Antoni KORCYL*

OPTIMISATION OF FACILITY LOCATION – HEURISTICS APPROACH

Abstract: The paper presents a heuristic algorithm for the Capacitated Facility Location Problem. The algorithm is based on a combination of Tabu Search and Filter-and-Fan metaheuristics.

Keywords: Capacitated Facility Location Problem, Tabu Search, Filter-and-Fan metaheuristics, heuristic approach.

1. Introduction

High costs related to the opening of a company's new branch, warehouse or customer service centre, justify taking into consideration decisions related to long-term strategy planning. Any attempt at achieving specified strategy planning, new production sites or service centres has to ensure not only the realisation of the set objective in short-term functioning function, but also profitable operation even in an unstable environment or population or market conditions. Decisions concerning the choice of a specified locations among the feasible ones are determined by the requirement to guarantee products and services of required quality. Location problems have been known for a long time and the related research has resulted in numerous different solutions, which in addition to purely knowledge-enriching and theoretical aspects also addressed real-life problems. Since the location problem is NP-hard not few of the proposed methods of solving it rae based on heuristic algorithms.

2. Classic location problem

The aim of the classic location problem is to minimise the aggregate cost of transporting products from the manufacturer to customers and of launching relevant facilities at the both ends of the route.

* AGH University of Science and Technology, Krakow, Poland

From among possible locations we choose those which meet the above requirements assuming that customers' demand is satisfied. Figure 1 shows the flow chart for the classic location problem.

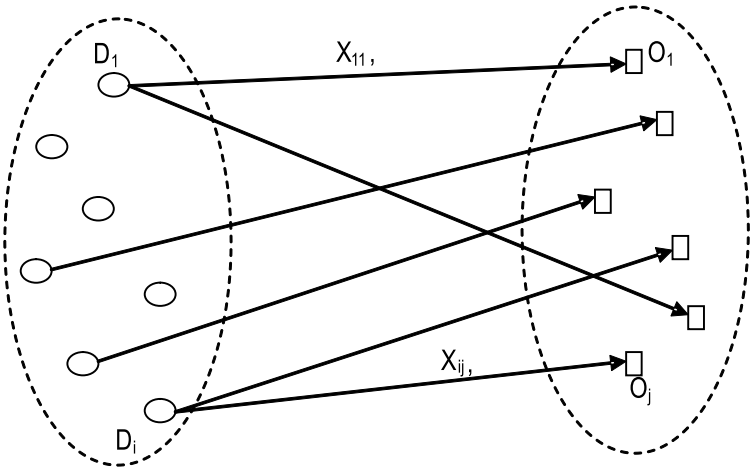

Fig. 1. *Flow chart for the classic location problem*

The classic location problem is also referred to as the warehouse location problem. Let us consider potential locations of new warehouses, that is a set $\boldsymbol{D} = \{1, \ldots, i, \ldots, D\}$, and the set of customers $\boldsymbol{O} = \{1, \ldots, j, \ldots, O\}$. The customers' demand for the product must be satisfied through adequate supplies from warehouses. The demand generated by the j-th customer will be denoted by b_j. The cost of transporting the product from the warehouse i to the customer j will be denoted by C_{ij}, F_i stands for the constant cost of launching the warehouse i. Let us introduce the following decision variables:

X_{ij} – describes servicing the j-th customer at the i-th warehouse;
Y_j – equals 1 if a warehouse is opened at the j-th location or 0 otherwise.

Supposing that every customer generates a normalised demand $b_j = 1$, the mathematical model of this problem takes the following form:
Minimise:

$$\sum_{i=1}^{D} \sum_{j=1}^{O} C_{ij} X_{ij} + \sum_{i=1}^{D} F_i Y_i \tag{1}$$

subject to:

$$X_{ij} \leqslant Y_i \quad i \in D, j \in O \tag{2}$$

$$\sum_{j=1}^{O} X_{ij} = 1 \quad i \in D \tag{3}$$

$$X_{ij} \in \{0,1\} \quad i \in D, j \in O \tag{4}$$

$$Y_i \in \{0,1\} \quad i \in O \tag{5}$$

The objective function minimises the aggregate costs of delivering products from warehouses to customers to satisfy the demand and launching new warehouses at specified locations (setup costs). Constraint (2) ensures that a warehouse is opened at a location from which there is a flow of the product to customer.

Constraint (3) ensures that each customer's demand is satisfied, but each customer is serviced by one warehouse only.

When the numbers of warehouses and customers are small, the solution of this problem may be found with use of mix-integer programming packages (Schrage *et al.* 1991). For the classic location problem with large numbers of suppliers and customers there is a simple heuristic algorithm has been proposed based on the Filter-and-Fan heuristics (Greistorfer *et al.* 2006).

3. Algorithm filter-and-fan

The search for new effective solutions to optimisation problems resulted in focusing efforts on intensive use of heuristic algorithms, and their implementations have successfully been applied in management, system engineering or medicine. The use of artificial feedback between the neighbourhood of the scanned area and feasible strategies has contributed enormously to the progress in the field analysed.

The objective of the proposed algorithms is to deliver such methods for examining the domain of solutions which would support the execution of the algorithm was effective at low computation cost.

The Filter-and-Fan algorithm proposed by F. Glover (1998) is a method based on the Scatter Search heuristics (Glover *et al.* 2000) and being a modification of the latter. This algorithm produces a purified (filtered) solution. The method became an alternative for the Ejection Chain metaheuristics. The Filter-and-Fan metaheuristics is an intermediate method between the Scatter Search and Ejection Chain ((Glover *et al.* 2006a, Glover *et al.* 2006b)).

This method consist in the integration of the *filtration* and *sequential dispersion* of candidates from the strategies listed in the Taboo Search metaheuristics and may be viewed as a reduced variant thereof, generating multiple paths within the *width* of the search strategy.

In terms of searching the neighbourhood, the method generates a mix of movements as a sequence of elementary moves. Graphically, the Filter-and-Fan algorithm may be illustrated with components of the neighbourhood tree, in which constraints are represented by elementary move components while nodes represent solutions obtained as a result of these moves. The rule does not apply to the root as it stands

for the initial solution to which the elementary moves refer. The maximum number L of levels allowed in a single move sequence determines the tree depth. In the neighbourhood tree, the width is first examined level by level. Each move is controlled by η_0, η_1, η - the parameters of the method.

The scheme of the Fan-and-Filter algorithm is shown below. X^* is the best solution found until the given time, $M(k)$ is a list of potential moves at the level k of the algorithm tree. L is an upper bound of the number of tree levels.

Build an initial solution X;

Step 0. ***Generating the list of component moves***

(a) Change X by executing single moves using local search until the local optimum X^* is found. Let $M(0)$ be the set of all moves determined during the most recent iteration of the local search procedure.

(b) Create a list of candidates $M(0)$ with η_0 highest move values in the set M.

(c) Put $X^* = X$, and let X be new start solution, that is the root of the search tree. Apply the best η_1 moves from the set $M(0)$ for X to create the first level of F&F algorithm tree with solutions $X_i(1)$ $(i = 1, \ldots, \eta_1)$. Put $k = 1$.

Step 1. ***Generating the filter and fan tree***

(a) Identify η_2 best derivative moves obtained from $M(0)$ for each of the solutions $X_i(k)(i = 1, \ldots, \eta_1)$ by computing values appropriate for the sample solution.

(b) If the best value found is better than X^*, then execute a compound move from $X_i(k)$ to a new solution X, improving the current one. Put $X^* = X$ and go back to *Step 0*.

(c) If the best value found is not better than X^*, choose n1 test derivative moves in order for them to become elements of the set $M(k)$.

(d) Apply the move set $M(k)$ to the corresponding solutions $X_i(k)$ to create $X_i(k+1)$.

(e) If $k = L$ then stop the search. Otherwise, put $k = k + 1$ and repeat *Step 1*.

The maximum number of levels considered in a single move sequence defines the tree depth. The algorithm runs until moves are executed which change the value of certain variable at each of the tree nodes. Exchange moves render the results implicit, until at two neighbouring nodes on a given branch of the tree; one of the variables changes from 0 to 1, while another one changes from 1 to 0. Since that point, the result of the move chosen at a tree node is transferred to the subsequence of nodes along the same branch.

The method is *adaptive*, because a type of neighbourhood and a move are chosen according to the current state of search. The method is dynamic until the moments that amount of single used *swift* (change of one's variable from zero to one or otherwise) build the movement depends of the level of the tree in which the best sample movement was found, depending one's iteration to other. The most frequently changes the value of one's variable from zero to one or from zero to one,

using for building the move depending of the tree level, in which their based best move was fund and his frequently is depends on the one iteration to the another. In different case the move depends on searching depth.

4. Unnormalized location problem

The non-normalised location problem is an extension of the classic location problem obtained by additionally specifying customers' demand and suppliers' supply. Every customer has to be serviced by at least one supplier, while in the classic location problem, every customer is serviced by exactly one supplier.

The flow chart for the non-normalised location problem is shown in Figure 2.

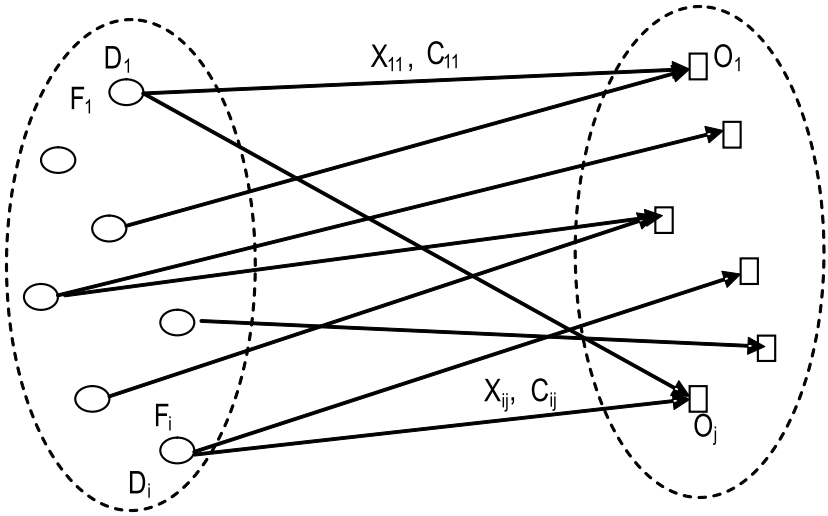

Fig. 2. *Flow scheme in the unnormalized location problem*

Where customers' demand in not normalised, additional parameters have to be included. Assume that every supplier has a limited quantity of goods, denoted by d_i, while customers generate demand denoted by b_i, as in the previous model. The mathematical model of this problem is as follows:
Minimise:

$$\sum_{i=D_1}^{D_i} \sum_{j=O_1}^{O_j} C_{ij} X_{ij} + \sum_{i=O_1}^{D_i} F_i Y_i \tag{6}$$

subject to:

$$\sum_{j=1}^{O} X_{ij} \leqslant d_i Y_i \quad i \in D \tag{7}$$

$$\sum_{i=1}^{D} X_{ij} \geqslant b_j \quad j \in O \tag{8}$$

$$\sum_{i=1}^{D} d_i \geqslant \sum_{j=O_1}^{O_j} b_j \tag{9}$$

$$X_{ij} \geqslant 0 \quad i \in D, j \in O \tag{10}$$

$$Y_i \in \{0,1\} \quad i \in O \tag{11}$$

The objective function minimises the aggregate costs of delivering products from warehouses to customers to satisfy the demand and launching new warehouses at specified locations. Constraint (7) ensures that the sum of all flows from a given warehouse to customers do not exceed supply capacity, provided that a warehouse is opened at a given location. Constraint (8) ensures that the sum of flows from all opened warehouses to a given customer satisfies the customer's demand. This constraint also includes the option of satisfying a given customer's demand of by multiple warehouses, unlike in the classic location problem, where each customer is to be supplied by one warehouse only. The supply and demand balance is guaranteed by constraint (9). Constraints (10) and (11) are the nonnegativity and binarity conditions for individual variables.

The presented mathematical model of the problem can also be solved with use of a standard discrete optimisation packet (Schrage *et al.* 1991).

5. Heuristic F&F–TS algorithm

Below, a heuristic algorithm for the non-normalised location problem is proposed. In the earlier simple F&F algorithm for normalised location problem (1), where a solution value obtained with use of a standard discrete optimisation package depended on the costs of opening a given location and transport therefrom to customers, in the case warehouse launch cost is significantly greater then transport cost (normalised location problem), the solution provides for a single location: the cheapest one. But when the costs of transport to individual customers and costs of opening locations are comparable, the solution is a set of multiple locations. In the non-normalised location problem, additional constraints are flows between trade participants. Flows to a given customer do not have to be arranged by a single supplier; a single customer's demand is to be satisfied by one or more suppliers. Consequently, this problem can be compared with the task assignment and machine load problem.

The proposed algorithm is based on the Filter-and-Fan and Tabu-Search heuristics (Korcyl, Sawik 1992, Glover 1989, Glover 1990, Glover 1995) and starts with checking whether the customers' demand is satisfiable by the suppliers (in accordance with constraint (9) in the model described in Section 4).

To each customer j, assign, if possible, a supplier i able to satisfy the customer's demand in full. If not possible, to each customer whose demand may not be satisfied by any single supplier, select suppliers in such a way that each customer's demand is satisfied and the least possible total cost. Compute the value of the solution thus obtained, which serves as a base solution X. Determine the parameters L and **taboo status** (i.e., the maximum number of iterations of the solutions of taboo status (it depends on the problem size), as well as η_1 and η_2.

Go to *Step 0* of the F&F algorithm and find the local optimum X^*. Define the set of moves which may be generated from the solution obtained. This set determines the starting **taboo list**, which depends on the numbers of customers and suppliers. The size of the taboo list may change during the start of algorithm under specified circumstances (especially at the time of finding a solution in Step 1 (b) of the F&F algorithm).

Generate a move list $M(0)$. Put $X^* = X$. Use η_1 moves from the set $M(0)$ to obtain the tree with solutions $X_i(1)(I = 1, \ldots, \eta_1)$.

Go to *Step 1* of the F&F algorithm. While executing consecutive stages of this step in search for a solution at the next tree node, apply the *qualification criterion function* to modify the tabu list. To intensify the search, use the long-term memory (the frequency of variable occurrence in the solutions having been examined), in the case of *Step 1 (b)*.

6. Computer simulations

Computer simulations for the proposed algorithm were executed on randomly generated data for different problem classes. The numbers of suppliers and customers fluctuated, as did the cost of transport and cost of warehouse setup. In the literature, no standard solution has been found against which simulation results could be verified. Therefore, the operation of algorithm was verified against the optimal solutions obtained for the non-normalised location problem modelled mathematically as in (6)–(11) with use of a standard discrete optimisation package.

The mathematical model of object location optimisation has been coded in the LINGO programming language, while the heuristic algorithm has been coded in the C++ language. Given the computational power of the computer used and the random nature of the generated data, the only quantities which admit reasonable comparison are average computation times and average deviations of values the obtained heuristic solutions from the optimal values.

For the optimal solutions, the computation time for tested examples ranged from few seconds (3 sec.) to over ten minutes (16 min. 18 sec.).

For the solutions generated by the heuristic algorithm, the computation time was shorter (depending on the problem complexity, from one second to one sixtieth of time for optimal solutions). The deviation was in the range from 0 to 10%, depending on the problem size and input parameters (mainly the parameter L).

7. Conclusions

The computer simulations carried out have proved that the computational complexity of the problem considered affects the decision concerning the selection of warehouse locations. Adopting a strategic decision may take time. On the other hand, a proper planning of facility locations for their multiyear profitable operation requires an enormous number of simulations to be run, with various considerations included, concerning economy, serviced population and anticipation of trends in customers' preferences. In such circumstances, heuristic methods should be used to carry so large a number of operations, even if the results so obtained deviate from the optimal values. This approach is an attempt at solving the location problem with limited resources.

Further work in the area should concentrate on intensifying the search for solutions generated in the both steps of the F&F algorithm and attempts at defining algorithm input parameters in relation to problem size. The introduced elements of the Tabu Search metaheuristics may be modified or work on using elements of other discrete optimisation heuristics may be carried.

References

Glover F. 1989. *Tabu Search – Part I*. ORSA Journal on Computing, Vol. 1, No. 3.

Glover F. 1990. *Tabu Search – Part II*. ORSA Journal on Computing, Vol. 32, No. 1.

Glover F. 1995.*Tabu Search fundamentals and uses*. Report Supported by National Science and Engineering Council of Canada under Grants 5-83998 and 5-84181.

Glover F., Laguna M. 2000. *Fundamentals of Scatter Search and Path Relinking*. Control and Cybernetics, Vol. 29, No. 3, pp. 653–684.

Glover F., Rego C. 2006. *Dynamic and adaptive neighborhood search in combinatorial optimization*. University of Colorado, Boulder, invited survey paper for *4OR*.

Glover F., Rego C. 2006. *Ejection Chain and Filter-and-Fan Methods in Combinatorial Optimization*. 4OR: A Quarterly Journal of Operations Research, 4(4), pp. 263–296.

Greistorfer P., Rego C. 2006. *A simple filter-and-fan approach to the facility location problem.* Computers & Operations Research 33, pp. 2590–2601.

Hesse O.S., Daskin M.S. 1998. *Strategic facility location: A review*. European Journal of Operational Research, Vol. 111, pp. 423–447.

Klose A., Drexl A. 2005. *Facility location models for distribution system design*. European Journal of Operational Research , Vol. 162, pp. 4–29.

Korcyl A., Sawik T. 1992. *Algorytm typu Tabu dla wyznaczania partii produkcyjnych i obciążenia maszyn w elastycznym systemie produkcyjnym*. Zeszyty Naukowe Politechniki Śląskiej.

Schrage L., Cunningham K. 1991. *LINGO, Optimization Modeling Language*. LINDO Systems Inc., Chicago.