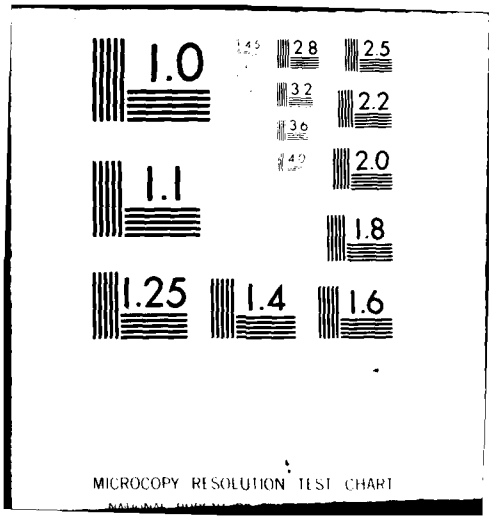
AD-A097 108 STANFORD UNIV CA DEPT OF OPERATIONS RESEARCH F/6 12/1
BINARY INTEGER LINEAR PROGRAMMING: A HYBRID IMPLICIT ENUMERATIO--ETC(U)
DEC 80 N E JACOMIN N00014-76-C-0418
UNCLASSIFIED TR-97 NL

2 of 2
AD A 097 108

END
DATE FILMED
5-81
DTIC


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6
MICROCOPY RESOLUTION TEST CHART

CHAPTER SIX

AREAS FOR FUTURE RESEARCH

6.1 Modifications of the Algorithm

Two types of modifications are available which might improve the execution times of our algorithm. One type would stick with the procedures detailed in Chapters 3 and 4 but attempt to carry them out more efficiently, while the other would make changes in the procedures themselves. In this section we examine the steps of the algorithm considering both types of changes.

It was mentioned in Section 4.2 that Hillier's heuristic procedure was modified in a rather rudimentary way to take advantage of (BILP). A more thorough adaptation would cut down on computer times for all of our problems.

As detailed in Section 5.4 the bounding technique has performed its function admirably. For this reason changes in the procedure itself seem inadvisable. However, there is a way to more quickly fathom some partial solutions for the J_N variables. Since the bounding inequality reads

$$(1) \qquad \sum_{j \in J_N} x_j / (x_j^{(j)}) \leq 1 - \rho_0^I ,$$

each time ρ_0^I is updated (whenever a new incumbent is found) we may fix any J_N variables satisfying

$$1/(x_j^{(j)}) > 1 - \rho_0^I$$

at zero for the remainder of the algorithm. This follows since any eligible partial solution must satisfy (1). In our summary in the Appendix we see that this added step would fathom a partial solution where such a variable is fixed at level one in Step ⑧ rather than in ⑨ (iii). Analyzing further the results of our computer tests we found that this additional test would have been satisfied in several cases. At the moment further modifications of the bounding technique are not anticipated.

Before considering changes in the hybrid portion of our algorithm it is useful to examine Geoffrion's (1969) algorithm in greater detail. Since we found our bounding technique eliminating essentially the same partial solutions as his surrogate constraints, but with considerably less computation effort, we might wonder what our results would be using the remainder of his algorithm with our bounding technique. As described in Section 4.5 our backtracking step is identical to his. His branching step, however, is different. Initially, he performs the following change of variables to define $\underline{x}'' \in R^n$ for any $\underline{x} \in R^n$:

$$x_j'' = \begin{cases} x_j & c_j \leq 0 \\ 1 - x_j & c_j > 0 \end{cases},$$

yielding the problem

$$\begin{aligned} & \text{maximize} && x_0'' = \underline{c}''\underline{x}'' \\ \text{(BILP)}'' \quad & \text{subject to:} && A''\underline{x}'' \leq b'' \\ & && \underline{x}'' \text{ binary} \end{aligned}$$

from (BILP). This change of variables assures $\underline{c}'' \leq 0$ and the branching step uses this information. In particular at a given partial solution S which is not fathomed he branches on the variable x_{j_0} satisfying

$$\max \sum_{i=1}^{m} \min\{0, b_i''^S - a_{ij}''\} \tag{2}$$

where $b_i''^S = b_i'' - \sum_{j \in S} a_{ij}'' x_j''^S$. The maximum in (2) is taken over all those j such that x_j'' is not binary-valued in the solution to $(\text{BILP})_R''^S$. The branching variable is set at level one yielding the new partial solution S'. The motivation for this choice is as follows: if the maximum in (2) is zero, the solution to $(\text{BILP})_R''^{S'}$ is $x_j'' = 0$, $j \notin S'$ (since $\underline{c}'' \leq 0$) and S' may be fathomed immediately.

Now we are ready to make some observations on this choice of branching variable in regard to our algorithm. First, we cannot in general achieve a form of the problem where the objective function coefficients are nonpositive. In the bounding technique we perform a necessary change of variables which may result in $c_j' > 0$ for some $j \in J_N$. Reversing this action would nullify our bounding inequality. However, the success of Geoffrion's more rudimentary branching procedure may still provide a lesson and a model for us. In both algorithms LP relaxations are solved as part of the fathoming process; in Geoffrion's algorithm these relaxations in addition provide surrogate constraints while in our case they provide the information to compute the Tomlin penalties. For this reason we may hypothesize that our execution times are suffering due to the time involved in computing the penalties and

not due to the actual solution of LP relaxations. Certainly it is more time consuming to use the penalties than Geoffrion's approach. In particular, in the case where the number of variables in J_B is large we compute a very large number of penalties. Hence, the problems least favorable to the bounding technique are also least favorable to the branching step.

We propose to experiment with a procedure in which the Tomlin penalties would still be used but not at each branching opportunity. In particular in the problems where the size of J_B is large we feel that a sophisticated choice of branching variable such as that dictated by the Tomlin penalties is desirable at those branching steps reached soon after a new partial solution for the J_N variable is reached. Even though this is where the greatest number of penalties is computed, getting off on the right track is crucial in such problems. However, instead of continuing to do this, we propose to shift to a more rudimentary procedure as the size of the partial solutions at hand gets larger. As those variables with the greatest effect (as indicated by the Tomlin penalties) on the objective function are fixed, the computational experience available indicates that the penalties become closer in magnitude and thus we tend to make choices between similar options. To do so at great computational expense is a mistake.

In summary, we will experiment with modifying our branching step as follows. Penalties will be used until they are within some tolerance of each other and then a quicker process will be applied until a new partial solution for the J_N variables is reached. In problems where J_B is small we may wish to abandon the use of penalties altogether

if computational testing indicates that better times can be achieved. The new branching process could be similar to Geoffrion's where we modify (2) to take into account both positive and negative coefficients in the objective function. This may be more computationally expensive than is necessary but only computational testing can answer that question.

Clearly more test problems need to be run both to further examine the performance of the algorithm in its present form and to test the utility of the aforementioned modifications. In particular, we want to use some large and sparse problems with a favorable constraint to variable ratio. Such problems would lead to LP relaxations best suited to MINOS so we could see the effect of appealing to the strong points of the two major parts of our algorithm.

6.2 Extension of the Algorithm to the Mixed Case

The most desirable extension for this algorithm would be a version that would handle the mixed binary integer linear programming problem. This problem may be stated as:

$$
\text{(MBILP)} \qquad
\begin{aligned}
\text{maximize} \quad & x_0 = \underline{c} \cdot \begin{bmatrix} \underline{x} \\ \underline{y} \end{bmatrix} \\
\text{subject to:} \quad & A \begin{bmatrix} \underline{x} \\ \underline{y} \end{bmatrix} \leq \underline{b} \\
& \begin{bmatrix} \underline{x} \\ \underline{y} \end{bmatrix} \geq \underline{0} \\
& \underline{y} \leq \underline{e} \\
& \underline{x} \text{ binary}
\end{aligned}
$$

where $\underline{x}$ is a column vector in R^s, $\underline{y}$ is a column vector in R^t, $\underline{e}$ is the unit vector in R^t, and $s + t = n$ with $\underline{c}$, A, $\underline{b}$ and $\underline{0}$ as in (BILP). This problem is referred to as mixed since $\underline{x}$ is a binary vector while the components of $\underline{y}$ vary continuously between 0 and 1. Of course, the major work involved in such an extension would be the adaptation of the bounding technique to this case. We may define $\mathcal{B}$ in the same way and find the extreme points $\underline{x}^{(j)}$. As a matter of fact, the bounding inequality itself still holds; however, there may be some components of $\underline{y}$ in the J_N variables, in which case partial solutions fixing only binary variables will not contain these continuous variables. However, we do have an additional inequality which the continuous variables must satisfy. Hence, depending on the number of nonbinary variables found amongst the J_N variables our bounding inequality may be weakened considerably. Hence, any adaptation to (MBILP) would depend heavily on further work regarding the nonbinary variables in J_N or, more generally, on the type of hybrid procedure used to handle the $\underline{y}$ variables.

6.3 Conclusions

The author intends to carry out the suggestions in Section 6.1 in the near future and results will be reported. After the fine tuning of the algorithm has been achieved, work on (MBILP) will be undertaken. In addition, any hybrid algorithm is always open to improvement achieved by utilizing new techniques presented in other algorithms. We will continue to watch the literature for such opportunities.

APPENDIX

DETAILED SUMMARY OF THE ALGORITHM

① Use the simplex method to find the solution $\underline{x}^*$ to $(\text{BILP})_R$.

② Execute the heuristic to obtain an initial feasible point $\underline{x}^F$ with objective function value x_0^F.

③ Perform the change of variables for variables at level 1 in $\underline{x}^*$, yielding (BILP)'.

④ Find the extreme points $\underline{x}^{(i)}$ $(i = 1, \ldots, n)$ of the region $\mathscr{B}$.

⑤ Determine the maximal number of variables in J_N which may be at a positive level in an eligible partial solution for these variables.

⑥ Initialize $\rho_0^I = 0$, $x_0^I = x_0'^F$, $\underline{x}^I = \underline{x}'^F$ and adjoin the constraint $\underline{c}'\underline{x}' \geq x_0^I$ to (BILP)'.

⑦ If $1/(x_j^{(j)}) > 1$, fix $x_j' = 0$ for the remainder of the algorithm.

⑧ Initially all variables in J_N are set at level 0. Subsequently we search for the first zero variable not permanently fixed at 0 and set it at level 1. All preceding variables in J_N are set at 0. If no zero entry is found, the algorithm terminates.

⑨ Check to see if the partial solution S from ⑧ is eligible:

(i) Check for binary and conditional binary infeasibility in

$$\underline{c}'\underline{x}' \geq x_0^I .$$

If the former occurs, go to ⑧. If the latter occurs, augment S appropriately to obtain S' and go to ⑯.

(ii) Check to see if the bound from (5) is satisfied. If not, go to (8).

(iii) Check the bounding inequality. If it is violated go to (8).

(10) Solve $(\text{BILP})_R'^S$; if infeasible go to (8).

(11) If the solution in (10) is binary we have a new incumbent. If not, go to (13).

(12) Update $\underline{x}^I$, x_0^I, ρ_0^I. Go to (8).

(13) Compute up and down Tomlin penalties for those fractional valued variables in 10 not at a quasi-integer level.

(14) Attempt to fathom using penalties. If successful, to to (8).

(15) Branch on the variable yielding

max(max(up penalty, down penalty))

in the direction giving the largest penalty. Denote the new partial solution by S'.

(16) Solve $(\text{BILP})_R'^{S'}$. If feasible, go to (19).

(17) Using the backtracking scheme of Section 4.5 attempt to generate a new partial solution S' without altering the values of the J_N variables. If this is not possible, go to (8).

(18) Go to (16).

(19) If the solution is not binary, go to (13).

(20) A new incumbent has been found. Update $\underline{x}^I$, x_0^I, ρ_0^I. Go to (17).

BIBLIOGRAPHY

Armstrong, Ronald D. and Prabhakant Sinha (1974), "Improved Penalty Calculations for a Mixed Integer Branch-and-Bound Algorithm," Mathematical Programming 6, pp. 212-223.

Balas, E. (1965), "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," Operations Research 13, pp. 517-546.

Balas, E. (1974), "Intersection Cuts from Disjunctive Constraints," Management Sciences Research Report No. 330, Carnegie-Mellon University.

Balas, E. and R. G. Jeroslow (1975), "Strengthening Cuts for Mixed Integer Programs," Management Sciences Research Report No. 359, Carnegie-Mellon University.

Balas, E. and Clarence H. Martin (1978), "Pivot and Complement - A Heuristic for 0-1 Programming," Management Sciences Research Report No. 414, Carnegie-Mellon University.

Balinski, M. L. (1965), "Integer Programming: Methods, Uses and Computation," Management Science 12, pp. 253-313.

Beale, E. M. L. (1965), "Survey of Integer Programming," Operational Research Quarterly 16, pp. 219-228.

Beale, E. M. L. and Small, R. E. (1965), "Mixed Integer Programming By a Branch and Bound Technique," Proceedings of the IFIP Conference, pp. 450-451.

Benichou, M., J. -M. Gauthier, P. Girodet, G. Hentges, G. Ribière and O. Vincent (1971), "Experiments in Mixed-Integer Linear Programming," Mathematical Programming 1, pp. 76-94.

Cooper, L. and M. W. Cooper (1978), "All-Integer Linear Programming - A New Approach via Dynamic Programming," Naval Research Logistics Quarterly 25, pp. 415-429.

Dakin, R. J. (1965), "A Tree-Search Algorithm for Mixed Integer Programming Problems," Computer Journal 8, pp. 250-255.

Dantzig, G. B. (1963), Linear Programming and Extensions, Princeton University Press.

Dantzig, G. B., D. R. Fulkerson, and S. M. Johnson (1954), "Solution of a Large-Scale Traveling-Salesman Problem," Operations Research 2, pp. 393-410.

Davis, Ronald E., David A. Kendrick and M. Weitzman (1971), "A Branch-and-Bound Algorithm for Zero-One Mixed Integer Programming Problems," Operations Research 19, pp. 1036-1044.

Driebeek, Norman J. (1966), "An Algorithm for the Solution of Mixed Integer Programming Problems," Management Science 12, pp. 576-587.

Etcheberry, J., Carlos Conca and Ennio Stacchetti (1978), "An Implicit Enumeration Approach for Integer Programming Using Subgradient Optimization," Technical Report, Universidade de Chile.

Forrest, J. J. H., J. P. H. Hirst and J. A. Tomlin (1974), "Practical Solution of Large Mixed Integer Programming Problems with UMPIRE," Management Science 20, pp. 736-773.

Garfinkel, R. S. and G. L. Nemhauser (1972), Integer Programming, John Wiley and Sons.

Gauthier, J. -M. and G. Ribière (1977), "Experiments in Mixed-Integer Linear Programming Using Pseudo-Costs," Mathematical Programming 12, pp. 26-47.

Geoffrion, A. M. (1967a), "Implicit Enumeration Using an Embedded Linear Program," Rand Memorandum RM-5406-PR.

Geoffrion, A. M. (1967b), "Integer Programming by Implicit Enumeration and Balas' Method," SIAM Review 9, pp. 178-190.

Geoffrion, A. M. (1969), "An Improved Implicit Enumeration Approach for Integer Programming," Operations Research 17, pp. 437-454.

Geoffrion, A. M. (1974), "Lagrangian Relaxation for Integer Programming," Mathematical Programming Study 2, pp. 82-114.

Geoffrion, A. M. (1976), "A Guided Tour of Recent Practical Advances in Integer Linear Programming," OMEGA 4, pp. 49-57.

Geoffrion, A. M. and R. E. Marsten (1972), "Integer Programming Algorithms: A Framework and State-of-the-Art Survey," Management Science 18, pp. 465-491.

Glover, F. (1965), "A Multiphase-Dual Algorithm for the Zero-One Integer Programming Problem," Operations Research 13, pp. 879-919.

Glover, F. (1968), "Surrogate Constraints," Operations Research 16, pp. 741-749.

Glover, F. (1969), "Integer Programming Over a Finite Additive Group," SIAM Journal of Control 7, pp. 213-231.

Gomory, R. E. (1958), "Outline of An Algorithm for Integer Solutions to Linear Programs," Bulletin of the American Mathematical Society 64, pp. 275-278.

Gomory, R. E. (1960), "Solving Linear Programming Problems in Integers," Bellman and Hall, Combinatorial Analysis, Proceedings of the 10th Symposium of the American Mathematical Society, pp. 211-215.

Gomory, R. E. (1963a), "An Algorithm for Integer Solutions to Linear Programs," Graves and Wolfe, Recent Advances in Mathematical Programming, pp. 269-302.

Gomory, R. E. (1963b), "An All-Integer Integer Programming Algorithm," Muth and Thompson, Industrial Scheduling, pp. 193-206.

Gomory, R. E. (1965), "On the Relation Between Integer and Noninteger Solutions to Linear Programs," Proceedings, National Academy of Science 53, pp. 260-265.

Gorry, G. A. and J. F. Shapiro, "An Adaptive Group Theoretic Algorithm for Integer Programming Problems," Management Science 17, pp. 285-306.

Hillier, F. S. (1969a), "A Bound-and-Scan Algorithm for Pure Integer Linear Programming with General Variables," Operations Research 17, pp. 638-679.

Hillier, F. S. (1969b), "Efficient Heuristic Procedures for Integer Linear Programming With an Interior," Operations Research 17, pp. 600-637.

Kochman, G. A. (1976), "Decomposition in Integer Programming," Technical Report No. 66, Department of Operations Research, Stanford University.

Land, A. H. and A. G. Doig (1960), "An Automatic Method of Solving Discrete Programming Problems," Econometrica 28, pp. 497-520.

Little, John D. C., Katta G. Murty, Dura W. Sweeney and Caroline Karel (1963), "An Algorithm for the Traveling Salesman Problem," Operations Research 11, pp. 972-989.

Markowitz, H. M. and A. S. Manne (1957), "On the Solution of Discrete Programming Problems," Econometrica 25, pp. 84-110.

Murtaugh, B. A. and M. A. Saunders (1977), "MINOS - A Large-Scale Nonlinear Programming System - User's Guide," Technical Report SOL 77-9, Department of Operations Research, Stanford University.

Petersen, C. C. (1967), "Computational Experience with Variants of the Balas Algorithm Applied to the Selection of R & D Projects," Management Science 13, pp. 736-750.

Preckel, P. V. (1980), "Modules for Use With MINOS/AUGMENTED in Solving Sequences of Mathematical Programs," SOL Report, Department of Operations Research, Stanford University, to appear.

Senju, S. and Y. Toyoda (1968), "An Approach to Linear Programming with 0-1 Variables," Management Science 15, pp. B196-B207.

Shapiro, J. F. (1968a), "Dynamic Programming Algorithms for the Integer Programming Problem - I: The Integer Programming Problem Viewed as a Knapsack Type Problem," Operations Research 16, pp. 103-121.

Shapiro, J. F. (1968b), "Group Theoretic Algorithms for the Integer Programming Problem II: Extension to a General Algorithm," Operations Research 16, pp. 928-947.

Shapiro, J. F. (1970), "Turnpike Theorems for Integer Programming Problems," Operations Research 18, pp. 432-440.

Tomlin, J. A. (1970), "Branch and Bound Methods for Integer and Non-Convex Programming," Abadie, Integer and Nonlinear Programming, pp. 437-450.

Tomlin, J. A. (1971), "An Improved Branch-and-Bound Method for Integer Programming," Operations Research 19, pp. 1070-1075.

Toyoda, Y. (1975), "A Simplified Algorithm for Obtaining Approximate Solutions to Zero-One Programming Problems," Management Science 21, pp. 1417-1427.

Trauth, C. A., Jr. and R. E. Woolsey (1969), "Integer Linear Programming: Study in Computational Efficiency," Management Science 15, pp. 481-493.

Wolsey, L. A. (1969), "Mixed Integer Programming: Discretization and the Group Theoretic Approach," Technical Report No. 42, Operations Research Center, MIT.

Zanakis, S. H. (1977), "Heuristic 0-1 Linear Programming: An Experimental Comparison of Three Methods," Management Science 24, pp. 91-104.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: 97	2. GOVT ACCESSION NO.: AD-A097108
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): BINARY INTEGER LINEAR PROGRAMMING: A HYBRID IMPLICIT ENUMERATION APPROACH	5. TYPE OF REPORT & PERIOD COVERED: TECHNICAL REPORT
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): NANCY E. JACQMIN	8. CONTRACT OR GRANT NUMBER(s): N00014-76-C-0418
9. PERFORMING ORGANIZATION NAME AND ADDRESS: OPERATIONS RESEARCH PROGRAM - ONR DEPARTMENT OF OPERATIONS RESEARCH STANFORD UNIVERSITY, STANFORD, CALIF.	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (NR-047-061)
11. CONTROLLING OFFICE NAME AND ADDRESS: OFFICE OF NAVAL RESEARCH OPERATIONS RESEARCH PROGRAM CODE 434 ARLINGTON, VA. 22217	12. REPORT DATE: DECEMBER 1980
	13. NUMBER OF PAGES: 100
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)
This document has been approved for public release and sale; its distribution is unlimited. Reproduction in whole or in part is permitted for any purpose of the United States Government.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES
Also issued as Technical Report No. 80-30, Dept. of Operations Research - Stanford University under NSF Grant MCS76-81259.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
INTEGER PROGRAMMING, INTEGER LINEAR PROGRAMMING,
BINARY LINEAR PROGRAMMING, IMPLICIT ENUMERATION, ALGORITHMS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
PLEASE SEE OTHER SIDE

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TECHNICAL REPORT NO. 97 - ABSTRACT

This report develops a hybrid algorithm to solve the binary integer linear programming problem. This problem involves optimizing a linear objective function subject to a set of linear inequalities where, in addition, we require the variables to take on the value 0 or 1.

The approach developed is that of implicit enumeration; that is, the set of all possible binary combinations of values for the variables is searched without considering each combination explicitly. This is accomplished by applying a series of tests which when satisfied allow immediate elimination of large subsets of these completions. It is in the choice of tests to be used that this algorithm may be termed a hybrid. Borrowing penalties and pseudocosts as well as binary infeasibility and conditional binary infeasibility tests from previous approaches, the algorithm is built to use each of their strengths. In addition, an existing heuristic procedure is used to generate a good feasible binary point at the outset. Thus, a good initial bound on the optimal function value is available.

There is, however, a general weakness in the aforementioned tests since they can really only be applied to one variable at a time. This weakness can lead to serious inefficiencies when dealing with large problems. Hence, a new technique to obtain "joint" bounds on the values of a subset of the variables has been developed. These bounds are obtained by applying the theory of convex analysis to the region defined by the constraints that are binding at the solution to the stated problem when the variables are allowed to vary continuously between 0 and 1. This theory allows immediate rejection of combinations of values for the subset of variables involved when the "joint" bound is violated. Furthermore, as better feasible solutions are found in the course of the algorithm, the bound strictly tightens. Once a combination of values is accepted, the corresponding subset of complete feasible solutions is examined using the tests referred to above.

Finally, it is demonstrated that this bounding technique is intrinsically well-suited to problems with large numbers of variables (in relation to the number of constraints). In other words, precisely when it would be most inefficient to use tests applied to one variable at a time the "joint" bound will be applied to subsets of larger cardinality. Hence the number of remaining variables to be tested individually can be kept at a reasonable level.

A discussion of computational aspects and results, as well as direction of future related research, are included.


DAT
FILM