*Proceedings of the 1997 Winter Simulation Conference*
*ed. S. Andradóttir, K. J. Healy, D. H. Withers, and B. L. Nelson*

# SIMULATION-BASED APPROACH TO THE WAREHOUSE LOCATION PROBLEM FOR A LARGE-SCALE REAL INSTANCE

Kazuyoshi Hidaka
Hiroyuki Okano

Tokyo Research Laboratory
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa, 242, Japan

## ABSTRACT

We simulated a large-scale Uncapacitated Warehouse (Facility) Location Problems (UFLP) in the real world on a digital map, and found an approximate solution. The problem was to find a near-optimal solution for the number and locations of warehouses that minimize the sum of the transportation cost and fixed cost, and meet the needs of 6,800 customers. The network data of the digital map were efficiently used to obtain candidate warehouse locations, to simulate the transportation cost, to simulate the warehouse fixed cost, and to find a near-optimal solution for the number and locations of warehouses. In this paper, we propose a simulation-based approach to the large-scale UFLP, including a new heuristic algorithm named "Balloon Search," and give the results of our experiments.

## 1 INTRODUCTION

A manufacturing company in Japan currently has 11 warehouses that deliver spare parts for it's products to 6,800 customers located in the Kanto area. The sum of the fixed warehouse cost and the transportation cost is about 13 million dollars a year. The manufacturer wants to reduce the total cost by relocating or closing some of the current warehouses, and wants to know the optimal warehouse number and locations for its warehouses.

This is an instance of an optimization problem called the Uncapacitated Facility Location Problem (UFLP). Over the last 30 years, large amount of research had been done on the UFLP, and several exact (e.g., Mathematical Programings (Krarup 1983), and DUALOC (Erlenkotter 1978) and approximate (e.g., Greedy and Interchange Heuristics (Kuehn 1963, Cornuejols 1977) ), and Lagrangean Heuristics (Beasley 1993) algorithms have been proposed.

However, much of this research has been based on artificial data, and there has been little research on how to solve large-scale instances of the UFLP in the real world. For example, almost all of the previous research gave candidate locations and fixed costs of the warehouses, and transportation costs from warehouses to customers. But in real-world problems, these data must be calculated on the basis of a "proper model" or "proper assumptions," because it is difficult to determine the all fixed and transportation costs for candidate warehouses. This is especially true when we try to take a large number of warehouse candidates into account, so as to improve the optimality of the solution.

Moreover, UFLP is a NP-hard problem. That is, if the number of candidate warehouses (m) and customers (n) are growing larger, the number of constraint-equations for Integer Programming explodes and it is very hard not only to solve the problem, but even to represent it in the memory of a computer. For example, in the case of our real problem, n = 6,800 and m = 380,000, and it is very difficult to manipulate the huge number of constraint equations for such a problem size. Therefore, we need a practical method for finding an approximately optimal number and locations from a large number of possible locations.

In this paper, we propose a simulation-based approach to large-scale instance of the UFLP in the real world, including a new heuristic algorithm named "Balloon Search," and give the results of our experiments.

## 2 STRATEGY FOR SOLVING A LARGE-SCALE UFLP IN THE REAL WORLD

In order to solve a real instance of a large-scale UFLP, we developed simulation-based methods. That is, we simulated our UFLP in the real world on a digital map.

First, all warehouses and customers were assumed to be located at nodes of a network covering the whole Kanto area. From the customer nodes, sets of candidate warehouse locations were selected, and the fixed and transportation costs for each candidate were calculated on the basis of the real data for the current 11 warehouses. The transportation cost was assumed to be proportional to the delivery time.

Next, we optimized the number and locations of the warehouses in two steps.

The first step was to apply Greedy - Interchange heuristics with the warehouse candidates selected in the above manner. Using these heuristics, we found an approximately optimal solution for the number and locations of warehouses selected from the warehouse candidates.

Next, in order to improve the solution obtained in the first step, we devised a heuristic procedure for finding medians, and named it "Balloon Search."

This procedure may be outlined as follows: We relocated each warehouse obtained in the first-step solution to an adjacent node of the network that was not selected as a warehouse candidate node. Warehouse $i$ was relocated by aiming to find one median of the corresponding subnetwork that included only the subset of customers who were assigned to warehouse $i$. In this process, we first reduced the subset of customers so that it included only customers whose transportation costs were relatively low. Then, we increased the number of the customers in the subset step by step, and repeatedly found medians. During these processes, we assumed that the fixed cost of the adjacent node was the same as that of the warehouse obtained in the first-step.

After the first and second optimization steps, a near-optimal solution for the number and locations of warehouses was expected to be found from the large number of network nodes of the digital map.

## 3- MATHEMATICAL FORMULATION

We now give a mathematical definition of the problem.

Let:

$I$ : set of warehouse candidates

$J$ : set of customers

$i : i \in I$

$j : j \in J$

$m$ : number of warehouse candidate $m = |I|$

$n$ : number of customers $n = |J|$

$C_{ij}$ : transportation cost of delivery from warehouse candidate $i$ to customer $j$

$F_i$ : fixed cost of warehouse candidate $i$

$x_{ij}$ : =1 if customer $j$ is supplied with all parts from warehouse candidate $i$; otherwise, $x_{ij}$ =0

$y_i$ : =1 if the warehouse candidate $i$ is open; otherwise, $y_i$ =0

The UFLP can then be formulated as an integer program as follows:

Minimize:

$$\sum_{i}^{m}\sum_{j}^{n} C_{ij}x_{ij} + \sum_{i}^{m} F_i y_i \tag{1}$$

Subject to:

$$\sum_{i}^{m} x_{ij} = 1 : j \in J\,\psi \tag{2}$$

$$x_{ij} \le y_i : i \in I, j \in J\psi \tag{3}$$

$$x_{ij}, y_i \in \{0, 1\} \leftarrow \tag{4}$$

The problem is to minimize (1) under the constraints of (2), (3), and (4).

If customer $j$ is assigned to warehouse candidate $i$, then, all that customer's parts are supplied by warehouse candidate $i$.

## 4- DATA PREPARATION

### 4.1- Transportation Cost

We used a digital map that includes network data composed of edges (corresponding to road segments), nodes (corresponding to crossing points and terminations of roads), and attributes of edges and nodes. There are about 780,000 nodes and 380,000 edges, and the covered area is an approximately 200 km x 200 km square region around Tokyo named the Kanto area.

We assumed that the transportation cost per delivery from a warehouse to a customer is proportional to the simulated delivery time, and we calculated the "simulated delivery time" by dividing the shortest distance between a warehouse and a customer on the road network by the average speed of a delivery vehicle. The average speed of a delivery vehicle was varied according to the road type and road width.

We then obtained the simulated transportation cost of a customer by multiplying the simulated delivery time by the demand of the customer (the frequency of requests). The number of requests in the last year was used as demand data of each customer.

Moreover, we assumed that a customer is supplied with all its parts by the warehouse from which the simulated delivery time is shortest.

On the basis of the above assumptions, we calculated the simulated transportation costs of customers in the case of the current 11 warehouses, and assigned all customers to the nearest warehouses. We believe that we simulated the transportation cost well, because the share of each warehouse in the simulated transportation cost reflects its share in the real data, when we compare the total simulated transportation costs for the current 11 warehouses.

### 4.2 Warehouse Candidates

The number of warehouse candidates affects the load of the simulation in term of, for example, the CPU time and the sizes of work areas, such as a Dijkstra table (a table for storing each value of the shortest path between a candidate and a customer). It may also affect the quality of the solution obtained by the simulation; that is to say, it requires a tradeoff between the load of the simulation and the quality of the solution. To investigate the tradeoff, we chose different sets of candidates with sizes $|I|$ between 128 and 1024 ($|I| = c \times 128$, where $c = \{1, 2, \ldots, 8\}$).

In the problem instance we are interested in, the number of customers is $6,800$, which is more than 100 times greater than the current number of the warehouses. In the area served by a warehouse, the warehouse may be located near the center (median) of the customers served; these customers are typically densest near the warehouse. Thus, we assume that each warehouse candidate is located at the same node as one of the customers, and obtain the candidates by the following procedure:

First, the customers are clustered into 128 rectangles (buckets), each of which covers almost the same number of customers, using the same technique as in a $k$-d tree by Bentley (1990). Candidates are then selected from each rectangle. When the number of candidates is greater than 128 (the number of buckets), more than one candidate is selected from each bucket. In this way, more candidates are selected from buckets in rural areas than from buckets in central Tokyo; this allows us to investigate solutions in sparsely populated areas in more detail.

### 4.3 Fixed Costs of Warehouses

The fixed cost of a warehouse consists of an occupancy cost, which depends heavily on the land price, and a labor cost, which depends on the size of the warehouse. For a close approximation, we need the actual land price and the actual relationship between the labor cost and the size of the warehouse. However, neither is available, because of the huge effort required to determine them. We were given, as an alternative, a total fixed cost for each current warehouse.

We assumed that each fixed cost is proportional to the number of customers covered by a circle of fixed radius centered at each warehouse depends. When the number of customers laying within a circle is great, the customers are dense around the candidate, which implies that the land price may be high. It also implies that the number of customers assigned to the candidate may be great. When the number of customers laying within the circle is small, the customers are sparse around the candidate, and the fixed cost of the candidate may be low. According to a comparison with the actual fixed costs of the current warehouses, the approximated fixed costs of the candidates in our assumption follow the same order as the actual ones, and are thus good enough to be used in our simulation.

## 5 OPTIMIZATION

Optimization was done in two steps, using the simulated data. The first step consisted of Greedy - Interchange heuristics, and the second step of Balloon Search.

### 5.1 Greedy - Interchange Heuristics

The Greedy - Bump and Shift (Interchange) heuristics proposed by Kuehn (1963) have three processes:
In the Greedy process, warehouses are located at the most economical positions, one at a time, until no additional warehouses can be added without increasing the total cost.
In the Bump process, those warehouses that became uneconomical as a result of the placement of subsequent warehouses are eliminated.
In the Shift process, a warehouse is shifted to another potential location in the same territory if the relocation causes a reduction of the total cost.

Through our experiments, we found cases in which the optimal number of interchange (Shift) processes was larger than that of Greedy processes. Therfore, we modified the above heuristics as follows, and established the Greedy - Interchange heuristics.

Let the set of warehouses at the $l$th addition be $I_l^w$ ( $|I_l^w| = l$ ).
In the Greedy heuristic, first, warehouses are located at the most economical positions, one at a time, until no additional warehouses can be added without increasing the total cost.

Next, if there are $k$ warehouses after the Greedy heuristic terminates, continue adding warehouses after $k$ at the most economical positions, one at a time, until the number of warehouses becomes $k + d : 0 < d < k$, even if this increases the total cost.

Let $\Phi = \{I_l^w | l = k-d, ..., k, ..., k+d\}$ .
In the Interchange process, we apply the best swap search to all $I_l^w \in \Phi$, and find the optimal number and locations of warehouses. The best swap search is executed as follows:
Exchange a selected warehouse with the one of non-selected candidate warehouses that gives the largest cost reduction.
Repeat this process for all selected warehouses, and iterate until the total cost cannot be reduced any more.

In our experiments, a small value of $d$ was always sufficient ($0 < d < 3$).

## 5.2- Balloon Search

We now describe the idea of Balloon Search.
An approximate solution obtained by Greedy - Interchange heuristics only includes warehouses from those candidates selected from the huge number of network nodes. Therefore, we can decrease the total transportation cost by relocating each warehouse to a nearby "non-candidate" node of the network on the assumption that the fixed cost of a near node is the same as that of the previous node.

The transportation cost of each customer assigned to a warehouse is, of course, different. We assume that a customer whose transportation cost is relatively large has a higher probability of changing the supplier (warehouse) than another whose transportation cost is relatively small.

Therefore, we first relocate a warehouse to an adjacent node that decreases the total transportation cost of the subset of customers who are assigned to this warehouse and whose transportation costs are relatively small. We apply this relocation once to each warehouse, and then reassign each customer to the new warehouse from which the transportation cost is smallest.

We repeat this "single relocation of each warehouse" and reassignment of customers by increasing the size of each subset of customers.

When no relocation decreases the transportation cost, the algorithm terminates.

We search for the optimal warehouse location as the median of an expanding "Balloon" that includes the assigned customers.

In the following description we use the word "Balloon" for a subset of customers who are assigned to a warehouse and who are included in the first $N_{Bi}$ members in ascending order of transportation cost.
Let:
$I_{gi}$ : set of warehouses in the solution of the Greedy and Interchange heuristics
$I_w$ : set of warehouses for a solution
$w_i$ : node of warehouse $i$
$S_i$ : set of customers assigned to warehouse $i$
$B_i$ : Balloon of warehouse $i$ ( $B_i \subseteq S_i$ )
$N_i$ : number of customers supplied from warehouse $i$
$N_{Bi}$ : numbers of customer in Balloon $i$
$V_i^{adj}$ : set of adjacent nodes of $w_i$
$v_i : v_i \in V_i^{adj}$
$\Delta^{T_i}(v_i)$ : decrease in the transportation cost of Balloon $i$ when $w_i$ moves to the adjacent node $v_i$
$r_{inc}$ : expansion ratio of Balloon
$R_b$ : ratio of the number of customers included in all Balloons to the total number of customers

We now give the
*Balloon Search Procedure*

```
1. $I_w = I_{gi}$, initialize $R_b, r_{inc}$
2. $N_{Bi} = R_b \times N_i, Update\ B_i : i \in I_w$
3. If $\Delta^{T_i}(v_i) \leq 0 : v_i \in V_i^{adj}, i \in I_w$
       Terminate with the solution of $I_w$
   Else
       $w_i = v_i : max\left\{\Delta^{T_i}(v_i)\right\}, \Delta^{T_i}(v_i) > 0 : i \in I_w$
4. Update the assignment for every customer
   and calculate the sum of the transportation costs
5. Update $N_i, V_i^{adj} : i \in I_w$
6. $R_b = r_{inc} \times R_b$; go to 2.
```

In our experiment, we used an initial value of $R_b$ from the set {0.05, 0.10, 0.30, 0.50, 0.70, 1.00}, and an initial value of $r_{inc}$ from the set {1.05, 1.10, 1.20, 1.40}.
The value of $\Delta^{T_i}(v_i)$ in the above step 3 is calculated as follows:
3.a. Make a tree $TREE_i$ whose root is $w_i$ and whose nodes $V_i^{tree}$ are composed of the nodes of those customers $V_i^c$ assigned to warehouse $i$ and the nodes on the paths to warehouse $i$ from $v \in V_i^c$ .
3.b.- Let $d(v)$ be the sum of the demand (the frequency of requests) of $v$ and the demand of the all descendants of $v$ in the $TREE_i$.
3.c. Then
$\Delta^{T_i}(v_i) = 2(d(v_i) - d(w_i)) \times uc(v_i, w_i)$
Here, $uc(v_i, w_i)$ is the delivery cost per request along the edge connecting $v_i$ and $w_i$, from $v_i$ to $w_i$.

## 6- EXPERIMENTAL RESULTS

First, in order to evaluate the simulated fixed and transportation costs, we applied Greedy - Interchange heuristics in three cases with different average speeds on highways and metropolitan expressways (see Table 1), for 512 warehouse candidates. Figure 1 shows the results in case 1, and Figure 2 the results in case 3. In both figures, the x-coordinate denotes the number of warehouses and the y-coordinate the reduction ratio of the total cost in relation to the total cost of current 11 warehouses. These figures show that the number of warehouses that gives the lowest value is larger in case 3 than in case 1. This result, indicating that if the transportation cost is growing larger (that is, the average speed is slower and we need more time for delivery) we should open more warehouses to reduce the total cost, matches the decision-making strategy in the real world.

Table 1: Sets of Average Speed (km/h). HW: Highway, ME: Metropolitan Expressway, NP: National/Prefectural Road.

| *Case No.* | *HW* | *ME* | *NP* $\geq 13m$ | *NP* $< 13m$ | *others* |
|---|---|---|---|---|---|
| 1 | 80 | 50 | 40 | 30 | 30 |
| 2 | 120 | 80 | 40 | 30 | 30 |
| 3 | 40 | 40 | 40 | 30 | 30 |

Through the following experiment, the set of average speeds in case 1 was assumed. We must remember that the solution of our experiment is affected by the set of average speeds.

In Figure 1, we must pay attention to the fact that the minimum value was obtained for nine warehouses after application of the Interchange heuristic, even though the best solution obtained by the Greedy heuristic was for eight warehouses. This means that the Kuehn's heuristic strategy, which claims Interchange (Bump and Shift) heuristic should begin with optimal number of warehouses given by the Greedy heuristic, does not achieve the final minimal value in solving our problem instance. Therefore, we decided the optimal number of warehouses by observing the change in the total cost as the number of warehouses varied from 5 to 15.

While changing the number of warehouse candidates in the sequence of 128, 256, 384, 512, 640, 768, 896, and 1024, we applied the Greedy - Interchange heuristics and Balloon Search. Balloon Search was applied with initial $R_b$ values of 0.05, 0.10, 0.30, 0.50, 0.70, and 1.00, and with $r_{inc}$ values of 1.05, 1.10, 1.20, and 1.40. When the initial value of $R_b$ is 1.00, the Balloon is not expanded; that is, all customers are considered while medians are found at each step of Balloon Search.

In every experiment, with various numbers of warehouse candidates, the optimal number of warehouses was 9. However, the locations (nodes) of these nine warehouses and the total cost differ depending on the number of warehouse candidates.

Figure 3 shows the relation of the number of warehouse candidates and the minimal total cost achieved by each heuristic when the number of warehouse is 9. The x-coordinate represents the number of warehouse candidates and the y-coordinate the reduction ratio of the total cost in relation to the total cost of the current 11 warehouses. The four graphs correspond to the result of Greedy and Interchange heuristics, Balloon Search with no-expanding, and Balloon Search. The graph of Balloon Search was plotted by using the best combination of $R_b$ and $r_{inc}$ for each number of warehouse candidates.

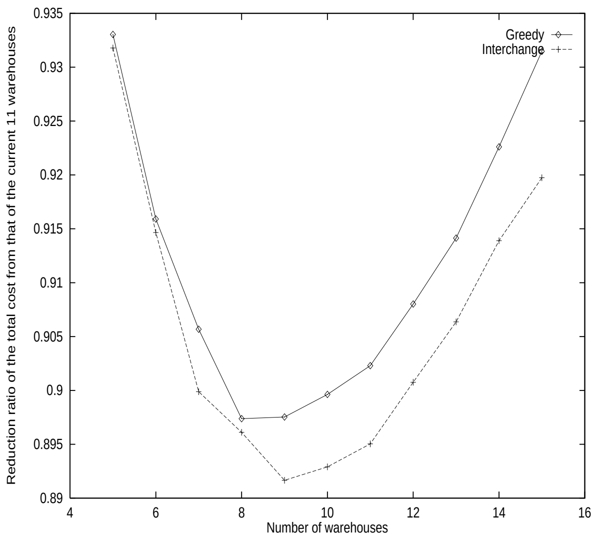

Figure 1: Number of Warehouses and the Minimal Total Cost with Average Speeds of (80, 50, 40, 40, 30) km/h.

The minimal value of the Greedy - Interchange heuristics is improved as the number of warehouse candidates grows larger, and it is inclined to be saturated when $x \geq 700$.

Table 2 shows how Balloon Search improved the solution of the Greedy - Interchange heuristics. At every number of warehouse candidates, Balloon Search improved the minimal cost of Greedy - Interchange heuristics, with an improvement ratio of 2%-11%.

Table 2: Improvement by Balloon Search.

| *Number of Candidates* | *Greedy-Interchange* | *Balloon Search* | *Improvement Ratio (%)* |
|---|---|---|---|
| 128 | 0.903461 | 0.896153 | 7.56 |
| 256 | 0.89427 | 0.889305 | 4.69 |
| 384 | 0.89157 | 0.886417 | 4.75 |
| 512 | 0.891656 | 0.879608 | 11.12 |
| 640 | 0.894176 | 0.89131 | 2.70 |
| 768 | 0.886699 | 0.88427 | 2.14 |
| 896 | 0.889275 | 0.884924 | 3.92 |
| 1024 | 0.889644 | 0.879394 | 9.28 |

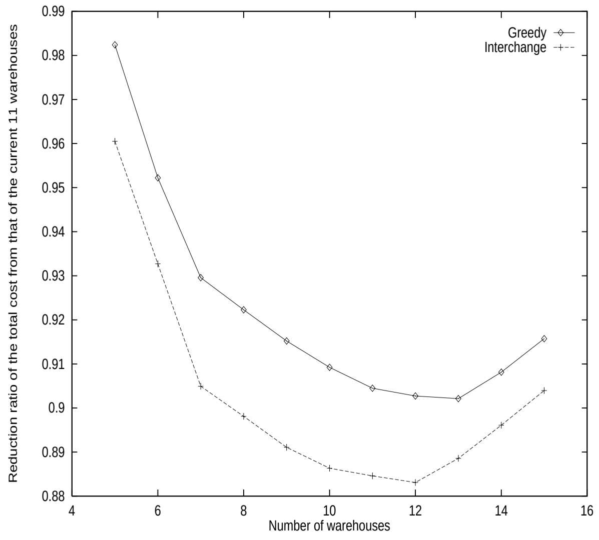

Figure 2: Number of Warehouses and the Minimal Total Cost with Average Speeds of (40, 40, 40, 40, 30) km/h.

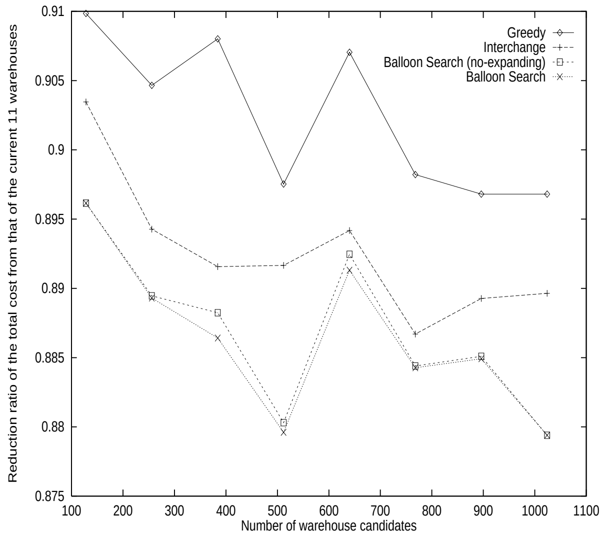

Figure 3: Number of Warehouse Candidates and the Minimal Total Cost.

However, the efficiency of the expanding the Balloon is not clear in this problem instance, because the expanded Balloon is slightly better than the unexpanded Balloon.

In Figure 3, we want to emphasize that, by using Balloon Search, we obtained as good a solution with 512 candidates as that with 1024 candidates. The result shows that we can find a good solution even if we select a relatively small number of warehouse candidates for the Greedy - Interchange heuristics. This is a very useful result, because a large number of warehouse candidates consumes a lot of computational time and space.

Figure 4 shows the relation between the number of iterations in Balloon Search and the total cost, with 512 warehouse candidates and initial $R_b$ values of 0.1 and 1.0. The x-coordinate represents the number of iterations in Balloon Search, and the y-coordinate the reduction ratio of the total cost in relation to the total cost of the current 11 warehouses. It can be trivially shown that Balloon Search with a small $r_{inc}$ value requires more iterations before terminating. Four curves of Balloon Search with an initial value of $R_b = 0.1$ (Expanding) start from a worse total cost than that of $R_b = 1.0$ (No-expanding) and terminate with a better total cost. From this figure, we can understand that the most careful and laborious increment ($r_{inc} = 1.05$) is not the best way. We thus have a chance to find a good solution with fast execution of Balloon Search.

We modified that Balloon Search procedure so that $R_b$ is increased after no warehouses can be moved any more with the current value of $R_b$. The aim of this modification is to execute Balloon Search very slowly, finding the exact medians at each value of $R_b$. In Figure 5, we plotted an example of the results of this modified Balloon Search, with 512 warehouse candidates, and with initial $R_b$ values of 0.10 (Expanding) and 1.0 (No-expanding). A comparison of Figures 4 and 5, shows that Balloon Search does not improve the optimal value, even if it is executed while finding exact medians at each value of $R_b$.

Our experiments led us to conclude that the optimal number of warehouses is 9, that the Greedy heuristic improved the total cost of the current 11 warehouses by 9% - 11%, that the Interchange heuristic improved the total cost by a further 0.5% - 1.5%, and and that the Balloon Search improved it a further 0.5% - 1.5%.

Finally, as shown in Figure 6, we were able to find approximately optimal warehouse locations after application of the three heuristics, which improved the total cost by about 12%.

## 7 CONCLUSIONS

We obtained an approximate solution of a large-scale UFLP in the real world, by simulating the transportation and fixed costs on a digital map and by applying a Balloon Search algorithm after the Greedy - Interchange Heuristic algorithms. We found approximately optimal solutions of the number and locations of warehouses, thus improving the total cost by about 12%. The solution was affected by the average speed of a delivery vehicle.

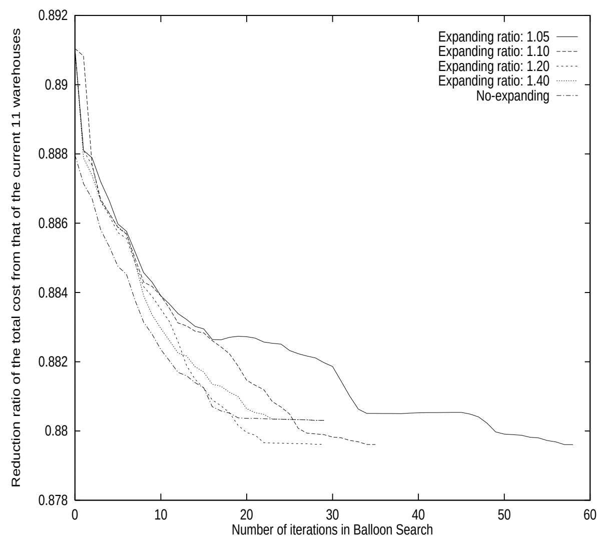

Figure 4: Number of Iterations in Balloon Search and the Minimal Total Cost with 512 Candidates, Initial $R_b = 0.1$.

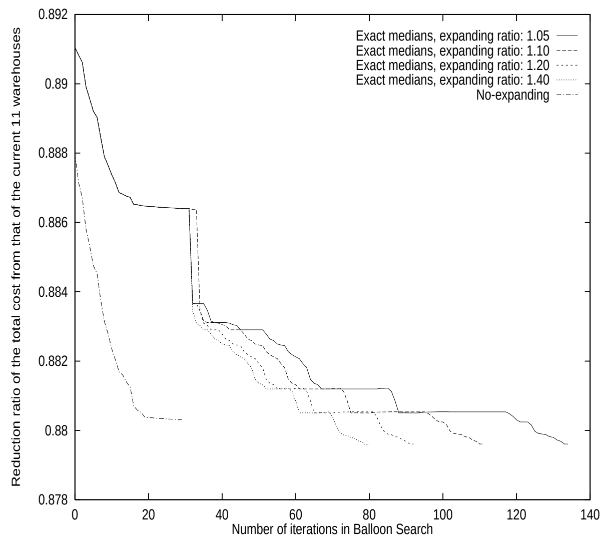

Figure 5: Number of Iterations in Balloon Search with Exact Medians and the Minimal Total Cost with 512 Candidates, Initial $R_b = 0.1$.

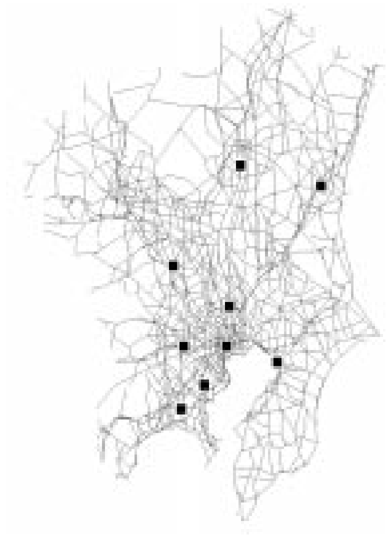

Figure 6: 9 Warehouse Locations of the Approximate Solution in the Kanto Area

We introduced the idea of Balloon Search, and applied it to solve a real large-scale instance of UFLP.

Through experiments on Balloon Search, we found that we can obtain a good solution even if we select a relatively small number of warehouse candidates for the Greedy - Interchange heuristics. Therefore, we can reduce the computational time and space of calculating Dijkstra table for warehouse candidates, which is the most laborious step in the whole procedure.

Our experiments also show that using a small expansion ratio of the Balloon or finding an exact median at the designated value of the Balloon size is not the best way to avoid local optima. These results indicate that we have a chance of finding a good solution quickly.

In the given problem instance, the efficiency of expanding the Balloon is not so clear. However, as we can see in Figure 4, Balloon Search is expected to be efficient for avoiding local optima.

Although Balloon Search improved the solutions, the strategy for defining the parameter values is not fixed. In further research, we want to establish a strategy for defining the parameter values depending on the size and the structure of the problem to be solved, and to use this new heuristic algorithm to solve the p-median problem on the graph.

Finally, we note that our customer (a manufacturing company in Japan) established its warehouse relocation plan on the basis of the simulated solution that we obtained in our experiments.

## REFERENCES

Krarup J., and P. M. Pruzan. 1983. The Simple Plant Location Problem: Survey and Synthesis. *European Journal of Operations Research* 12: 36–81.

Kuehn A., and M.J. Hamburger. 1963. A Heuristic Program for Locating Warehouses. *Management Science* 9: 643–666.

Cornuejols G., M. Fisher, and G. Nemhauser. 1977. Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms. *Management Science* 23(8): 789–810.

Erlenkotter D. 1978. A Dual-Based Procedure for Uncapacitated Facility Location. *Operations Research* 26(6): 992–1009.

Beasley J.E. 1993. Lagrangean Heuristics for Location Problems. *European Journal of Operational Research* 65: 383–399.

Bentley J.L. 1990. *k-d* Trees for Semidynamic Point Sets. *Sixth Annual ACM Symposium on Computational Geometry* 187–197.

## AUTHOR BIOGRAPHIES

**KAZUYOSHI HIDAKA** is a Manager of Optimization Solution Project of Exploratory Technology and Applications of Tokyo Research Laboratory of IBM Japan. He received a B.S. degree in applied physics from Science University of Tokyo in 1982, he received a M.S. degree in energy science from Tokyo Institute of Technology in 1984, and he received Ph.D. from Waseda University in 1996. His research interests include optimization, logistics, and educational computing.

**HIROYUKI OKANO** is a researcher of Tokyo Research Laboratory of IBM Japan. He received a B.S. degree (1988) and a M.S. degree (1990) in Information Science from the Tokyo University of Agriculture and Technology. He joined Tokyo Research Laboratory, IBM Japan, in 1990, and researched on user interface and system software. In 1995, he joined operations research group, and started to research on combinational optimization. His research interests include the traveling salesman problem and the vehicle routing problem.