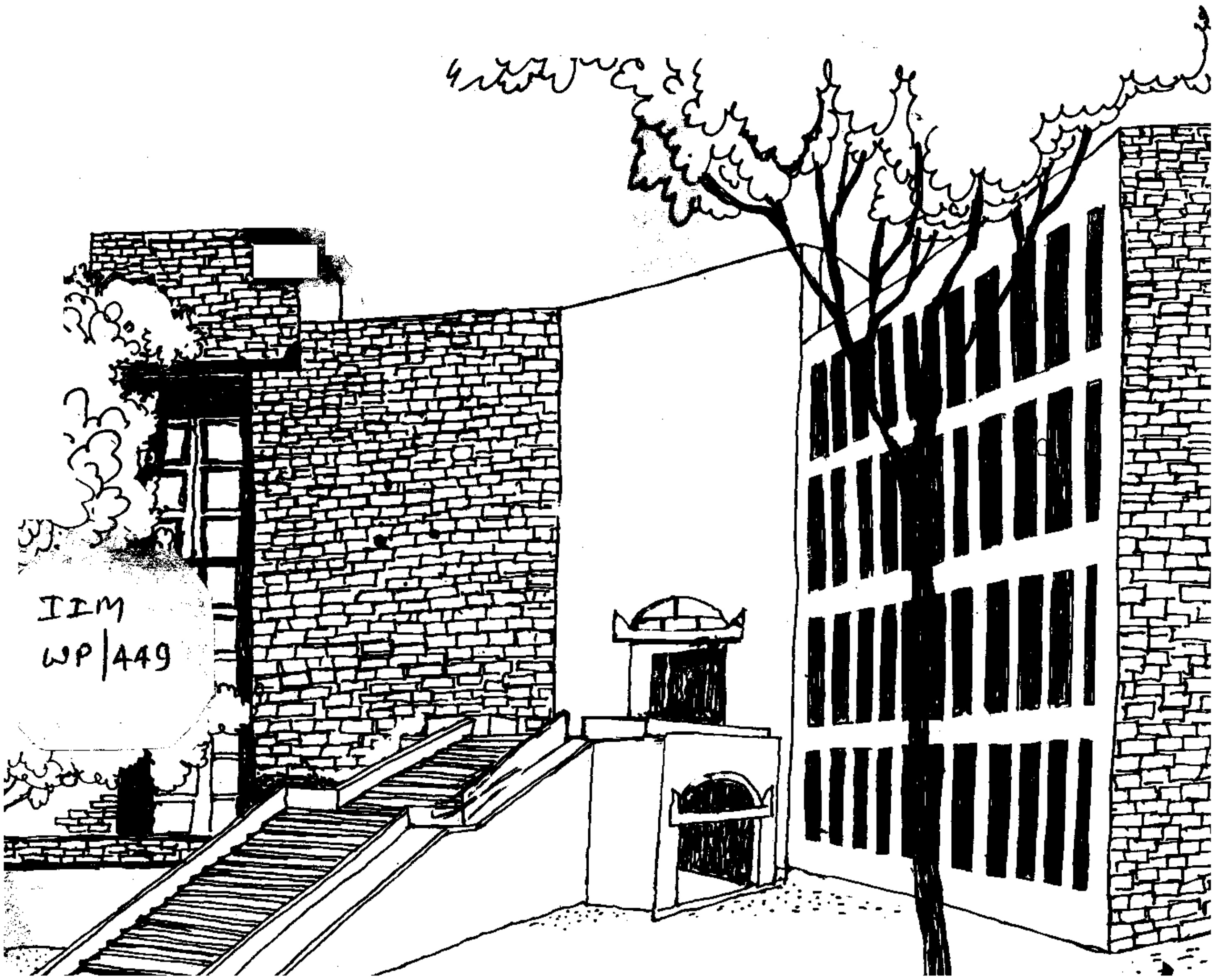
विद्याविनियोगाद्विकासः
IIM
AHMEDABAD

W. P.: 449

Working Paper


IIM
WP|449

AN UPPER BOUNDING HEURISTIC FOR NONLINEAR INTEGER PROGRAMS

By

Omprakash K. Gupta

&

A. Ravindran

W P No. 449

January, 1983

The main objective of the working paper series of the IIMA is to help faculty members to test out their research findings at the pre-publication stage.

INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD-380015
INDIA

AN UPPER BOUNDING HEURISTIC FOR NONLINEAR INTEGER PROGRAMS

By

Omprakash K Gupta
Indian Institute of Management
Vastrapur Ahmedabad 380 015

and

A Ravindran
School of Industrial Engineering
University of Oklahoma
Norman OK 73019
USA

CT: Many optimization problems are formulated as nonlinear mixed integer programming problems. Often practioners, as well as theoreticians, are interested in finding an upper bound on the objective minimum as fast as possible. An upper bound can be found by locating an integer feasible solution first and then evaluating the corresponding value of the objective function. Given that an algorithm A exists which can generate integer feasible solutions, this paper suggests a heuristic so that the computational efforts are reduced in locating an integer feasible solution. Using the branch and bound procedures, this heuristic is tested on a number of test problems and the computational results are reported.

1. INTRODUCTION

Mathematical programming techniques have long been of interest to management scientists, systems analysts, engineering designers, applied mathematicians and many others. A number of applied optimization problems can be formulated as nonlinear mixed integer programming problems. The overall interest in the area of nonlinear integer programming has recently grown up due to its wide applicability. The following is a small portion of the long list of applications:

resource allocation problems (5, 10), capacity expansion problems (2), inventory planning problems (11), engineering design problems (4,7,12) and system reliability problems (10). Several attempts have been made to solve nonlinear integer programming problems, more specifically in the case of problems with some special structures. Surveys on such procedures can be seen in (3,9).

In general, a nonlinear mixed integer programming problem P can be stated as follows:

$$\text{Minimize } f(x) \qquad x=(x_1, x_2 \ldots\ldots x_n) \quad \in R^n \quad \text{(i)}$$

$$\text{Subject to } g_j(x) \geq 0, \quad j=1,2,\ldots,NI, \quad \text{(ii)}$$

$$h_k(x) = 0, \quad k=1,2\ldots,NE, \quad \text{(iii)}$$

$$x_j \text{ is an integer, } j = 1,2, \ldots, m \leq n \quad \text{(iv)}$$

It should be noted that out of the n decision variable x_1, x_2 ..., x_n a total of m variables are identified as integer variables and these integer variables are indexed first. This is done only for the sake of convenience and causes no loss of generality.

In the above statement of problem p, a vector $x \in R^n$ is called a continuous feasible solution if it satisfies constraints (ii) and (iii). If a continuous feasible solution x also satisfies (iv), we would define x as an integer feasible solution (IFS) to p.

Quite often one is interested in finding only an upper bound on the objective as fast as possible. One way to accomplish this is by first locating an IFS to the given problem. The purpose of this paper is to present such an upper bounding heuristic.

2. UPPER BOUNDING HEURISTIC

We will assume that there exists an algorithm A which can generate an IFS to the problem p. The basic idea of this heuristic is to find an IFS to a relatively small size problem and then extend it to an integer feasible solution to the given problem. A sequence of smaller size partial problems is constructed wherein the number of integer variables increments by one as we move from one partial problem to another. Once a partial problem is constructed, it is solved by the algorithm A to locate an IFS to it. We can now carry out the heuristic in the following steps.

For the sake of notational convenience, let us assume that all the variables are integer variables.

1. Solve the nonlinear continuous problem.

Let $x = (x_1^0, X_2^0, \ldots, x_n^0)$ be the continuous optimal solution.

Find the integer y_i nearest to X_i^0 for $i = 1, 2, \ldots n$, and form a vector $y = (y_1, y_2, \ldots y_n)$. Also initialize a counter, say i=0.

2. If y is a continuous feasible solution, go to step 4. Otherwise, go to step 3.

3. Reset the counter i to $i+1$, and replace the value of the variable x_i by x_i^0. Also replace the vector y and $y = (x_1^0, x_2^0, \ldots x_i^0, y_{i+1} \ldots, y_n)$, and return to step 2.

4. If y is integer feasible, stop. Otherwise, go to step 5.

5. Construct an i-dimensional problem p_i by substituting the variables $x_{i+1}, x_{i+2}, \ldots, x_n$ by the integer values $y_{i+1} \ldots, y_n$ in the objective and the constraint functions of the given problem. Thus an i-dimensional partial nonlinear integer programming problem$(\tilde{p}_i)$ is constructed with $x_1, x_2 \ldots x_i$ as its integer variables. Apply algorithm A to get an IFS to p_i. If an IFS, say $x=(z_1, z_2, \ldots, z_i)$, is obtained, stop. The vector $x = (z_1, z_2 \ldots z_i, y_{i+1}, \ldots y_n)$ would be an integer solution to our problem p. Otherwise, return to step 3.

It should be noted that in the beginning the problems are similar in size, and hence they may be solved quickly. However, as one moves along the sequence of the partial problems they become larger in size , and in the worst possible situation the original problem itself would be solved. Therefore, if the original problem has an integer feasible solution, it certainly would have been discovered by this heuristic.

3. COMPUTATIONAL RESULTS AND ANALYSIS

A computational study was carried out to evaluate the utility of the heuristic developed in the previous section. Branch and bound methodology was implemented in solving nonlinear integer programming problems. A computer code BBNLMIP was developed. The code BBNLMIP uses computer program OPT, developed by Gabriele and Ragsdell (6), to solve nonlinear continuous problems. Readers are urged to refer to (8) for details on the branch and bound application to nonlinear integer programming problems.

The code BBNLMIP was used to solve a set of 22 test problems. These test problems consisted of both real-life and randomly generated problems (13). The test problems ranged from two to sixteen in number of variables, and zero to eight on number of constraints. Eighteen of the problems were of pure integer type, and the remaining four of mixed integer type. Additional problems of greater complexity were considered, but were dropped due to [illegible] computational cost. A detailed FORTRAN listing of the test problems can be seen in (8). These test problems were solved on CDC 6500 with and without the heuristic to obtain integer feasible solutions. The solution times and the number of nonlinear continuous problems solved were recorded. Table 1 presents the solution times with and without the heuristic and their ratio. These ratios range from a low of .081 to a high of 1.000 with mean .578.

Table 1

Comparision of Solution Times (Seconds)

Problem	With Heuristic (Y)	Without Heuristic (N)	Ratio: Y/N
1	.634	.632	1.000
2	.207	2.541	.081
3	.478	.479	1.000
4	.095	.170	.559
5	1.585	2.397	.244
6	.105	.529	.198
7	.307	.463	.663
8	1.677	1.681	1.000
9	.247	2.528	.098
10	.534	.521	1.000
11	.819	1.406	.583
12	1.193	2.829	.422
13	7.219	7.223	1.000
14	3.154	3.173	1.000
15	.159	.981	.162
16	.187	.428	.437
17	5.497	61.069	.090
18	32.725	32.494	1.000
19	80.939	89.728	.002
20	40.599	40.599	1.000
21	.463	.464	1.000
22	.562	5.981	.094
		Sum	12.723
		Average	.578

Since the solution time often depends on the type of computer used, and the art of programming itself, it was decided to use an alternate method to evaluate the performance of the heuristic. This method compares the number of nonlinear continuous problems solved instead of the solution times. The continuous problems which are solved under a branch and bound strategy in fact define the corresponding branch and bound tree and therefore these problems remain the same irrespective of the computer system and the art of programming. The total number of nonlinear continuous problems solved with and without the heuristic and their ratios are displayed in Table 2. The results are somewhat similar to those obtained earlier. On an average the heuristic solved 62.5% of the nonlinear continuous problems when compared with the number of problems solved without it.

Table 2

Comparison of Number of Nonlinear Continuous Problems Solved

Problem	With Heuristic (Y)	Without Heuristic (N)	Ratio: Y/N
1	5	5	1.000
2	1	11	.091
3	4	4	1.000
4	1	2	.500
5	3	4	.750
6	1	5	.200
7	3	3	1.000
8	4	4	1.000
9	1	11	.091
10	4	4	1.000
11	4	7	.571
12	4	9	.444
13	12	12	1.000
14	9	9	1.000
15	1	4	.250
16	1	4	.250
17	4	44	.091
18	30	30	1.000
19	26	58	.448
20	6	6	1.000
21	2	2	1.000
22	1	17	.059
		Sum	13.745
		Average	.625

3.1 Quality of Heuristic Bounds

It was felt that the quality of bounds obtained by the heuristic should be compared with those obtained without the heuristic. Therefore, an attempt was made to measure the gap between the bounds and the optimal values.

Definition: Let ZOPT denote the optimal value, and let ZFIRST denote the upper bound corresponding to the first integer feasible solution by a branch and bound strategy. The Absolute Percentage Gap (APG) of the bound ZFIRST is defined by:

$$APG = (ZFIRST - ZOPT) \times 100/ZOPT$$

The absolute percentage gaps were computed for test problems with and without the heuristic. The number of problems for which the APG was within various specific limits are shown in Table 3. The results indicate that the number of problems solved within a given specific APG limit do not seem to differ significantly. It may therefore be concluded that the quality of the bounds provided by the heuristic do not differ significantly from those provided without the heuristic.

Table 3

Number of Test Problems Solved within various APG

APG	With Heuristic	Without Heuristic
0	9	14
1	15	17
5	16	18
10	16	18
25	17	19
50	18	20
100	19	21

4. CONCLUDING REMARKS

We have reported a heuristic procedure to find upper bounds to the nonlinear mixed integer programming problems. It has been shown that on the average the heuristic takes about 60% of the effort in locating integer feasible solutions than without the heuristic. The quality of bounds obtained by implementing the heuristic did not differ significantly than the quality of bounds obtained without implementing the same.

REFERENCES

1. Benichou et al., "Experiments in Mixed-Integer Linear Programming" Mathematical Programming, Vol. 1, No. 1971, pp 76-94.

2. Bickel, T.C. "The Optimal Capacity Expansion of a Chemical Plant via Nonlinear Integer Programming," Ph.D. dissertation, The University of Texas at Austin, 1978.

3. Cooper, Mary W., "A Survey of Methods for Pure Nonlinear Integer Programming," to appear in Management Science

4. Dinkel, J.J. and Kochenberger, "Discrete Solutions to Engineering Design Problems," Journal of Engineering Mathematics, Vol.9, No.1, January 1975. pp.29-38.

5. Everest III, Hugh, "Gneralized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," Operations Research, Vol. 11, 1973, pp.399-417.

6. Gabriele, G.A., and Ragsdell, K.M., "OPT, A Nonlinear Programming Code in Fortran IV," The Modern Design Series, Vol.1, Purdue Research Foundation, 1976.

7. Gisvold, K.M., and Moe, J., "A Method for Nonlinear Mixed Integer Programming and its Application to Design Problems," Journal of Engineering for Industry, May 1972, pp.353-364.

8. Gupta, Omprakash K. "Branch and Bound Experiments in Nonlinear Integer Programming," Ph.D. Thesis, School of Industrial Engineering, Purdue University, West Lafayette, Indiana, December 1980.

9. GuptaO., and Ravindran, A., "Nonlinear Integer Programming Algorithms: A Survey," Working Paper No.8005, College of Business and Economics, Washington State University, Pullman, Washington 99164.

10. Kettelle, J.C., "Least Cost Allocation of Reliability Investment," Operations Research Vol.10, 1963, pp.245-265.

11. Miller, Bruce L., "On Minimizing Non-separable Functions defined on the Integers with an Inventory Application,," SIAM Journal of Applied Mathematics, Vol.21, No.1, July 1971, pp.166-185.

12. Ragsdell, K.M. and Phillips, D.T. "Optimal Design of a Class of Welded Structures using Geometric Programming," ASME Journal of Engineering for Industry , Vol. 98, series B, No.3, 1975, pp.1021-1025.

13. Rosen, J.B., and Suzuki, S., "Construction of Nonlinear Programming Test Problems," Communications of the ACM, Vol.8, No.2, pp.113.