J Math Model Algor (2014) 13:301–313
DOI 10.1007/s10852-013-9233-9

An Iterated Local Search for the Budget Constrained Generalized Maximal Covering Location Problem

Majid Salari

Received: 11 February 2013 / Accepted: 30 July 2013 / Published online: 13 September 2013

Abstract Capacitated covering models aim at covering the maximum amount of customers' demand using a set of capacitated facilities. Based on the assumptions made in such models, there is a unique scenario to open a facility in which each facility has a pre-specified capacity and an operating budget. In this paper, we propose a generalization of the maximal covering location problem, in which facilities have different scenarios for being constructed. Essentially, based on the budget invested to construct a given facility, it can provide different service levels to the surrounded customers. Having a limited budget to open the facilities, the goal is locating a subset of facilities with the optimal opening scenario, in order to maximize the total covered demand and subject to the service level constraint. Integer linear programming formulations are proposed and tested using ILOG CPLEX. An iterated local search algorithm is also developed to solve the introduced problem.

Keywords Location · Covering · Emergency response · Iterated local search

1 Introduction

Introduced by Revelle et al. [27], the location set covering problem (LSCP) is a basic problem in the field of location theory. Given two sets of customer and facility vertices, the goal of the LSCP is to cover all the customers' demand using the minimum number of uncapacitated facilities [27]. Essentially, each facility i has a predetermined covering radius r_i within that all of the located customers will be covered.

The maximal covering location problem (MCLP) is one of the most celebrated problems and is close to the LSCP. Here, we seek the optimal selection of a given number of facilities, say p, from the set of potential facilities, in order to maximize

M. Salari (✉)
Department of Industrial Engineering, Ferdowsi University of Mashhad, Mashhad, Iran
e-mail: msalari@um.ac.ir

the total covered demand. Formulations and algorithms for both planar and network versions of this problem have been proposed by many researchers (see [10, 21] and [9, 14, 31] for network and planar versions, respectively). Berman et al. [5] developed a generalization of the MCLP in which the weights corresponding to some customer vertices could be negative in sign. They referred to several applications and developed some formulations and heuristics to solve their developed problem [5].

The LSCP and MCLP have several practical applications in different areas including: telecommunications, navigation and disaster management. For a complete survey on different areas of applications, we refer the readers to the papers by Current et al. [12], Kolen and Tamir [18] and Plastria [25].

The basic underlying assumption of the LSCP and MCLP is that the facilities are uncapacitated. Although this assumption has many practical applications in different areas, the capacitated version of these problems is being widely used. In this variation of the problem, we are given a value C_j as the maximum capacity of each facility j and we are not allowed to exceed this threshold, when assigning different customers' demand to the facility j. Formulations and solution approaches for the capacitated covering models have been proposed in the paper by Current and Storbeck [13].

Several variations of these two problems, i.e., LSCP and MCLP, have been introduced by researchers. In the remainder of this section, we will focus on different variants and some generalizations of these two classical models, followed by the description of our developed problem.

Berman and Krass [2] introduced the idea of gradual covering in which the assumption of "all or nothing coverage" is not valid. Essentially, the proportion of the covered demand is a non-increasing function of the covering radius [2]. There are also some other variants of the gradual covering in which there are two input parameters l_1 and l_2 ($l_1 < l_2$). Those customers located within distance l_1 of at least one constructed facility will be fully covered, while those beyond l_2 will not be covered at all. Finally, the customers with the distance between l_1 and l_2 from an already located facility, will be covered partially [3, 15].

Berman et al. [4] introduced the variable radius covering problem. In this problem, depending on the budget invested to open a facility, it will be able to cover different covering area. Having an unlimited amount of resources, the goal of this problem is to minimize the cost of opening different facilities to cover all the customers' demand through finding the optimal number, location and coverage radii of the facilities [4].

The idea of cooperative covering refers to some applications where each facility emits a "signal" that dissipates over the distance and each customer will receive the aggregate signal sent from all facilities. A customer is covered, whenever, its cumulative signals are greater than a given threshold. Exact and heuristic methods have been developed for the planar and discrete versions of this problem [6, 8].

In the capacitated p-median problem, the goal is to partition a given number of vertices, say n, into m different clusters with a side constraint. In particular, each cluster is represented by a certain vertex, called the facility, and the demand allocated to each facility j, cannot exceed a given threshold C_j. The goal is to minimize the total distance traveled by the non-facility vertices to reach their allocated facility [20, 23, 24, 28].

The recent survey by Berman et al. [7] reviews the most recent ideas for different classes of the covering problems and for more details we refer the interested readers to this paper [7].

The aim of the most presented problems is providing full coverage or maximizing the total covered demand of the potential customers with a set of side constraints. On the other hand, there are several problems seeking to minimize the total covered area by locating some undesirable facilities. Locating of garbage dumps and nuclear firms are examples of such problems. Many papers have addressed different versions of this problem. In particular, the case of locating a single facility [1, 11, 29] or multiple facilities [19, 22] have been introduced by different researches for which many solution techniques, based on exact and heuristic ideas, have been developed. For a complete survey on this problem and its variants, we refer the readers to the paper by Erkut and Neuman [16].

The solution maximizing the total covered demand is the one in which a facility is located at the same location of all available customers. In fact, availability of some restrictions, e.g. resource or time constrained, makes this solution impossible to be implemented in practice. In the most articles dealing with the MCLP and its variations, we are given a set of facilities from which we have to select a subset of facilities to cover the desirable amount of demand. Essentially, in some variations, e.g. LSCP, the goal is minimizing the total located facilities while in some others, that is opening a pre-specified number of facilities. To the best of our knowledge and as stated in Berman et al. [4], the idea of having different scenarios to locate the facilities with the presence of budget constraint has not yet been investigated in the literature. For example, let us consider a situation in which we have to provide the demand of different demand zones in an affected area by an earthquake. In such a situation, there may be a demand zone with the small amount of demand where just a small size facility with a minimum number of staff and resources could be sufficient to cover the demand available in this zone. On the other hand, there may be a facility that needs a larger amount of resources to cover the allocated demands. As a result, we propose a model in which a subset of facilities could be constructed, each by choosing one scenario from the set of available scenarios. Having a limited amount of resources to construct different facilities, the goal of the problem is to maximize the total covered demand.

The rest of the paper is organized as follows: in Section 2, we present a formal description of the problem and some of its generalizations. Section 3 introduces the proposed heuristic method to solve the problem, while the computational results are provided in Section 4. Concluding remarks and future research directions are described in Section 5.

2 Problem Statement

Suppose we are given a set $V = \{I \cup J\}$ of vertices, where, $I = \{1, 2, \cdots, m\}$ and $J = \{1, 2, \cdots, n\}$ are the set of customers and facilities, respectively. Each customer $i \in I$ has a pre-specified demand d_i that must be provided by at most one facility $j \in J$. In other words, we are not allowed to split the demand of each customer among different facilities. Moreover, we assume that each facility $j \in J$ can be constructed using different scenarios and depending on the selected scenario, the facility could have its own capacity and covering radius. Let us define $S_j = \{1, 2, \cdots, t_j\}$ as the set of different scenarios to open the facility $j \in J$. We represent by B_j^s and B, the cost corresponding to opening facility $j \in J$ using scenario $s \in S_j$, and the total

available budget to open different facilities, respectively. Moreover, let C_j^s represent the capacity of facility $j \in J$, established using scenario $s \in S_j$. The goal is to open a subset of potential facilities and to allocate a subset of the customers to the constructed facilities, in order to maximize the covered demand.

To model the proposed problem, we introduce some variables as follows:

Let z_{ij} to be the demand percentage of customer i allocated to facility j, and assume x_j^s and y_{ij} be two binary variables, defined as follows:

$$x_j^s = \begin{cases} 1 & \text{if facility } j \in J \text{ is located using scenario } s \in S_j, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$$y_{ij} = \begin{cases} 1 & \text{if demand of customer } i \in I \text{ is allocated to facility } j \in J, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Finally, for each $i \in I$, $j \in J$ and $s \in S_j$, a_{ij}^s is an input parameter, taking the value 1 if customer i is within the pre-specified covering radius of facility j which is constructed using scenario $s \in S_j$ and 0, otherwise.

In the following, we introduce a model which we call the single demand budget constraint generalized MCLP.

– **Single demand budget constraint generalized MCLP**

$$\max \sum_{i \in I} \sum_{j \in J} d_i z_{ij} \tag{3}$$

subject to:

$$\sum_{s \in S_j} x_j^s \leq 1 \qquad \forall j \in J \tag{4}$$

$$\sum_{j \in J} \sum_{s \in S_j} B_j^s x_j^s \leq B \tag{5}$$

$$y_{ij} \leq \sum_{s \in S_j} a_{ij}^s x_j^s \qquad \forall i \in I,\ j \in J \tag{6}$$

$$z_{ij} \leq y_{ij} \qquad \forall i \in I,\ j \in J \tag{7}$$

$$\sum_{i \in I} d_i z_{ij} \leq \sum_{s \in S_j} C_j^s x_j^s \qquad \forall j \in J \tag{8}$$

$$\sum_{j \in J} y_{ij} \leq 1 \qquad \forall i \in I \tag{9}$$

$$x_j^s \in \{0, 1\} \quad \forall j \in J, s \in S_j, \tag{10}$$

$$y_{ij} \in \{0, 1\}, 0 \leq z_{ij} \leq 1 \quad \forall i \in I,\ j \in J. \tag{11}$$

The objective function 3 is to maximize the total covered demand. For each $j \in J$, the first set of constraints, imposes that each facility to be located using at most one scenario $s \in S_j$. Constraint 5 limits the maximum available budget (B) to construct

different facilities. For each customer $i \in I$ and facility $j \in J$, constraint set 6 assures that we are not allowed to assign customer i to facility j, whenever, it has not yet been located using one of the available scenarios. For each pair of customer and facility vertices, constraint 7 models the relation between the z_{ij} and y_{ij} variables. Particularly, the demand of customer i could be allocated to facility j, whenever, y_{ij} is equal to 1, otherwise the allocation is not allowed. For each located facility j, using a given scenario $s \in S_j$, constraint 8 imposes that the total demand assigned to facility j cannot exceed its corresponding capacity, C_j^s. Finally, for each customer $i \in I$, constraint set 9 indicates that we are not allowed to split the demand of customer i among different facilities because splitting the demand of each customer over different facilities is not functional in reality. The decision variables are defined in Eqs. 10 and 11.

There may be situations in which each customer could have more than one demand type. Essentially, we assume that each customer $i \in I$ to have some demands from the set of different demand types $K_i = \{1, 2, \cdots, l_i\}$. In the following, we introduce a new model which is a generalization of the proposed model for the case of single demand budget constraint generalized MCLP. Before introducing the model, we give an updated version of some parameters and variables introduced in the latter model.

In a situation like emergency management, some items such as drugs could have a higher priority than some others, like blankets or tents. Based on this fact, we have assigned a weight to the different demand types available in our problem. In particular, for each customer $i \in I$, the demand type $k \in K_i$ is associated with a profit p^k which we call it as the priority of demand type k.

We define d_i^k, as the amount of the demand type $k \in K_i$ of customer i and represent by z_{ij}^k, the demand percentage type $k \in K_i$ of customer i allocated to facility j. Moreover, y_{ij}^k which is a generalization of the y_{ij}, is defined as follows:

$$y_{ij}^k = \begin{cases} 1 & \text{if demand type } k \in K_i \text{ of customer } i \text{ is allocated to facility } j, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

The humanitarian products could be different in size and weight. In our new model, the weight corresponding to the demand of type k is represented by w^k. The other notations and definitions are as those introduced for the case of single demand type. The generalization of the latter model reads as follows.

– **Multi demand budget constraint generalized MCLP**

$$\max \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} p^k d_i^k z_{ij}^k \tag{13}$$

subject to:

$$\sum_{s \in S_j} x_j^s \leq 1 \quad \forall j \in J \tag{14}$$

$$\sum_{j \in J} \sum_{s \in S_j} B_j^s x_j^s \leq B \tag{15}$$

$$y_{ij}^k \leq \sum_{s \in S_j} a_{ij}^s x_j^s \quad \forall i \in I,\ j \in J,\ k \in K_i \tag{16}$$

$$z_{ij}^k \leq y_{ij}^k \quad \forall i \in I,\ j \in J,\ k \in K_i \tag{17}$$

$$\sum_{i \in I} \sum_{k \in K} w^k d_i^k z_{ij}^k \leq \sum_{s \in S_j} C_j^s x_j^s \quad \forall j \in J \tag{18}$$

$$\sum_{j \in J} y_{ij}^k \leq 1 \quad \forall i \in I, k \in K_i \tag{19}$$

$$x_j^s \in \{0, 1\} \quad \forall j \in J, s \in S_j, \tag{20}$$

$$y_{ij}^k \in \{0, 1\}, 0 \leq z_{ij}^k \leq 1 \quad \forall i \in I,\ j \in J, k \in K_i. \tag{21}$$

The objective 13 is to maximize the total weighted covered demand. Constraint sets 14 and 15 are the same as that we have already explained for the single demand type (i.e. constraints 4 and 5). The remaining six constraint sets 16–21 are the generalization of constraints 6–10 in the single demand budget constraint generalized MCLP, respectively, in which different demand types and the weight corresponding to each unit of product k have been taken into account. Essentially, for each $j \in J$ constraint 18 shows that the weighted demand allocated to facility $j \in J$ cannot exceed its available capacity.

In the proposed models, it may lead to situations where the demand of a given customer is not served by the closest facility in the sense that it might be that two demands can be switched between two suppliers bringing both closer. To prevent this situation one could apply the following closest assignment constraint proposed by Wanger and Falkson [30] for a location-allocation model.

$$\sum_{r: d_{ir} > d_{ij}} y_{ir}^k + \sum_{s \in S_j} x_j^s \quad \forall i \in I,\ j \in J, k \in K_i. \tag{22}$$

in which d_{ij} is the distance between customer i and facility j. We do not apply this set of constraints in our final experiments because we have introduced the covering radius as an acceptable distance between any pairs of facility and customer vertices. Moreover, tests on a set of instances show that adding this constraint set to the model, will result in a worse objective function for some of the tested instances.

3 Iterated Local Search Algorithm

Running the instances to optimality by CPLEX was not successful in some cases (see Section 4). So, in order to have an efficient method to face large size instances in a reasonable CPU time, we propose a heuristic procedure for the developed problems in Section 2. The introduced heuristic is an iterated local search (ILS) algorithm for which the general framework is depicted in Algorithm 1. Essentially, the ILS algorithm consists of three major parts, namely *initialization*, *local search* and *perturbation*. The algorithm starts by constructing an initial feasible solution and tries to improve it by applying different moves developed in the *local search* phase. Moreover, to try to escape from the local optimum, a *perturbation* procedure

Algorithm 1 The general framework of the iterated local search algorithm

```
Current_Solution := Initialization();
Best_Solution := Current_Solution;
Iter = 0;
while (Iter <= Max_iter) do
    Current_Solution := local search(Current_Solution);
    if (Cost(Current_Solution) < Cost(Best_Solution)) then
        Best_Solution := Current_Solution;
        Iter = 0;
    end
    else
        Iter = Iter + 1;
    end
    Current_Solution := Perturbation(Current_Solution)
end
```

is designed. The algorithm iterates until it is not able to improve the solution's quality for a given number of successive iterations, Max_{iter}. Algorithm 1 gives the pseudocode of the proposed heuristic, while the details of each procedure are provided in the following sections. It is worth mentioning that general metaheuristics such as Tabu Search, genetic algorithms, variable neighborhood search, etc., could also obtain very good results.

3.1 Initialization

To construct an initial feasible solution, facilities and their different opening scenarios are taken into account and sorted in a non-increasing order of their capacities. Starting from the fist facility in the ordered list, they are added to the solution one by one until we have not exceeded the total available budget (B). Each time a facility is added to the solution, and as long as there is enough capacity at the facility location $j \in J$, the covered customers by j are selected as follows. An eligible customer $i \in I$ (i.e., a customer i with $a^s_{ij} = 1$) and one of its unsatisfied demand types $k \in K_i$ are selected, randomly, and allocated to the facility j. In this manner, customers and their corresponding different demand types have the same probability to be selected for being allocated to an opened facility.

3.2 Local Search

The local search consists of two main components, namely *swap* and *extraction-reinsertion* procedures. The algorithm iteratively applies these two procedures until no improvement could be achieved for the LK successive iterations. Moreover, the idea of *threshold accepting* (TA) is applied to try to diversify the search process. In particular, by using the TA idea, the non-improving moves are accepted if their corresponding objective values are within α percentage gap from the best known solution. In the following, the details of these procedures are given.

– *Swap*: the moves developed through the *swap* procedure aim at improving the objective function by substituting an existing facility in the solution with a new one out of the current set of visited facilities. To do so, the algorithm extracts a

facility from the solution, and replaces it with the best feasible candidate facility. Particularly, among all of the unvisited facilities, with their different opening scenarios, the best feasible move with respect to the objective value is selected. There may be a situation for which the new substitution results in a worse solution. In this case the new solution is accepted as the current solution if its objective value is within α percentage gap from the best known solution.

– *Extraction-reinsertion*: In this procedure, a given number of facilities, *ExtPar*, are randomly extracted from the solution. Upon this step and in a heuristic manner, the algorithm selects the best substitution from those unvisited facilities having the maximum impact on the objective function. Particularly, as long as there is enough budget and once at a time, this procedure continues to add the new facilities to the solution from those having the most improvement in the solution's cost. Non-improving moves will be accepted, if their cost is within the α percentage gap of the best solution's cost.

The general framework of the local search procedure is given in Algorithm 2.

Algorithm 2 The general framework of the *local search*

```
BestSolution := The solution obtained by applying the initialization phase;
CurrentSolution := BestSolution;
Allow = 1;
while (Allow) do
    Allow = 0;
    Randomly order the set of opened facilities;
    j = 0;
    while (j < number of the opened facilities) do
        j = j + 1;
        NewSolution := Apply the swap procedure using the j^th facility of the
        CurrentSolution;
        if (Cost(NewSolution) < Cost(BestSolution)) then
            BestSolution := NewSolution;
            CurrentSolution := NewSolution;
            Allow = 1;
        end
        else if (Cost(NewSolution) <= (1+α)*Cost(BestSolution)) then
            CurrentSolution := NewSolution;
        end
    end
    j = 0;
    while (j < number of the opened facilities) do
        j = j + 1;
        NewSolution := Apply the extraction-reinsertion procedure by extracting
        the j^th facility of the CurrentSolution;
        if (Cost(NewSolution) <= Cost(BestSolution)) then
            BestSolution := NewSolution;
            CurrentSolution := NewSolution;
            Allow = 1;
        end
        else if (Cost(NewSolution) <= (1+α)*Cost(BestSolution)) then
            CurrentSolution := NewSolution;
        end
    end
end
```

3.3 Perturbation

To escape from the local optimal solution, a *perturbation* phase has been designed. Taking into account the best solution, obtained during the search process, a given number of facilities, *PertPar*, are randomly extracted from the solution. Following this step and as long as there is enough budget to add some new facilities, the algorithm selects unvisited facilities and adds them to the solution. It is worth mentioning that all the unvisited facilities with their different opening scenarios have the same probability to be selected for addition into the set of visited facilities.

4 Computational Results

The multi-demand budget constraint generalized MCLP is a generalization of the single-demand type. As a result, we have performed our computational tests on the multi-demand version of the problem. To do so, we have designed several test instances. Data are characterized by four major factors, namely the number of customers (m) and facilities (n), the number of different demand types for each $i \in I$ (l_i), and the number of scenarios to open each facility $j \in J$ (s_j). To generate the instances, a subset of Euclidean data are taken from the TSP library [26]. Particularly, we have used three instances, namely kroA100, gil200 and d262, and using each of these instances, some new data which are tailored according to our studied problem are obtained. The number of the customers in each instance is a factor of the total number of vertices. In particular, three new instances are generated by using each of the initial datum in which the first 30 %, 50 % and 70 % of the vertices of each datum are supposed to be the customers and the rest of the vertices are considered as facility vertices. Moreover, four different random budget values are used as the maximum threshold on the total budget available to open different facilities. Finally, the number of scenarios to open each facility $j \in J$ (s_j), is a random integer value taken from [1, 10]. Moreover, the profit (p_k), and the weight (w_k) of each product type k is supposed to be a random integer value taken from [1, L] in which L is the total number of demand types. We have generated two groups of instances in which each customer could have 3 or 5 different demand types, respectively.

All experiments have been performed on a computer with an Intel Core 2 Duo Q8400 processor (2.6 GHz) and 3.5 GB of RAM. To run the experiments, we used CPLEX 12.1 with the default set of parameters [17], allowed to run for up to 600 s, and the heuristic method has been implemented in C++.

In order to find the reasonable values of the parameters, used for the ILS algorithm, we performed several runs on a set of test instances and the results are represented in Table 1, in which different tested values and the best selected values to

Table 1 Parameters setting for the ILS algorithm

Parameter	Tested values	Final value
Max_{iter}	{100, 200}	100: 3 demand types 200: 5 demand types
LK	{5, 15, 25}	25
ExtPar	{10, 30, 50}	50
PertPar	{30, 50, 70}	30
α	{1, 2, 3}	1

run the final experiments are given for all available parameters. Tables 2 and 3 report the results obtained by our proposed model for the multi demand budget constraint generalized MCLP with three and five different demand types, respectively. In these tables, the first column gives the index of the instances, while columns named by $|V|$, m and n, give the number of vertices, customers and facilities in each instance, respectively. The next three columns represent the CPLEX results, while for each datum, the objective function, the gap to optimality and the running time of CPLEX (in seconds) are given. Finally, the last three columns report the heuristic results,

Table 2 Computational results for the budget constraint generalized MCLP with three different demand types

Index	$\|V\|$	m	n	CPLEX			ILS		
				Objective	Gap	Time	Objective	Gap	Time
1	100	30	70	42.33	0.00	0.02	42.33	0.00	0.08
2	100	30	70	25.00	0.00	0.02	25.00	0.00	0.09
3	100	30	70	35.67	0.00	0.00	35.67	0.00	0.08
4	100	30	70	36.00	0.00	0.02	36.00	0.00	0.10
5	100	50	50	20.00	0.00	0.02	20.00	0.00	0.04
6	100	50	50	23.50	0.00	0.00	23.50	0.00	0.05
7	100	50	50	228.17	0.12	600.00	228.17	0.00	1.00
8	100	50	50	215.83	0.95	600.00	215.83	0.00	0.87
9	100	70	30	57.00	0.00	0.00	57.00	0.00	0.06
10	100	70	30	62.00	1.51	600.00	62.00	0.00	0.06
11	100	70	30	171.80	0.80	600.00	171.80	0.00	0.39
12	100	70	30	287.33	0.50	600.00	287.33	0.00	0.92
13	200	60	140	74.50	3.87	600.00	74.50	0.00	0.38
14	200	60	140	71.00	1.26	600.00	71.00	0.00	0.38
15	200	60	140	78.00	1.62	600.54	78.00	0.00	0.30
16	200	60	140	462.00	0.16	600.54	462.00	0.00	12.62
17	200	100	100	49.00	0.00	0.03	49.00	0.00	0.17
18	200	100	100	49.33	0.00	0.03	49.33	0.00	0.19
19	200	100	100	58.00	5.17	600.00	58.00	0.00	0.16
20	200	100	100	333.00	0.38	600.00	333.00	0.00	4.98
21	200	140	60	133.00	0.34	600.00	133.00	0.00	0.36
22	200	140	60	107.00	0.41	600.00	107.00	0.00	0.26
23	200	140	60	135.33	0.68	600.00	135.33	0.00	0.46
24	200	140	60	317.00	0.07	600.00	317.00	0.00	2.92
25	262	78	184	97.00	0.07	600.00	97.00	0.00	0.58
26	262	78	184	95.00	0.00	0.11	95.00	0.00	0.70
27	262	78	184	93.00	0.59	600.00	93.00	0.00	0.82
28	262	78	184	104.50	0.00	0.12	104.50	0.00	0.88
29	262	131	131	73.33	1.09	600.00	73.33	0.00	0.34
30	262	131	131	62.00	2.69	600.00	62.00	0.00	0.26
31	262	131	131	59.00	2.95	600.00	59.00	0.00	0.29
32	262	131	131	69.67	0.00	0.05	69.67	0.00	0.35
33	262	184	78	34.67	0.00	0.02	34.67	0.00	0.15
34	262	184	78	33.00	0.00	0.00	33.00	0.00	0.13
35	262	184	78	345.33	0.16	600.00	345.33	0.00	4.10
36	262	184	78	309.00	0.54	600.00	309.00	0.00	3.31
Average				123.51	0.72	366.68	123.51	0.00	1.08

Table 3 Computational results for the budget constraint generalized MCLP with five different demand types

Index	$\|V\|$	m	n	CPLEX			ILS		
				Objective	Gap	Time	Objective	Gap	Time
1	100	30	70	66.59	0.00	0.01	66.59	0.00	0.23
2	100	30	70	43.33	0.00	0.02	43.33	0.00	0.16
3	100	30	70	56.80	0.00	0.00	56.80	0.00	0.21
4	100	30	70	56.00	0.93	600.00	56.00	0.00	0.21
5	100	50	50	31.00	0.00	0.00	31.00	0.00	0.13
6	100	50	50	53.75	0.00	0.00	53.75	0.00	0.11
7	100	50	50	223.00	0.78	600.00	223.00	0.00	1.81
8	100	50	50	213.00	0.00	0.06	213.00	0.00	1.20
9	100	70	30	106.73	0.00	0.02	106.73	0.00	0.17
10	100	70	30	309.00	0.00	0.03	309.00	0.00	0.65
11	100	70	30	229.00	0.93	600.00	229.00	0.00	0.73
12	100	70	30	312.50	0.48	600.00	312.50	0.00	1.37
13	200	60	140	101.95	0.60	600.00	101.95	0.00	0.76
14	200	60	140	122.83	1.11	600.00	122.83	0.00	0.99
15	200	60	140	157.00	0.62	0.08	157.00	0.00	1.07
16	200	60	140	126.80	0.16	0.08	126.80	0.00	0.84
17	200	100	100	73.00	1.92	600.00	73.00	0.00	0.53
18	200	100	100	75.46	0.84	600.00	75.46	0.00	0.41
19	200	100	100	117.00	0.00	0.05	117.00	0.00	0.48
20	200	100	100	90.25	0.00	0.05	90.25	0.00	0.42
21	200	140	60	199.25	0.12	600.00	199.25	0.00	1.00
22	200	140	60	174.40	0.00	0.04	174.40	0.00	0.94
23	200	140	60	226.00	0.93	600.00	226.33	0.00	0.91
24	200	140	60	232.00	0.00	0.06	232.00	0.00	1.19
25	262	78	184	151.60	0.00	0.12	151.60	0.00	1.43
26	262	78	184	148.00	0.29	600.00	148.00	0.00	2.27
27	262	78	184	176.87	0.00	0.13	176.87	0.00	1.56
28	262	78	184	134.00	0.55	600.00	134.00	0.00	2.14
29	262	131	131	89.40	0.00	0.08	89.40	0.00	0.72
30	262	131	131	144.27	0.00	0.28	142.70	1.09	0.84
31	262	131	131	154.32	0.19	600.00	154.32	0.00	1.20
32	262	131	131	70.50	0.00	0.23	69.32	1.67	0.97
33	262	184	78	74.00	0.00	0.01	74.00	0.00	0.31
34	262	184	78	59.00	0.00	0.02	59.00	0.00	0.36
35	262	184	78	53.75	0.00	0.02	53.75	0.00	0.30
36	262	184	78	64.70	0.00	0.02	64.70	0.00	0.38
Average				132.62	0.25	205.75	130.95	0.08	0.81

including the objective value, the gap with respect to the best found solution by CPLEX, and the total running time of the ILS algorithm (in seconds).

Based on the results reported in Tables 2 and 3, CPLEX could reach the optimal solutions in 49 % of the instances within 600 s of CPU time, while in the remaining data the gap is below 5.17 %. Moreover, the proposed heuristic method could obtain the best found solutions in 100 % and 94 % of the instances for the case of three and five demand types, respectively.

From the computing time point of view, the heuristic method performs very rapidly. Particularly, the average running time of the ILS algorithm, over the instances with three and five demand types, is 1.08 and 0.81 s, respectively, while this is 366.68 and 205.75 s corresponding to the results obtained by CPLEX, respectively.

5 Conclusions and Future Research

Some generalizations of the maximal covering location problem introduced and integer linear programming models were developed. Assuming the total budget to locate different facilities is available in advance, the goal was to open a subset of facilities to maximize the total covered demand of customers. Moreover, the idea in opening facilities is that they could be located using different scenarios, leading to different covering radii and service levels with direct relation with the amount of the budget invested for this purpose. An iterated local search algorithm was proposed to solve the problem. Comparing the results of the developed algorithm with those obtained by CPLEX, clearly indicated the effectiveness of the method in obtaining good quality solutions within a very short running time. As future research, we propose developing a dynamic or multi-period version of this model in which the whole budget to locate different facilities is available for different time periods and depending on the period at which we are located the selected scenarios to open different facilities could be changed.

References

1. Berman, O., Drezner, Z.: A note on the location of an obnoxious facility on a network. Eur. J. Oper. Res. **120**, 215–217 (2000)
2. Berman, O., Krass, D.: The generalized maximal covering location problem. Comput. Oper. Res. **29**, 563–591 (2002)
3. Berman, O., Krass, D., Drezner, Z.: The gradual covering decay location problem on a network. Eur. J. Oper. Res. **151**, 474–480 (2003)
4. Berman, O., Drezner, Z., Krass, D., Wesolowsky, G.O.: The variable radius covering problem. Eur. J. Oper. Res. **196**, 516–525 (2009)
5. Berman, O., Drezner, Z., Wesolowsky, G.O.: The maximal covering problem with some negative weights. Geogr. Anal. **41**, 30–42 (2009)
6. Berman, O., Drezner, Z., Krass, D.: Cooperative cover location problems: the planar case. IIE Trans. **42**, 232–246 (2010)
7. Berman, O., Drezner, Z., Krass, D.: Generalized coverage: new developments in covering location models. Comput. Oper. Res. **37**, 1675–1687 (2010)
8. Berman, O., Drezner, Z., Krass D.: Discrete cooperative covering problems. J. Oper. Res. Soc. **62**(11), 2002–2012 (2011)
9. Canovas, L., Pelegrin, B.: Improving some heuristic algorithms for the rectangular p-cover problem. In: Moreno-Perez, J.A. (ed.) Proceedings of the VI Meeting of the EURO Working Group on Locational Analysis, Universidad De La Laguna, Tenerife, pp. 23–31. Spain (1992)
10. Church, R.L., ReVelle, C.: The maximal covering location problem. Pap. Reg. Sci. Assoc. **32**, 101–118 (1974)
11. Church, R.L., Garfinkel, R.S.: Locating of an obnoxious facility on a network. Transp. Sci. **12**, 107–118 (1978)
12. Current, J., Daskin, M., Schilling, D.: Discrete network location models. In: Drezner, Z., Hamacher, H. (eds.) Facility Location: Applications and Theory. Springer, Berlin (2002)
13. Current, J.R., Storbeck, J.E.: Capacitated covering models. Environ. Plann. B **15**(2), 153–163 (1988)
14. Drezner, Z.: On a modified one-center problem. Manag. Sci. **27**, 848–851 (1981)

15. Drezner, Z., Wesolowsky, G.O., Drezner, T.: The gradual covering problem. Nav. Res. Logist. **51**, 841–855 (2004)
16. Erkut, E., Neuman, S.: Analytical models for locating undesirable facilities. Eur. J. Oper. Res. **40**, 275–291 (1989)
17. IBM ILOG CPLEX: http://www.ilog.com. Accessed 11 Feb 2013
18. Kolen, A., Tamir, A.: Covering problems. In: Mirchandani, P.B., Francis, R.L. (eds.) Discrete location theory, pp. 263–304. Wiley-Interscience, New York, NY (1990)
19. Kuby, M.J.: Programming models for facility dispersion: the p-dispersion and maxisum dispersion problems. Geogr. Anal. **19**, 315–329 (1987)
20. Lorena, L.N., Senne, E.: A column generation approach to capacitated P-median problems. Comput. Oper. Res. **31**(6), 863–876 (2004)
21. Megiddo, N., Zemel, E., Hakimi, S.L.: The maximum coverage location problems. SIAM J. Algebra Discr. **4**, 253–261 (1983)
22. Moon, I.D., Chaudhry, S.: An analysis of network location problems with distance constraints. Manag. Sci. **30**, 290–307 (1984)
23. Osman, I., Christofides, N.: Capacitated clustering problems by hybrid simulated annealing and tabu search. Int. Trans. Oper. Res. **1**(3), 317–336 (1994)
24. Osman, I., Ahmadi, S.: Guided construction search metaheuristics for the capacitated p-median problem with single source constraint. J. Oper. Res. Soc. **18**, 339–348 (2006)
25. Plastria, F.: Continuous covering location problems. In: Drezner, Z., Hamacher, H. (eds.) Facility Location: Applications and Theory. Springer, Berlin (2002)
26. Reinelt, G.: A traveling salesman problem library. ORSA J. Comput. **3**, 376–384 (1991)
27. ReVelle, C., Toregas, C., Falkson, L.: Applications of the location set covering problem. Geogr. Anal. **8**, 67–76 (1976)
28. Santos-Correa, E., Steiner, M.T., Freitas, A.A., Carnieri, C.: A genetic algorithm for solving a capacitated p-median problem. Numer. Algor. **35**(4), 373–388 (2004)
29. Tamir, A.: Obnoxious facility location on graphs. SIAM J. Discret. Math. **4**, 550–567 (1991)
30. Wanger, J.L., Falkson, L.M.: The optimal nodal location of public facilities with price-sensitive demand. Geogr. Anal. **7**, 69–83 (1975)
31. Watson-Gandy, C.: Heuristic procedures for the m-partial cover problem on a plane. Eur. J. Oper. Res. **11**, 149–157 (1982)