Journal of the Operations Research Society of Japan
Vol. 56, No. 3, September 2013, pp. 157–165

SOLVING CONSTRAINED TWO-FACILITY LOCATION PROBLEMS

Atsuo Suzuki
Nanzan University

Zvi Drezner
California State University-Fullerton

(Received May 8, 2013; Revised October 6, 2013)

Abstract A general approach to optimally solve multiple facility location problems based on the "Big Triangle Small Triangle" approach to solving single facility problems is proposed. The proposed procedure is especially effective when the solution is constrained to a given polygon such as the convex hull of demand points. The procedure is tested on the two facilities Weber problem with attraction and repulsion (WAR) with excellent computational results.

Keywords: Facility planning, global optimization, Weber location problem, attraction and repulsion, big triangle small triangle algorithm

1. Introduction

There are several global optimization algorithms designed for optimally solving location problems. The first algorithm was the "Big Square Small Square" (BSSS) proposed by Hansen et al. [9] and improved by Plastria [14]. The basic idea is to start with a "big square" that contains the optimal solution and perform a branch and bound in the area of that square. Branching is performed by partitioning the big square to four small squares and eliminate squares by evaluating upper and lower bounds in each square until the last remaining square is small enough and is eliminated. Drezner and Suzuki [6] partitioned the feasible area to triangles by the Delauney triangulation [11] and followed the same process using triangles rather than squares. They call it "Big Triangle Small Triangle" (BTST) algorithm. Schöbel and Scholz [15] generalized the procedure for solving problems in three or more dimensions by the "Big Cube Small Cube" (BCSC) algorithm. Berman et al. [1] suggested the "Big Segment Small Segment" to solve location problems on a network when the facility can be located anywhere on the network.

Both BSSS and BTST are designed for solving the same type of problems. One advantage of BTST is the ease of implementation when the solution is restricted to a set of convex polygons such as the convex hull of demand points. When implementing BSSS, part of a square may be infeasible. A lower bound in the square can serve as a lower bound for the feasible part of the square but may be "loose". However finding an upper bound may not be simple. Usually, a point in the square, such as its center, is selected and the value of the objective function at that point serves as an upper bound. For the upper bound to be correct, the value of the objective function must be evaluated at a feasible point. The user may need to find the intersection between the feasible area and the square in order to implement the algorithm. To the best of our knowledge this complication is not addressed in the literature. In many applications the optimal solution is known to be in the convex hull of demand points so there is no feasibility issue. However, when points outside the convex hull may have a better value of the objective function but are unacceptable, this issue must be addressed. On the other hand, if points outside the convex hull, such as in a

square, are allowed, the four vertices of the square can be added to the triangulation and the whole square is feasible.

Hansen et al. [9] solved the obnoxious facility location problem when the nuisance caused by the facility declines by the square of the distance from the facility. The unrestricted solution for this problem is clearly at infinity and allowing the facility to be located outside the convex hull may not be acceptable. Another advantage of the BTST over BSSS is that there are no demand points in the interior of the triangles while there may be demand points in the interior of squares. The presence of demand points in the interior of a polygon may complicate the establishment of an effective lower bound in a polygon. The obnoxious facility location problem has spikes (infinite values) at demand points which makes the surface of the objective function in the polygon may be very irregular when demand points exist in the interior of the polygon. When demand points are not present in the interior of a polygon, the surface is well behaved.

An example when the feasibility requirement is an issue is the "Weber problem with attraction and repulsion" (WAR) [7] which is the standard Weber problem [5, 12, 17] of minimizing the sum of weighted distances when some of the weights can be negative. Church and Garfinkel [3] investigated maximizing the sum of weighted distances when weights are positive which is equivalent to minimizing the sum when all weights are negative. Drezner and Wesolowsky [7] proved that when the sum of all weights is negative, the solution is at infinity and when the sum of all weights is positive, the solution point is finite (but can be outside the convex hull). It is well known that if all weights are positive, the solution must be in the convex hull. Therefore, restricting the solution to the convex hull of demand points or a union of polygons may be required.

This issue may exist in higher dimensions as well. Schöbel and Scholz [15] solved the WAR problem in three dimensions in a cube but did not consider any restrictions on the location of the solution point. They also solved the 2-median and 3-median problems in the plane but for these problems it is known that the solution is in the convex hull of the demand points and therefore the feasibility requirement is not necessary.

2. Formulation and Analysis

Consider a minimization problem of locating two facilities such that each demand point receives its services from the closest facility. Let n demand points be located in the plane. The Euclidean distance between demand point i and point X is $d_i(X)$. Two facilities X_1, X_2 need to be located in the convex hull of demand points. The objective function is a sum of monotonic functions of the minimum distance to the two facilities $\phi_i\left(\min\left\{d_i(X_1), d_i(X_2)\right\}\right)$. Some of the functions can be monotonically increasing and others are monotonically decreasing in the distance. The objective function to be minimized is:

$$F(X_1, X_2) = \sum_{i=1}^{n} \phi_i\left(\min\{d_i(X_1), d_i(X_2)\}\right) \tag{2.1}$$

This is a common type of the objective function for many location problems. Other structures such as replacing the sum operator by a max operator may require problem specific derivation of the lower bound in a triangle.

This problem can be solved by BCSC [15] by defining the solution space in four dimensions but since the locations of the facilities are restricted to the convex hull of demand points, the implementation of BCSC may be quite complicated as explained above. To implement BTST [6], a list of triangles is generated by the Delauney triangulation [11]. All

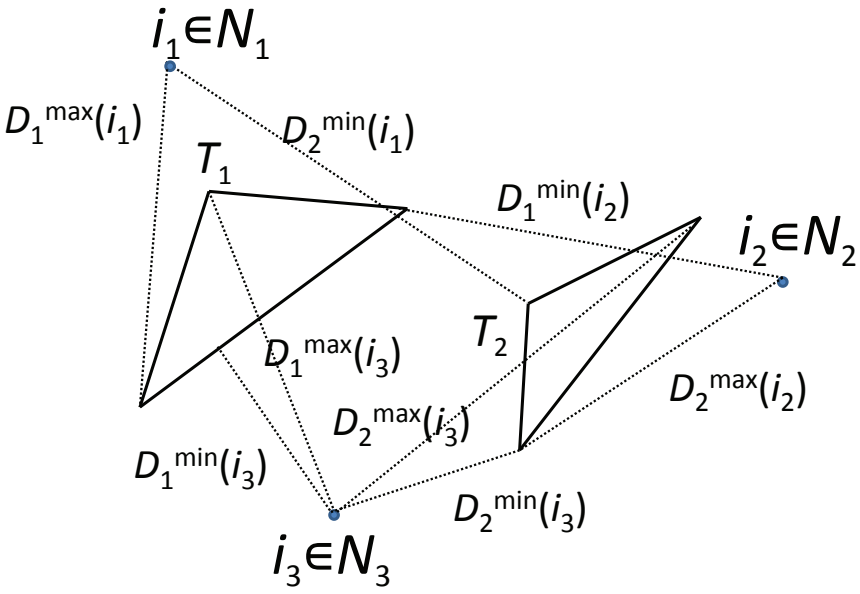

Figure 1: Examples of the demand points of N_1, N_2 and N_3

pairs of triangles are considered. Suppose that the two facilities must be located in two triangles, T_1 and T_2, one facility in each triangle. An upper bound is the value of the objective function when each facility is located at any point in the triangle such as the center of gravity of the triangle (unweighted average of its three vertices). A lower bound is more difficult to construct.

2.1. A General Approach to Constructing a Lower Bound

In order to implement BTST to the two facility location problem we assume that an effective lower bound for the single facility objective function when one facility is located in a triangle is available. Suppose that we need a lower bound when one facility is located in triangle T_1 and the second facility located at a triangle T_2.

Let $D_k^{\max}(i)$ and $D_k^{\min}(i)$ be the maximum and minimum distances, respectively, to triangle $k = 1, 2$ for demand point $i = 1, \ldots, n$. $D_k^{\max}(i)$ is the maximum distance to the three vertices of the triangle and calculating $D_k^{\min}(i)$ is detailed in [4].

The set N of demand points is divided into three subsets.

N_1 : Demand points i that satisfy $D_1^{\max}(i) \leq D_2^{\min}(i)$, i.e., $\min\{d_i(X_1), d_i(X_2)\} = d_i(X_1)$ for any $X_1 \in T_1$, $X_2 \in T_2$.

N_2 : Demand points i that satisfy $D_2^{\max}(i) \leq D_1^{\min}(i)$, i.e., $\min\{d_i(X_1), d_i(X_2)\} = d_i(X_2)$ for any $X_1 \in T_1$, $X_2 \in T_2$.

N_3 : All other demand points, i.e., $\min\{d_i(X_1), d_i(X_2)\}$ can be either $d_i(X_1)$ or $d_i(X_2)$.

In Figure 1, demand point i_1 belongs to N_1 because $D_1^{\max}(i_1) \leq D_2^{\min}(i_1)$ holds, demand point i_2 belongs to N_2 because $D_2^{\max}(i_2) \leq D_1^{\min}(i_2)$ holds, and demand point i_3 belongs to N_3 because neither $D_1^{\max}(i_3) \leq D_2^{\min}(i_3)$ nor $D_2^{\max}(i_3) \leq D_1^{\min}(i_3)$ holds.

By these definitions the objective function (2.1) can be expressed as:

$$F(X_1, X_2) = \sum_{i \in N_1} \phi_i \{d_i(X_1)\} + \sum_{i \in N_2} \phi_i \{d_i(X_2)\} + \sum_{i \in N_3} \phi_i \{\min\{d_i(X_1), d_i(X_2)\}\} \quad (2.2)$$

$$= F_1(X_1) + F_2(X_2) + F_3(X_1, X_2) \quad (2.3)$$

defining the three functions $F_1(X_1), F_2(X_2)$ and $F_3(X_1, X_2)$.

The first two functions $F_1(X_1)$ and $F_2(X_2)$ are equivalent to the single facility objective and thus can be bounded by the available lower bound for the single facility problem. For bounding $F_3(X_1, X_2)$, we observe that

$$\min\{D_1^{\min}(i), D_2^{\min}(i)\} \ \leq \ \min\{d_i(X_1), d_i(X_2)\} \ \leq \ \min\{D_1^{\max}(i), D_2^{\max}(i)\}$$

For a monotonically increasing function $\phi_i(d_i(X))$ the term $\phi_i\{\min\{d_i(X_1), d_i(X_2)\}\}$ is replaced by $\phi_i\left\{\min\{D_1^{\min}(i), D_2^{\min}(i)\}\right\}$ and for a monotonically decreasing function $\phi_i\{\min\{d_i(X_1), d_i(X_2)\}\}$ is replaced by $\phi_i\{\min\{D_1^{\max}(i), D_2^{\max}(i)\}\}$. The lower bound for $F(X_1, X_2)$ is the sum of these three lower bounds.

Once the first two bounds are effective, the effectiveness of the lower bound depends on the third term. The third bound may not be that effective but as the sizes of the two triangles shrink, the cardinality of N_3 decreases significantly as well. Therefore, the lower bound is effective for smaller triangles.

2.2. The Algorithm

Phase 1: Triangulate the convex hull of the demand points by the Delauney triangulation [11] obtaining a list T of triangles. Evaluate the value of the objective function for X_1 and X_2 located at the centers of all pairs of triangles. The best of these values is UB^*, an upper bound on the optimal value of the objective function.

Phase 2: Create a list of potential pairs of triangles. Scan all pairs of triangles in T and calculate the lower bound for each pair. All pairs of triangles for which $LB < UB^*(1-\epsilon)$ are compiled in a list of potential pairs of triangles.

Phase 3: Select the pair of triangles with the lowest LB. If $LB \geq UB^*(1-\epsilon)$, terminate the algorithm with UB^* as the solution. The "size" of a triangle can be defined in many ways. We define it as the sum of squares of distances between the center and the three vertices. Select the triangle with the larger size and split it into four small triangles defining four pairs of "small" triangles. The vertices of these four triangles include the original vertices of the triangle and the center of the sides of the triangle. Note that the size of the small triangle is one quarter of the size of the big triangle. Calculate four upper bounds and four lower bounds. Update UB^* if necessary. Remove the pair from the list of pairs of triangles and add up to four pairs as long as their lower bound satisfies $LB < UB^*(1-\epsilon)$. ϵ is a parameter which determines the accuracy of the optimal solution [6].

It is well known that the number of triangles generated by n demand points is $2n-\nu-2$ where ν is the number of vertices (or edges) of the convex hull of the demand points. For completeness we provide here a short proof. Let t be the number of triangles, E be the number of edges in the triangulation, and F be the number of faces which is equal to $t+1$ because the convex hull itself is a face.

Lemma 2.1. $E = n + t - 1$

Proof. By the "old" Euler's formula $n - E + F = 2$ (see Hilbert and Cohn-Vossen [10], p.290). By definition $F = t + 1$ and the Lemma follows.

Lemma 2.2. $t = 2n - \nu - 2$.

Proof. Each internal edge is common to two triangles and the ν edges on the circumference of the convex hull belong to only one triangle. Therefore, $3t + \nu$ is double the number of edges. By Lemma 2.1 $\frac{3t+\nu}{2} = n + t - 1$ leading to $t = 2n - \nu - 2$.

For example, for $n = 3$ we have $\nu = 3$ and thus $t = 1$.

2.3. Implementation Considerations

The list of pairs of triangles can be managed by utilizing the binary heap [2, 8] which is an efficient way to find the pair of triangles with the lowest lower bound. The binary heap data structure represents N values in a binary tree of depth $\log_2 N$ [2, 8]. Finding the smallest value of LB is done in $O(1)$ and adding or removing an element requires $O(\log N)$ time.

We construct the binary heap of the pairs of triangles with the key of their lower bounds. A maximum of K pairs of triangles is allowed. For example, we can reserve memory for $K = 1,000,000$ pairs. We keep in memory for every pair of triangles 13 values: the coordinates of the vertices of both triangles, and the lower bound for that triangle. The size of this array $A = \{a_{ij}\}$ is up to K rows and 13 columns. In addition, a pointer vector $\{v_k\}$ of up to length K containing the row number in the array of element k of the tree is maintained. For example, v_1 is the row number in A with the lowest lower bound. As explained above we just need a procedure to change the values in a row of the matrix A. To perform that we identify the parent of element k as $k_1 = \lfloor \frac{k}{2} \rfloor$ and its row in the matrix as row v_{k_1} and its two offspring as $2k$ and $2k+1$ and their rows as v_{2k} and v_{2k+1}. Suppose that element i is exchanged with element j. We prefer to retain the physical location of rows in the matrix A. We therefore only change the pointers v_i and v_j. v_j is moved to location i and v_i is moved to location j. When deleting an item, the last node of the tree (row v_k of the last k) needs to replace the deleted item. We simply leave the row of the deleted item unused and do not reduce the number of rows in the matrix A. Only when all K rows are used, a scan of the matrix A deleting all unused rows is performed.

The Algorithm Using the Binary Heap

1. Set the heap size to zero and calculate UB^* by evaluating the value of the objective function when the facilities are located at the centers of all $\frac{t(t-1)}{2}$ pairs of triangles.
2. Scan all pairs of triangles and add all pairs for which $LB < UB^*(1-\epsilon)$.
3. Select the pair with the lowest value of LB which is in position 1 of the tree. If its $LB \geq UB^*(1-\epsilon)$, stop with UB^* as the solution.
4. Select the larger of the two triangles of the pair and generate 4 small triangles each paired with the other triangle in the pair. Update UB^* if necessary. Add all pairs of triangles which satisfy $LB < UB^*(1-\epsilon)$. The first such pair replaces the pair in position 1 and the tree reorganized. Additional additions are added at the end of the heap (and the heap reorganized). If no small pairs of triangles are added, the big pair in position 1 is deleted.
5. If the number of elements in the heap reaches K, delete all unused rows in A and scan all heap members and delete those for which $LB \geq UB^*(1-\epsilon)$. If none is deleted, the algorithm is terminated with an "overflow" error message. UB^* can be used as a heuristic result.

3. An Example: The Two Facility WAR Problem

The Weber objective with attraction and repulsion (WAR) is defined as follows. Let n points be located in the plane each with an associated weights w_i, $i = 1, \ldots, n$. The weights can be positive or negative. The Euclidean distance between demand point i and point X is $d_i(X)$. Two facilities X_1, X_2 need to be located in the convex hull of demand points so that the objective function

$$F(X_1, X_2) = \sum_{i=1}^{n} w_i \min\{d_i(X_1), d_i(X_2)\} \tag{3.1}$$

is minimized.

This can be embedded in the general framework (2.1) by defining $\phi_i(d_i(X)) = w_i d_i(X)$. The function $\phi_i(d_i(X))$ is monotonically increasing for $w_i > 0$ and monotonically decreasing for $w_i < 0$.

3.1. A General Lower Bound

To implement the general approach (2.2), the objective function (3.1) can be expressed as:

$$F(X_1, X_2) = \sum_{i \in N_1} w_i d_i(X_1) + \sum_{i \in N_2} w_i d_i(X_2) + \sum_{i \in N_3} w_i \min\{d_i(X_1), d_i(X_2)\} \quad (3.2)$$

$$= F_1(X_1) + F_2(X_2) + F_3(X_1, X_2) \quad (3.3)$$

defining the three functions $F_1(X_1), F_2(X_2)$ and $F_3(X_1, X_2)$.

The first two functions $F_1(X_1)$ and $F_2(X_2)$ are equivalent to the single facility WAR objective. Each one can be represented as a difference between two convex functions and bounded from below by replacing the sum of positive terms with their tangent plane thus defining a sum of two concave functions which obtains its minimum at a vertex of their respective triangle. This lower bound is proven to be very effective. For details see [4, 6, 15] among many others. For bounding $F_3(X_1, X_2)$, we follow the procedure in Section 2.1:

For a positive w_i the term $w_i \min\{d_i(X_1), d_i(X_2)\}$ is replaced by $w_i \min\{D_1^{\min}(i), D_2^{\min}(i)\}$ and for a negative w_i it is replaced by $w_i \min\{D_1^{\max}(i), D_2^{\max}(i)\}$. The lower bound for $F(X_1, X_2)$ is the sum of these three lower bounds.

3.2. Improving the Lower Bound

We propose to improve the lower bound for the subset N_3.

Lemma 3.1. *Let a, b be two numbers. Then,* $\min\{a, b\} + \max\{a, b\} = a + b$.

Proof. If a is the minimum then b must be the maximum and vice versa.

Let the set N_3 be partitioned to the set N^+ of demand points with positive weights and N^- the set of demand points with negative weights ($-w_i$ where $w_i > 0$).

The WAR objective function for N_3 is:

$$F_3(X_1, X_2) = \sum_{i \in N^+} w_i \min\{d_i(X_1), d_i(X_2)\} - \sum_{i \in N^-} w_i \min\{d_i(X_1), d_i(X_2)\} \quad (3.4)$$

Applying Lemma 3.1 to the second sum in (3.4) we obtain

$$F_3(X_1, X_2) = \sum_{i \in N^+} w_i \min\{d_i(X_1), d_i(X_2)\} + \sum_{i \in N^-} w_i \max\{d_i(X_1), d_i(X_2)\}$$

$$- \sum_{i \in N^-} w_i \{d_i(X_1) + d_i(X_2)\} = F_4(X_1, X_2) + F_5(X_1, X_2) - F_6(X_1, X_2) \quad (3.5)$$

We will obtain the improved lower bound of $F_4(X_1, X_2)$ and $F_5(X_1, X_2)$. Let the center of the triangle of T_k for $k = 1, 2$ be X_k^c and $T_i(X_k)$ be the tangent plane to $d_i(X_k)$ for $k = 1, 2$ at X_k^c.

$$T_i(X_k) = d_i(X_k^c) + \frac{x_k^c - x_i}{d_i(X_k^c)}(x_k - x_k^c) + \frac{y_k^c - y_i}{d_i(X_k^c)}(y_k - y_k^c) \qquad \text{for } k = 1, 2 \quad (3.6)$$

By the convexity of the distances $d_i(X_k) \geq T_i(X_k)$. Therefore,

$$F_4(X_1, X_2) \geq L_1(X_1, X_2) = \sum_{i \in N^+} w_i \min\{T_i(X_1), T_i(X_2)\} \quad (3.7)$$

Table 1: Results

	Iterations			Max Pairs of Triangles			Time (sec.)		
n	Min	Max	Ave	Min	Max	Ave	Min	Max	Ave
10	18	330,374	39,863.1	39	131,976	15,879.2	0.00	42.39	8.49
20	38	10,453	2,872.7	68	3,779	1,302.6	0.00	0.30	0.05
50	551	99,202	12,699.2	544	36,124	4,951.9	0.03	9.98	2.15
100	1,160	140,867	19,878.7	720	66,085	8,935.9	0.28	25.37	5.16
200	822	22,335	7,895.2	514	6,659	3,655.1	0.45	7.00	2.71
500	1,949	548,045	76,668.7	766	206,703	29,918.4	19.02	481.31	85.44
1,000	2,782	523,273	91,283.9	1,703	353,009	52,944.6	161.07	955.40	286.20
2,000	21,332	420,201	144,602.8	8,563	178,459	67,966.7	1,256.63	2,350.90	1,585.68
5,000	18,442	3,954,543	924,237.4	10,446	1,657,627	387,259.9	19,171.60	51,100.50	25,690.11

$L_1(X_1, X_2)$ is concave as a sum of minima between two linear functions.

$\max\{d_i(X_1), d_i(X_2)\}$ is convex as a maximum between two convex functions. Therefore, it is bounded from below by the tangent hyperplane (in four dimensions) at any pair of points. Suppose that $d_i(X_1^c) \geq d_i(X_2^c)$. $T_i(X_1)$ is the tangent plane for the maximum function (and X_2 is ignored) because at X_k^c, $k = 1, 2$, the pair of points at which the tangent hyperplane is constructed, $\max\{d_i(X_1), d_i(X_2)\} = d_i(X_1)$. The same argument leads to $T_i(X_2)$ being the tangent hyperplane if $d_i(X_2) \geq d_i(X_1)$. Another way to show that the discussion above leads to a lower bound is the observation that

$$\max\{d_i(X_1), d_i(X_2)\} \geq \max\{T_i(X_1), d_i(X_2)\} \geq T_i(X_1)$$

and the same for X_2. Therefore, either $T_1(X_1)$ or $T_2(X_2)$ are lower bounds and we select the one for which $d_i(X_k^c)$ is larger. We call the tangent hyperplane as $T_i^*(X_1, X_2)$, and the weighted sum of these individual tangent planes as $L_2(X_1, X_2)$. Then,

$$F_5(X_1, X_2) \geq L_2(X_1, X_2) = \sum_{i \in N^+} w_i \{T_i^*(X_1, X_2)\} \tag{3.8}$$

$L_2(X_1, X_2)$ is linear and thus concave. In conclusion,

$$F_3(X_1, X_2) \geq L(X_1, X_2) = L_1(X_1, X_2) + L_2(X_1, X_2) - F_6(X_1, X_2) \tag{3.9}$$

$L(X_1, X_2)$ is concave because $L_1(X_1, X_2)$, $L_2(X_1, X_2)$ are concave and $F_6(X_1, X_2)$ is convex. Thus it obtains its minimum at one of the 9 pairs of vertices of the two triangles which is a better lower bound compared to the general lower bound.

4. Computational Experiments

The program was compiled by an Intel 11.1 Fortran Compiler with no parallel processing and run on a desktop with the Intel 870/i7 2.93GHz CPU Quad processor and 8GB RAM. Only one thread was used. The Delauney triangulation was coded using the Fortran program based on Sugihara and Iri [16] subroutines first developed by Ohya et al. [13].

Problems for various values of n demand points (See Table 1) were generated in a unit square. Weights were randomly generated in $[-1, 1]$. Thus, the number of positive and negative weights is about the same. This is considered the most difficult problem to solve [6]. When the sum of the weights is skewed to be either negative or positive the problems have

fewer local optima and are easier to solve. Ten problems were generated for each value of n. Since the value of the objective function is very close to zero, we did not use relative accuracy but used an accuracy of $\epsilon = 10^{-6}$ times the sum of the absolute values of all the weights. It means that the accuracy of the optimal solutions becomes lower as the number of demand points n becomes large. It is to avoid the serious numerical error based on the floating point number system for the computation.

In Table 1 we report the minimum maximum and average for: (i) the number of iterations, (ii) the maximum number of triangles throughout the search, and (iii) run time in seconds. As is shown in Table 1, the variations of these values are very large. It is an unexpected result. It means that that the solution of the Two Facility WAR problem deeply depends on both the locations and the weights of the demand points. We applied BTST algorithm which are expected to solve the problem effectively without the variation of the location and the weight. We think that the problem is essentially difficult even by the BTST algorithm which is proved to be the most effective method [6].

These problems were solved in a reasonable computer time and the maximum number of pairs of triangles is manageable. Several problems required a relatively large number of pairs of triangles and many iterations. This resulted in a large variance of the maximum number of triangles, number of iterations, and consequently run time because of the difficulty of the problem.

5. Conclusions

A general approach to optimally solve multiple facilities location problems was proposed. The procedure was tested on the two facilities WAR problem (Weber with attraction and repulsion). Computational experiments proved the efficiency of the procedure.

As future research we will test the new procedure on other non convex location problems with two facilities for which the solution is not known to be in the convex hull of demand points. For such problems the "Big Cube Small Cube" approach is difficult to implement.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 24241054.

References

[1] O. Berman, Z. Drezner, and D. Krass: Big segment small segment global optimization algorithm on networks. *Networks*, **58** (2011), 1–11.

[2] S. Carlsson: Improving worst-case behavior of heaps. *BIT Numerical Mathematics*, **24** (1984), 14–18.

[3] R.L. Church and R.S. Garfinkel: Locating an obnoxious facility on a network. *Transportation Science*, **12** (1978), 107–118.

[4] Z. Drezner: A general global optimization approach for solving location problems in the plane. *Journal of Global Optimization*, **37** (2007), 305–319.

[5] Z. Drezner, K. Klamroth, A. Schöbel, and G.O. Wesolowsky: The Weber problem. In Z. Drezner and H.W. Hamacher (eds.): *Facility Location: Applications and Theory* (Springer, Berlin, 2002), 1–36.

[6] Z. Drezner and A. Suzuki: The big triangle small triangle method for the solution of non-convex facility location problems. *Operations Research*, **52** (2004), 128–135.

[7] Z. Drezner and G.O. Wesolowsky: The Weber problem on the plane with some negative weights. *INFOR*, **29** (1991), 87–99.

[8] H.N. Gabow, Z. Galil, T. Spencer, and R.E. Tarjan: Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica*, **6** (1986), 109–122.

[9] P. Hansen, D. Peeters, and J.-F. Thisse: On the location of an obnoxious facility. *Sistemi Urbani*, **3** (1981), 299–317.

[10] D. Hilbert and S. Cohn-Vossen: *Geometry and the Imagination* (Chelsea Publishing Company, New York, 1956), English translation of Anschauliche Geometrie (1932).

[11] D.T. Lee and B.J. Schachter: Two algorithms for constructing a Delaunay triangulation. *International Journal of Parallel Programming*, **9** (1980), 219–242.

[12] R.F. Love, J.G. Morris, and G.O. Wesolowsky: *Facilities Location: Models & Methods* (North Holland, New York, 1988).

[13] T. Ohya, M. Iri, and K. Murota: Improvements of the incremental method of the Voronoi diagram with computational comparison of various algorithms. *Journal of the Operations Research Society of Japan*, **27** (1984), 306–337.

[14] F. Plastria: GBSSS, The generalized big square small square method for planar single facility location. *European Journal of Operational Research*, **62** (1992), 163–174.

[15] A. Schöbel and D. Scholz: The big cube small cube solution method for multidimensional facility location problems. *Computers and Operations Research*, **37** (2010), 115–122.

[16] K. Sugihara and M. Iri: A robust topology-oriented incremental algorithm for Voronoi diagram. *International Journal of Computational Geometry and Applications*, **4** (1994), 179–228.

[17] A. Weber: *ÜBer Den Standort Der Industrien, 1. Teil: Reine Theorie Des Standortes. English Translation: on the Location of Industries* (University of Chicago Press, Chicago, 1929), Originally published in Tübingen, Germany in 1909.

Atsuo Suzuki
Department of Information Systems and
Mathematical Sciences
Nanzan University
27 Seirei-cho, Seto-shi, Aichi 489-0863, Japan
E-mail: atsuo@nanzan-u.ac.jp