A Note on the Robust International Sourcing Algorithm of Gutiérrez and Kouvelis

Lawrence V. Snyder
Lehigh University

Report No. 05T-004

A Note on the Robust International Sourcing Algorithm of Gutiérrez and Kouvelis

Lawrence V. Snyder*

Department of Industrial and Systems Engineering, Lehigh University

February 2, 2005

*Please send correspondence to Lawrence V. Snyder, Dept. of Industrial and Systems Engineering, Lehigh University, 200 West Packer Ave., Bethlehem, PA 18015, lvs2@lehigh.edu, 610 758 6696.

Abstract

We discuss an error contained in Gutérrez and Kouvelis's (1995) branch-and-bound algorithm for a robust international sourcing problem, which reduces to a robust version of the uncapacitated fixed-charge location problem (UFLP). The algorithm takes parameters p and N and attempts to find either the N most robust solutions (i.e., the N solutions with smallest maximum regret) or all solutions with maximum regret less than or equal to p if there are fewer than N of them. The crux of the error is that optimization is performed based on one objective (expected cost) while fathoming is performed based on another (robustness).

Keywords: robust optimization, regret, international sourcing, facility location

Gutiérrez and Kouvelis (1995) present a model for choosing among a set of global suppliers in an environment in which currency exchange rates (or, equivalently, transportation costs) are uncertain. Using several reasonable assumptions and transformations, the authors show that the *deterministic* international sourcing problem is equivalent in form to the uncapacitated fixed-charge location problem (UFLP). They model uncertainty using discrete scenarios and seek to find the most *robust* solution, where robustness is measured by the minimax regret criterion. (The *regret* of a solution in a given scenario is the percentage difference between the cost of the solution in that scenario and the cost of the optimal solution for that scenario. The minimax regret solution is the one that has the smallest regret across all scenarios.) The model that Gutiérrez and Kouvelis finally arrive at, then, is a minimax-regret version of the UFLP.

The authors present a novel branch-and-bound algorithm for solving this problem, which they call the robust international sourcing algorithm (RISA). RISA maintains multiple branch-and-bound trees (one for each scenario), and the trees are branched and fathomed simultaneously so that they all have the same structure at the same time. The algorithm takes parameters p and N (an integer) and returns the N "most robust" solutions (i.e., the N solutions with smallest maximum regret), or all solutions with maximum regret less than or equal to p if there are fewer than N of them. RISA is presented as an exact, i.e., not heuristic, algorithm.

It is our contention that RISA uses incorrect fathoming rules, causing the algorithm to return solutions that are suboptimal. There are two sources for the error in the algorithm. The first is an implicit assumption that child nodes never contain "more robust" solutions than their parent nodes. This assumption is false, as we will show below. Second, when a node is fathomed from one tree, the corresponding nodes are fathomed from all trees, meaning that in some cases good branches may be inappropriately thrown out with bad ones.

In Section 1 of this note, we briefly describe RISA and introduce the required notation. Then, in Section 2, we provide a numerical example showing that RISA returns a suboptimal solution.

1 The Robust International Sourcing Algorithm (RISA)

Let I be the set of potential suppliers that the firm is considering establishing relationships with, and let P be the set of factories operated by the firm. Establishing a relationship with a supplier entails a substantial fixed cost, and once relationships have been established, the set of available suppliers remains fixed for the duration of the model's time horizon. For each supplier $i \in I$, let P_i be the set of factories that supplier i is eligible to serve.

Let S be the set of scenarios, each of which determines both fixed and variable costs. In particular, let F_i^s be the fixed cost of establishing a relationship with supplier $i \in I$ under scenario $s \in S$ and let c_{ij}^s be the variable cost to procure and transport all product required by factory $j \in P$ from supplier $i \in I$ to factory j in scenario $s \in S$. (These costs are derived in Gutiérrez and Kouvelis (1995) by transforming different cost parameters, but for the purposes of this note, it is sufficient to assume that these are the given costs.)

Define decision variables

$$y_i = \begin{cases} 1, & \text{if a relationship is established with supplier } i \in I \\ 0, & \text{otherwise} \end{cases}$$

and

$$x_{ij}^s = \text{the fraction of factory } j\text{'s demand that is served by supplier } i, \text{ for } i \in I, j \in P_i.$$

Note that suppliers must be chosen now, before it is known which scenario will occur, but factories may be assigned to suppliers once the scenario is known.

For a given scenario $s \in S$, the deterministic international sourcing problem (DISP) is then formulated as follows:

$$\begin{array}{lllll}
(1) & \text{(DISP)} & \text{minimize} & \displaystyle\sum_{i\in I}\sum_{j\in P_i} c_{ij}^s x_{ij}^s + \sum_{i\in I} F_i^s y_i & \\
(2) & & \text{subject to} & \displaystyle\sum_{i\in I} x_{ij}^s \geq 1 & \forall j \in P \\
(3) & & & x_{ij}^s \leq y_i & \forall i \in I,\ j \in P \\
(4) & & & 0 \leq x_{ij}^s \leq 1 & \forall i \in I,\ j \in P \\
(5) & & & y_i \in \{0,1\} & \forall i \in I
\end{array}$$

This is equivalent to the uncapacitated fixed-charge location problem (UFLP; Balinski 1965). (The summation over $j \in P_i$ in (1) is nonstandard for the UFLP, but this can easily be replaced by $j \in P$ by setting $c_{ij}^s = \infty$ for all s if $j \notin P_i$.)

For a given vector Y of y_i variables, let $Z_s(Y)$ be the cost of the solution under scenario s. (The assignment variables x_{ij}^s can be determined easily once the y_i variables are known.) Let Y_s^* be the optimal vector of y_i variables for scenario s; then $Z_s(Y_s^*)$ is the optimal cost for (DISP) under scenario s. The *relative regret* of solution Y in scenario s is

$$\frac{Z_s(Y) - Z_s(Y_s^*)}{Z_s(Y_s^*)}.$$

The robust international sourcing problem (RISP) is then

$$(6) \qquad \text{(RISP)} \qquad \min_Y \left\{ \max_{s \in S} \frac{Z_s(Y) - Z_s(Y_s^*)}{Z_s(Y_s^*)} \right\}$$

The robust international sourcing algorithm (RISA) maintains $|S|$ separate branch-and-bound trees, each representing (DISP) for a given scenario. A node k_s in the tree for scenario s is said to *correspond* to node k_t in the tree for scenario t if they occupy the same place in their respective trees. Branching is done in all trees simultaneously, i.e., if node k_s is branched on, then node k_t is branched on in scenario t's tree for all $t \in S \setminus \{s\}$. Similarly, if node k_s is fathomed, then k_t is fathomed for all $t \in S \setminus \{s\}$, as well. The algorithm maintains a list of nodes under consideration at any given time; branching means adding the child nodes to the list and fathoming means removing nodes from the list.

We will describe the steps of the algorithm in brief here; the reader is refereed to Gutiérrez and Kouvelis (1995) for the details. The algorithm involves computing a lower bound $\underline{Z}_s^k$ at a given node k for a given scenario s. Gutiérrez and Kouvelis suggest using the "weak relaxation" proposed by Efroymson and Ray (1966). The lower bound takes into account the variables currently forced to 0 or 1. Any all-integer solution to the lower-bound problem is feasible for (DISP). In addition, as a pre-processing step, (DISP) must be solved to optimality to obtain $Z_s(Y_s^*)$ for each s; Gutiérrez and Kouvelis suggest using Efroymson and Ray's algorithm for the UFLP to solve these problems.

For parameters p and N, RISA attempts to return every solution with maximum regret less than or equal to p if there are fewer than N of them, or the N solutions with smallest maximum regret otherwise. (In RISA, p is progressively reduced so that there are only N solutions with maximum regret less than or equal to p.)

Step 0. *Initialization.* Select parameters N and p and initialize an empty list of solutions L_R. (L_R will contain potential "most robust" solutions.) Compute the optimal solutions Y_s^* to (DISP) for each scenario s and the corresponding costs $Z_s(Y_s^*)$. Create node 1 in each tree.

Step 1. *Branching.* Select a scenario $\bar{s}$, a node $k_{\bar{s}}$ in scenario $\bar{s}$'s tree, and a variable i to branch on (using a straightforward branching rule we will omit here). This entails removing $k_{\bar{s}}$ and its corresponding nodes from all trees, creating two child nodes for the node corresponding to $k_{\bar{s}}$ in each tree, and storing the child nodes in a new list called L_{New}. (Let $k_s^{[0]}$ represent the "$y_i = 0$" child of node k_s and $k_s^{[1]}$ the "$y_i = 1$" child.) If forcing $y_i = 0$ means that there is now some factory all of whose eligible suppliers are forced closed, then forcing $y_i = 0$ results in an infeasible problem, so remove $k_s^{[0]}$ from L_{New} for each s.

Step 2. *Robustness test for* [0] *nodes.* If the $k_s^{[0]}$ nodes were not removed in Step 1, then for each s compute $\underline{Z}_s^{k_s^{[0]}}$, a lower bound on (DISP) for scenario s given the current suppliers forced open or closed. If $\left(\underline{Z}_s^{k_s^{[0]}} - Z_s(Y_s^*)\right)/Z_s(Y_s^*) > p$ for any s, then the regret for scenario s in any feasible solution will exceed p, so remove $k_s^{[0]}$ from L_{New} for all s.

Step 3. *Robustness test for* [1] *nodes.* For each s, compute $\underline{Z}_s^{k_s^{[1]}}$. If $\left(\underline{Z}_s^{k_s^{[1]}} - Z_s(Y_s^*)\right)/Z_s(Y_s^*) > p$ for any s, then the regret for scenario s in any feasible solution will exceed p, so remove $k_s^{[1]}$ from L_{New} for all s. If L_{New} is now empty, go to Step 1. Otherwise, let L_{Int} be the list of nodes in L_{New} whose lower-bound solutions happen to be all-integer. (Note that k_s may be in L_{Int} while k_t is not, even if k_s and k_t correspond to each other.)

Step 4. *Robustness test for integral solutions.* For each $k \in L_{\text{Int}}$, compute the maximum regret of the solution for node k across all scenarios. If the maximum regret exceeds p, then remove k from L_{Int}. (Note that dropping k from L_{Int} does not mean fathoming node k.) If L_{Int}

is now empty, go to step 1.

Step 5. *Update list L_R of robust solutions.* For each $k \in L_{\text{Int}}$, add k to L_R. If $|L_R| > N$, then we have too many robust solutions. Drop the $|L_R| - N$ solutions with the largest regret from L_R, remove the corresponding nodes (for all scenarios) from all trees, and set p equal to the largest maximum regret of any solution in L_R. Go to step 1.

The algorithm terminates when there are no unexplored nodes left. The authors state:

> Notice that the algorithm only eliminates nodes from the different scenario trees when a given lower bound does not meet the robustness criterion (step 2), or when we tighten the robustness criterion (reduce p) because we have already identified N robust solutions (step 4).[1] Hence when the algorithm finishes executing, for a given prespecified robustness parameter p, it will either have identified the best N robust solutions, or if it identifies $n < N$, possibly $n = 0$ robust solutions, then we can guarantee that these are the only robust solutions for the given p. (p. 184)

We disagree with the conclusion of this quote. In the next section, we give an example in which the algorithm will return a solution even though a more robust solution exists.

2 Counterexample to RISA

Consider the network pictured in Figure 1. There are 5 suppliers, 2 factories, and 2 scenarios. The figure shows supplier eligibility and costs. The numbers next to the links give scenario-specific transportation costs (scenario 1, then scenario 2). All suppliers have fixed costs of 50 (in both scenarios), and both factories have a demand of 1. There are no minimum procurement requirements of the type described in section 3.1 of Gutiérrez and Kouvelis (1995).

[Insert Figure 1 Here.]

Let $p = 1$ and $N = 1$ (that is, we want to find the single solution with smallest maximum regret, and start with relatively large p.) By inspection one can confirm that the optimal scenario solutions are $Y_1^* = (1, 0, 1, 0, 0)$ with objective value $Z_1(Y_1^*) = 200$ and $Y_2^* = (0, 1, 0, 1, 0)$ with

$Z_2(Y_2^*) = 200$. The minimax regret solution is $Y = (1,0,0,0,1)$ with $R_1 = R_2 = 0.125$ (where R_i is the percentage regret if scenario i occurs). Since each supplier is eligible to serve only a single factory, the weak relaxation of (DISP) is equivalent to (DISP) itself, so $\underline{Z}_s^k$ is simply the optimal objective value of the LP relaxation of (DISP) for scenario s when variables are fixed as in node k.

We will walk through the RISA algorithm step by step. The branch-and-bound trees are shown in Figure 2, with (single-scenario) objective value, solution vector, and regret displayed next to the nodes. One can easily verify that the solutions to the LP relaxations discussed in the example below are correct.

[Insert Figure 2 Here.]

Step 0. Solve the root nodes (with no variables fixed). The optimal solution for scenario 1 is to choose suppliers 1 and 3, with cost 200 and regret $R_1 = 0$ if scenario 1 occurs and $R_2 = 0.375$ if scenario 2 occurs, since the cost of this solution if scenario 2 occurs is 275. Similarly, the optimal solution for scenario 2 is to choose suppliers 2 and 4. This solution has cost 200 and regret $R_1 = 0.4$ and $R_2 = 0$.

Step 1. Choose a scenario, node, and variable to branch on. We'll choose scenario 1, node 1, and the variable y_1. (These choices are consistent with the branching rules described on p. 182 of Gutiérrez and Kouvelis (1995).) We remove node $k = 1$ from both trees and create nodes $1_s^{[0]}$ and $1_s^{[1]}$ for $s = 1, 2$, with y_1 fixed to 0 and 1 in these nodes, respectively. Since setting $y_1 = 0$ does not make the problem infeasible, at the end of this step $L_{\text{New}} = \{1_1^{[0]}, 1_1^{[1]}, 1_2^{[0]}, 1_2^{[1]}\}$ and we proceed to step 2. (Recall that the notation $1_2^{[0]}$, for instance, means child node [0] of node 1, scenario 2.)

Step 2. The optimal solution at node $1_1^{[0]}$ (child 0 for scenario 1) is $Y = (0,1,1,0,0)$ with cost 205 and regret $R_1 = 0.025$, $R_2 = 0.35$. For scenario 2, the optimal solution at the root node already had $y_1 = 0$, so this solution remains optimal for the child node. Both solutions pass the lower-bound robustness test since the lower bounds are within p $(= 1)$ of the optimal solution for the scenario.

Step 3. The optimal solution for scenario 1 is the same as at the root node since this solution already had $y_1 = 1$. For scenario 2, the optimal solution is $Y = (1,0,0,1,0)$ with cost 205 and regret $R_1 = 0.375$, $R_2 = 0.025$. Both solutions pass the lower-bound robustness test. All nodes are added to their respective trees, and since all solutions are integral, $L_{\text{Int}} = L_{\text{New}}$.

Step 4. The maximum regret for all solutions across all scenarios is less than p, so no nodes are removed from L_{Int}.

Step 5. Set $L_R = L_{\text{Int}}$. Since $|L_R| = 4$ and $N = 1$, 3 solutions must be dropped from the list. The best solution is at node $1_1^{[0]}$ with maximum regret 0.35, so we drop nodes $1_1^{[1]}$, $1_2^{[0]}$, and $1_2^{[1]}$.

When we drop $1_2^{[0]}$, we must also drop $1_1^{[0]}$ since nodes are dropped from all scenario trees simultaneously. Therefore, there are no nodes left to explore, and the algorithm terminates, returning $Y = (0,1,1,0,0)$ with maximum regret 0.35 as the "optimal" solution.

Two problems arise at this point. The first is that dropping $1_1^{[0]}$ because we've dropped $1_2^{[0]}$ is throwing out the good with the bad. Had we continued to explore this branch, we would have found the solution $Y = (0,1,0,0,1)$, which has maximum regret 0.15 ($R_1 = 0.15$, $R_2 = 0.1$). This solution, while not optimal, is still better than the solution returned.

Second, when we drop nodes $1_1^{[1]}$ and $1_2^{[1]}$, we fathom the section of the tree that contains the optimal (minimax regret) solution, so the solution returned is not optimal. The fathoming rule in step 5 implicitly assumes that child nodes never contain more robust solutions than their parents do. In the standard integer programming branch-and-bound algorithm, for example, it is true that the LP bound at a node is never strictly better than the LP bound at its parent node. This reasoning does not apply in RISA, though the algorithm is designed as though it does. The crux of the error is that optimization at each node is performed based on one objective (expected cost) while fathoming is performed based on another (robustness).

The fathoming rule in Step 5 is therefore incorrect. One could modify the algorithm so that nodes were not fathomed in step 5, but one would suspect that the algorithm would require much more branching and much larger computation times than those reported by Gutiérrez and Kouvelis.

3 Notes

[1]This actually occurs in step 5.

References

[1] Balinski, M. L. (1965). "Integer Programming: Methods, Uses, Computation," *Management Science* 12, 253–313.

[2] Efroymson, M. A. and T. L. Ray. (1966). "A Branch and Bound Algorithm for Factory Location," *Operations Research* 14, 361–368.

[3] Gutiérrez, Genaro J. and Panagiotis Kouvelis. (1995). "A Robustness Approach to International Sourcing," *Annals of Operations Research* 59, 165–193.

4 Figure Legends

Figure 1. Example network.

Figure 2. Branch-and-bound trees for example network.

5 Figures

Figure 1:

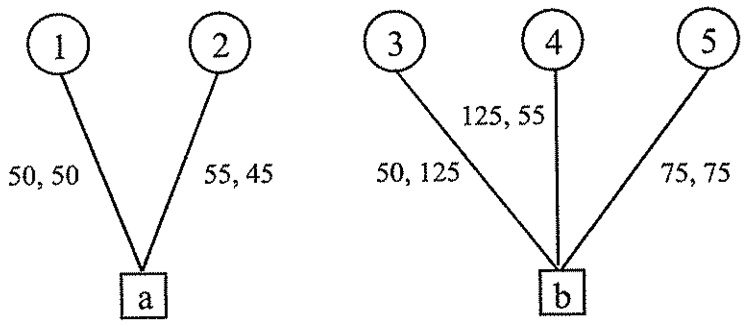
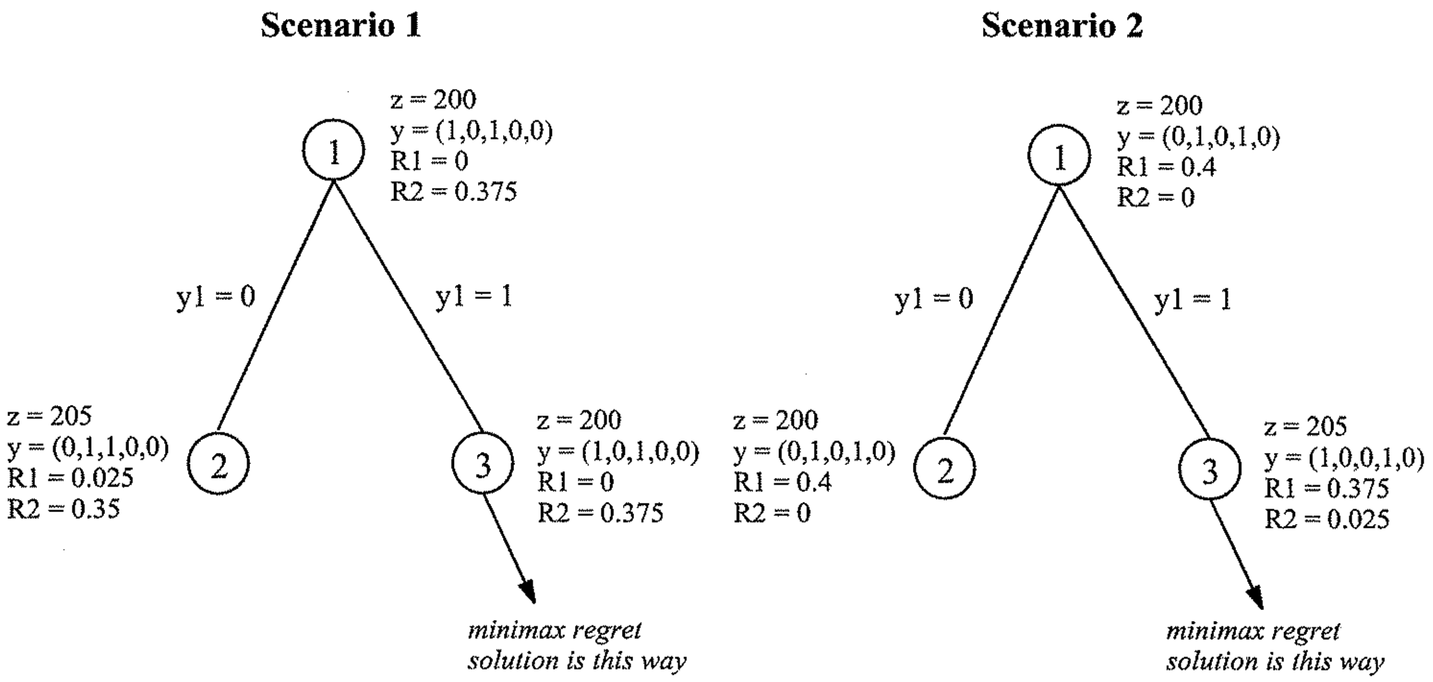


Figure 2:


Scenario 1
1
z = 200
y = (1,0,1,0,0)
R1 = 0
R2 = 0.375
y1 = 0
y1 = 1
z = 205
y = (0,1,1,0,0)
R1 = 0.025
R2 = 0.35
2
3
z = 200
y = (1,0,1,0,0)
R1 = 0
R2 = 0.375
minimax regret
solution is this way
Scenario 2
1
z = 200
y = (0,1,0,1,0)
R1 = 0.4
R2 = 0
y1 = 0
y1 = 1
z = 200
y = (0,1,0,1,0)
R1 = 0.4
R2 = 0
2
3
z = 205
y = (1,0,0,1,0)
R1 = 0.375
R2 = 0.025
minimax regret
solution is this way