Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Discrete Mathematics 39 (2012) 121–128

Electronic Notes in DISCRETE MATHEMATICS

www.elsevier.com/locate/endm

Solving Multifacility Huff Location Models on Networks Using Variable Neighborhood Search and Multi-Start Local Search Metaheuristics

Sanja Roksandić [a,1] Emilio Carrizosa [b,2] Dragan Urošević [a,3] Nenad Mladenović [c,4]

[a] *Mathematical Institute, Serbian Academy of Sciences and Arts, Belgrade, Serbia*

[b] *Faculty of Mathematics, University of Seville, Seville, Spain*

[c] *Department of Mathematics, SISCM, Brunel University, London, UK*

Abstract

We consider multifacility Huff location problems on networks. The mixed integer nonlinear optimization problem is solved using Variable Neighborhood Search and Multi-Start Local Search metaheuristics. Computational experience is reported.

Keywords: network, facility location, Huff model, variable neighborhood search, multi-start local search, global optimization.

1 Introduction

Location optimization problems on a network in a competitive environment have been extensively studied in OR. The problem was first presented by

[1] Email: `sanja@mi.sanu.ac.rs`
[2] Email: `ecarrizosa@us.es`
[3] Email: `draganu@mi.sanu.ac.rs`
[4] Email: `nenad.mladenovic@brunel.ac.uk`

1571-0653/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.endm.2012.10.017

Hakimi [2]. He formulated the competitive problem under the assumption that consumers deterministically choose the nearest store. In the real world, however, this assumption is not always acceptable, because consumers do not always choose the nearest store. Rather, consumers probabilistically choose among several stores. This probabilistic choice behavior is modeled by Huff, known as the Huff model. Huff formulated a model for capturing market share assuming that the probability that a consumer patronizes a shopping center is proportional to the attractiveness of the center and inversely proportional to a power of the distance to it. Although the original Huff model was based on an assumption that a market area is represented by a continuous plane with Euclidean distance, the model was extended to the network Huff model by Okabe and Kitamura [6] which was defined on a network with the shortest-path distance. Ghosh and McLafferty [1] considered their problem under the same assumption of discrete demand (nodal demand). Okunuki and Okabe [7] considered link based demand with slightly changed objective function.

In this paper we apply the network Huff model to a competitive location problem optimizing new facility locations on a network. We apply Variable Neighborhood Search and Multi-Start Local Search metaheuristics to solve this problem assuming that new facilities can be located at any point on the network and demand generated in the nodes.

1.1 Problem definition

We assume that customers are located in the vertices of a network $\mathcal{N} = (V, E)$, $V = \{v_1, \ldots, v_n\}$, $n \in \mathbb{N}$, $E \subseteq V^2$. The customers raise demand. We also assume that there are m facilities already located on the network. The facilities provide service and hence satisfy the raised demand. They are located at points $y_1, \ldots, y_m$ on network $\mathcal{N}$. The demand $w(v_i) = w_i$ associated with the vertex v_i, $i \in \{1, \ldots, n\}$, has the following properties:

$$w(v_i) \geq 0 \quad \text{and} \quad \sum_{v_i \in V} w(v_i) = 1\,.$$

The demand w may vary from vertex to vertex. For instance, it can be distributed uniformly to each vertex.

Our goal is to *locate p new facilities* $x_1, \ldots, x_p$ on the network which will respond to the customers' demand so that *the captured demand is maximal.*

To state the above location optimization problem more explicitly, let us formulate the network Huff model on $\mathcal{N}$. Firstly, let us introduce facility attractiveness, a property of each facility in the system. Facility attractiveness of a specific facility may be measured by the floor area, by the number of services/items that specific facility offers or in any other predefined way. Therefore, let us denote with $a_{y_1}, \dots, a_{y_m}$ and $a_{x_1}, \dots, a_{x_p}$ the attractiveness of the existing and new facilities, respectively. In order to unify the notations and simplify formulas, let us denote with a_{f_j} the attractiveness of

- the existing facility when $f \equiv y$, $j \in \{1, \dots, m\}$ and
- the new facility when $f \equiv x$, $j \in \{1, \dots, p\}$

located at point f_j. Let $d(v_i, f_j)$ be the distance from the customer located in vertex v_i to the facility at f_j on network $\mathcal{N}$. Let us now introduce distance deterrence function $F(d(v_i, f_j))$ of the customer in v_i from the facility at f_j. The distance deterrence function is a monotonically decreasing function with respect to $d(v_i, f_j)$. In his original model, Huff specified the distance deterrence function F as a power function, i.e.

$$F(d(v_i, f_j)) = d(v_i, f_j)^{-\lambda}, \ \lambda > 0\,.$$

Eventually, let $P(v_i, f_j)$ be the probability of customer in v_i choosing facility at f_j among the $m + p$ possible facilities. In these terms, the network Huff model is as follows

$$P(v_i, f_j) = \frac{a_{f_j} d(v_i, f_j)^{-\lambda}}{\sum_{f_k} a_{f_k} d(v_i, f_k)^{-\lambda}}\,.$$

Using the network Huff model, we proceed with formulating a problem for obtaining the demand $D(f_j)$ captured by facility at f_j. Let $D(v_i, f_j)$ be the demand in v_i captured by facility at f_j. Since the Huff model gives us the choice probability of customer in v_i choosing the facility at f_j, $D(v_i, f_j)$ is obtained from multiplying the probability $P(v_i, f_j)$ by $w(v_i)$, i.e.

$$D(v_i, f_j) = P(v_i, f_j) w(v_i) = \frac{a_{f_j} d(v_i, f_j)^{-\lambda}}{\sum_{f_k} a_{f_k} d(v_i, f_k)^{-\lambda}} w(v_i)\,. \tag{1}$$

To obtain the demand $D(f_j)$ captured by facility at f_j we need to sum the equation (1) over all vertices $v_i \in V$, i.e.

$$D(f_j) = \sum_{v_i \in V} D(v_i, f_j) = \sum_{v_i \in V} \frac{a_{f_j} d(v_i, f_j)^{-\lambda}}{\sum_{f_k} a_{f_k} d(v_i, f_k)^{-\lambda}} w(v_i)\,.$$

With m existing facilities located at points $y_1, \dots, y_m$ of network $\mathcal{N}$ we are supposed to locate p new facilities at points $x_1, \dots, x_p$ in order to compete them and capture maximal demand. The total demand captured only by new facilities is given by formula

$$\sum_{j=1}^{p} D(x_j) = \sum_{j=1}^{p} \sum_{v_i \in V} \frac{a_{x_j} d(v_i, x_j)^{-\lambda}}{\sum_{f_k} a_{f_k} d(v_i, f_k)^{-\lambda}} w(v_i) ,$$

where $f \in \{y, z\}$; $j \in \{1, \ldots, m\}$ if $f = y$, and $j \in \{1, \ldots, p\}$ if $f = z$. Since it has to be maximal, problem we have to solve is

$$\max_{x_1, \ldots, x_p \in \mathcal{N}} \sum_{j=1}^{p} \sum_{v_i \in V} \frac{a_{x_j} d(v_i, x_j)^{-\lambda}}{\sum_{f_k} a_{f_k} d(v_i, f_k)^{-\lambda}} w(v_i) .$$

2 Variable Neighborhood Search and the application to Huff location problem

2.1 Variable Neighborhood Search concept

Variable Neighborhood Search (VNS) ([3],[5]) is a well known metaheuristic method. It is designed for solving various combinatorial optimization problems. It uses local search procedure as one of its basic tools. Moreover, it involves systematic change of neighborhoods in the search. Unlike metaheuristics based on local search methods, VNS does not follow a trajectory, but explores increasingly distant neighborhoods of the current incumbent solution. Then, a local search routine is applied repeatedly to find local optima starting from these neighboring solutions. The search is re-centered around a new solution if and only if an improvement has been made with respect to global best solution.

Therefore, to construct different neighborhood structures and to perform a systematic search, we need to have a way for finding the distance between any two solutions, i.e., one needs to supply the solution space with some metric (or quasi-metric) and then induce neighborhoods from it. In the following sections we answer this problem-specific question for our particular problem.

The basic idea of VNS metaheuristic is to use more than one neighborhood structure and to proceed to a systematic change of them within a local search. The algorithm remains in the same solution until another solution better than the incumbent is found and then jumps there. Neighborhoods are usually ranked in such a way that intensification of the search around the current solution is followed naturally by diversification. The level of intensification or diversification can be controlled by a few easy to set parameters. We may view the VNS as a "shaking" process, where a movement to a neighborhood further from the current solution corresponds to a harder shake. Unlike random restart, the VNS allows a controlled increase in the level of the shake.

2.2 The application to the Huff network model

In order to implement VNS for the specific variant of the Huff location problem, we need to define solution representation as well as neighborhood structures and local search strategy. A particular solution consists of the location set for the new facilities on the given network. The location of each facility is uniquely determined by the edge, i.e. by the pair of vertices, and the position on the edge. The position on the edge is given by 1-dimension coordinate belonging to the $[0, 1]$ interval with respect to one of the vertices of the edge. Therefore, the location of the particular facility is given by the ordered pair $(x, (u, v))$, where the first entry refers to the position on the edge given by the second entry of the pair. The position x is calculated with respect to the first vertex of the pair related to the edge. As the particular solution consists of p facility locations, it will be presented as a list $[(x_1, (u_1, v_1)), \ldots, (x_p, (u_p, v_p))]$ of p ordered pairs where the ith pair corresponds to the ith facility location.

Let us now define a neighborhood structure in the solution space we introduced. If $s = [(x_1, (u_1, v_1)), \ldots, (x_p, (u_p, v_p))]$ is a solution, we may chose at random one of the p facilities and move it to one of the adjacent edges. Then we perform local search on the new edge by some of the well known techniques (line search, Fibonacci search, etc.) in order to reach the location which influences the objective function the most. We call this operation the *rank* 1 *stepping.* If we repeat this operation k times, $k <= p$, we call it the *rank k stepping.* We say that a solution s' is at the *step-distance* k from the solution s if s can be transformed into s' by applying the rank k stepping.

In order to improve the implementation performances, we have introduced another type of neighborhood structures. If $s = [(x_1, (u_1, v_1)), \ldots, (x_p, (u_p, v_p))]$ is a solution, we may chose at random two of the p new facilities of the solution and swap their locations. We call this operation the *rank* 1 *swapping.* If we repeat this operation k times, $k < \lfloor p/2 \rfloor$, we call it the *rank k swapping.* We say that a solution s' is at the *swap-distance* k from the solution s if s can be transformed into s' by applying the rank k swapping. The best results are obtained combining these two types of neighborhood structures.

To complete the VNS implementation, we have to define local search strategy. A *first improvement* local search strategy is performed: starting from a solution s we move a new facility to each of the adjacent edges until the first improvement of the objective function value is found. After repeating this for each of the p new facilities, the best of the obtained p locations is kept.

Let us denote by N_k, $k = 1, \ldots, k_{max}$ a finite sequence of pre-selected neighborhood structures, and by $N_k(x)$ the set of feasible solutions corresponding to neighborhood structure N_k at the point x, where x is an initial solution. Let us note that most local search metaheuristics use one neighborhood structure, i.e. $k_{max} = 1$. The algorithm presented here demonstrates the application of the basic VNS heuristic to the multifacility Huff location model on a network.

Algorithm 1 *VNS algorithm for Huff location model.*

```
1. Find an initial solution
2. Choose a stopping criterion
3. repeat
4.     $k = 1$
5.     while ($k <= k_{max}$)
6.         (Shaking) Choose at random either stepping or
                     swapping neighborhood type with equal probability.
                     Generate randomly a point $x'$ from $N_k(x)$
7.         (Local search) Apply first improvement local search method
                          with $x'$ as the initial solution;
                          the obtained local minimum denote by $x''$
8.         (Move or not) if $x''$ is better than the incumbent
9.                            move to $x''$ ($x = x''$)
10.                           $k = 1$
11.                       else $k = k + 1$
12. until the stopping criterion is met
```

Usually, the initial solution is determined by some constructive heuristic and then improved by local search before the beginning of actual VNS procedure. In this case the initial solution is generated randomly and then improved by Fibonacci local search method. The stopping criterion may be e.g. the predetermined maximal allowed CPU time, the maximal number of all iterations or the iterations between two improvements. Here the stopping criterion is maximal allowed CPU time. Often successive neighborhoods N_k are nested, but it is not necessary to be always the case. Let us note that the point x' is generated at random in order to avoid cycling which might occur if any deterministic rule was used. Basic VNS is a simple metaheuristic and its only parameter is k_{max} the preselected number of neighborhoods. Although, for each particular problem the solution representation, number and order of neighborhoods, and stopping condition should be defined in a way to ensure

Table 1
Computational results

	Instance				VNS				MSLS	
No.		n	q	p	best	avg	st.dev	best.time	best	best.time
1	lin105.tsp	105	13	8	50.00	50.00	0.00	3077.89	20.77	102.06
2	pr124.tsp	124	15	10	49.71	49.65	0.07	3844.65	22.55	2599.89
3	pr136.tsp	136	17	11	41.50	41.22	0.30	4205.92	23.41	3572.44
4	kroA150.tsp	150	18	12	41.67	41.39	0.22	4783.25	19.68	2157.19
5	kroB200.tsp	200	25	16	47.64	47.28	0.21	14687.06	17.96	8959.79

efficient execution of the search.

3 Computational results

The VNS algorithm for the specific Huff location model is implemented in C programming language on Linux platform. Test instances have been run on the computer with the i686 Intel Core 2 Duo CPU E6750 at 2.66GHz and 8GB RAM.

Since there does not exists the set of benchmark problems for the Huff location model, we have chosen small problems from the TSPLIB library where network dimension varies from 100 to 200. The number q of existing facilities is set to $n/8$ and the number of new ones to $2/3q$. The locations of existing facilities are created in the following way. Firstly, they have been chosen randomly. Then the VNS method was applied with 10% of total running time planned for the VNS algorithm execution for the particular test instance. In the end, randomly chosen p out of q existing facility locations were switched with the new facility locations obtained by the VNS algorithm. The attractiveness of each facility has been chosen randomly. Our experience shows that the best results are obtained if the probability of choosing either stepping or swapping shaking strategy is set to 0.5. k_{max} should be set to $p/2$. Maximal running time depends on the size of the particular test instance and it varies from 1 to 5 hours.

The results obtained by VNS are compared with the results obtained by the Multi-Start Local Search metaheuristic (MSLS). It is an iterative approach where a single iteration consists of generating a random solution and performing a local search strategy with the random solution as a starting point. In case there was the improvement of the objective function value, the incumbent is updated. We applied first improvement local search strategy. Initial solution was generated randomly. The algorithm is implemented in C programming language on Linux platform. The obtained results are presented in the Table 1. Both of the algorithms were given the same total execution time, although, only time when the best solution was reached is reported. Solutions (either best or average) are expressed as the percentage of the total demand.

4 Conclusion

We may conclude that VNS behaves better than MSLS in all tested examples in the sense of the objective function value, although for the same running period MSLS reaches its best solution in less time.

References

[1] Ghosh, A., McLafferty, S., and C. S. Craig, *Multifacility retail networks*, in Facility Locations, (Drezner, Z., ed). (1995), 301-330.

[2] Hakimi, S. L., *On locating new facilities in a competitive environment*, European Journal of Operational Research. **12**, (1983), 29–35.

[3] Hansen, P., and N. Mladenović, *Variable neighborhood search methods*, Encyclopedia of Optimization, 2nd Ed., Springer. **part(22)**, (2009), 3975–3989.

[4] Huff, D. L., *A Probabilistic Analysis of Shopping Center Trade Areas*, Land Economics. **39**, (1963), 81–90.

[5] Mladenović, N., and P. Hansen, *Variable neighborhood search: principles and applications*, European Journal of Operational Research. **130**, (1997), 449–467.

[6] Okabe, A., and M. Kitamura, *A computational method for market area analysis on a network*, Geographical Analysis. **28**, (1996), 330-349.

[7] Okunuki, K.-I., and A. Okabe, *Solving the Huff-Based Competitive Location Model on a Network with Link-Based Demand*, Annals of Operations Research. **111**, (2002), 239-252.