Journal of Industrial Engineering 5 (2010) 73-83

Maximal Benefit Location Problem for A Congested System

Reza Rabieyan, Mehdi Seifbarghy*

Islamic Azad University,Qazvin Branch, Department of Industrial Engineering, Qazvin, Iran

Received 5 Oct., 2009; Revised 25 Oct., 2009; Accepted 11 Nov., 2009

Abstract

Some servers are to be located at nodes of a network. Demand for services of these servers is located at each node and a subset of the nodes is to be chosen to locate one server in each. Each customer selects a server with a probability dependent on distance and a certain amount of benefit is achieved after giving service to the customer. Customers may waive receiving service with a known probability. The objective is to maximize the total benefit. In this paper, the problem is formulated, three solution algorithms are developed and applied to some numerical examples to analyze the results.

Key words: Location; Congested; Maximal Benefit; Network; Poisson.

1. Introduction

Much research has been carried out on location problems of minimizing total travel time, physical distance or some other travel related 'costs'. A common assumption in the addressed problems is that facilities are sufficiently large to meet any demand likely to be encountered. Optimization models for location with (nearly) constant demand are derived from the Location Set Covering Problem (Toregas et al. [26]), the p - median (Hakimi [15]; ReVelle and Swain [25]), and the Maximal Covering Location Problem (MCLP) (Church and ReVelle [11]).

Real world situations very often have variable and random demands for their services. Although the facility may be able to cope with average demands, there are times of heavy demands when it does not cope; such a facility is called congested (Boffey et al. [9]).

In congested systems, a facility will not be able to cope at times of very heavy demand. When this is the case, it may be possible for users to wait until the facility is free to serve them whereas in some other cases such as, for example maternity homes, it is not possible to wait. When waiting is not permitted or only limited waiting is allowed, then a user is lost to the fully occupied facility and their demand is either satisfied elsewhere or not at all. An obvious objective to minimize in such situations will thus be the total amount of demand lost (Boffey et al. [9]).

*Corresponding Author E-mail: m.seifbarghy@qiau.ac.ir

Location of congested systems is studied with regard to both mobile servers and immobile servers (fixed serves). In the case of immobile servers, users travel to facilities for service but in the case of mobile servers, the servers travel from facilities to the users. Facilities are often identical; otherwise, they differ regarding the number of servers. The studied model in this paper is mainly concerned with immobile servers. (For an excellent account of the location problems of mobile servers, see Berman and Krass [4]).

As regards immobile servers that are subject to congestion, it is reasonable to assume that each user's demand follows a time homogeneous Poisson process. Users' Prolonged waiting at the sites of facilities is undesirable and in some situations (e.g. emergency services) may be unacceptable. When queuing, though highly undesirable, is permitted, a (b,α) service level constraint may be introduced (Boffey et al. [9]).

This takes the form

$$P(\textit{facility } j \textit{ has} \le b \textit{ users in queue}) \ge \alpha, \qquad (1)$$

Where $0 < \alpha < 1$. If this constraint is satisfied for all facilities, the solution is said to be (b,α) *reliable*. An alternative service level constraint (Marianov and Serra [18]) is

$$P(\textit{waiting time at facility } j \le t) \ge \alpha. \qquad (2)$$

Several studies have been done on queuing systems in facilities. Grass and Harris [14] studied a location model

which included the given constraint in (1) in the case of M/M/1 queuing system in which arrival and service follow time homogeneous Poisson distribution and there is only a single server in each facility . In another study, the M/M/m queuing system in which m is the number of servers in each facility was formulated by Marianov and Serra [18]. Boffey et al. [9] focused on the corresponding models with single- and multi-server facilities using M/Er/m/N queuing systems with queues limited to no more than N but with the objective being the total number of users lost to the system per unit time. Here Er denotes the order r Erlang distribution which specializes to an exponential distribution when r = 1. The M/G/1 queuing system has also been studied by Rajagopalan and Yu [21].

All these studies ensured a desired service quality by including a suitable constraint in the formulation. An alternative way effectively achieves the same goal by incorporating an appropriate term in the objective. It, however, leads to a nonlinear mathematical program with integer constraints. Desrochers et al. [13] studied such a model in which the congestion effects are expressed in the objective in this way, but within an SLP-like setting rather than a pMP-like one. Assuming M/M/1 queuing systems for the servers at the facilities, the objective is to minimize the total weighted sum of the travel time of the users and queuing time. The corresponding M/M/m queue problem has been extended (Marianov [16]; Marianov et al. [17]).

Supposing a limit of b individual users queuing in line, each facility acts as an M/G/1/(b +1) queue. In the special case that service at a facility follows a time homogeneous Poisson distribution then we have an M/M/1/ (b + 1) queue. A reasonable objective to minimize is the total demand that is lost through queues being 'full'. This can be generalized to the case of m server facilities. In this regard, the case of the location of airline hubs modeled as M/D/m queues was studied by Marianov and Serra [19]. The cost of facility set up is taken into account as in SLP and the service level constraint is of the form of (1).

In many situations (e.g. bank branches, ATMs, preventative health care), quality of service, the travel time, and time waiting at a facility before receiving service would also be of importance for individual users. Usually the current state of the queuing system of a facility is unknown to the user before traveling to it and just some knowledge of the general condition may be available. Some conditions in which the congested location problem is formulated when there is a type of knowledge for the user are as follows:

(a) When users have very little knowledge of queue characteristics, (b) when users have estimates of mean queue length for all facilities and (c) when users have knowledge of the current state of relevant queues.

Moving forward from the above explanations on congested systems, now a more relevant and specific literature for the research in this paper is reviewed.

Daskin [12] states that the Maximum Expected Covering Location Problem (MEXCLP) maximizes the expected coverage by a free server. Berman et al. [7] developed a heuristic algorithm to locate optimally one server on a congested network. Based on the one-server location algorithm of Berman, two heuristics for locating p servers on a congested network were developed (Berman et al. [7]; Berman et al. [8]). According to ReVelle and Hogan [22], the Probabilistic Location Set Covering Problem forces all demands to be covered with a pre-specified reliability. Besides, the model maximizes the population covered with a pre-specified reliability (ReVelle and Hogan [24]). Batta [2] studied a model to investigate the effect of using expected service time dependent queuing disciplines on optimal location of a single server.

It should be noted that server independence and system-wide server busy probability is a common assumption used in many studies such as Daskin's [12] maximum expected coverage location problem (MEXCLP) and ReVelle and Hogan's [23] maximum availability location problem (MALP).

When dealing with the design of service networks such as health and EMS services, banking or distributed ticket selling services, the location of service centers has a strong influence on the congestion, and consequently on the quality of service. The Queuing Maximal Covering Location-Allocation Model with co-location of *m* servers per center can be stated as: "Locate *p* service centers, each with *m* servers and allocate users to them so to maximize covered population usually subject to a service quality." Marianov and Serra [18] formulated several maximal coverage models with one or more servers per service center and developed heuristics to solve the models for a 30-nodes service network. The main constraint of the models is that nobody stands on line for a time longer than a given time limit. An extension of the previous research (Marianov and Serra [19]) sought to cover all population and included server allocation to the facilities. The model suggested is a Set Covering formulation, which locates the least number of facilities and allocates the minimum number of servers (clerks, tellers, machines) to them in order to minimize queuing effects. Based on the number of servers allocated to each facility being constant or variable, two different models are proposed and some heuristics with good performance on a 55-node service network are developed and tested.

More recently, Berman and Drezner [3] introduced a multiple server location problem in a stochastic environment. Specifically, in their study, demand for service generated at nodes of the network was the random variable and the time to service calls was stochastic. Multiple servers were allowed to be located

on any potential location (assumed the set of nodes).The objective was to minimize the sum of travel time and average waiting time for all servers in this system.
Berman and Drezner [3] is a generalization of two recent papers. In the studies by Wang et al. [27] and Berman and Krass [5], at most one server can be located at any potential location and thus the system is modeled as an *M/M/*1 queue rather then an *M/M/k* queue. A similar model with additional constraints on demand that is lost because of insufficient coverage or congestion was studied by Berman et al. [6]. Its objective function is to minimize the number of facilities. Besides, Aboolian et al. [1] introduced the multiple server center location problem. In the problem, p servers are to be located at nodes of a network. Demand for services of these servers is located at each node and a subset of nodes is to be chosen to locate one or more servers in each. Each customer selects the closest server. The objective is to minimize the maximum time spent by any customer, including travel time and waiting time at the server sites. In fact, in Aboolian 's study, the problem was formulated and a heuristic was developed to solve it. The studies by Aboolian et al. [1] and Berman and Drezner [3] have some similarities. In both, the facilities are fixed, there are demands for service, service times are stochastic, and one or more servers is allowed to be located at any potential location. However, in Aboolian's study, the objective is to minimize the maximum travel time plus the average waiting time spent at the service facility for all customers. Applications for such a model include the locations of post offices, banks' branches, and medical facilities where the number of medical care personnel must be determined at each location (Aboolian et al. [1]).
In this paper, a network of several nodes is considered. Each node which could be considered as a customer has a demand for service. The demand follows a time homogeneous Poisson process. Some servers are to be located at the nodes of the network which means a subset of per node is to be chosen to locate one server in each. The service distribution is also Poisson and a maximum probability is considered for each server occupancy. Each customer selects the closest server and a certain amount of benefit is achieved after giving service to the customer. Customers may waive receiving service with a set probability. The objective is to maximize the total benefit.

The rest of the paper is ogranizied in this way: notation and problem formulation is given in section 2, and in section 3, solution algorithms including genetic algorithm, simulated annealing and heuristic algorithm are presented. Section 4 includes the reults of solving some numerical problems based on the designed algorithms, and section 5 is on the results and further research.

2. Notation and problem formulation

The notation is as follows:
n **:** Number of the network nodes (customers)
M **:** Number of service centers
φ_i **:** Demand rate for ith node (i=1,2,3,…,*n*)
μ : Service rate of the service centers
d_{ij} **:** Distance between nodes i and j
p_{ij} **:** The recourse probability of the ith node demand to service center at the node j
α **:** Probability of joining the queue line when the service center is occupied
β **:** Minimum probability of idleness at a service center in the long term
ρ_j **:** The occupancy probability (occupancy coeficcient) of the service center at the jth node
B_{ij} **:** obtained benefit rate in service center at node j from recoursing demand at node i
y_j is the model decision variable and is1 if a server is located at node j and is 0 otherwise.
The system under study is a network where arcs are the possible paths between nodes and nodes are demand centers (or customers) which could also be candidate locations for locating servers. Most ''central planning'' location models assume that if there is a facility at a particular node, all demands originating at that node are served by the same facility(server). According to the logit function (McFadden [20]) and assuming that customers travel to each facility with a probability that depends on the distance between the customer and facility, p_{ij} is computed is

$$p_{ij} = \frac{y_j e^{-d_{ij}}}{\sum_{j=1}^{n} y_j e^{-d_{ij}}} \qquad \forall\, i, j = 1,2,...,n$$

P_{ij} is necessarily zero if there is no open facility at node j. Demand rate at the jth service center can be stated as (4) since it is the sum of all demand percentages of the nodes which recourse to service center at the node j.

$$\sum_{i=1}^{n} p_{ij}.\varphi_i \quad \forall\ j = 1,2,3,...,n$$

Since each service center acts as an M/M/1 queuing system and β is the minimum probability of idleness, constraint (5) ensures that the occupancy probability of each service center is not greater than 1-β**.**

$$\rho_j = \sum_{i=1}^{n} \frac{p_{ij}.\varphi_i}{\mu} \le 1-\beta \quad \forall\, j = 1,2,3,...,n \tag{5}$$

Furthermore, refering to the definition of B_{ij}, the total benefit achieved at service centers would be equal to $\sum_{j=1}^{n}\sum_{i=1}^{n} B_{ij}.p_{ij}.\varphi_i.y_j$ if no demand is lost. There is a lost demand at server j with probability (1-α) when the server is occupied (with probability ρ_j). Based on this, the total benefit achieved at the server centers is as in Equation (6).

$$Z = \sum_{j=1}^{n}\sum_{i=1}^{n} B_{ij}.p_{ij}.\varphi_i.y_j.\left(\rho_j.\alpha + (1-\rho_j)\right) \tag{6}$$

The optimization model to locate servers can be stated as follows:

$$Z = \sum_{j=1}^{n}\sum_{i=1}^{n} B_{ij}.p_{ij}.\varphi_i.y_j.\left(\rho_j.\alpha + (1-\rho_j)\right) \tag{7}$$

$$\sum_{j=1}^{n} y_j = M \tag{8}$$

$$\rho_j = \sum_{i=1}^{n} \frac{p_{ij}.\varphi_i}{\mu} \le 1-\beta \quad \forall\, j = 1,2,3,...,n \tag{9}$$

$$p_{ij} = \frac{y_j e^{-d_{ij}}}{\sum_{j=1}^{n} y_j e^{-d_{ij}}} \qquad \forall\, i, j = 1,2,...,n \tag{10}$$

$$y_j = 0 \text{ or } 1 \tag{11}$$

3. Solution algorithms

The set covering and maximal covering problems are known NP-hard combinatorial optimization problems. Here, the model can be reduced to maximal covering and therefore it is NP-hard. Three heuristic algorithms are developed to solve the problem. Two of them , i.e. GA(Genetic Algorithm) and SA(Simulated Annealing) are meta heuristic.

3.1. Genetic Algorithm

Genetic Algorithm (GA) is a class of evolutionary algorithms and is based on a population of solutions. GA is a generic optimization method which can be applied to any problem if the feasible solutions of the problem can be represented as strings of binary or real numbers which are called chromosomes. Each chromosome has a fitness value that corresponds to the objective function value of the associated solution. Initially, there is a population of chromosomes that are randomly generated. Then, a number of chromosomes are selected as parents for mating in order to produce new chromosomes (solutions) that are called offsprings. The mating of parents is performed through applying the GA operators such as crossover and mutation. The selection of parents and producing offsprings are repeated until the stopping rule (for example, a certain number of iterations) is satisfied.
Before giving a general outline of the proposed genetic algorithm, some additional notations are defined as follows:
Population_size: Size of the population of solutions which is constant during the algorithm performance.
Max_iteration: Number of generations which should be produced until the algorithm stops.
p_c : Crossover rate (which is the probability of selecting a chromosome in each generation for crossovering)
p_m : Mutation rate (which is the probability of selecting a gen or bit inside a chromosome for mutating)
Fitness_function: Fitness value or the objective function value

The general outline of the proposed GA is like this:
Step 0: Initialize population_size, max_iteration, p_c and p_m.
Step 1: Randomly generate the initial population.
Step 2: Repeat until the max_iteration is reached:
Step 2.1: Perform the reproduction operator according to the roulette wheel rule and make a newer population.
Step 2.2: Select the parent chromosomes from the population with probability p_c.
Step 2.3: cross over:
a. Determine the pairs of parents among the parent chromosomes.
b. Apply the crossover operator to produce two offsprings for each pair.
c. Replace the offsprings in the population instead of the parents.
Step 2.4: Apply the mutation operator on the population with probability p_m.
Step 2.5: Save the best value in bv(best value).
Step 3: Print the bv solution.
In the proposed GA algorithm, each chromosome is represented by an n-dimensional vector like $A = [y_1, y_2, ..., y_n]$ whose *j th* entry stands for the *j th* node in the network. The value of the *ith* entry is the

number of servers at node *j*. $y_j = 1$ means that a server is located at node j and $y_j = 0$ means that no server is located at node *j*.

While randomly making the initial solutions (chromosomes) and also during the algorithm running, constraint (8) must be satisfied which means the number of servers at the nodes does not have to be greater than *M*.

Two-point crossover operator, that is, generating two random numbers for each chromosome to split it into three parts and then combining the parts of two parent chromosomes together is used and simple mutation operator (replacing bit 0 with 1 and vice versa with the probability p_m for each gen) is applied.

3.2. Simulated Annealing

Simulated Annealing (SA) is a random-search technique which simulates the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) in optimizing combinatorial optimization problems.

SA was first developed in 1983 to deal with highly nonlinear problems. SA approaches the global maximum like a bouncing ball which bounces over mountains from valley to valley. It begins at a high "temperature" which enables the ball to make very high bounces. As the temperature declines, the ball cannot bounce much high and settles to become trapped in relatively small ranges of valleys. A generating distribution generates possible valleys or states which are explored. An acceptance distribution is also defined which depends on the difference between the function value of the present generated valley to be explored and the last saved lowest valley. The acceptance distribution decides probabilistically whether to stay in a new lower valley or to bounce out of it. All the generating and acceptance distributions depend on the temperature .It has been proved that by carefully controlling the rate of cooling the temperature, SA can find the global optimum. However, this requires infinite time. Fast annealing and Very Fast Simulated Reannealing (VFSR) or Adaptive Simulated Annealing (ASA) are each in turn exponentially faster and overcome this problem.

Before giving a general outline of the proposed SA, some additional notations are defined as follows:

Max_iteration: Number of generations which should be produced until the algorithm is stoped.

In_loop_iteration: Maximum number of iterations at each temperature

$t\alpha$: Rate of cooling

T_{in} : Initial temperature

T_f : Final temperature

Fitness_function: Fitness value or the objective function

The general outline of the proposed SA is as follows:

Step1: Randomly generate the initial population of an arbitrary size and calculate the best fitness value $((fitness\_function)_0)$;*Solution* $= ((fitness\_function)_0)$

Step 2**:** Parameter initialization;

Step2.1: Set the annealing parameters; initial temperature T_{in}, final temperature T_f, maximum number of iterations (Max_iteration), maximum number of iterations at each loop (In_loop_iteration) and rate of cooling $t\alpha$

Step2.2: Initialize the iteration counter; iter=0;

Step 3: Annealing Schedule;

Step3.1: Inner loop initialization; il=0;

Step3.2: At every temperature achieve equilibrium. Execute inner loop until the condition in 3.2.5 is met;

Step3.2.1: il=il+1;

Step3.2.2: Generate a neighborhood solution (which satisfies the constraints). Calculate the fitness value $(fitness\_function)_{il}$

Step3.2.3:$\epsilon = (fitness\_function)_{il} - (fitness\_function)_{il-1}$

Step3.2.4: IF ($\epsilon \geq 0$) or (Random(0,1) $\leq e(-\epsilon/Tel)$)

THEN accept the new solution, *Solution* $= (fitness\_function)_{il}$

ELSE reject the new solution; *Solution* $= (fitness\_function)_{il-1}$

Step3.2.5: IF (il ≥ in_loop_iteration)

THEN terminate inner loop and GOTO Step 3.3

ELSE continue inner loop and GOTO Step 3.2

Step3.3: $T_{iter+1} = t\alpha . T_{iter}$

Step3.4: IF ($T_{iter+1} \leq T_f$)

THEN T_{iter+1} =max_iteration

ELSE iter = iter +1;

Step3.5: IF (iter ≥ max_iteration)

THEN terminate inner loop and GOTO Step 4

ELSE continue inner loop and GOTO Step 3.1

Step4:Terminate the best solution, *Solution* and stop.

3.3. Heuristic Algorithm

This heuristic algorithm begins with an initial solution with *n* (Number of nodes) bits which *M* (number of servers to be located) of them are 1 and the rest are 0. The initial solution is designed so that the servers are located at the nodes with higher demands. This would lie the total covered demand and the corresponding total benefit (the objective function value) in a high position

and likely closer to optimum. Based on the initial solution, a set of *M(n-M)* more solutions is generated from relocating each server (shifting 1 values in the initial solution to the locations with zero values). Since the initial solution has *M* bits with 1 values and each of them could be shifted to *(n-M)* locations, *M(n-M)* possible solutions are developed. The objective function values for the solutions are computed and the maximum value is selected. If the maximum value is greater than or equal to the initial solution's objective function value, the algorithm continues, otherwise stops.

4. Numerical examples and the results

Numerical examples are designed for the case of n =30 and n =60 nodes. B_{ij} is assumed to be 1 for each customer i at each node j, α equals to0.7 or 0.9 and β equals to0.05 or 0.15 and the values are constant for all the examples.

For the case of n =30, demand rates of the nodes (φ_i's) and distances between the nodes (d_{ij}'s) are randomly generated from [50, 100] and [2,10] respectively, service rate (μ) is 400 or 500 and the number of service centers (M) to be located is assumed to be 10 or 15. Randomly generated demand rates are 100, 56, 64, 87, 68, 97, 67, 60, 98, 89, 78, 87, 69, 96, 72, 69, 50, 99, 98, 97, 85, 84, 83, 82, 81, 71, 72, 73, 74, 75 respectively for the nodes.

The GA parameters, Max_iteration, Population_size, p_c and p_m are assumed 1000, 60 or 80, 0.6 or 0.8 and 0.01 or 0.001 respectively.

The SA parameters including Max_iteration, In_loop_iteration, $t\alpha$, T_{in} and T_f are supposed to be 4000, 40, 0.9 or 0.95, 0.0000001 and 80 or 100.

Considering the GA parameters for the case of n =30, there is seven parameters of α , β, μ, M, Population_size, p_c and p_m with two-level values and accordingly it is possible to design 2^7 numerical problems. Tuning the GA parameters, the partial factorial designs with 2^{k-2} have been used. Each numerical problem is run three times, therefore 3×2^5=96 experiments are designed and run and the fitness functions are obtained. Regarding the fitness function as the response variable and the seven parameters as the independent variables and using Minitab 14 statistical package, the Regression equation is obtained and optimized for each numerical problem and also the optimized values of the GA parameters are obtained.

The same method is used for tuning the parameters of the proposed SA algorithm.

Table 1 gives the results of running the proposed genetic algorithm for three times with considering just four problems from the total of 32 problems.

Table 1
results from three times running the proposed genetic algorithm considering 4 sample numerical problems

NO	α	B	μ	*M*	Population_size	P_c	P_m	GA Fitnessfunction Value
1	0.7	0.15	500	15	60	0.8	0.010	1428.42 1427.22 1429.30
2	0.9	0.15	400	15	60	0.8	0.010	1516.50 1517.07 1526.04
3	0.7	0.05	400	15	60	0.6	0.010	1380.55 1392.29 1385.35
4	0.7	0.05	500	10	80	0.8	0.010	1562.88 1556.48 1555.49

Considering the parameters α, β, μ, M, Population_size, p_c and p_m as the independent variables and replacing them with $n_1, n_2, n_3, n_4, n_5, n_6, n_7$ respectively, the fitness function value as a Regresion equation is as in (12).

$$f(n_1, n_2, n_3, n_4, n_5, n_6, n_7) = -11248.3000 + 14750.6000\ n_1 + 95380.5000\ n_2 + 24.4552n_3 - 19.4181n_5 + 3775.2400n_6 + 72504.0000\ n_7 - 23.1700n_1n_3 - 112650.000\ 0n_1n_2 - 0.1465n_1n_5 - 4028.3600n_1n_6 - 3875.6000n_1n_7 - 87.5031n_1n_4 - 203.2720n_3n_2 - 0.0020n_3n_5 - 8.2242n_3n_6 - 21.7123n_3n_7 - 0.0398n_3n_4 + 161.2780n_2n_5 - 27484.4000\ n_2n_6 + 63303.30n_2n_7 - 188.24n_2n_4 + 43.36n_5n_6 - 1197.72n_5n_7 + 0.6n_5n_4 + 213.516n_1n_2n_3 + 27566.80n_1n_2n_6 + 7122.40n_1n_6n_7 + 54.27n_2n_3n_6 + 34.76n_3n_6n_7 - 270.12n_2n_5n_6 \quad (12)$$

Table 2 presents the fitness function values for the 32 numerical problems after determining the optimal values of the algorithm parameters. The algorithm is run by 4000 iterations for each problem. Figures 1 and 2 illustrate the increasing trends of the fitness function against the iteration for problems 14 and 28. The improvement percentage in average fitness function value is about 6.33 from the first generation to the last one. Table 3 shows the fitness function values for the 32 numerical problems after determining the optimal values of the SA algorithm parameters.

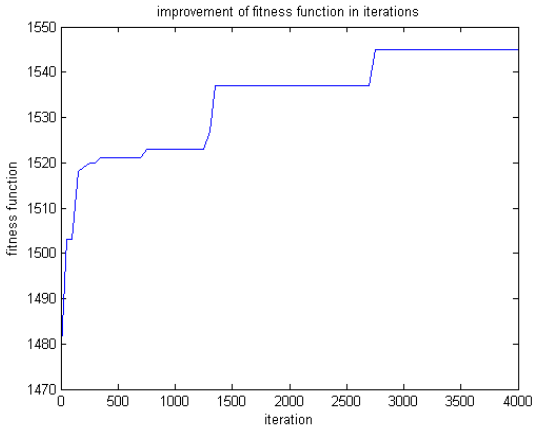

Fig. 1. Fitness function variations for problem 14

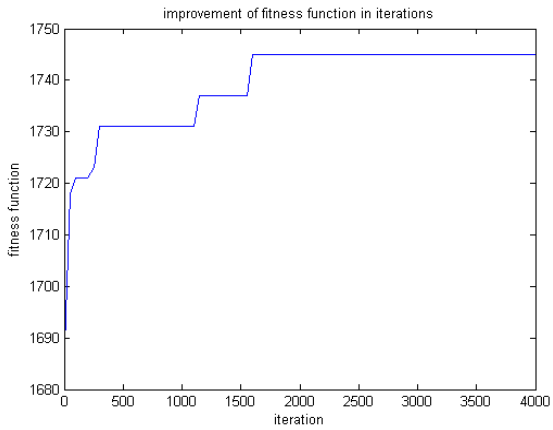

Fig. 2. Fitness function variations for problem 28

Table 4 gives the optimal values of the objective function obtained from running the 3 algorithms for 16 problems out of the 32 previous problems).

For the case of n =60, demand rates of the nodes (φ_i's) and distances between the nodes (d_{ij}'s) are randomly generated from [70, 100] and [1,25] respectively, service rate (μ) is 900 or 1200 and the number of service centers (M) to be located is assumed to be 30 or 40.

The GA parameters, Max_iteration, Population_size, p_c and p_m are assumed to be 4000, 80, 0.8 and 0.01 respectively.

The SA parameters including Max_iteration, In_loop_iteration, $t\alpha$, T_{in} and T_f are supposed to be 4000, 40, 0.9 or 0.95, 0.0000001 and 80 or 100.

Table 5 indicates the optimal values of the objective function obtained from running the 3 algorithms for 16 problems.

The softwares used in the study included MATLAB release 2008a package for running the algorithms, Minitab 14 for designing the experiments and for regression analysis, and Lingo 8.0 for optimizing. A computer with 2.00 GHz CPU and 3.00 GB RAM was used.

Table 2
Fitness function values after tuning the GA parameters

NO	α	β	μ	M	Population_size	p_c	p_m	GA Fitness function Value
1	0.7	0.15	500	15	60	0.8	0.01	1431.34
2	0.9	0.15	400	15	60	0.8	0.01	1531.58
3	0.7	0.05	400	15	80	0.8	0.001	1387.97
4	0.7	0.05	500	10	80	0.8	0.001	1566.35
5	0.9	0.15	500	15	60	0.8	0.01	1533.93
6	0.7	0.05	500	15	80	0.8	0.001	1427.90
7	0.7	0.15	400	15	80	0.6	0.001	1391.48
8	0.9	0.15	500	15	60	0.8	0.01	1544.28
9	0.9	08.15	400	15	60	0.8	0.01	1520.58
10	0.9	0.05	500	10	80	0.6	0.001	1754.47
11	0.9	0.05	500	15	80	0.6	0.001	1538.43
12	0.7	0.05	500	10	80	0.8	0.001	1530.45
13	0.9	0.05	400	15	80	0.8	0.001	1522.37
14	0.9	0.05	500	15	80	0.6	0.001	1545.19
15	0.7	0.15	400	15	80	0.6	0.001	1647.39
16	0.9	0.15	500	10	60	0.8	0.01	1749.17
17	0.9	0.05	400	15	80	0.8	0.001	1521.19
18	0.7	0.05	400	10	80	0.8	0.001	1496.49
19	0.7	0.15	500	15	60	0.8	0.01	1430.71
20	0.7	0.05	400	10	80	0.8	0.001	1499.30
21	0.9	0.15	400	10	60	0.8	0.01	1728.66
22	0.7	0.05	400	10	80	0.8	0.001	1497.40
23	0.7	0.05	400	15	80	0.8	0.001	1387.18
24	0.9	0.05	500	10	80	0.6	0.001	1750.47
25	0.9	0.15	400	10	60	0.8	0.01	1725.55
26	0.9	0.15	400	10	60	0.8	0.01	1725.55
27	0.9	0.15	500	10	60	0.8	0.01	1753.91
28	0.9	0.15	500	10	60	0.8	0.01	1745.20
29	0.7	0.15	500	10	60	0.8	0.01	1573.89
30	0.7	0.05	400	15	80	0.8	0.001	1400.71
31	0.9	0.05	500	10	80	0.6	0.001	1741.52
32	0.7	0.15	500	10	60	0.8	0.01	1562.81

Table 3
Fitness function values after tuning the SA parameters

NO	α	β	μ	M	Population_size	tα	T_f	SA Fitness function Value
1	0.7	0.15	400	10	80	0.9	100	1358.66
2	0.7	0.05	500	10	60	0.95	100	1505.26
3	0.7	0.05	400	10	60	0.95	100	1379.06
4	0.7	0.05	500	10	60	0.95	100	1412.33
5	0.9	0.05	500	15	60	0.95	100	1413.21
6	0.9	0.15	400	10	80	0.90	100	1543.55
7	0.9	0.15	400	15	80	0.90	100	1364.50
8	0.9	0.15	500	10	80	0.90	100	1634.36
9	0.9	0.05	500	10	60	0.95	100	1680.24
10	0.9	0.15	500	10	80	0.90	100	1573.72
11	0.7	0.15	400	15	80	0.9	100	1268.44
12	0.9	0.15	500	15	80	0.9	100	1410.11
13	0.9	0.05	500	10	60	0.95	100	1619.67
14	0.7	0.05	400	15	60	0.95	100	1275.97
15	0.9	0.05	400	10	60	0.95	100	1606.56
16	0.7	0.15	500	15	80	0.9	100	1301.45
17	0.7	0.15	500	10	80	0.9	100	1485.28
18	0.7	0.15	400	10	80	0.90	100	1406.27
19	0.9	0.05	400	10	60	0.95	100	1644.78
20	0.7	0.05	500	15	60	0.95	100	1281.73
21	0.7	0.05	500	15	60	0.95	100	1299.81
22	0.9	0.05	400	15	60	0.95	100	1436.54
23	0.9	0.15	500	15	80	0.9	100	1455.39
24	0.9	0.05	400	15	60	0.95	100	1445.85
25	0.9	0.05	500	15	60	0.95	100	1445.24
26	0.7	0.05	400	15	60	0.95	100	1220.13
27	0.7	0.15	400	10	80	0.90	100	1432.20
28	0.9	0.15	400	10	80	0.90	100	1607.94
29	0.7	0.05	400	10	60	0.95	100	1407.69
30	0.7	0.15	500	15	80	0.9	100	1327.78
31	0.9	0.15	400	15	80	0.9	100	1426.90

32	0.7	0.15	400	15	80	0.9	100	1282.78

Table 4
optimal values of the objective function obtained from running the algorithms for the case of n=30

NO	α	β	μ	M	GA fitness function	SA fitness function	Heuristic objective value
1	0.7	0.15	500	15	1431.34	1327.78	1445.67
2	0.9	0.15	400	15	1531.58	1327.78	1540.08
3	0.7	0.05	400	15	1400.71	1275.97	1405.50
4	0.7	0.05	500	10	1566.35	1505.26	1556.50
5	0.9	0.15	500	15	1544.28	1455.39	1553.8۳
6	0.7	0.05	500	15	1427.90	1299.81	1445.67
7	0.7	0.15	400	15	1391.48	1299.81	1405.50
8	0.9	0.05	500	10	1754.47	1680.24	1763.48
9	0.9	0.05	500	15	1545.19	1445.24	1553.83
10	0.9	0.05	400	15	1522.37	1445.85	1540.08
11	0.9	0.15	500	10	1753.91	1634.36	1763.48
12	0.7	0.05	400	10	1499.30	1407.69	1489.47
13	0.9	0.15	400	10	1728.66	1607.94	1739.26
14	0.7	0.15	500	10	1573.89	1485.28	1556.50
15	0.7	0.15	400	10	1494.87	1485.28	1489.47
16	0.9	0.05	400	10	1730.75	1644.78	1739.26

Table 5
optimal values of the objective function obtained from running the algorithms for the case of n=60

NO	α	β	μ	M	GA fitness function	SA fitness function	Heuristic objective value
1	0.7	0.15	1200	40	3422.92	3032.54	3570.81
2	0.9	0.15	900	40	3541.56	3206.93	3680.64
3	0.7	0.05	900	40	3537.23	3014.53	3505.54
4	0.7	0.05	1200	30	3738.76	3599.39	3893.41
5	0.9	0.15	1200	40	3528.75	3200.42	3702.61
6	0.7	0.05	1200	40	3445.88	3113.55	3570.81
7	0.7	0.15	900	40	3371.97	3210.42	3505.54
8	0.9	0.05	1200	30	3906.34	3588.49	4077.90
9	0.9	0.05	1200	40	3529.15	3168.59	3702.61
10	0.9	0.05	900	40	3545.18	3143.97	3680.64
11	0.9	0.15	1200	30	3916.63	3662.15	4077.90
12	0.7	0.05	900	30	3662.44	3189.05	3800.81
13	0.9	0.15	900	30	3892.48	3358.82	4047.73
14	0.7	0.15	1200	30	3751.61	3472.62	3893.41
15	0.7	0.15	900	30	3618.94	3203.27	3800.81
16	0.9	0.05	900	30	3891.89	3547.29	4047.73

5. Conclusions and further research

In this research, a network of several nodes is considered. Each node could be considered as a customer with a certain demand rate. The demand follows a time homogeneous Poisson process. Some servers are to be located at nodes of the network. The service distribution is also Poisson and a maximum probability is considered for each server occupancy. Each customer selects the closest server and a certain amount of benefit is achieved after giving service to the customer. Customers may waive receiving service with a set probability. The objective is to maximize the total benefit. Three heuristic algorithms with two of them being meta heuristic are developed and applied on several numerical problems in the case of a network with 30 and 60 nodes . The results show that the heuristic algorithm gives closer to optimum solutions rather than the two meta heuristic algorithms. In the case of n=30 nodes and considering the results of Table 4, the heuristic algorithm is 0.4 % and 7 % stronger than GA and SA respectively while in the case of n=60 nodes and considering the results of Table 5 , the heuristic algorithm is 11 % and 43 % stronger than GA and SA respectively. These findings indicate that the heuristic algorithm becomes stronger as the number of network nodes increases.

References

[1] R. Aboolian, O. Berman, and Z. Drezner, The multiple server center location problem. Annals of Operations Research; doi:10.1007/s10479-008-0341-2, 2008.

[2] R. Batta, A queueing-location model with expected service time dependent Queueing Disciplines. European Journal of Operational Research, 39, 192 – 205, 1989.

[3] O. Berman, Z. Drezner, The multiple server location problem. Journal of the Operational Research Society, 58, 91–99, 2007.

[4] O. Berman, D. Krass, Facility location problems with stochastic demands and congestion, in Z. Drezner, H.W. Hamacher, (Eds.), Facility Location, Applications and Theory, Springer-Verlag, 2001.

[5] O. Berman, D. Krass, Recent developments in the theory and applications of location models, Part II. Annals of Operations Research, 111, 15–16, 2002.

[6] O. Berman, D. Krass, and J. Wang, Locating service facilities to reduce lost demand. IIE Transactions, 38, 933–946, 2006.

[7] O. Berman, R. Larson, and S. Chiu, Optimal server location on a network Operating as a M/G/1 Queue. Operations Research, 12(4), 746 – 771, 1985.

[8] O. Berman, R. Larson, and C. Parkan, The Stochastic queue *p* – Median location problem. Transportation Science, 21, 207 – 216, 1987.

[9] B. T. Boffey, R. D. Galvao, and L. Espejo, A review of congestion models in the location of facilities with immobile servers. European Journal of Operational Research, 178, 643-662, 2007.

[10] B. T. Boffey, R. D. Galvao, and V. Marianov, Location of single-server immobile facilities with service time variance and loss minimization. Computers & Operations Research, submitted for publication.

[11] R. Church, C. ReVelle, The Maximal Covering Location Problem. Papers of the Regional science Association, 32, 101 – 118, 1974.

[12] M. S. Daskin, A maximum expected covering location model: Formulation, properties and heuristic solution. Transportation Science, 17, 48-70, 1983.

[13] M. Desrochers, P. Marcotte, and M. Stan, The congested facility location problem. Location Science, 3, 9–23, 1995.

[14] D. Gross, C. M. Harris, Fundamentals of Queueing Theory, Wiley, New York, 1985.

[15] S. L. Hakimi, Optimal locations of switching centers and the absolute centers and medians of a Graph. Operations Research, 12, 450 – 459, 1964.

[16] V. Marianov, Location of multiple-server congestible facilities for maximising expected demand when services are non-essential. Annals of Operations Research, 123, 125–141, 2003.

[17] V. Marianov, M. Rı´os, and F.J. Barros, Allocating servers to facilities, when demand is elastic to travel and waiting times. Operational Research, 39, 143–162, 2005.

[18] V. Marianov, D. Serra, Probabilistic maximal covering location-allocation for congested system, Journal of Regional Science, 38, 401–424, 1998.

[19] V. Marianov, D. Serra, Location models for airline hubs behaving as M/D/c queues. Computers & Operations Research , 30, 983-1003, 2001.

[20] D. McFadden, Conditional logit analysis of qualitative choice behaviour, in P. Zarembka, (Eds.), Frontiers in Econometrics, Academic Press, New York, 1974.

[21] S. Rajagopalan, H. L. Yu, Capacity planning with congestion effects. European Journal of Operational Research, 134, 365-377, 2001.

[22] C. ReVelle, K. Hogan, A reliability-constrained siting model with local estimates of busy fractions. Environment and Planning B: Planning and Design, 15, 143–152, 1988.

[23] C. ReVelle, K. Hogan, a. The maximum availability location problem. Transportation Science, 23, 192-200, 1989.

[24] C. ReVelle, K. Hogan, b. The maximum reliability location problem and α-Reliable *p*-Center problem: Derivatives of the probabilistic location Set Covering problem. Annals of Operations Research, 18, 155–174, 1989.

[25] C. ReVelle, S. Swain, Central Facilities Location. Geographical Analysis, 2, 30 – 42, 1970.

[26] C. Toregas, R. Swain, C. ReVelle, and L. Bergman, The location of emergency service facilities. Operations Research, 19, 1363–1373, 1971.

[27] Q. Wang, R. Batta, and C. M. Rump, Algorithms for a facility location problem with stochastic customer demand and immobile servers. Annals of Operations Research, 111, 17–34, 2002.