Advances in Decision Technology and Intelligent Information Systems, Volume IV, K.J. Engemann and G.E. Lasker, Eds.
The International Institute for Advanced Studies in Systems Research and Cybernetics, 2003, pp. 51-55

Highly Efficient Algorithm for Two-Switch Location Problem

Boris S. Verkhovsky, Yuriy S. Polyakov
Computer Science Department, College of Computing Sciences
New Jersey Institute of Technology, Newark, NJ 07102

Keywords: optimal search algorithm; binary partitioning; topology design; two-switch location; separating line

Introduction

To communicate through satellites many large business corporations and governmental and military offices now rely on earth stations (ES), which are more complex and more expensive than relatively cheap, low-throughput, and low-quality "dishes" capable of simultaneous communication with only one or very few satellites. At the same time, medium or small "size" companies usually cannot financially afford an individualized ES and thus they have to share an ES with several other users to reduce the costs. Therefore an important role in acquring, installing, and maintaining ES's is now played by communications companies.

Communications industry has recently become very competitive. To reduce its expenses and make it more economically attractive for potential customers, a communication company needs to determine how many ES's of each type it needs, where to allocate them, and how the customers need to be interconnected with these earth stations. A correct decision can save dozens or even hundreds of millions of dollars annually, and hence can attract more users. The present paper describes an algorithm to solve this type of optimization problems.

Most of the existing approaches to solve such problems can be split into three categories: exact, where the optimal clusters and switch locations are mathematically guaranteed to be correct; heuristic, which are much faster than exact methods, but do not always yield the optimal solution; and hybrid, where heuristic methods are used to approach the optimal partition and then exact methods are used to obtain the final solution. A good review of all three types of approaches is presented in (Brimberg *et al.*, 2000)

In this paper, we focus on the two-switch (ES) case for Euclidian distances. The following exact methods have been studied to solve this problem: exponential (Kakusho, 1982), separation line (Drezner, 1984), and *D.-C.* (Differences of Convex functions) programming (Chen *et al.*, 1998). Although the proposed heuristic algorithm is in some sense similar to the minisum algorithm studied in (Drezner, 1984), it provides much faster, O(log$n$), solutions as it is demonstrated in the paper.

Problem definition for multi-switch location problem

For the sake of generality, we refer to ES's as switches.

1) Consider locations of n users which are specified by coordinates $P_i = (a_i, b_i)$, i=1,…, n. Each user is characterized by a "volume" of incoming and outgoing communication flow w_i ("weight" of i-th user's flow).
2) Let m be the number of switches, S_k be a set of all users P_i connected with the k-th switch C_k, and (u_k, v_k) be coordinates of C_k.
3) Let $f(w_i, P_i, C_k)$ be a cost function for the transmission link connecting the i-th user P_i and k-th switch C_k.
4) The objective is to minimize the total cost of all links and all switches:

$$\min W(C) = \sum_{k=1}^{m}\left[\sum\nolimits_{i\in S_k} f\left(w_i, P_i, C_k\right) + q_k \sum\nolimits_{i\in S_k} w_i\right] \tag{1}$$

where $q_k \sum_{i\in Sk} w_i$ is the cost of the k-th switch as a function of all outgoing and incoming flows. In this case, we do not consider the situation when the switches interact with each other because earth stations interact directly with a satellite. A detailed survey of various methods to solve the clustering problems, similar to the one described above, is presented in (Brimberg *et al.*, 2000).

Special cases of the problem

Suppose the number m of switches is fixed, a cost of every switch is either small or flow independent, and switch locations are specified. Then it is easy to find S_k. Indeed

$$S_k = \{i: \min_{1\le j\le m} f(w_i, P_i, C_j) = f(w_i, P_i, C_k)\}. \tag{2}$$

In the other special case, when all S_k are known, it is easy to find the optimal location for the k-th switch:

$$g_k\left(S_k\right) = \min_{(u_k, v_k)} \sum\nolimits_{i\in S_k} f\left(w_i, P_i, C_k\right) \text{ for } k = 1, \ldots, m \tag{3}$$

If $f(w_i, P_i, C_k) = w_i * l_2 (P_i , C_k)$, where $l_2(P_i, C_k) = [|a_i - u_k|^2 + |b_i - v_k|^2]^{1/2}$, then the problem (3) is known as the **Fermat-Weber problem**. It has been studied by many authors over the last forty years (Love *et al.*, 1988; Verkhovsky and Polyakov, 2002). Difficulties appear if the clusters S_k are not known, the cost of a switch is neither small nor flow independent, and the number m of switches and/or their best locations (u_k, v_k) for all k are not known either.

Linear switching cost function

If $q_k \sum_{i\in S_k} w_i = q_k \sum_{i\in S_{2k}} w_i + q_k \sum_{i\in S_{2k+1}} w_i$ for all S_{2k} and S_{2k+1}, such that $S_{2k} \cap S_{2k+1} = \varnothing$ and $S_{2k} \cup S_{2k+1} = S_k$ for all k=1, …, m, then the switching cost is a linear function of the network flow. In this case, it is clear that the optimal partition is not affected by the switching costs. Therefore the more is the number of clusters, the less is the total cost, that is $g_k > g_{2k} + g_{2k+1}$.

Two-switch location problem

It is important to emphasize that there is a substantial difference between the following two cases: m=1 and m=2. In the first case, only one single-switch minisum problem needs to be solved. In the case when m=2, multiple single-switch problems are solved by repetitive application of a specific algorithm, such as the Weiszfeld procedure (Love *et al.*, 1988) for the **Fermat-Weber problem**. The “naïve” approach is to check all possible pairs of clusters S_1 and S_2. Then there are $2^{n-1}–1$ different ways to partition n points into two clusters S_1 and S_2 and for each configuration two single-switch location problems need to be solved. Thus, the total time complexity of such a brute force algorithm is $O(2^n)$, (Kakusho, 1982).

Now let us briefly consider existing exact methods for the cases when the switch costs are neglected. Drezner (Drezner, 1984) suggests to use a straight line passing through a pair of user points and mathematically proves that such method of separation is guaranteed to find the optimal partition when the cost function is proportional to distances. In this case, the number of partitions is equal to the number of user pairs, that is $n(n–1)$. Obviously, two single-switch minisum problems have to be solved for each pair. Therefore the number of single-switch minisum problems that has to be solved is $O\left(n^2\right)$. Another exact method (Chen *et al.*, 1998) uses *d.-c.* programming, a recent technique of global optimization, to obtain a near-linear increase in the computation time as n increases. The approach is restricted to convex functions and thus has a limited scope of application to real-world problems.

Binary partitioning algorithm

The major idea of the proposed algorithm is based on experimental results demonstrating that a straight line, going through the optimal location for the single-switch minisum problem and one of the user points, in many cases can provide the optimal partition for a two-switch location problem with the same set of user points.

Let us start with a general clustering algorithm and then derive an efficient procedure based on this algorithm.

Clustering algorithm

1. Find a center of rotation R.
2. Convert all points/users to the polar system of coordinates (α, r) where R is the center of the coordinate system.

Comment: All points are also divided into the ones above and below the horizontal line going through R.

3. If α>180° then α:= α – 180°; (α,r):=(α, r, *below*), else (α,r):=(α, r, *above*).
4. Lexicographically sort all points: Let $U_1 := (\alpha_1, r_1, flag_1)$; $U_2 := (\alpha_2, r_2, flag_2)$. If $\alpha_1 < \alpha_2$, then $U_1 < U_2$; if $\alpha_1 = \alpha_2$ and $r_1 < r_2$, then $U_1 < U_2$; if $\alpha_1 = \alpha_2$ and $r_1 = r_2$ and $flag_1$=*above*, then $U_1 < U_2$.
5. Rotate straight line L around R for all x between 0° and 180°. Let $i = 1,\dots, n$. If ($\alpha_i <= x$ and $flag_i$ = *above*) or ($\alpha_i > x$ and $flag_i$ = *below*), then U_k belongs to cluster S_2; else U_k belongs to cluster S_1.

It is noteworthy that the user points are not simply sorted by angles, like in (Drezner, 1984), but are lexicographically ordered by angle, radius, and orientation. This approach excludes the special situation when two or more demand points lie on the same line, which had to be specially considered for the approach in (Drezner, 1984). It is obvious that n rotations are needed for the above algorithm.

A good candidate for the pivot in the case of a two-switch location problem is the optimal switch location obtained by solving the single-switch location problem for the same set of users. In some cases, less than 25% of the conducted experiments, however, this method does not yield the optimal partition. It mostly happens when we get one or few “fugitive” users, for which the cost function value for the current cluster is higher than if the user would be placed into the other cluster. In the majority of cases, a simple loop similar to the heuristic algorithm developed in (Cooper, 1964) resolves the issue and results in the optimal partition.

Now let us list the complete algorithm for the two-switch location problem. For simplicity, the points and indexes for these points are going to be considered as the same.

Binary partitioning algorithm

Step 1: Solve the single-switch minisum problem for all n users to find the pivot for rotation.

$$\min_{(u_0,v_0)} \sum_{i=1}^{n} f\left(w_i, P_i, C_0\right) \tag{4}$$

Step 2: Do steps 2 – 4 of the clustering algorithm with C_0 used instead of R.

Step 3: For all points i with $flag_{\mathrm{i}}$ = *above*, add their Cartesian coordinates to cluster S_1.
The rest of the points are added to cluster S_2.
Step 4: Let h_{min} be an arbitrary large number, $SO_1 := S_1$, $SO_2 := S_2$.
For all users $i = 1,\ldots,n$ do the following (rotate the line L) {
Step 4*a*: Let $x := \alpha_i$. If $flag_{\mathrm{i}}$ = *above*, then reassign the point i from cluster S_1 to S_2.
Else reassign the point i from cluster S_2 to S_1.
Step 4*b*: Compute

$$g_k\left(S_k(x)\right) := \min_{(u_k,v_k)} \sum_{i\in S_k(x)} f\left(w_i, P_i, C_k\right),\ k=1,2. \tag{5}$$

Step 4*c*: Compute

$$h(x) := g_1\left(S_1(x)\right) + g_2\left(S_2(x)\right) + q_1 \sum_{i\in S_1(x)} w_i + q_2 \sum_{i\in S_2(x)} w_i\,. \tag{6}$$

Step 4*d*: If $h(x) < h_{min}$ then $h_{min} := h(x)$, $SO_1 := S_1$, $SO_2 := S_2$.}
Step 5*:* {"Fugitive" handling after the main algorithm is complete}
Let *count* := 1;
While *count* > 0 {reassignments were done in the previous step or this part is called the first time}
{ *count* := 0;
Step 5*a:* If $f(w_i, P_i, C_1) > f(w_i, P_i, C_2)$ for $i\in SO_1$, then reassign $i\in SO_2$; *count*++.
If $f(w_i, P_i, C_2) > f(w_i, P_i, C_1)$ for $i\in SO_2$, then reassign $i\in SO_1$; *count*++.
Step 5*b:* Using (5) find the optimal locations of C_1 and C_2 for the new values of SO_1 and SO_2.}
Comment: C_1, SO_1, C_2, SO_2 is the final solution for the two-switch location problem.

As can be seen from the pseudocode above, the algorithm requires solving O(n) single-switch location problems. Step 5 is very similar to the heuristic algorithm developed in (Cooper, 1964), and is used to handle the situation when a straight line going through C_0 does not provide the optimal partition.

Computer experiments and optimization

We ran three hundred experiments with coordinates for the users and the associated weights generated using uniform random distribution on the interval (0,1). Cost functions $\left[w_i\, l_2\left(P_i, C\right)\right]^{\beta}$, where $\beta = 0.7,\ldots,1.0$, were used. Switching costs were considered to be linearly dependent on the network flow and thus were not taken into account.

According to (Veroy, 1989), the function $h(x)$ sometimes demonstrates a bimodal behavior and a special optimal search algorithm developed in (Veroy, 1989) for any discrete periodic bimodal function can be applied. The optimal search algorithm reduces time complexity for such a function from $\mathrm{O}(x)$ to $\mathrm{O}(\log x)$. In all the calculations, the minimal value obtained using that optimal search algorithm is compared with the minimal value produced by the non-accelerated algorithm before Step 5.

The developed binary partitioning algorithm is compared with the exact algorithm proposed by Drezner (Drezner, 1984) and the well-known heuristic algorithm described by Cooper (Cooper, 1964), for which an initial random partition was used.

For n = 15,..,50, both the binary partitioning and Cooper's algorithm are not always able to give the exact solution for the two-switch location problem. At the same time, Cooper's algorithm converges much faster (1–6 iterations versus n) to some stable condition, when no more reassignments could be made. The average relative deviation from the exact solution is slightly less for the binary partitioning algorithm. The function $h(x)$ does not always demonstrate bimodal behavior and thus the values of the minimum cost obtained using the non-accelerated method and the optimal search algorithm are sometimes different by 5%.

For n>50 (up to 1000 – highest calculated), both algorithms fail again to always provide the optimal solution. However, Cooper's algorithm is sometimes giving values of $h(x)$ up to 8% higher than the value found using the binary partitioning algorithm, which are always very close to the exact value due to the nature of the algorithm. Although the function $h(x)$ generated by the binary partitioning algorithm is not always purely bimodal, it always demonstrates a good overall bimodal behavior with a deep clearly-seen minimum and, in majority of the cases, the optimal search algorithm is coming to the global minimum. In cases when it is not getting to the global minimum, the optimal search algorithm lands very close, never exceeding 0.50% from the minimal value of $h(x)$. Considering the fact that Cooper's algorithm is run after the main part of the binary partitioning algorithm is done (in Step 5), additional one or two reassignment-location steps usually compensate for the differences in the values of h_{min} obtained using the non-accelerated algorithm and the optimal search algorithm version. Therefore, for n>50 the binary partitioning algorithm accelerated using the optimal search procedure developed in (Veroy, 1989) almost always provides more accurate solution than Cooper's algorithm. The number of iterations needed for the accelerated version of the binary partitioning algorithm is usually 1.2–4 more than for Cooper's algorithm. However, Cooper's algorithm would have to be run for several, if not dozens, of initial random partitions to attain the same average accuracy as the accelerated binary partitioning algorithm.

References

Brimberg, J., Hansen, P., Mladenovic, M. and E. Taillard (2000); Improvements and Comparison of Heuristics for Solving the Uncapacitated Multisource Weber Problem; Operations Research, Vol. 48, No. 3 (pp. 444-460).

Chen, P.-C., Hansen, P., Jaumard, B. and H. Tuy (1998); Solution of the Multisource Weber and Conditional Weber Problems by d.-c. Programming; Oper. Res., Vol. 46, No 4 (pp. 548-562).

Cooper, L. (1964); Heuristic Methods for Location Allocation Problems; SIAM Review, Vol. 6 (pp. 37-53).

Drezner, Z. (1984); The Planar Two-Center and Two-Median Problems; Transportation Science, Vol. 18, No. 4 (pp. 351-361).

Kakusho, O. (Oct. 1982); Cluster analysis: Definition, Algorithm, and Validation; Proc. 6th Int. Conf. Patt. Recogn., Munich (pp. 131-133).

Love, R. F., Morris, J. G. and G. O. Wesolowsky (1988); Facilities Location, Models and Methods; North-Holland. New York.

Veroy (now Verkhovsky), B. (1989); Optimal Search Algorithm for Extrema of a Discrete Periodic Bimodal Function; J. Complexity, Vol. 5 (pp. 238-250).

Verkhovsky, B. and Polyakov, Yu. (2002); Feedback Algorithm for the Single-Facility Minisum Problem; Annals of the European Academy of Sciences (pp. 127-136).