. However, no rigorous guarantee on global optimality can be given since no special structure on the functions has been assumed to guarantee validity of the modified outer-approximations and uniqueness of the solutions of the NLP subproblems. Finally, it is interesting to note that when nonconvexities are not involved in the MINLP, the strategy will be able to automatically identify this situation and terminate at phase I with the global optimum. The two-phase strategy has been recently implemented in DICOPT and tested successfully with several nonconvex test problems (see Kocis and Grossmann, 1987b) where the global optimum was found in 24 out of 28 cases as seen in Table HI.

Another interesting aspect is the effect that the model formulation of the MINLP can have on the OA/ER algorithm. The effect can be seen both in the NLP subproblem stage and the MELP master problem. Firstly, different algebraic formulations can have a great impact on the efficiency of the NLP solver. For instance, an NLP solver may find it difficult to find even a feasible solution for one NLP formulation, while for an equivalent formulation it may find the optimal solution very quickly. The latter will typically be the case when all the variables are properly bounded and scaled and the constraints are written in mostly linear form. Model formulation can also influence the quality of the lower bounds predicted by the MILP master problem. For example, in the linearization of some nonlinear functions (e.g. bilinearities), terms may vanish when variables take zero values and as a result, substantially cut into the feasible region. By redefining variables and setting non-zero lower bounds, one can often avoid these difficulties. However, the modelling of MINLP problems still remains an area that is not very well understood. This is currently a subject of our research work and will be reported in a future paper.

## CONCLUSIONS

While in the recent past, the solution of MINLP problems was regarded as a very difficult and expensive task, this situation has undergone a considerable change. The combination of recent developments in MINLP algorithms (e.g. Outer-Approximation algorithm and its extension with the Equality-Relaxation algorithm) along with the improved efficiency of NLP and MILP solvers and increased computing power has led to the capability to solve large scale mixed-integer nonlinear programming problems as was shown in this paper with the computer code DICOPT. Furthermore, the availability of modelling languages, such as GAMS, provides model builders with a powerful tool which allows for compact representation of large problem formulations of mixed-integer optimization models. Although there are still some unresolved issues, the ability to routinely perform MINLP calculations in process systems engineering (e.g. structural flowsheet optimization) is now at hand.

## ACKNOWLEDGEMENT

The authors would like to acknowledge financial support from the National Science Foundation under grant CPE-8351237 and partial support from the Engineering Design Research Center at Carnegie Mellon. We would like to thank Soren Nielsen, Tony Brooke, and Alexander Meeraus of the World Bank for their assistance and cooperation interfacing the OA/ER algorithm with the modelling language GAMS. The authors are also grateful to the Cornell Theory Center for access to their EBM-3090/600 supercomputer.

## Appendix A. Equality-Relaxation Algorithm

Based on the nonlinear programming subproblem ($NLP^K$) and the MELP master problem ($M^K$) presented in Appendix B, the steps in the outer-approximation/equality relaxation algorithm can be stated as follows (it is assumed that the NLP subproblems in Step 2 have a feasible solution):

Step 1 Select initial binary assignment $y^1$, set K=l.
Initialize lower and upper bounds, z^-∞ $z_U$=∞.

Step 2 Solve ($NLP^k$) for fixed $y^K$ yielding $z(y^K)$, $x^K$, and $X^K$.
If $z(y^K)$<Zjj, then set $y^*=y^K$, $x^*=x^K$, and Zu=$z(y^K)$.
Define the diagonal matrix $T^K$ as:

$$t_{ii}^k = \begin{cases} -1 \text{ if } \lambda_i^k < 0 \\ +l \text{ if } Xf > 0 \\ 0 \text{ if } \lambda_i^k = 0 \end{cases} \qquad *=l,2,...r$$

where ^fare the associated optimal lagrange multipliers for the nonlinear equations hj(x)=O, i=l,2...r, in the subproblem ($NLP^k$).

Step 3 Derive at $x^K$ the linear approximations for f(x), h(x), and g(x), and set up the master program given by problem ($M^K$) (see Appendix B). The linear coefficients and right hand side constants in ($M^K$) are then given by:

$$w^* = Vf(x^*)^r \quad w_o^* = Vf(x^*)^r\,[x^*] - f(x^*)$$

$$R^k = \nabla h(x^k)^T \quad r^k = \nabla h(x^k)^T\,[x^k]$$

$$S^* = Vg(x^*)^r \quad s^* = Vg(x^*)^r\,[x^*] - g(x^*)$$

Step 4 Solve the master program ($M^K$):

[a] If a feasible solution $y^{K+1}$ exists with objective value $z_L^K < z_U$; set K=K+1, go to Step 2.

[b] If z£^zy or no feasible solution exists, stop.
Optimal solution is Zy at y*,x*.

## Appendix B. MILP Master Problem ($M^K$)

$$
\begin{aligned}
& zf=\min \quad c^r y+\mu \\
s.t.\ & (w^k)^T x-\mu \le w^k{}_o \\
& T^k R^k x \le T^k r^k \qquad k=1,2...K \\
& S^k x \le s^k \\
& Ax=a \qquad (M^K) \\
& By+Cx \le d \\
& Ey \le e \\
& z_L^{K-1} \le c^r y+\mu \\
& x \in X \\
& y \in \{0,1\}^m \cap \{\text{integer cuts}\}^K \\
& \mu \in R^1
\end{aligned}
$$

where $z_L^{K-1}$ is the predicted lower bound at iteration K, ji is the largest linear approximation to the nonlinear objective function, and the integer cuts correspond to constraints which eliminate the assignments of binary variables analyzed at the previous K iterations:

$$
-\sum_{i \in B^k} y_i \le |B^k|-1 \qquad k=1,2...K
$$

where for any integer combination $y^k$, the index sets are such that $B^k=\{j:yj^=l\}$ and N^ljty^K)}. The lower bound $z_p^1$ of the constraint on $c^T y+n$ is introduced to expedite the solution of the MILP.

## Appendix C. Illustration of GAMS Modelling Language in DICOPT

```
$TITLE BATCH PROCESSING PROBLEM la.
*
*  MINLP FORMULATION OF EXAMPLE PROBLEM FROM:
*       Grossmann, I.E. and Sargent, R.W.H. , 1979.
*       Ind. Eng. Chem. Process Des. Dev. , 18: 343-348.

 SETS          I          PRODUCTS                        / A , B /
               J          STAGES                          / 1 * 3 /
               K          BINARY VARIABLES                / 1 * 2 /

 SCALAR        H          HORIZON TIME (HRS)              / 6000. /
               ALPHA      COST COEFFICIENT FOR UNITS      / 250.  /

 PARAMETERS    Q(I)       PRODUCTION RATE OF PRODUCT I           /
                            A = 200000. ,  B = 100000.           /

               YCOEFF(K)  COEFFICIENTS IN EQUATION: UNITS        /
                            1 = 1        , 2 = 2                 /

 TABLE         S(I,J)     SIZE FACTOR FOR PRODUCT I IN STAGE J

                        1         2         3
             A          2.        3.        4.
             B          4.        6.        3.

 TABLE         T(I,J)     PROCESSING TIME OF PRODUCT I IN STAGE J

                        1         2         3
             A          8.       20.        8.
             B         16.        4.        4.


 VARIABLES     Y(K,J)     BINARY VARIABLE DENOTING UNIT EXISTENCE
               V(J)       VOLUME OF STAGE J
               B(I)       BATCH SIZE OF PRODUCT I
               TL(I)      CYCLE TIME OF PRODUCT I
               N(J)       NUMBER OF UNIT IN PARALLEL STAGE J
               COST       TOTAL COST OF BATCH PROCESSING UNITS   ;

 BINARY   VARIABLES       Y(K,J)                                  ;
 POSITIVE VARIABLES     V(J) , B(I) , TL(I) , N(J)                ;

 EQUATIONS     VOL(I,J)     CALCULATE VOLUME OF STAGE J
               CYCLE(I,J)   CALCULATE CYCLE TIME OF PRODUCT I
               TIME         TIME CONSTRAINT
               UNITS(J)     NUMBER OF PROCESSING UNITS PER STAGE
```

```
              OBJ         OBJECTIVE FUNCTION DEFINITION        ;

VOL(I,J)..    V(J)  =G=  S(I,J) * B(I)                         ;
CYCLE(I,J).. N(J)  * TL(I)  =G=  T(I,J)                        ;
TIME..        SUM(I , Q(I) * TL(I) / B(I) )  =L=  H            ;
UNITS(J)..    N(J)  =E=  1. + SUM(K , YCOEFF(K) * Y(K,J) )     ;
OBJ..         COST  =E=  ALPHA * SUM(J , N(J) * V(J) ** 0.6)   ;


*  BOUNDS SECTION
V.LO(J)   =  250.   ;
V.UP(J)   =  2500.  ;
N.LO(J)   =  1.     ;
N.UP(J)   =  3.     ;

PARAMETERS   TLLOW(I)       LOWER BOUND ON TL(I)
             TLUPP(I)       UPPER BOUND ON TL(I)
             BLOW(I)        LOWER BOUND ON B(I)
             BUPP(I)        UPPER BOUND ON B(I)             ;
   TLLOW(I)  = SMAX(J, T(I,J) / N.UP(J) )                    ;
   TLUPP(I)  = SMAX(J, T(I,J) )                              ;
   BLOW(I)   = Q(I) * ( SMAX(J, T(I,J) / N.UP(J)) ) / H      ;
   BUPP(I)   = MIN( Q(I) , SMIN(J , V.UP(J) / S(I,J) ) )     ;

TL.LO(I)  =  TLLOW(I)                                        ;
TL.UP(I)  =  TLUPP(I)                                        ;
B.LO(I)   =  BLOW(I)                                         ;
B.UP(I)   =  BUPP(I)                                         ;


*  INITIAL POINT SELECTED FOR BINARY VARIABLES
Y.L(K,J)     = 0                                             ;
Y.L('1','1') = 1                                             ;
Y.L('1V3') = 1                                               ;

*  INITIAL POINT PROVIDED TO NLP SOLVER
N.L(J)    =  1. + SUM( K , YCOEFF(K)  *  Y.L(K,J) )          ;
B.L(I)    =  ( B.LO(I) + B.UP(I) ) / 2.                      ;
V.L(J)    = SMAX(I ,  S(I,J) * B.L(I) )                      ;
TL.L(I)   = SMAX(J ,  T(I,J) / N.L(J) )                      ;

MODEL   BATCH   / ALL / ;
Y.FX(K,J)  =  Y.L(K,J)                    ;
SOLVE BATCH USING MIDNLP MINIMIZING COST   ;
```

# References


[I] Benders, J. F.
Partitioning Procedures for Solving Mixed-variables Programming Problems.
*Numerische Mathematik* 4:238-252,1962.

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

[3] Duran, M. A., Grossmann, I. E.
A Mixed-Integer Nonlinear Programming Approach for Process Systems Synthesis.
*AIChE Journal* 32(4):592-606,1986b.

[4] Floudas, C. A., Paules G.E.
A Mixed-Integer Nonlinear Programming Formulation for the Synthesis of Heat Integrated Distillation Sequences,
presented at Annual AIChE Meeting (New York), 1987.

[5] Geoffrion, A. M.
Generalized Benders Decomposition.
*Journal of Optimization Theory and Applications* 10(4), 1972.

[6] Grossmann, I. E., Floudas, C. A.
Active Constraint Strategy for Flexibility Analysis in Chemical Processes,
*Comp. andChem. Engr.* ll(6):675-693,1987.

[7] Harsh, M. G.
The Development of Performance Models and Optimal Process Retrofits.
Master's thesis, Carnegie-Mellon University, 1987.

[8] Henley, E. J., Kumanoto, J. H.
*Designing for Reliability and Safety Control.*
Prentice-Hall Inc., Englewood Cliffs, NJ, 1985.

[9] IBM.
*IBM Mathematical Programming System Extended 1370 (MPSX1370), Basic Reference Manual.*
Technical Report, IBM, White Plains, NY, 1979.

[10] Kendrick, D., Meeraus, A.
*GAMS, An Introduction.*
Technical Report, Development and Research Department at the World Bank, Washington, DC, 1985.

[II] Kettani, O., Oral, M.
Equivalent Formulations of Nonlinear Integer Problems for Efficient Optimization,
submitted for publication (1987).

[12] Kocis, G. R., Grossmann, I. E.
Relaxation Strategy for the Structural Optimization of Process Flowsheets.
*Ind. Eng. Chem. Research* 26(9):1869-1880,1987a.

[13] Kocis, G. R., Grossmann, I. E.
Global Optimization of Nonconvex MINLP Problems in Process Synthesis, presented at Annual AIChE Meeting (New York), paper 96b, 1987b.

[14] Kocis, G.R.
*A Mixed-Integer Nonlinear Programming Approach to Structural Flowsheet Optimization.*
PhD thesis, Carnegie-Mellon University, 1988.

[15] Marsten,R.
*Users Manual for ZOOMIXMP*
The Department of Management Information Systems, University of Arizona, 1986.

[16] Mawengkang, H., Murtagh, B. A.
Solving Nonlinear Integer Programs with Large-Scale Optimization Software.
*Annals of Oper. Res.* 5:425-437,1986.

[17] Murtagh, B. A., Saunders, M. A.
*MINOS User's Guide.*
Technical Report SOL 83-20, Systems Optimization Laboratory, Department of Operations Research, Stanford University, 1985.

[18] Paules, G. E., Floudas, C A.
APROS: A Discrete-Continuous Optimizer for Solution of Mixed-Integer Nonlinear Programming Problems,
presented at ORSA/ITMS Meeting (St. Louis), 1987.

[19] Vaselenak, J. A., Grossmann, I. E., Westerberg, A. W.
Optimal Retrofit Design of Multiproduct Batch Plants.
*Ind. Eng. Chem. Research* 26:718-726,1987.

**Figure 1. OA/ER Implementation in DICOPT**

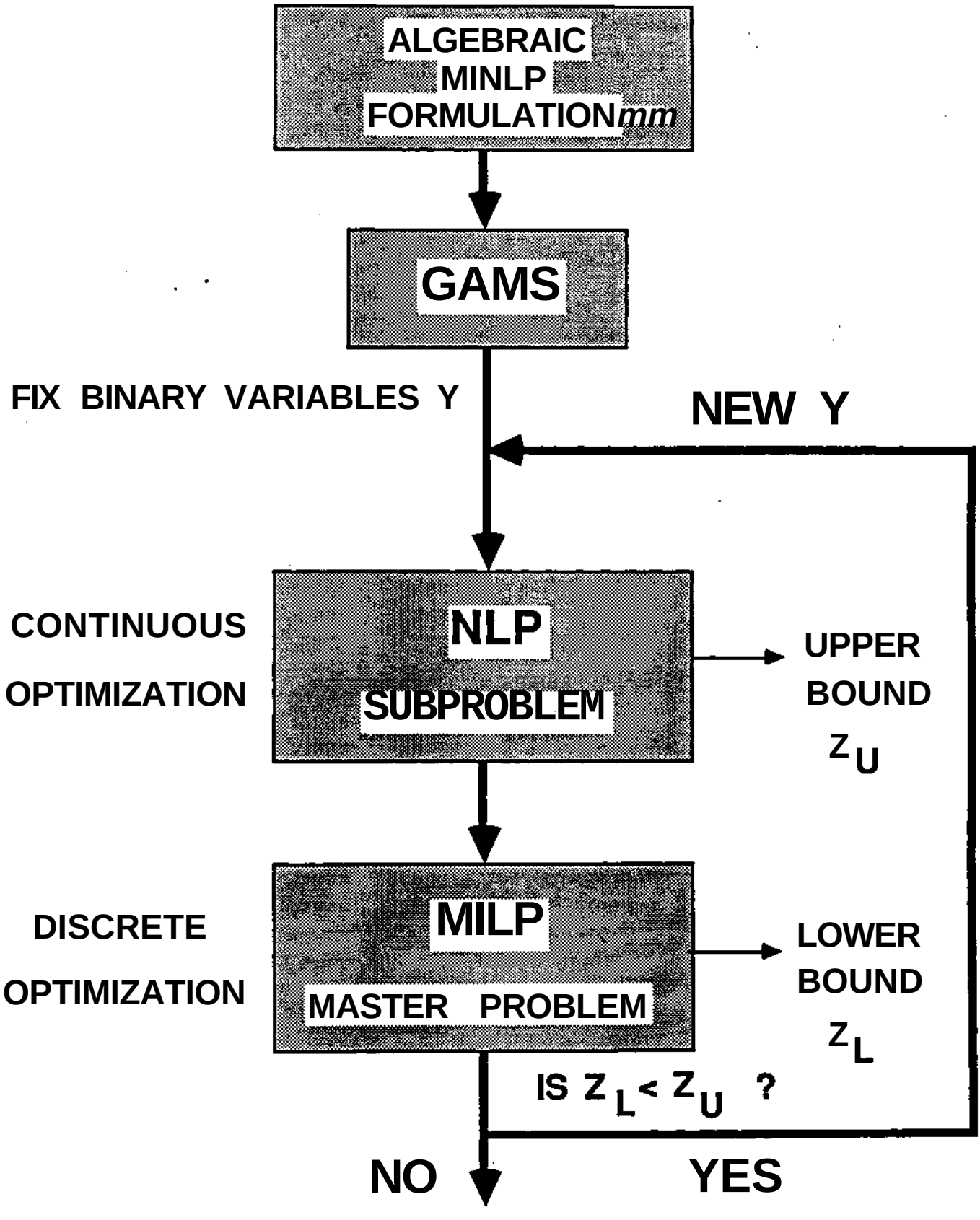

**Table** I. **Computational Results**

| Problem | Binary Variables | Cont. Variables | Constr. | Nonlin-earities[1] | Iterations | Total Time[2] | Solver Times[3] |
|---|---|---|---|---|---|---|---|
| REAC | **2** | **8** | **6** | **(L.2,0)** | **2** | 1.4 | 0.2/1.2 |
| PLAN | **3** | **7** | **8** | **(LAO)** | **2** | 1.4 | 0.2/1.2 |
| FLEX | **4** | **12** | **15** | **(LAO)** | **3** | 2.3 | 0.4/1.9 |
| CAP1 | **4** | **0** | **1** | **(N.0,0)** | **4** | 2.5 | 0.3/2.2 |
| CAP2 | **10** | **0** | **4** | **(N.0,0)** | **3** | 2.0 | 0.2/1.8 |
| EX3 | **8** | **25** | **30** | **(L.5,0)** | **3** | 2.5 | 0.4/2.1 |
| REL | **12** | **48** | **77** | **(N.8,0)** | **3** | 7.3 | 1.1/6.2 |
| EX4 | **25** | **5** | **30** | (N.0,25) | **3** | 8.2 | 1.4/6.8 |
| TFY | **14** | **34** | **53** | **0U8.O)** | **2** | 2.8 | 0.9/1.9 |
| BATCH5 | **24** | 22 | **73** | (NAD | **4** | 5.4 | 1.0/4.4 |
| BATCH8A | **40** | **32** | 141 | (NAD | **4** | 10.4 | 1.6/8.8 |
| BATCH8B | **40** | **32** | 141 | **(N,0,D** | **7** | 17.3 | 3.6/13.7 |
| BATCH8C | **40** | **32** | 141 | (NAD | **9** | 24.4 | 5.2/19.2 |
| BATCH12 | **60** | **40** | 217 | (NAD | **3** | 9.8 | 1.8/8.0 |
| FLOW | **9** | **410** | 421 | **(L.14Q.0)** | **3** | 105.5 | 96.95 / **8.5f** |

**^objective function:L-linear, N-nonlinear, equalities , inequalities)**

**^PU-seconds, IBM-3090/600**

**[3]NLP subproblems:MINOS / MILP master problems:MPSX**

**Table n. Comparison of MPSX Versus ZOOM in Solving MILP Master Problem**

| Problem | CPU-time[1] | CPU-time[2] | Ratio[3] |
|---|---|---|---|
| PLAN | 1.41 (0.19,1.22) | 0.52 (0.15,0.37) | 0.30 |
| EX3 | 2.52(0.41,2.11) | 1.95 (0.43,1.52) | 0.72 |
| EX4 | 8.20 (1.41,6.79) | 57.28 (1.43.55.85[4]) | 8.23 |
| BATCH5 | 5.36 (0.99,4.37) | 17.59 (1.00,16.59) | 3.80 |
| BATCH8A | 10.41(1.56,8.85) | 39.31(1.58,37.73) | 4.26 |

[1]CPU seconds, IBM-3090/600: Total (NLP subproblem - MINOS5.1, MILP Master problem - MPSX)

2CPU seconds, IBM-3090/600: Total (NLP subproblem - MINOS5.1, MILP Master problem - ZOOM)

[3]Ratio of ZOOM/MPSX

[4]Solver error encountered with ZOOM at final iteration

**Table III. Computational Results with Nonconvex MINLP Problems**

| Problem | Binary Variables | Continuous Variables | Initial Points Tested | Global Optimum Found |
|---|---|---|---|---|
| CAP1 | 4 | 0 | 8 | 8 |
| EX3 | 3 | 2 | 8 | 7 |
| BATCH | 12 | 22 | 10 | 7 |
| FLOW | 9 | 410 | 2 | 2 |