A Computational Study of Search Strategies for Mixed Integer Programming


Abstracted by: Lin Wan

This paper surveyed and evaluated the effectiveness of different search strategies in the branch-and-bound procedure for solving mixed integer programming problem using linear programming relaxations, regarding the advances that have taken place over the years in the fields of computer hardware, computer software, and mathematics. Such search strategies include branching rules and node selection.

The most common division strategies, or “branching rules”, are to partition the feasible region by changing variables’ bounds. This paper discussed two most popular kinds of such schemes, namely variable dichotomy and GUB dichotomy. When using variable dichotomy, we must choose one variable to define the division if there are many fractional variables in the solution to the LP relaxation problem. So we need to predict which fractional variables will most improve the upper bound when required to be integral by either estimating the change of the objective function value or providing a lower bound on the degradation of the objective function value. Branching rule experiments are conducted on 5 kind of such variable dichotomy methods: estimation methods, lower bounding methods, combining estimates and lower bounds, using degradation estimates, and non-estimate-based branching rules. The results show that:

R The use of pseudocosts in an intelligent manner is essential to solve many of the problems;
R Combining pseudocosts with lower bounding information seems to improve the robustness of the branching method at a relatively small computational price;
R There is no branching method which clearly dominates the others, so a sophisticated MIP solver should allow many different options for selecting the branching variable.

When the problem has generalized upper bound GUB) constraints of the form \( \sum_{j \in T} x_j \leq 1 \) for some \( T \), another problem subdivision scheme used in practice is called branching on a GUB Dichotomy. The advantages to branch on a GUB constraint instead of a variable are that the branch and bound tree is more “balanced and when the GUB is actually a special ordered set, the fractional LP solution may suggest which variable will be one in the optimal solution, and thereby demonstrate a good set \( T \) on which to base the branching dichotomy. But from the experiment results, a straightforward approach to GUB branching seems to not be as effective as branching on a variable if no logical order to the variables in a GUB.

About node selection strategies, since our purpose for selecting a node is to find good integer feasible solutions or to prove that no solution better than our current one with value
exists, the quality of current solution value is an important factor in determining which node to select for evaluation. In this paper, node selection methods are categorized as static methods, estimate-based methods, two-phase methods, and backtracking methods. Although it is difficult to tell which is the best method, we can still draw some conclusions from the results of computational tests:

- Pseudocost-based node estimate methods or combining a pseudocost-based estimate method in backtracking seems to be the best idea for node selection;
- Backtracking methods the select the node with the best estimate generally outperform those methods that choose the best bound node when backtracking;
- There is no node selection method that clearly dominates the others, so a sophisticated MIP solver should allow many different options for selecting the node to evaluate;
- Even in the presence of a primal heuristic, depth-first search performs poorly in practice.