(MINLP): BRANCH AND BOUND METHODS

A general *mixed integer nonlinear programming problem* (*MINLP*) can be written as

$$\begin{array}{rrcl} \min & f(\mathbf{x},\mathbf{y}) & & \\ \text{subject to} & \mathbf{h}(\mathbf{x},\mathbf{y}) & = & \mathbf{0} \\ \text{(MINLP)} & \mathbf{g}(\mathbf{x},\mathbf{y}) & \leq & \mathbf{0} \\ & \mathbf{x} & \in & R^n \\ & \mathbf{y} & \in & Z^m. \end{array}$$

Here $\mathbf{x}$ is a vector of n continuous variables and $\mathbf{y}$ is a vector of m *integer variables*. In many cases, the integer variables $\mathbf{y}$ are restricted to the values 0 and 1. Such variables are called *binary variables*. The function f is a scalar valued objective function, while the vector functions $\mathbf{h}$ and $\mathbf{g}$ express linear or nonlinear constraints. Problems of this form have a wide variety of applications, in areas as diverse as IR spectroscopy [5], finance [2], chemical process synthesis [8], topological design of transportation networks [11], and marketing [9].

The earliest work on *branch and bound* algorithms for mixed integer linear programming dates back to the early 1960's [6, 12, 14]. Although the possibility of applying branch and bound methods to mixed integer nonlinear programming problems was apparent from the beginning, actual work on such problems did not begin until later. Early papers on branch and bound algorithms for mixed integer nonlinear programming include [10, 13].

A branch and bound algorithm for solving MINLP requires the following data structures. The algorithm maintains a list L of unsolved subproblems. The algorithm also maintains a record of the best integer solution that has been found. This solution, ($\mathbf{x}^*$, $\mathbf{y}^*$), is called the incumbent solution. The incumbent solution provides an upper bound, ub, on the objective value of an optimal solution to the MINLP.

The basic branch and bound procedure is as follows.

1. Initialize: Create the list L with MINLP as the initial subproblem. If a good integer solution is known, then initialize $\mathbf{x}^*$, $\mathbf{y}^*$, and ub to this solution.
2. Select: Select an unsolved subproblem, S from the list L. If L is empty, then stop. If there is an incumbent solution, then that solution is optimal. If there is no incumbent solution, then MINLP is infeasible.
3. Solve: Relax the integrality constraints in S and solve the resulting nonlinear programming relaxation. Obtain a solution $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and a lower bound, lb, on the optimal value of the subproblem.
4. Fathom: If the relaxed subproblem was infeasible, then S will clearly not yield a better solution to MINLP than the incumbent solution. Similarly, if $lb \geq ub$, then the current subproblem cannot yield a better solution to MINLP than the incumbent solution. Remove S from L, and return to step 2.
5. Integer Solution: If $\hat{\mathbf{y}}$ is integer, then a new incumbent integer solution has been obtained. Update $\mathbf{x}^*$, $\mathbf{y}^*$, and ub.
6. Branch: At least one of the integer variables y_k takes on a fractional value in the solution to the current subproblem. Create a new subproblem, S_1 by adding the constraint
$$y_k \leq \lfloor \hat{y}_k \rfloor.$$
Create a second new subproblem, S_2 by adding the constraint
$$y_k \geq \lceil \hat{y}_k \rceil.$$
Remove S from L, add S_1 and S_2 to L, and return to step 2.

The following example demonstrates how the branch and bound algorithm solves a simple MINLP.

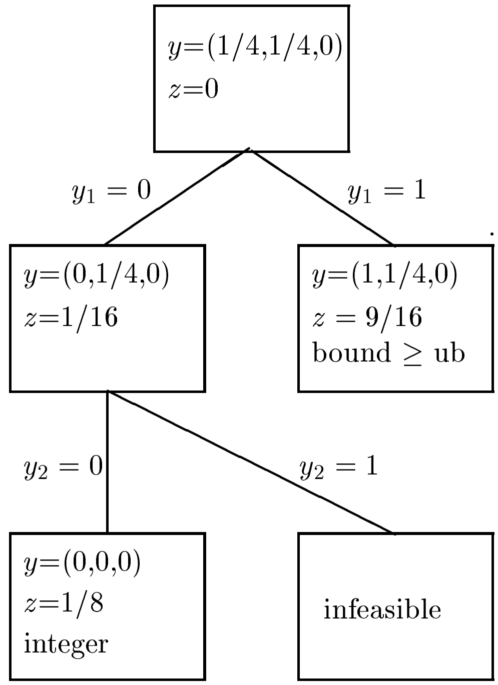
$$\begin{array}{rrcl} \min & (y_1 - \frac{1}{4})^2 + (y_2 - \frac{1}{4})^2 + y_3^2 & & \\ & -2y_1 + 2y_2 & \leq & 1 \\ & \mathbf{y} & & \text{binary.} \end{array}$$

mixed integer nonlinear programming problem
MINLP
integer variables
binary variables
branch and bound

The optimal solution to the initial nonlinear programming relaxation is $y = (1/4, 1/4, 0)$, with an objective value of $z = 0$. Both y_1 and y_2 take on fractional values in this solution, so it is neccesary to select a branching variable. The algorithm selects y_1 as the branching variable, and creates two new subproblems in which y_1 is fixed at 0 or 1. In the subproblem with y_1 fixed at 0, the optimal solution is $y = (0, \frac{1}{4}, 0)$, with $z = 1/16$. Since the optimal value of y_2 is fractional, the algorithm again creates two new subproblems, with y_2 fixed at 0 and 1. The optimal solution to the subproblem with $y_1 = 0$ and $y_2 = 0$ is $y = (0, 0, 0)$, with $z = 1/8$. This establishes an incumbent integer solution. The subproblem with $y_1 = 0$ and $y_2 = 1$ is infeasible and can be eliminated from consideration. The subproblem with $y_1 = 1$ has an optimal solution with $y = (1, 1/4, 0)$ and objective value $z = 9/16$. Since 9/16 is larger than the objective value of the incumbent solution, this subproblem can be eliminated from consideration. Thus the optimal solution to the example problem is $y^* = (0, 0, 0)$ with objective value $z^* = 1/8$.

Since each subproblem S creates at most two new subproblems, the set of subproblems considered by the branch and bound algorithm can be represented as a binary tree. The following figure shows the branch and bound tree for the example problem.

indefinite quadratic programming
bilinear programming
fractional linear programming
convex underestimators
BARON



Branch and Bound Tree for a Sample Problem

There are a number of important issues in the implementation of a branch and bound algorithm for MINLP.

The first important issue is how to solve the nonlinear programming relaxations of the subproblems in step 3. If the objective function f and the constraint functions $\mathbf{g}$ are convex, while the constraint functions $\mathbf{h}$ are linear, then the nonlinear programming subproblems in step 3 are convex and thus relatively easy to solve. A variety of methods have been used to solve these subproblems including generalized reduced gradient (GRG) methods [10], sequential quadratic programming (SQP) [3], active set methods for quadratic programming [7], and interior point methods [15].

However, if the nonlinear programming subproblems are nonconvex, then it can be extremely difficult to solve the nonlinear programming relaxation of S or even obtain a lower bound on the optimal objective function value. For some specialized classes of nonconvex optimization problems, including *indefinite quadratic programming*, *bilinear programming*, and *fractional linear programming*, convex functions which underestimate the nonconvex objective function are known. These *convex underestimators* are widely used in branch and bound algorithms for nonconvex nonlinear programming problems. Branch and bound techniques for nonconvex continuous optimization problems can also be used within a branch and bound algorithm for nonconvex mixed integer nonlinear programming problems. The *BARON* system implements this approach to solve a variety of nonconvex mixed integer nonlinear programming problems [16, 17].

The choice of the next subproblem to be solved in step 2 can have a significant influence on the performance of the branch and bound algorithm. In mixed integer linear programming, a variety of heuristics are employed to select the next subproblem [1]. One popular heuristic used in branch and bound algorithms for MILP is the "*best bound rule*", in which the subproblem with the smallest lower bound is selected. The best bound rule is widely used within branch and bound algorithms for MINLP [3, 10, 17]

In step 6, there may be a choice of several variables with fractional values to be the branching variable. A simple approach is to select the variable whose value $\hat{y}_k$ is furthest from being an integer [3, 10]. In mixed integer linear programming, estimates of the increase in the objective function that will result from forcing a variable to an integer value are often made. These estimates, called "*pseudocosts*" or "*penalties*", are used to select the branching variable. Penalties have also been used in branch and bound algorithms for mixed integer nonlinear programming problems [10, 17].

The performance of the branch and bound algorithm can be improved by computing lower bounds on the optimal value of a subproblem without actually solving the subproblem. In [7], lower bounds on the optimal objective value of a subproblem are derived from an optimal dual solution to the subproblem's parent problem. If this lower bound is larger than the objective value of the incumbent solution, then the subproblem can be eliminated from consideration. In [3], Lagrangean duality is used to compute lower bounds during the solution of a subproblem. When the lower bound exceeds the value of the incumbent solution, the current subproblem can be discarded.

Another way to improve the performance of a branch and bound algorithm for MINLP is to tighten the formulation of the nonlinear programming subproblems before solving them. In the BARON package, dual information from the solution to a nonlinear programming subproblem is used to restrict the ranges of variables and constraints in the children of the subproblem [16, 17].

In *branch and cut* approaches, constraints called cutting planes are added to the nonlinear programming subproblems [2, 18]. These additional constraints are selected so that they reduce the size of the feasible region of nonlinear programming subproblems without eliminating any integer solutions from consideration. This tightens the formulations of the subproblems and thus increases the probability that a subproblem can be fathomed by bound. Furthermore, the use of cutting planes can make it more likely that an integer solution will be obtained early in the branch and bound process. A variety of cutting planes developed for use in branch and cut algorithms for integer linear programming have been adapted for use in branch and cut algorithms for nonlinear integer programming. These include *mixed integer rounding cuts* [2], *knapsack cuts* [2], *intersection cuts* [2], and *lift-and-project cuts* [18].

best bound rule
pseudocosts
penalties
branch and cut
mixed integer rounding cuts
knapsack cuts
intersection cuts
lift-and-project cuts

To date, little work has been done to compare the performance of branch and bound methods for MINLP with other approaches such as *outer approximation* and *generalized Bender's decomposition*. Borchers and Mitchell compared an experimental branch and bound code with a commercially available outer approximation code on a number of test problems [4]. This study found that the branch and bound code and outer approximation code were roughly comparable in speed and robustness. Fletcher and Leyffer compared the performance of their branch and bound code for mixed integer convex quadratic programming problems with their implementations of outer approximation, generalized Bender's decomposition, and an algorithm that combines branch and bound and outer approximation approaches [7]. Fletcher and Leyffer found that their branch and bound solver was consistently faster than the other codes by about an order of magnitude.

References

[1] Beale, E. M. L.: 'Integer Programming', in D. Jacobs (ed.): *The State of the Art in Numerical Analysis*, Academic Press, 1977, pp. 409–448.

[2] Bienstock, Daniel: 'Computational Study of a Family of Mixed–Integer Quadratic Programming Problems', *Mathematical Programming* **74**, no. 2 (1996), 121–140.

[3] Borchers, Brian, and Mitchell, John E.: 'An Improved Branch and Bound Algorithm for Mixed Integer Nonlinear Programs', *Computers and Operations Research* **21**, no. 4 (1994), 359–367.

[4] Borchers, Brian, and Mitchell, John E.: 'A Computational Comparison of Branch and Bound and Outer Approximation Algorithms for 0–1 Mixed Integer Nonlinear Programs', *Computers and Operations Research* **24**, no. 8 (1997), 699–701.

[5] Brink, Anders, and Westerlund, Tapio: 'The Joint Problem of Model Structure Determination and Parameter Estimation in Quantitative IR Spectroscopy', *Chemometrics and Intelligent Laboratory Systems* **29** (1995), 29–36.

[6] Dakin, R. J.: 'A Tree–Search Algorithm for Mixed Integer Programming Problems', *Computer Journal* **8** (1965), 250–255.

[7] Fletcher, Roger, and Leyffer, Sven: Numerical Experience with Lower Bounds for MIQP Branch–and–bound, Tech. Rep. NA/151, University of Dunndee, 1994, To Appear in SIAM Journal on Optimization.

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

[9] Gavish, Bezalel, Horsky, Dan, and Srikanth, Kizhanathamy: 'An Approach to the Optimal Positioning of a New Product', *Management Science* **29**, no. 11 (1983), 1277–1297.

[10] Gupta, Omprakash K., and Ravindran, A.: 'Branch and Bound Experiments in Convex Nonlinear Integer Programming', *Management Science* **31**, no. 12 (1985), 1533–1546.

[11] Hoang, Hai Hoc: 'Topological Optimization of Networks: A Nonlinear Mixed Integer Model Employing Generalized Benders Decomposition', *IEEE Transactions on Automatic Control* **27** (1982), 164–169.

[12] Land, A., and Doig, A.: 'An Automatic Method of Solving Discrete Programming Problems', *Econometrika* **28**, no. 3 (1960), 497–520.

[13] Laughhunn, D. J.: 'Quadratic Binary Programming with Applications to Capital-Budgeting Problems', *Operations Research* **18**, no. 3 (1970), 454–461.

[14] Lawler, E. L., and Wood, D. E.: 'Branch and Bound Methods: A Survey', *Operations Research* **14**, no. 4 (1966), 699–719.

[15] Lee, Eva K., and Mitchell, John E.: 'Computational experience of an interior point algorithm in a parallel branch-and-cut framework': *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing, Minneapolis, March 1997*, 1997.

[16] Ryoo, Hong S., and Sahinidis, Nikolaos V.: 'A Branch–and–Reduce Approach to Global Optimization', *Journal of Global Optimization* **8**, no. 2 (1996), 107–139.

[17] Sahinidis, Nikolaos V.: 'BARON: A General Purpose Global Optimization Software Package', *Journal of Global Optimization* **8**, no. 2 (1996), 201–205, BARON is available at http://archimedes.me.uiuc.edu/sigma/baron.html.

[18] Stubbs, Robert A., and Mehrotra, Sanjay: A Branch–and–Cut Method for 0–1 Mixed Convex Programming, Tech. Rep. 96-01, Northwestern University, Evanston, IL, 60208, 1996.

---

outer approximation
generalized Bender's decomposition
Borchers
Mitchell
Fletcher
Leyffer

Brian Borchers
New Mexico Tech
Department of Mathematics
Socorro, NM 87801, USA
E-mail address: borchers@nmt.edu

AMS 1991 Subject Classification: 90C11.
Key words and phrases: mixed integer programming, branch and bound, MINLP.