A Computational Comparison of Branch and Bound and Outer Approximation Algorithms for 0-1 Mixed Integer Nonlinear Programs

Brian Borchers

Mathematics Department, New Mexico Tech, Socorro, NM 87801, U.S.A.

John E. Mitchell

Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, U.S.A.

In this paper we compare the performance of two codes for convex 0-1 mixed integer nonlinear programs on a number of test problems. The first code uses a branch and bound algorithm. The second code is a commercially available implementation of an outer approximation algorithm. The comparison demonstrates that both approaches are generally capable of solving the test problems. However, there are significant differences in the robustness of the two codes and their performance on different classes of problems.

1 Introduction

Mixed integer nonlinear programming problems (MINLP) arise in many applications, including chemical process synthesis [5,9,12] and portfolio selection [10,11].

In this note we consider MINLP of the form:

$$
\begin{aligned}
(MINLP) \qquad & \min f(\mathbf{x},\mathbf{y}) \\
& \text{subject to } \mathbf{g}(\mathbf{x},\mathbf{y}) \le 0 \\
& \mathbf{x} \in \{0,1\}^m \\
& \mathbf{y} \le \mathbf{u} \\
& \mathbf{y} \ge \mathbf{l}
\end{aligned}
$$

Here $\mathbf{x}$ is a vector of m binary variables, $\mathbf{y}$ is a vector of n continuous variables, and $\mathbf{u}$ and $\mathbf{l}$ are vectors of upper and lower bounds for the continuous variables $\mathbf{y}$.

We will assume that f and $\mathbf{g}$ are convex functions. Although the algorithms discussed in this note can be applied to nonconvex problems, there is no guarantee that an optimal solution will be obtained. Fortunately, many problems of interest are convex.

Branch and bound techniques have traditionally been used to solve mixed integer linear programming problems (MILP). However, branch and bound techniques can also be applied to convex MINLP [2,7,8]. In this note, we use the branch and bound code discussed in [2].

In recent years, outer approximation has become a popular technique for the solution of MINLP [4,9,12,13]. The outer approximation algorithm operates by solving a series of mixed integer linear programming problems that are obtained as outer approximations to the mixed integer nonlinear programming problem. A series of nonlinear programming subproblems are solved to obtain the outer approximations. In this note, we use GAMS/DICOPT, a commercially available implementation of the outer approximation algorithm [3,6].

2 Computational Comparison

In order to compare the performance of our branch and bound approach with the performance of an outer approximation algorithm, we solved a number of convex 0-1 MINLP test problems with our branch and bound code and with GAMS/DICOPT. In this section we present the results of this computational comparison.

To date, only a limited number of MINLP test problems have been made available by researchers. A publicly available suite of test problems in a standardized format similar to MIPLIB [1] would be invaluable in developing and testing codes for MINLP. Since no such library yet exists, we have selected a number of test problems from previous papers and randomly generated portfolio selection problems of various sizes.

Table 1 gives the characteristics of the test problems, including the number of integer and continuous variables and the number of linear and nonlinear constraints. The problems range from very small problems with fewer than five integer variables to large problems with as many as 150 integer variables. The problems **batchdes**, **meanvarx**, and **procsel** are supplied by GAMS Development Corporation with the GAMS/DICOPT software. The **batchdes** and

procsel problems are chemical process synthesis problems, while **meanvarx** is a portfolio selection problem. The problems **batch5**, **batch8**, **batch12**, and **ex3** are chemical engineering process design problems supplied by Ignacio Grossmann. The problems **port10** through **port150** are randomly generated portfolio selection problems that are similar to but somewhat larger than **meanvarx**.

Both codes were run on a Sun SPARC 10/30 workstation under SunOS. The branch and bound code, BB, is described in [2]. We used version 2.25.078 of GAMS/DICOPT, with OSL as the MILP solver and MINOS5.3 as the NLP solver [3,6]. Default settings for DICOPT parameters were used with two exceptions. CPU time and iteration limits were increased to solve the larger problems in our test set. Also, DICOPT stopping option 1, which stops DICOPT when a solution has been proven optimal was selected instead of the default, stopping option 2, which uses a heuristic to stop DICOPT with a solution that is not guaranteed to be optimal.

Table 1 shows the results of the test runs. For GAMS/DICOPT, the CPU times include only the time used by the OSL and MINOS5.3 solvers and exclude the time needed to compile the GAMS model and overhead associated with starting and stopping the solvers.

In all cases, the optimal objective function values obtained by GAMS and BB were very close. GAMS reports the objective value to four digits beyond the decimal point. In each case the solution obtained by BB matches the solution obtained by GAMS to at least 8 significant digits or to the number of digits reported by GAMS.

On the portfolio selection problems, **meanvarx**, **port10**, **port25**, **port50**, **port100**, and **port150**, the branch and bound code was generally somewhat faster than GAMS/DICOPT. GAMS/DICOPT was not able to solve **port100** or **port150** in a reasonable amount of CPU time. In both cases, the runs were stopped after one hour of CPU time, without GAMS/DICOPT having found an optimal solution. An examination of the output shows that GAMS/DICOPT had discovered a large number of nearly optimal solutions. However, the outer approximation algorithm was unable to generate a sufficiently accurate outer approximation to the nonlinear objective function. As a result, the outer approximation algorithm could not discover an optimal solution to the problem.

On the chemical process synthesis problems, **batch5**, **batch8**, and **batch12**, GAMS/DICOPT was roughly twice as fast as our branch and bound code. One explanation for the relatively poor performance of our branch and bound code on the batch problems is that these problems have sparse Hessian and constraint matrices. MINOS 5.3 is designed to exploit this sparsity, while the routine used in our branch and bound code is not designed for sparse problems.

Table 1
Computational Results.

Problem	0–1	Continuous	Linear	Nonlinear	BB	GAMS
Problem	Vars	Variables	Constraints	Constraints	CPU	CPU
batch5	24	22	72	1	19.4	7.4
batch8	40	33	140	1	144.0	76.1
batch12	60	40	216	1	423.6	190.6
batchdes	9	10	18	1	0.3	2.0
ex3	8	9	19	4	1.1	3.8
meanvarx	14	21	44	0	0.9	3.1
port10	10	10	13	0	0.4	6.3
port25	25	25	28	0	6.2	31.1
port50	50	50	53	0	25.8	16.7
port100	100	100	103	0	150.6	> 1hr
port150	150	150	153	0	555.0	> 1hr
procsel	3	7	5	2	0.1	2.0

In solving the nonlinear programming relaxation of **batch12**, MINOS 5.3 used 1.1 CPU seconds, while the NAG routine E04VCF used 19.5 CPU seconds. A faster nonlinear programming subroutine such as MINOS could improve the performance of the branch and bound code.

3 Summary and Conclusions

In selecting an appropriate technique for the solution of a mixed integer nonlinear programming problem, there are several important issues, including flexibility, reliability, accuracy, and efficiency. It is also important to understand what kinds and sizes of problems can reasonably be solved.

The failure of GAMS/DICOPT to solve the two largest portfolio selection models demonstrates a potential problem with the outer approximation approach. In these cases, the outer approximation code found a large number of near optimal solutions without ever finding an optimal solution. It appears that this failure to converge was caused by the inability of the outer approximation algorithm to generate a sufficiently accurate outer approximation to the MINLP. In contrast, the branch and bound code was able to solve all of the problems in our test set without difficulty.

In terms of efficiency, some problems were solved more quickly by GAMS/DICOPT while other problems were solved more quickly by the branch and bound code. Since no problem that could be solved by both codes required more than 600 seconds of CPU time for solution by either of the two codes we can conclude that it is generally relatively easy to solve problems of this size and complexity using either approach. However, it is not clear whether either method will be effective for much larger problems.

Acknowledgement

The research of both authors was partially supported by ONR Grant number N00014–94–1–0391. We would like to thank the GAMS Development Corporation for providing us with a copy of the GAMS/DICOPT software and some of the test problems used in this note. We would also like to thank Ignacio Grossmann for providing a number of test problems.

References

[1] Robert E. Bixby, E. Andrew Boyd, and Ronni R. Indovina. MIPLIB: A test set of mixed integer programming problems. *SIAM News*, 25(20):16, March 1992.

[2] Brian Borchers and John E. Mitchell. An improved branch and bound algorithm for mixed integer nonlinear programs. *Computers and Operations Research*, 21(4):359–367, 1994.

[3] Anthony Brooke, David Kendrick, and Alexander Meeraus. *GAMS: A User's Guide*. Scientific Press, San Francisco, 1988.

[4] Marco A. Duran and Ignacio E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36:307–339, 1986.

[5] Christodoulos A. Floudas. *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press, New York, 1995.

[6] GAMS. *GAMS – The Solver Manuals*. GAMS Development Corporation, 1217 Potomac Street N.W., Washington, D.C. 20007, September 1993.

[7] Omprakash K. Gupta and A. Ravindran. Nonlinear integer programming algorithms: A survey. *OPSEARCH*, 20(4):189–206, 1983.

[8] Omprakash K. Gupta and A. Ravindran. Branch and bound experiments in convex nonlinear integer programming. *Management Science*, 31(12):1533–1546, 1985.

[9] G. R. Kocis and I. E. Grossmann. Computational experience with DICOPT solving MINLP problems in process systems engineering. *Computers and Chemical Engineering*, 13:307–315, 1989.

[10] D. J. Laughhunn. Quadratic binary programming with applications to capital-budgeting problems. *Operations Research*, 18(3):454–461, 1970.

[11] J. C. T. Mao and B. A. Wallingford. An extension of Lawler and Bell's method of discrete optimization with examples from capital budgeting. *Management Science*, 15(2):51–60, 1968.

[12] Granville E. Paules and Christodoulos A. Floudas. APROS: Algorithmic development methodology for discrete-continuous optimization problems. *Operations Research*, 37(6):902–915, 1989.

[13] J. Viswanathan and I. E. Grossmann. A combined penalty function and outer approximation method for MINLP optimization. *Computers and Chemical Engineering*, 14:769–782, 1990.