ARAB REPUBLIC OF LIBLIC

THE INSTITUTE OF NATIONAL PLANNING

التخطيط القومي

Memo. No. 1212

ON THE USE OF FICTITIOUS BOUNDS IN TREE SEARCH ALGORITHMS

BY

MOKHTAR S. BAZARAA

AND

ALWALID N. ELSHAFEI

Dec. 1977

CAIRO

SALAH SALEM St.-NASR CITY

MANAGEMENT SCIENCE
Vol. 23, No. 8, April, 1977
Printed in U.S.A.

Notes* II

ON THE USE OF FICTITIOUS BOUNDS IN TREE SEARCH ALGORITHMS†

MOKHTAR S. BAZARAA‡ AND ALWALID N. ELSHAFEI§

One of the strategies used by many tree search algorithms is to follow down one path in the tree until either a feasible solution is found or else fathoming occurs. For a minimization problem, the lower bounds calculated at various tree nodes tend to be well below the optimal value of the objective function, let alone the values of the successively improving upper bounds. As a result fathoming usually occurs only deep in the tree, and consequently, the search becomes rather lengthy. The purpose of this note is to describe a procedure for using and updating fictitious upper bounds in a systematic way so that optimal and suboptimal solutions can be obtained with a smaller computational effort. The procedure is illustrated by two examples: the traveling salesman problem and the quadratic assignment problem.

Introduction

Tree search algorithms for a minimization problem can be classified under two major categories. The first (breadth) category attempts to search the tree by branching from the node with the smallest lower bound. This serves to minimize the portion of the tree which is explicitly explored, but on the other hand requires storing all the nodes whose lower bounds are less than the current best known upper bound. The second (depth) strategy follows down one path in the tree until a feasible solution is found or else the lower bound exceeds the upper bound and fathoming occurs.

In the depth strategy it becomes of great importance to have a tight upper bound so that fathoming would occur without the need to go deep in the tree. One way of doing this is to use a fictitious upper bound which is smaller than the current bound at hand. The reader may refer to Lawler and Wood [4] for the use of fictitious bounds and for a survey of branch and bound procedures.

The purpose of this paper is to present a systematic way of choosing and updating the fictitious upper bounds so that both optimal and suboptimal solutions can be obtained by expending less computational effort. The method is particularly suited for in-depth tree search algorithms.

The Use of Fictitious Bounds

Let UB and LB be the upper and lower bounds on the minimal objective value of the overall problem. The upper bound UB is given by the smallest objective value of all known feasible solutions. Given a partial solution during the search process, let LB′ be a lower bound on the objective values of all feasible completions of this solution. Obviously, if $LB' \geq UB$ then the partial solution is fathomed since it cannot lead to an improved feasible solution.

Now construct the following fictitious upper bound $FUB = \alpha\ UB + (1-\alpha)LB$, where $\alpha \in (0,1]$. Since $UB > LB$ (if $UB = LB$ we are through), and if $\alpha < 1$, then $FUB < UB$. If we fathom when $LB' \geq FUB$ rather than when $LB' \geq UB$, then the

* All Notes are refereed.

† Accepted by Arthur M. Geoffrion; received April 28, 1975. This paper has been with the authors 3 months, for 1 revision.

‡ Georgia Institute of Technology.
§ Institute of Planning, Cairo, Egypt.

portion of the tree explicitly searched would be reduced, and the speed of fathoming would increase. If a feasible solution with objective f such that $f <$ FUB is obtained, the new feasible solution and corresponding pertinent information[1] are stored, UB is replaced by f, a new fictitious upper bound is computed, and the process is continued starting with the newly found solution. If, on the other hand, the overall tree is searched without finding a feasible solution with $f <$ FUB, then we conclude that the overall lower bound is at least FUB, and so LB is replaced by FUB. In this case one of the following alternative actions may be taken.

(a) If UB and LB are sufficiently close we may stop with a "qualified" suboptimal solution.

(b) If an exact optimal solution is desired, and if UB and LB are sufficiently close, then α is switched to 1, and the search is continued from the node with the best known solution whose objective is UB. The effect of this is to abandon the fictitious bound approach and only fathom when LB′ $\geqslant$ UB.

(c) If UB and LB are not close, then the search is continued from the node with the best known feasible solution. A new FUB is computed with the same α or with a larger α (recall that LB was raised).

Obviously the strategy discussed above will lead to an optimal solution, or to a suboptimal solution with any *a priori* desired degree of accuracy. Even though parts of the tree will be searched more than once, the quick fathoming outweighs the repeated efforts, as indicated by the computational results in [1] and [2] and the two examples to be discussed later in this paper.

The Choice of α

The choice of α depends largely on the quality of the initial bound LB available, whether an optimal or suboptimal solution is required, and the quality of the latter. The following discussion may serve only as a general guideline. If α is chosen close to 1, then we are in effect fathoming on a number close to UB, thus the frequency of fathoming would not be high and the search may be entangled in detecting complete solutions which are eventually not very valuable. On the other hand, if α is close to zero, then fathoming will be substantially speeded, but it would be less likely to obtain feasible solutions with objective less than the fictitious upper bound.

Hence, if the lower bound LB is known *a priori* to be close to the optimal objective, small values of α are recommended, e.g., α in the range from 0.1 to 0.4. If, on the other hand, a tight lower bound is not available, then larger values of α should be used. For instance, if the optimal objective is known to be positive, we may fix LB at zero, and use values of $\alpha \geqslant 0.5$.

If the quality of the available lower bound cannot be ascertained beforehand, values of α in the neighborhood of $\alpha = 0.5$ are recommended. If $\alpha = 0.5$ is used, whether a feasible solution with objective less than FUB is found or not, the interval of uncertainty is at least reduced by half at each iteration, as shown in Figure 1.

[1] For example, in the traveling salesman problem we may store the information $1 \to 5(\bar{3}) \to 4 \to 2 \to 3 \to 1$ corresponding to the tour $1 \to 5 \to 4 \to 2 \to 3 \to 1$ where the notation $(\bar{3})$ means that the lower bound of all completions of the partial tour $1 \to 5 \to 3$ is at least equal to the cost of a known tour.

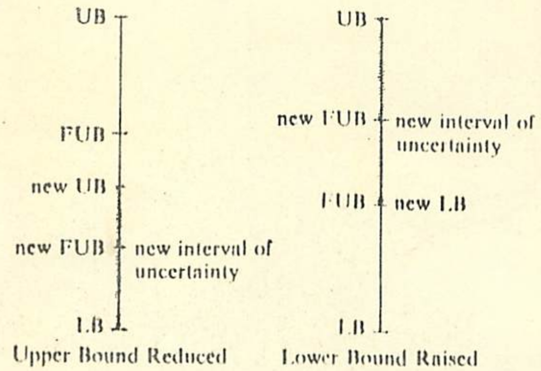

FIGURE 1. Using $\alpha = 0.5$.

Examples

We will now illustrate the effectiveness of using fictitious upper bounds in speeding the branch and bound search by two examples: the traveling salesman problem and the quadratic assignment problem. The reader may refer to [1] and [2] for the details of the algorithms used.

The Traveling Salesman Problem

The use of fictitious upper bounds was incorporated in a code for solving the traveling salesman problem. The following are some highlights of the procedure used. First, the dual problem is solved by a subgradient optimization technique, and the dual variables are used to form a new cost matrix. An in-depth branch and bound search was used to solve the primal problem. Given a partial solution, the node with the cheapest connection to the most recently assigned node is chosen, and LB′ is computed by finding the minimal spanning tree of the unassigned nodes. For more details the reader may refer to [1].

The 42 city problem of Dantzig, et al. [3] is used for illustration here. LB was *fixed* at zero while computing FUB, and the problem was solved three times independently on a Univac 1108, with $\alpha = 0.95$, $\alpha = 0.97$, and then $\alpha = 1.0$ (since LB was fixed at 0, large values of α were chosen). The results are shown in Table 1. Notice that the CPU time increases substantially as α increases with a relatively small improvement, or none at all, in the value of the best tour. With $\alpha = 0.95$, a tour with cost 708 was obtained in 5.9 seconds, while $\alpha = 0.97$ produced the same tour but in 32.8 seconds. Only when $\alpha = 1.0$ was used, the optimum tour with cost 699 was obtained and verified in 65.1 seconds.

TABLE 1
Experience with the 42 City Problem

α	Overall Lower Bound from the Dual Problem	Length of Best Tour	Optimum Obtained and Verified	Computational Time in Seconds
0.95	693	708	NO	5.9
0.97	693	708	NO	32.8
1.0	693	699	YES	65.1

In another run, fictitious bounds were used to find the optimal solution. LB was fixed at zero, and the following successive values of α (all in the same run) were used: 0.93, 0.95, 0.97, and 1.0. With a given α the tree is searched, fathoming whenever LB′ > FUB. After the tree is exhausted, α is increased as indicated above, and the search is continued from the node with the best solution. As a result, the optimal tour was obtained and verified in 27.7 seconds as opposed to 65.1 seconds without the use of fictitious bounds.

With the value of the optimal solution known to be 699, another run was executed with UB = 700 and $\alpha = 1.0$. The purpose of this run was to identify the optimal tour and to examine the effect of using the smallest valid upper bound on speeding the search. The running time was 5.2 seconds, approximately 4.4 seconds of which were used in the dual phase. in other words, the branch and bound scheme used approximately 0.8 seconds to identify the optimal tour and verify its optimality.

The Quadratic Assignment Problem

As another example, the use of fictitious bounds was incorporated in a code for solving the quadratic assignment problem. First an overall lower bound LB is calculated by solving a linear assignment problem. Then an in-depth branch and bound algorithm is used. Given a partial assignment, a lower bound LB′ on all its feasible completions is calculated by means of a linear assignment problem. The next object to be assigned was that with the maximum interactions with the most recently assigned object. For further details the reader may refer to [2].

Here we give our experience with a 6×8 quadratic assignment problem, where 6 objects are to be assigned to 6 of 8 available locations. The interactions among the objects and the arrangements of the locations are depicted below (rectilinear distance is used).

Object	1	2	3	4	5	6
1	0	30	5	15	10	5
2	0	0	13	5	41	15
3	0	0	0	6	22	70
4	0	0	0	0	7	25
5	0	0	0	0	0	5
6	0	0	0	0	0	0

Locations

1	2	3
4	5	6
7		8

The problem was run twice on an IBM 370/155. In the first run $\alpha = 1.0$ was used, i.e., no fictitious bounds. The optimal solution with objective 281 was obtained and verified in 7.5 seconds. In the second run $\alpha = 0.5$ was used until the difference between the lower and upper bounds was $\leqslant 10$, in which case we switched to $\alpha = 1.0$. The optimal solution was obtained and verified in 3.4 seconds.[2]

[2] Professor Bazaraa's work was partially supported by NSF Grant GK-38337. Professor Elshafei's work was done while visiting North Carolina State University.

References

1. BAZARAA, M. S. AND GOODE, J. J., "The Traveling Salesman Problem: A Duality Approach," School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia (1974).
2. ——— AND ELSHAFEI, A. N., "Exact and Heuristic Procedures for the Quadratic Assignment Problem," School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia (1974).
3. DANTZIG, G. B., FULKERSON, R. AND JOHNSON, S. M., "Solution of a Large Scale Traveling Salesman Problem," *Operations Research*, Vol. 2 (1954), pp. 393–410.
4. LAWLER, E. L. AND WOOD, D. E., "Branch-and-Bound Methods: A Survey," *Operations Research*, Vol. 14, (1966), pp. 699–719.