Yugoslav Journal of Operations Research
28 (2018), Number 2, 201–218
DOI: https://doi.org/10.2298/YJOR170120011J

# AN EFFICIENT GENETIC ALGORITHM FOR THE UNCAPACITATED *R*-ALLOCATION *P*-HUB MAXIMAL COVERING PROBLEM

Olivera JANKOVIĆ
*Faculty of Mathematics, University of Belgrade, Belgrade,Serbia*
*Faculty of Economics, University of Kragujevac, Kragujevac, Serbia*
*jankovic.olivera@gmail.com*

Received: January 2017 / Accepted: February 2018

**Abstract:** This paper deals with the Uncapacitated *r*-allocation *p*-hub Maximal Covering Problem (UrApHMCP) with a binary coverage criterion. This problem consists of choosing *p* hub locations from a set of nodes so as to maximize the total demand covered under the *r*-allocation strategy. The general assumption is that the transportation between the non-hub nodes is possible only via hub nodes, while each non-hub node is assigned to at most *r* hubs. An integer linear programming formulation of the UrApHMCP is presented and tested within the framework of a commercial CPLEX solver. In order to solve the problem on large scale hub instances that cannot be handled by the CPLEX, a Genetic Algorithm (GA) is proposed. The results of computational experiments on standard *p*-hub benchmark instances with up to 200 nodes demonstrate efficiency and effectiveness of the proposed GA method.

**Keywords:** *p*-hub Maximal Covering Problem, Binary Coverage, Heuristic, Genetic Algorithm.

**MSC:** 90B80, 90B10, 68T20.

## 1. INTRODUCTION

The hub covering problem is one of the fundamental and most studied problem in facility location theory. It is widely used within the design of supply chains, airline transportation networks, DHL services, postal delivery networks, computer and communication systems, etc. The hub covering problem was first

introduced by Campbell [7]. Let $N = \{1, 2, ..., n\}$ be a set of nodes in the network, each of them with a certain demand assigned, and let $H \subset N$ be a set of potential hub nodes. The goal of the hub covering problem is to choose hubs from the set $H$ and to allocate demand nodes to them, in order to minimize the number of hubs while covering all origin-destination (O–D) pairs, or to maximize the total demand covered by a given number of hubs. The O–D pair $i - j$ ($i, j \in N$) is considered to be covered by hubs $h_i, h_j \in H$ if nodes $i$ and $j$ are assigned to hubs $h_i$ and $h_j$, meaning that the path $i \rightarrow h_i \rightarrow h_j \rightarrow j$ is established, and the preselected covering criterion is fulfilled.

From the early work by Campbell [7], where different types of hub covering formulations were introduced, hub covering problems developed in two main directions: the hub set covering, and the hub maximal covering problems. The goal of the hub set covering problem is to locate the hub nodes such that all demands are covered while the cost of opening hub facilities is minimized. On the other hand, the hub maximal covering problem maximizes the demand covered with a fixed number of hubs to be located.

Another criterion for the classification of hub covering problems is the allocation scheme, which defines the way of allocating the non-hub nodes to the hubs. Two main allocation schemes are distinguished: a single allocation (each non-hub node is allocated to exactly one established hub) and multiple allocation (each non-hub node may communicate to other non-hub nodes via more than one located hubs). The *r*–allocation scheme was introduced by Yaman in [30], allowing each non-hub node to be assigned to at most $r$ located hubs.

The single allocation hub set covering problem was studied by Kara in [18], who proposed an improved problem formulation compared to the one from [7]. Wagner [28] further improved models for the single and multiple allocation hub set covering problem, and proposed a new model involving quantity-dependent transport time functions for links in the single allocation case. Ernst in [11] tightened up the formulations from [28] by lifting some of the constraints, and proposed a new formulation for the single allocation version of the hub set covering problem. Alumur and Kara in [2] considered the problem of cargo transportation and formulated it as a single allocation hub set covering problem, which relaxes the complete hub network assumption and minimizes the cost of establishing both hubs and links between the hubs.

Regarding the allocation strategy, the following variants of the Uncapacitated $p$-hub Maximal Covering Problem can be distinguished: the Uncapacitated Single Allocation $p$-hub Maximal Covering Problem (USApHMCP) [17], the Uncapacitated $r$-allocation $p$-hub Maximal Covering Problem (UrApHMCP), and the Uncapacitated Multiple Allocation $p$-hub Maximal Covering Problem (UMApHMCP) [28]. In the USApHMCP each node is assigned to exactly one out of $p$ established hubs, while in the UMApHMCP, each node can send and receive traffic through any of the $p$ established hubs in the network.

According to the *binary coverage criterion*, an origin-destination pair is considered covered if the cost of the established path via hubs is within the maximum allowed limit (see [8]). The *partial coverage* criterion involves the degree

of coverage of the path between origin-destination pairs that is determined by a non-increasing decay function based on the path length (see [24]).

The first mathematical formulations of USApHMCP and UMApHMCP proposed in [7] assume the binary coverage criterion. Hwang [17] also investigated USApHMCP using the binary coverage criterion and proposed a new model for USApHMCP, as well as two heuristics to tackle the problem: a distance based allocation and a volume based allocation heuristics. Weng [29] developed a new mathematical formulation for the multiple assignment version of the problem. Recently, Peker and Kara [24] proposed new mathematical models for the single and multiple $p$-hub maximal covering problem that can be applied in both binary and partial coverage cases.

This paper deals with UrApHMCP with the binary coverage criterion. The goal is to maximize the total covered demand with respect to the binary coverage criterion under the following assumptions:

- the number of hubs is fixed to $p$,
- each non-hub node can be assigned to at most $r$ hubs ($1 \leqslant r \leqslant p$),
- a direct transportation between non-hub nodes is not allowed,
- the amount of flow collected at each hub is not limited.

The UrApHMCP with binary coverage is an $NP$-hard optimization problem as a generalization of both single allocation ($r = 1$) and multiple allocation ($r = p$) $p$-HMCP, which are proved to be $NP$-hard (see [3, 7]).

This paper is organized as follows: In Section 2, the mathematical formulation for the UrApHMCP with a binary coverage criterion is given. In addition, in the same section, it is proved that single and multiple allocation $p$-hub covering problems are special cases of $r$-allocation $p$-hub covering problem. The details of the proposed GA-based method are given in Section 3, while in Section 4, the experimental results obtained by the proposed GA are presented. In Section 5, conclusions and future directions are given.

## 2. MATHEMATICAL FORMULATION

In order to present Integer Linear Programming (ILP) model for the UrApHMCP with binary coverage, the following notation is used:

- $G = (N, E)$ is a complete undirected graph, where $N = \{1, 2, ..., n\}$ is a set of nodes and $E = \{\{i, j\} : i, j \in N\}$ is a set of edges with no capacity restrictions;
- $d_{ij}$ is the distance (cost, time, etc.) between a pair of nodes $i, j \in N$. It is assumed that $d_{ij} = d_{ji}, d_{ii} = 0$, and $d_{ij} \leq d_{ik} + d_{kj}$, for all $i, j, k \in N$.
- $T_{ij}$ is the flow of demand between O–D pair $i - j$;

- The cost of total path length from origin $i \in N$ to destination $j \in N$ via hubs $k \in H$ and $m \in H$ is denoted as $c_{ij}^{km}$ and calculated as

$$c_{ij}^{km} = \chi d_{ik} + \alpha d_{km} + \delta d_{mj}, \tag{1}$$

  where $\chi$, $\delta$, and $\alpha$ represent cost factors for flow collection from an origin to a hub, distribution from a hub to a destination, and for the interhub transportation, respectively. It is assumed that $\alpha < \chi$ and $\alpha < \delta$.

According to the *binary coverage criterion* [8] used in this paper, the established path from $i$ to $j$ via hubs $k$ and $m$ is considered as *covered* if the cost of the total path length is equal or less than a predefined limit $\beta_{ij}$. As in paper by Peker and Kara [24], for each path $i \to k \to m \to j$, a binary parameter $a_{ij}^{km}$ is introduced to indicate whether the origin-destination pair $i$ and $j$ is covered by using hubs $k$ and $m$:

$$a_{ij}^{km} = \begin{cases} 1 & \text{if } c_{ij}^{km} \leq \beta_{ij} \\ 0 & \text{otherwise} \end{cases} \quad \forall i, j \in N \quad \forall k, m \in H. \tag{2}$$

The following decision variables are introduced:

- A binary variable $x_{ik}$ takes the value of 1 if node $i \in N$ is allocated to hub $k \in H$, and 0 otherwise. In case that $x_{kk} = 1$, the node $k$ is chosen as a hub, and is naturally assigned to itself.
- A binary variable $y_{ijkm}$ is equal to 1 if there is traffic that travels from origin node $i \in N$ to hub $k \in H$, then from hub $k \in H$ to hub $m \in H$, and finally from $m \in H$ to the destination node $j \in N$.

Using the above notation, the UrApHMCP with binary coverage can be represented by the following ILP model:

$$\max \sum_{i \in N} \sum_{j \in N} \sum_{k \in H} \sum_{m \in H} a_{ij}^{km} T_{ij} y_{ijkm} \tag{3}$$

such that

$$x_{ik} \leq x_{kk} \quad \forall i \in N \quad \forall k \in H, \tag{4}$$

$$\sum_{k \in H} x_{ik} \leq r \quad \forall i \in N, \tag{5}$$

$$\sum_{k \in H} x_{kk} = p, \tag{6}$$

$$\sum_{k \in H} \sum_{m \in H} y_{ijkm} = 1 \quad \forall i, j \in N, \tag{7}$$

$$y_{ijkm} \leq x_{ik} \quad \forall i, j \in N \quad \forall k, m \in H, \tag{8}$$

$$y_{ijkm} \leq x_{jm} \quad \forall i, j \in N \quad \forall k, m \in H, \tag{9}$$

$$x_{ik} \in \{0, 1\} \quad \forall i \in N \quad \forall k \in H, \tag{10}$$

$$y_{ijkm} \in \{0, 1\} \quad \forall i, j \in N \quad \forall k, m \in H. \tag{11}$$

The objective function (3) maximizes the covered demand of all origin-destination pairs. Constraints (4) and constraints (5) ensure that each node is allocated to at most $r$ hubs. Constraint (6) indicates that exactly $p$ hubs are located. Constraints (7) ensure that the flow from the origin to the destination node is routed entirely via some pair of located hubs. Constraints (8) and (9) impose the requirement that if the flow is routed from the origin $i$ to destination $j$ through the established hubs $k$ and $m$, respectively, than node $i$ is assigned to hub $k$ and node $j$ to hub $m$. Constraints (10) and (11) indicate the binary type of variables $x_{ik}$ and $y_{ijkm}$.

The proposed model can be applied, for example, to optimizing fast delivery services, cargo applications, emergency services, food delivery, etc., where customers are generally very sensitive to delivery time and prefer the companies with fast and on-time delivery. Companies are interested in satisfying customers' needs, and therefore, they are trying to decrease delivery time as much as possible, and to guarantee that packages are delivered on time. However, these requirements cannot be satisfied completely for every situation. So, their primary objective is to design a network such that the amount of cargo delivered on-time is maximized. The structure used by these companies is precisely the same as the hub network structure. Transportation of cargo packages from origin to destination node is handled by the operation centers that can be identified as hubs. Each demand center is assigned to an operation center that handles collection and distribution operations for the assigned demand center. A typical route for cargo consists of three segments: from the origin node to the operation center serving it, then to the consignees' operation center, and finally, to the consignee destination node. There are some $i \to k \to m \to j$ paths in which some nodes coincide, which cases actually correspond to some special real-life situations. More precisely, the following formal movements are possible:

- Case 1: The path contains only a single hub node that is different from both the origin and the destination nodes, which are also different, i.e., $i \neq j$, $k = m$ and $i \to k \to k \to j$. In practice, the both origin and destination nodes are served by the same operation center. Assuming that travel time from a node to itself is zero, it is clear that $c_{ij}^{kk} = \chi d_{ik} + \delta d_{kj}$. If $c_{ij}^{kk} \leq \beta_{ij}$, then according to the binary coverage criterion, origin-destination pair $i - j$ is covered by using hub $k$, $a_{ij}^{kk} = 1$ and $a_{ij}^{kk} T_{ij} = T_{ij}$. Otherwise, $a_{ij}^{kk} = 0$ and $a_{ij}^{kk} T_{ij} = 0$.

- Case 2: This case is similar to Case 1 but the origin and destination nodes coincide, i.e., $i = j$ and $i \to k \to k \to i$. This scenario occurs when a customer sends a package to a recipient situated in the same service area, but it first has to be processed at the sorting hub before being delivered to the recipient. In

this case, $c_{ii}^{kk} = \chi d_{ik} + \delta d_{ki} = (\chi + \delta) d_{ik}$. If $c_{ii}^{kk} \leq \beta_{ij}$, then $a_{ii}^{kk} = 1$ and $a_{ii}^{kk} T_{ii} = T_{ii}$. Otherwise, $a_{ii}^{kk} = 0$ and $a_{ii}^{kk} T_{ii} = 0$.

- Case 3: The origin node is a hub node (operation center) and the destination node is a non-hub node, i.e., $i = k = m$ and $i \to i \to i \to j$. In this case, $c_{ij}^{ii} = \delta d_{ij}$. If $c_{ij}^{ii} \leq \beta_{ij}$, then $a_{ij}^{ii} = 1$ and $a_{ij}^{ii} T_{ij} = T_{ij}$. Otherwise, $a_{ij}^{ii} = 0$ and $a_{ij}^{ii} T_{ij} = 0$.
- Case 4: The origin node is a non-hub node and the destination node is a hub node (operation center), i.e., $k = m = j$ and $i \to j \to j \to j$. In this case, $c_{ij}^{jj} = \chi d_{ij}$. If $c_{ij}^{jj} \leq \beta_{ij}$, then $a_{ij}^{jj} = 1$ and $a_{ij}^{jj} T_{ij} = T_{ij}$. Otherwise, $a_{ij}^{jj} = 0$ and $a_{ij}^{jj} T_{ij} = 0$.
- Case 5: The origin node and the destination node are the same hub node, i.e., $i = k = m = j$ and $i \to i \to i \to i$. This means that the package is sent and delivered in the same zone (or zip code). As the travel time from a node to itself is zero, $c_{ii}^{ii} = 0$. In this case, $a_{ii}^{ii} = 1$ and $a_{ii}^{ii} T_{ii} = T_{ii}$.
- Case 6: The origin node is a hub node and the destination node is a hub node (the both origin and destination nodes are operation centers), i.e., $i = k = m$ and $i \to i \to i \to j$ or $k = m = j$ and $i \to j \to j \to j$ or $i = k, m = j$ and $i \to i \to j \to j$. These situations are as those in the practice, but $c_{ij}^{ii} = \delta d_{ij}, c_{ij}^{jj} = \chi d_{ij}$ and $c_{ij}^{ij} = \alpha d_{ij}$. The path from $i$ to $j$ is considered as covered if $c_{ij}^{ii} \leq \beta_{ij}$ or $c_{ij}^{jj} \leq \beta_{ij}$ or $c_{ij}^{ij} \leq \beta_{ij}$. As $\alpha$ is the economies of scale factor used to discount the distance between hubs (for example, the delivery between two operation centers is done by means of larger, more specialized, or faster vehicles), it is clear that $c_{ij}^{ij} \leq c_{ij}^{ii}$ and $c_{ij}^{ij} \leq c_{ij}^{jj}$. In practice, if $c_{ij}^{ij} \leq \beta_{ij}$, then according to the binary coverage criterion, origin-destination pair $i - j$ is covered, $a_{ij}^{ij} = 1$ and $a_{ij}^{ij} T_{ij} = T_{ij}$. Otherwise, $a_{ij}^{ij} = 0$ and $a_{ij}^{ij} T_{ij} = 0$.

From the previous discussion, it can be seen that the paths in cases 1-4 and 6 have no influence on the objective function value as long the corresponding coefficients in the expression for the objective function are equal to zero. In case 5, the objective function value is increased by the flow of demand $T_{ii}$ that corresponds to the origin and destination nodes (which are the same).

In the remaining part of this section, the relationship of the model (3) – (11) with four-index formulations for the single and multiple allocation $p$-hub maximal covering problems is discussed. It will be proved that USApHMCP and UMApHMCP can be transformed to model (3) – (11) for $r = 1$ and $r = p$, respectively.

In the paper [17], the USApHMCP model is formulated as (3) such that (4), (6), (10)–(11), and

$$\sum_{k \in H} x_{ik} = 1 \quad \forall i \in N, \tag{12}$$

$$2y_{ijkm} \leq x_{ik} + x_{jm} \quad \forall i, j \in N \quad \forall k, m \in H. \tag{13}$$

For $r = 1$, it is obvious that constraints (5) directly follow from (12). By constraints (12), the origin node $i$ is allocated to exactly one hub $k$ (i.e. $x_{ik} = 1$), and the destination node $j$ is assigned to exactly one hub $m$ (i.e. $x_{jm} = 1$). Since (13) is satisfied, and the objective function of USApHMCP model is maximized, then in an optimal solution of this model $y_{ijkm} = 1$, and $(k, m)$ is the only hub pair that cover O–D pair $i - j$ . Therefore, the constraints (7) are satisfied. If $y_{ijkm} = 0$, then inequalities $y_{ijkm} \leq x_{ik}$ and $y_{ijkm} \leq x_{jm}$ obviously hold. In the case of $y_{ijkm} = 1$, from (13) it can be concluded that $x_{ik} + x_{jm} \geq 2$ and, thus, $x_{ik} = x_{jm} = 1$. Therefore, constraints (8) and (9) follow from (13).

Recall that UMApHMCP model from [7] uses variables $h_k$, $k \in H$, which are defined as follows: $h_k = 1, \forall k \in H$ if and only if a hub is established at location $k$, and 0 otherwise. The UMApHMCP model is formulated as (3) under the constraints (7), (11), and

$$\sum_{k \in H} h_k = p, \tag{14}$$

$$y_{ijkm} \leq h_k \quad \forall i, j \in N \quad \forall k, m \in H, \tag{15}$$

$$y_{ijkm} \leq h_m \quad \forall i, j \in N \quad \forall k, m \in H, \tag{16}$$

$$h_k \in \{0, 1\} \quad \forall k \in H. \tag{17}$$

In the model defined by an objective (3) and constraints (7), (11), (14)–(17), let $h_k = x_{kk}$, $k \in H$. If variables $x_{ik}$, $i \in N$, $k \in H$ are introduced adding constraints (4), (8) and (9), then (14) is transformed directly to (6). Constraints (5) for $r = p$ follow from constraints (4) and (6). From (4), (8) and (9) it follows that constraints (15) and (16) are redundant and can be omitted. In this way, model defined by (3), (7), (11), (14)–(17) can be transformed to model defined by (3)–(11).

## 3. GENETIC ALGORITHM FOR SOLVING UrApHMCP

Genetic Algorithm (GA) is a stochastic search technique proposed by Holland in [16]. Over the last two decades, GA has been proved as an efficient search technique for solving various optimization problems (see [9, 10, 14, 21]). GA is inspired by Darwin's principle of natural evolution and starts with a set of randomly generated candidate solutions, denoted as population. Each individual in

the population corresponds to a candidate solution to the problem, and it is represented by a genetic code. Parts of a genetic code are denoted as genes. In each iteration (i.e. generation), the following processes are performed: i) Evaluation of individuals in the population is done by calculating the fitness function, which measures the quality of an individual; ii) Selecting promising individuals from the current population according to their fitness, which will take part in producing new generation; iii) Creating offspring individuals by combining genetic codes of selected parent individuals, which is done by crossover operator; iv) Slight modification of some randomly selected offspring individuals by mutation operator. The generated offspring individuals enter the next generation of individuals. These steps are repeated until certain stopping criterion is satisfied. Through GA generations, the (average) fitness of individuals is improved, and the best fitted individual corresponding to the best found GA solution is returned when the GA finishes its run.

The basic scheme of GA is represented in Algorithm 1. More details on GA and its variants can be found in [4, 5, 15].

**Algorithm 1:** Basic scheme of GA

```
Input Data();
Population Init();
while not Finish() do
    Fitness Function();
    Selection();
    Crossover();
    Mutation();
end
Output Data();
```

In the literature, there are examples of successful GA applications to various hub location problems, such as [1, 19, 20, 25, 26, 27], etc. These studies indicate that GA is a promising metaheuristic approach for solving hub location problems. This was a motivation to design a GA metaheuristic for solving the variant of hub covering problem considered in this study.

### *3.1. Representation of individuals and fitness function calculation*

The proposed GA implementation uses a binary representation of individuals. A genetic code of an individual consists of $n$ genes, each corresponding to one node of the network. Each gene takes the value of 1 if the corresponding node is chosen as a hub, and 0 otherwise.

The initial population consists of $N_{ind} = 150$ randomly generated individuals. Each individual is created by randomly selected exactly $p$ genes, taking value of 1, while the remaining genes are set to 0. After generating the initial population, the fitness value of each individual is calculated. In this GA implementation, the fitness value of the individual is equal to its objective function value, calculated as follows.

Let $H$ be the set of indices $\{h_1, h_2, ..., h_p\}$ of $p$ hub nodes, obtained from the genetic code of an individual. Each hub node is naturally assigned to itself, while for the remaining non-hub nodes, their adequate allocations to the chosen hubs have to be found. This is performed by the function $alloc(i,h)$, proposed in [23], which is used to construct a greedy solution. This function reflects the influence of allocating non-hub node $i$ to hub $h$, and it is calculated as follows:

$$alloc(i,h) = \chi d_{ih} + \sum_{j \in N} \delta d_{hj}. \tag{18}$$

The first term in $alloc(i,h)$ represents the cost from origin node $i \in N$ to hub $h \in H$, while the second term measures the cost associated with the arcs from hub $h \in H$ to destination nodes. For each non hub node $i$, the values $alloc(i,h), h = h_1, h_2, \ldots, h_p$ are computed. Each non hub node $i$ is assigned to exactly $r$ hubs with the best allocation values $alloc(i,h)$, building a sequence of $r$ hubs in a greedy way. Now, it is easy to find traffic routes for each O–D pair $i-j$, and to calculate the objective function value. The key point of the UrApHMCP is to maximize cases of $c_{ij}^{km}$ within the coverage radius $\beta_{ij}$. Let $K$ and $M$ be sets of $r$ hubs assigned to nodes $i$ and $j$, respectively. Then, for each pair $i-j$, a promising solution can be obtained when hubs $k \in K$ and $m \in M$, through which the traffic is routed from $i$ to $j$, are chosen such that the cost $c_{ij}^{km} = \chi d_{ik} + \alpha d_{km} + \delta d_{mj}$ is minimized. The procedure considers all possible paths of type $i \to k \to m \to j$ and binary variable $y_{ijkm}$ is equal to 1 if there is traffic that is routed from origin node $i$ to hub $k$, then from hub $k$ to hub $m$, and finally, from $m$ to the destination node $j$. Naturally, some of the nodes $i, j, k$, and $m$ may coincide. Also, even if sets $K$ and $M$ have a common hub, the traffic from $i$ to $j$ is not necessarily routed through that hub. The straightforward calculation of all O–D shortest path pairs requires $O(n^2r^2)$ operations. In the case of $r = p$, the calculation of all shortest paths may be performed in $O(n^2r)$ operations using a modification of the Floyd-Warshall shortest path algorithm. The fitness value is then simply evaluated by summing up the demands over covered paths. The individual is considered as *correct* if there are no unused hubs, i.e., the ones with no non-hub nodes assigned. If this is not the case, the individual is considered *incorrect*, and its fitness value is set to 0.

*Example 1.* The left hand side of Figure 1 shows a network with $n = 5$ nodes denoted as $1, 2, 3, 4, 5$ with the following coordinates in the plane: (0,0), (2,0), (4,1), (1,3), and (0,2). The corresponding distance matrix is

$$D = \begin{pmatrix} 0 & 2 & \sqrt{17} & \sqrt{10} & 2 \\ 2 & 0 & \sqrt{5} & \sqrt{10} & 2\sqrt{2} \\ \sqrt{17} & \sqrt{5} & 0 & \sqrt{13} & \sqrt{17} \\ \sqrt{10} & \sqrt{10} & \sqrt{13} & 0 & \sqrt{2} \\ 2 & 2\sqrt{2} & \sqrt{17} & \sqrt{2} & 0 \end{pmatrix}, \tag{19}$$

where $D_{ij}$ is a distance between each pair of nodes $i, j \in \{1, 2, 3, 4, 5\}$. Let the flow of demand between each pair of nodes $i, j \in \{1, 2, 3, 4, 5\}$ be equal to 2. The optimal

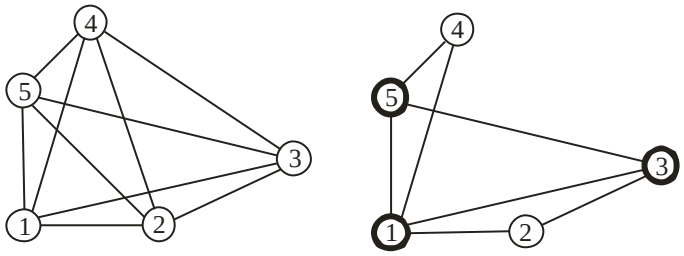


Figure 1: Hub network with $n = 5$, $p = 3$, $r = 2$, $\alpha = 0.25$

solution of the UrApHMCP that is obtained for $p = 3$, $r = 2$, $\alpha = 0.25$, $\chi = \delta = 1$ and $\beta = 2$ is presented to the right hand side of Figure 1. Hubs are located at nodes **1**, **3** and **5**, while non-hub nodes 2 and 4 are assigned to their closest established hubs. As it can be seen from the right hand side of Figure 1, non-hub node 2 is allocated to hubs 1 and 3, while non-hub node 4 is allocated to hubs 5 and 1. This optimal solution is encoded as 10101. After the locations of hubs are chosen and each node is assigned to $r$ hubs, previously described procedure is applied to find traffic routes for each pair $i - j$ and to calculate the objective function value. Namely, the transportation costs obtained as the shortest paths between each pair of nodes are: '$\mathbf{1}-\mathbf{1}$' 0, '$\mathbf{1}-2$' 2, '$\mathbf{1}-\mathbf{3}$' $0.25\cdot\sqrt{17}=1.03$, '$\mathbf{1}-\mathbf{5}-4$' $0.25\cdot 2+\sqrt{2}=1.91$, '$\mathbf{1}-\mathbf{5}$' $0.25\cdot 2=0.5$, '$2-\mathbf{1}$' 2, '$2-\mathbf{1}-2$' $2+2=4$, '$2-\mathbf{3}$' $\sqrt{5}$, '$2-\mathbf{1}-\mathbf{5}-4$' $2+0.25\cdot 2+\sqrt{2}=3.91$, '$2-\mathbf{1}-\mathbf{5}$' $2+0.25\cdot 2=2.5$, '$\mathbf{3}-\mathbf{1}$' $0.25\cdot\sqrt{17}=1.03$, '$\mathbf{3}-2$' $\sqrt{5}$, '$\mathbf{3}-\mathbf{3}$' 0, '$\mathbf{3}-\mathbf{5}-4$' $0.25\cdot\sqrt{17}+\sqrt{2}=2.44$, '$\mathbf{3}-\mathbf{5}$' $0.25\cdot\sqrt{17}=1.03$, '$4-\mathbf{5}-\mathbf{1}$' $\sqrt{2}+0.25\cdot 2=1.91$, '$4-\mathbf{5}-\mathbf{1}-2$' $\sqrt{2}+0.25\cdot 2+2=3.91$, '$4-\mathbf{5}-\mathbf{3}$' $\sqrt{2}+0.25\cdot\sqrt{17}=2.44$, '$4-\mathbf{5}-4$' $\sqrt{2}+\sqrt{2}=2.82$, '$4-\mathbf{5}$' $\sqrt{2}$, '$\mathbf{5}-\mathbf{1}$' $0.25\cdot 2=0.5$, '$\mathbf{5}-\mathbf{1}-2$' $0.25\cdot 2+2=2.5$, '$\mathbf{5}-\mathbf{3}$' $0.25\cdot\sqrt{17}=1.03$, '$\mathbf{5}-4$' $\sqrt{2}$, '$\mathbf{5}-\mathbf{5}$' 0. According to the binary coverage criterion, it may be concluded that 15 paths are considered as covered. The objective function value is then simply evaluated by summing up the demands over covered paths. Since the flow of demand between each pair of nodes is equal to 2, the objective function value that corresponds to the optimal solution is 30.

### *3.2. Genetic operators*

The proposed GA uses elitist strategy (see e.g. [20, 25, 26]). In order to ensure elitism, in each iteration, 30% of best fitted solutions from the current population is directly copied to the next population ($N_{el} = 45$). These individuals are denoted as elite ones, and their role is to preserve high-quality genetic material in the population. The rest of the next population is generated by applying GA operators: selection, crossover, and mutation.

The *selection* operator chooses the individuals that will produce offspring in the next generation, according to their fitness. Low fitness-valued individuals have

a lower chance of being selected than the ones with higher fitness values. The proposed GA uses fine-grained tournament selection (FGTS), an improvement of standard tournament selection [13]. This operator uses a real parameter $F_{tour}$, which denotes the desired average tournament size. The first type of tournaments is held $k_1$ times and its size is $[F_{tour}] + 1$, while the second type is realized $k_2$ times with $[F_{tour}]$ individuals participating. Numerical experiments performed for different hub location problems indicate that FGTS gives the best results for $F_{tour} = 5.4$ (see [19, 25, 26]). In the proposed GA implementation, FGTS operator is applied to $N_{nonel} = 105$ non-elitist individuals, tournaments are held $k_1 = 63$ and $k_2 = 42$ times with sizes 5 and 6, respectively. In practice, the running time for the FGTS operator is $O(N_{nonel} * F_{tour})$.

A *crossover* operator uses all non-elitist individuals to produce offspring for the next generation. The standard one-point crossover appears to be ineffective, and not appropriate for the applied representation scheme because the number of hubs in offspring individuals may be different from $p$. The implemented modified crossover operator randomly selects a pair of parent-individuals and produces one offspring individual, such that the set od established hubs in the offspring is obtained as the union of hubs in the parent individuals. Obviously, the offspring individual may have more than $p$ hubs. In order to correct the obtained offspring, it is necessary to set some bits to 0, i.e., to close some hubs. Therefore, a new matrix $E$ with $n$ rows and $q$ columns is constructed, where $q$ is the number of hubs of the offspring. Each entry $E(i, j)$ represents the closeness of hub $h_j$ to node $i$, which is estimated as in [23]:

$$E(i, j) = \chi d_{ij}. \tag{20}$$

The reason for using (20) is the fact that as the value of $E(i, j)$ is smaller, it is more convenient to send traffic from the node $i$ via node $j$. By using matrix $E$, hubs are removed iteratively by the following procedure. Let $m(E)$ denote the sum of minima computed for each row of matrix $E$, i.e.,

$$m(E) = \sum_{i=1}^{n} \min_{j \in \{1,2...q\}} E(i, j), \tag{21}$$

and let $E_j$ denote the matrix obtained from $E$ by deleting the column $j$. Since the removal of the hub $h_j$ corresponds to the deletion of the $j$-th column of the matrix $E$, the hub $h_j$ to be removed is chosen as the one that corresponds to the $j$-th column so that the difference between $m(E_j)$ and $m(E)$ is the least possible. After deleting hub $h_j$, matrix $E$ is replaced by matrix $E_j$. The described process is repeated until the number of hubs in the offspring becomes $p$.

Each produced offspring is subject to a *mutation* operator. Mutation is a process that reverses the structure of an individual and serves as a policy that helps in preventing the GA from being trapped in local optima. The simple mutation operator is implemented in the proposed GA. It is performed by changing the bit value at each bit position with the mutation rate $p_{mut}$, which is set to 0.03. After the mutation of an offspring, it is possible that the number of hub becomes greater

or less than $p$. In these cases, the offspring is corrected by adding or removing the necessary number of hubs. Hubs to be added or deleted are chosen randomly.

The proposed GA uses the combination of two stopping criteria: the maximal number of GA iterations without improvement ($max_1 = 300$), and the maximal number of performed iterations ($max_2 = 1000$). When at least one of these stopping criteria is met, the proposed GA finishes its work. The structure of the proposed GA for the UrApHMCP is presented in Algorithm 2.

**Algorithm 2:** Proposed GA heuristic for the UrApHMCP

```
Initialize the parameters;
Generate initial population P;
Calculate fitness values for all individuals in population;
while the number of iter. without impr. < max_1 and the number of iter. < max_2 do
    Choose N_el elite individuals;
    Set the next population C = ∅;
    Add N_el elite individuals to C;
    Apply FGTS to non-elitist individuals;
    for i = 1 to N_nonel do
        Select two parent-individuals: P_1, P_2;
        C_1 ← Crossover (P_1, P_2);
        C_1 ← Mutate (C_1, p_mut);
        Calculate fitness value of C_1;
        Insert C_1 into C;
    end
    P ← C;
end
Return the best individual from P and its fitness value;
```

## 4. COMPUTATIONAL RESULTS

In this section, the computational results of the GA method are presented. The proposed GA method is coded in C++ and all the tests are executed on an Intel Core I7 with 3.0 GHz CPU and 8GB of RAM. The performance of GA method is tested on the standard CAB and AP benchmark instances that are used for various hub location problems. A short description of data sets used in these computational experiments is given in Table 1.

Table 1: Data sets used in this computational experiments

| Name | Dimension | $\alpha$ | Description |
|---|---|---|---|
| CAB | $10 \leq n \leq 25$ | ranges from 0.2 to 1.0 | Introduced in [22]; Based on airline passenger flow between 25 cities in the United States. |
| AP | $10 \leq n \leq 200$ | 0.75 | Introduced in [12]; Derived from data from Australian Post; Involve up to 200 nodes corresponding to postal districts in Australia. |

For all considered hub instances, the collection and distribution cost parameters, $\chi$ and $\delta$, are set to 1. For CAB data set, parameter $\alpha$ ranges from 0.2 to 1.0, while for AP data set, $\alpha$ is set to 0.75.

The parameter $a_{ij}^{km}$ for the binary coverage is defined as in [24]:

$$a_{ij}^{km} = \begin{cases} 1, & \text{if } \ c_{ij}^{km} \leq 0.75\beta, \ \ \forall i, j \in N, \ \forall k, m \in H \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

For CAB instances, the values of the parameter $\beta$ are taken from [24], while for the other instances, the parameter $\beta$ is set to be equal to the corresponding solution value for the Uncapacitated Multiple Allocation $p$-hub Center Problem (UMApHCP) given in [6]. For each considered instance, the value of $\beta$ used in this paper represents the upper limit on the objective value of the UMApHCP with the same number of hubs $p$.

The quality of results obtained by the proposed GA heuristic is assessed by comparing them with the results obtained solving the ILP formulation (3) - (11) by a commercial CPLEX 12.6 MIP solver. In order to provide a fair comparison, the CPLEX was run on the same platform as GA. The maximum CPU time allowed for the CPLEX solver was set to 1 h (3600 s) on each test instance.

Since GA is a stochastic search method, it was run several times on each test instance. On each tested CAB and AP instance, the proposed GA was run 10 times.

Tables 2 – 3 provide results of the GA approach for small-size problem instances ($n = 25, 40$), while Tables 4 – 5 contain results obtained on medium and large size instances ($n = 50, 100, 200$).

In the first three columns of Tables 2 – 3, the number of nodes ($n$) and hubs to be located ($p$) are given, as well as the values of parameters $r$ and $\alpha$. The next two columns contain the optimal solutions (*Opt.sol*) obtained by the CPLEX or lower bounds, marked with superscripts $^*$, in cases when the CPLEX solver was unable to find an optimal solution within the imposed time limit of 1h and the corresponding running time ($t(s)$) in seconds. In the following two columns, the best GA solution (*Best sol*), and average total GA running time in seconds ($t_{tot}$) are given. Column *gen.avg* contains the average number of GA generations. The solution quality in all 10 executions is evaluated as a percentage gap $agap = \frac{1}{10}\sum_{i=1}^{10} gap_i$, where $gap_i$

is evaluated as percentage deviation from the optimal solution *Opt.sol*, or the best solution obtained by GA, in cases where no optimal solution is known. Standard deviation $\sigma$ of the average gap is presented in the last column.

From the results presented in Table 2, it can be seen that CPLEX failed to provide optimal solution for 3 out of 24 CAB instances. On these 3 instances (CAB25.4 for $r = 2$, $\alpha = 0.4$, CAB25.5 for $r = 2$, $\alpha = 0.6$, CAB25.4 for $r = 2$, $\alpha = 0.8$), CPLEX provided feasible solutions within 1h of CPU time. On the other hand, the proposed GA quickly reaches all known optimal solutions and improves lower bounds provided by CPLEX on three instances unsolved to optimality. The average CPLEX running time on instances that are solved to optimality is 1606.21 seconds, while the corresponding average GA running time is 0.21 seconds. On CAB instances with no optimal solution known, the proposed GA provided a solution within 0.12 seconds (on average).

The GA results on AP instances with $n = 25$ nodes (AP25), and AP instances with $n = 40$ nodes (AP40) are presented in Table 3. On these instances, CPLEX failed to provide optimal solution for 4 out of 6 instances from the group AP25, and for all AP40 instances that were tested. For AP25 instances, GA reached all optimal solutions and improved lower bounds provided by CPLEX on instances unsolved to optimality. The average CPLEX running time on instances that are solved to optimality is 1611.4 seconds, while the corresponding GA running time is 0.19 seconds, on average. On AP40 instances GA provided better solutions than lower bounds obtained by CPLEX, and the total GA running time on this data set was 0.85 seconds on average.

Medium and large size AP instances with $n = 50, 100, 200$ nodes remained out of reach of CPLEX when solving the considered problem. Not even a feasible solution was obtained within 1 h of running time, due to memory limits. Therefore, Tables 4 – 5 contain only the results obtained by the proposed GA. As it can be seen, the GA was able to provide the best solutions in an efficient manner. On average, for tested AP instances with $n = 50$ nodes (AP50), GA obtained solution in 20.26 seconds, while for large size AP instances with $n = 100, 200$ nodes, the GA running time was 70.36 seconds on average. On all instances that were solved by GA, small values of average gap and $\sigma$ indicate the reliability of the proposed GA approach.

Table 2: Comparison of CPLEX and GA solutions obtained on CAB data set

| | | | **CPLEX** | | **GA** | | | | |
|---|---|---|---|---|---|---|---|---|---|
| n.p | r | $\alpha$ | Opt.sol(%) | t(s) | Best sol(%) | $t_{tot}(s)$ | *gen.avg* | agap(%) | $\sigma$(%) |
| 25.3 | 2 | 0.2 | 96.58 | 1805.86 | 96.58 | 0.08 | 303.10 | 0.00 | 0.00 |
| 25.4 | 2 | 0.2 | 95.71 | 1541.47 | 95.71 | 0.13 | 330.20 | 0.00 | 0.00 |
| 25.4 | 3 | 0.2 | 95.71 | 2100.11 | 95.71 | 0.16 | 331.50 | 0.00 | 0.00 |
| 25.5 | 2 | 0.2 | 92.70 | 2087.15 | 92.70 | 0.18 | 420.20 | 0.00 | 0.00 |
| 25.5 | 3 | 0.2 | 92.70 | 2849.29 | 92.70 | 0.25 | 385.20 | 0.00 | 0.00 |
| 25.5 | 4 | 0.2 | 92.70 | 2431.20 | 92.70 | 0.31 | 409.20 | 0.00 | 0.00 |
| 25.3 | 2 | 0.4 | 96.24 | 2305.22 | 96.24 | 0.76 | 300.80 | 0.00 | 0.00 |
| 25.4 | 2 | 0.4 | 93.80* | 3600.09 | 95.01 | 0.10 | 302.70 | 0.00 | 0.00 |
| 25.4 | 3 | 0.4 | 95.01 | 2049.70 | 95.01 | 0.14 | 303.50 | 0.00 | 0.00 |
| 25.5 | 2 | 0.4 | 91.45 | 1267.85 | 91.45 | 0.15 | 340.60 | 0.25 | 0.25 |
| 25.5 | 3 | 0.4 | 91.84 | 1105.92 | 91.84 | 0.25 | 367.50 | 0.00 | 0.00 |
| 25.5 | 4 | 0.4 | 91.84 | 1110.36 | 91.84 | 0.28 | 351.50 | 0.00 | 0.00 |
| 25.3 | 2 | 0.6 | 93.17 | 354.23 | 93.17 | 0.08 | 300.70 | 0.00 | 0.00 |
| 25.4 | 2 | 0.6 | 93.62 | 1055.33 | 93.62 | 0.11 | 318.80 | 0.00 | 0.00 |
| 25.4 | 3 | 0.6 | 93.63 | 831.95 | 93.63 | 0.13 | 320.50 | 0.01 | 0.01 |
| 25.5 | 2 | 0.6 | 90.00* | 3600.05 | 90.12 | 0.14 | 314.10 | 0.00 | 0.00 |
| 25.5 | 3 | 0.6 | 90.19 | 2727.55 | 90.19 | 0.23 | 342.30 | 0.05 | 0.04 |
| 25.5 | 4 | 0.6 | 90.19 | 3531.83 | 90.19 | 0.26 | 353.50 | 0.00 | 0.00 |
| 25.3 | 2 | 0.8 | 90.08 | 682.41 | 90.08 | 0.08 | 303.30 | 0.00 | 0.00 |
| 25.4 | 2 | 0.8 | 89.02* | 3600.07 | 89.60 | 0.11 | 311.00 | 0.00 | 0.00 |
| 25.4 | 3 | 0.8 | 89.61 | 1829.71 | 89.61 | 0.15 | 322.20 | 0.00 | 0.01 |
| 25.5 | 2 | 0.8 | 89.03 | 728.15 | 89.03 | 0.16 | 329.00 | 0.29 | 0.28 |
| 25.5 | 3 | 0.8 | 89.05 | 605.09 | 89.05 | 0.24 | 350.90 | 0.00 | 0.00 |
| 25.5 | 4 | 0.8 | 89.05 | 730.11 | 89.05 | 0.27 | 345.30 | 0.00 | 0.00 |
| **Average:** | | | 92.20 | 1855.45 | 92.22 | 0.20 | 335.73 | 0.03 | 0.02 |

Table 3: Comparison of CPLEX and GA solutions obtained on AP25 and AP40 data set

| | | | **CPLEX** | | **GA** | | | | |
|---|---|---|---|---|---|---|---|---|---|
| n.p | r | $\alpha$ | Opt.sol(%) | t(s) | Best sol(%) | $t_{tot}(s)$ | *gen.avg* | agap(%) | $\sigma$(%) |
| 25.3 | 2 | 0.75 | 95.13 | 765.58 | 95.13 | 0.08 | 302.00 | 0.15 | 0.15 |
| 25.4 | 2 | 0.75 | 96.94* | 3000.50 | 96.94 | 0.12 | 306.50 | 0.00 | 0.00 |
| 25.4 | 3 | 0.75 | 97.16 | 1068.12 | 97.16 | 0.16 | 305.50 | 0.05 | 0.05 |
| 25.5 | 2 | 0.75 | 97.47* | 3600.07 | 97.87 | 0.17 | 338.50 | 0.00 | 0.00 |
| 25.5 | 3 | 0.75 | 98.03* | 3600.10 | 98.03 | 0.28 | 394.10 | 0.00 | 0.00 |
| 25.5 | 4 | 0.75 | 98.06* | 3600.09 | 98.06 | 0.35 | 389.80 | 0.12 | 0.20 |
| 40.3 | 2 | 0.75 | 84.25* | 3600.36 | 97.77 | 0.22 | 309.10 | 0.00 | 0.00 |
| 40.4 | 2 | 0.75 | 26.92* | 3600.35 | 96.85 | 0.42 | 356.00 | 0.03 | 0.04 |
| 40.4 | 3 | 0.75 | 28.64* | 3600.36 | 97.22 | 0.70 | 392.60 | 0.00 | 0.00 |
| 40.5 | 2 | 0.75 | 38.30* | 3600.36 | 97.04 | 0.60 | 376.60 | 0.00 | 0.00 |
| 40.5 | 3 | 0.75 | 38.05* | 3600.45 | 97.25 | 1.31 | 483.70 | 0.16 | 0.09 |
| 40.5 | 4 | 0.75 | 36.04* | 3600.35 | 97.44 | 1.84 | 516.90 | 0.10 | 0.05 |
| **Average:** | | | 69.58 | 3103.06 | 97.23 | 0.52 | 372.61 | 0.05 | 0.05 |

Table 4: GA results obtained for UrApHMCP on AP50 data set

| | | | GA | | | | |
|---|---|---|---|---|---|---|---|
| n.p | r | $\alpha$ | Best sol (%) | $t_{tot}$(s) | *gen.avg* | agap(%) | $\sigma$(%) |
| 50.3 | 2 | 0.75 | 97.86 | 0.44 | 333.60 | 0.14 | 0.05 |
| 50.4 | 2 | 0.75 | 97.53 | 0.80 | 333.90 | 0.07 | 0.07 |
| 50.4 | 3 | 0.75 | 98.05 | 1.56 | 446.00 | 0.02 | 0.01 |
| 50.5 | 2 | 0.75 | 97.66 | 1.40 | 514.50 | 0.06 | 0.02 |
| 50.5 | 3 | 0.75 | 97.89 | 2.87 | 528.70 | 0.06 | 0.05 |
| 50.5 | 4 | 0.75 | 97.90 | 5.32 | 488.50 | 0.03 | 0.04 |
| 50.10 | 2 | 0.75 | 99.20 | 3.60 | 740.10 | 0.02 | 0.03 |
| 50.10 | 3 | 0.75 | 99.27 | 9.73 | 802.20 | 0.04 | 0.04 |
| 50.10 | 4 | 0.75 | 99.29 | 18.76 | 778.80 | 0.02 | 0.01 |
| 50.10 | 5 | 0.75 | 99.32 | 30.47 | 787.70 | 0.03 | 0.02 |
| 50.10 | 6 | 0.75 | 91.11 | 43.41 | 773.80 | 0.04 | 0.02 |
| 50.10 | 7 | 0.75 | 91.08 | 80.06 | 831.70 | 0.02 | 0.01 |
| 50.10 | 8 | 0.75 | 99.35 | 64.92 | 809.80 | 0.02 | 0.02 |
| **Average:** | | | 97.35 | 20.26 | 628.41 | 0.05 | 0.03 |

Table 5: GA results obtained for UrApHMCP on large size AP instances

| | | | GA | | | | |
|---|---|---|---|---|---|---|---|
| n.p | r | $\alpha$ | Best sol(%) | $t_{tot}$(s) | *gen.avg* | agap(%) | $\sigma$(%) |
| 100.3 | 2 | 0.75 | 97.74 | 3.76 | 365.33 | 0.02 | 0.03 |
| 100.4 | 2 | 0.75 | 97.39 | 10.54 | 456.67 | 0.05 | 0.04 |
| 100.4 | 3 | 0.75 | 97.85 | 19.71 | 410.67 | 0.03 | 0.05 |
| 100.5 | 2 | 0.75 | 98.03 | 15.94 | 610.67 | 0.00 | 0.00 |
| 100.5 | 3 | 0.75 | 98.35 | 36.49 | 720.00 | 0.00 | 0.00 |
| 100.5 | 4 | 0.75 | 98.15 | 51.45 | 494.00 | 0.09 | 0.09 |
| 200.3 | 2 | 0.75 | 98.83 | 56.50 | 603.50 | 0.01 | 0.01 |
| 200.4 | 2 | 0.75 | 98.62 | 97.47 | 703.00 | 0.00 | 0.00 |
| 200.4 | 3 | 0.75 | 98.81 | 207.09 | 631.50 | 0.01 | 0.01 |
| 200.5 | 2 | 0.75 | 98.54 | 204.69 | 825.00 | 0.05 | 0.05 |
| 200.5 | 3 | 0.75 | 98.67 | 458.91 | 501.00 | 0.02 | 0.02 |
| **Average:** | | | 98.23 | 70.36 | 574.67 | 0.03 | 0.03 |

## 5. CONCLUSION

In this paper, an integer linear programming mathematical formulation for the Uncapacitated *r*-allocation *p*-hub Maximal Covering Problem (UrApHMCP) with a binary coverage criterion is presented. In order to solve the problem instances of larger problem dimensions, a Genetic Algorithm (GA) adapted to the considered problem is developed. Computational results on CAB and AP instances with up to 200 nodes demonstrate the efficiency and practicability of the proposed GA with respect to both computation time and solution quality. The obtained results clearly indicate that GA represents a promising heuristic approach to UrApHMCP. Future research should be conducted to examine the parallelization of the GA and its hybridization with other heuristics for solving this problem.

## REFERENCES


[1] Abdinnour-Helm, S., and Venkataramanan, M.A., "Solution approaches to hub location problems", *Annals of Operations Research*, 78 (1998) 31-50.

[2] Alumur, S., and Kara, B., "A Hub Covering Network Design Problem for Cargo Applications in Turkey", *Journal of Operational Research Society*, 60 (10) (2009) 1349-1359.

[3] Alumur, S., and Kara, B., "Network hub location problems: The state of the art", *European Journal of Operational Research*, 190 (1) (2008) 1-21.

[4] Beasley, D., Bull, D., and Martin, R., "An overview of genetic algorithms: Part 1. Fundamentals", *University computing*, 15 (1993) 58-58.

[5] Beasley, D., Bull, D., and Martin, R., "An overview of genetic algorithms: Part 2, research topics", *University computing*, 15 (4) (1993) 170-181.

[6] Brimberg, J., Mladenović, N., Todosijević, R., and Urošević, D., "A basic variable neighborhood search heuristic for the uncapacitated multiple allocation p-hub center problem", *Optimization Letters*, (2015) 1-15.

[7] Campbell, J.F., "Integer programming formulations of discrete hub location problems", *European Journal of Operational Research*, 72 (2) (1994) 387-405.

[8] Church, R., and ReVelle, C., "The maximal covering location problem", *Papers in Regional Science*, 32 (1) (1974) 101-118.

[9] Eremeev, A.V., and Kovalenko, J.V., "Optimal recombination in genetic algorithms for combinatorial optimization problems-Part I", *Yugoslav Journal of Operations Research*, 24 (1) (2014) 1-20.

[10] Eremeev, A.V., and Kovalenko, J.V., "Optimal recombination in genetic algorithms for combinatorial optimization problems-Part II", *Yugoslav Journal of Operations Research*, 24 (2) (2014) 165-186.

[11] Ernst, A.T., Jiang, H., Krishnamoorthy, M., and Baatar, H., "Reformulations and computational results for uncapacitated single and multiple allocation hub covering problems", *Working Paper Series*, 1 (2011) 1-18.

[12] Ernst, A.T., and Krishnamoorthy, M., "Exact and heuristic algorithms for the uncapacitated multiple allocation p-hub median problem", *European Journal of Operational Research*, 104 (1) (1998) 100-112.

[13] Filipović, V., "Fine-grained tournament selection operator in genetic algorithms", *Computing and Informatics*, 22 (2) (2012) 143-161.

[14] Goldberg, D.E., *Genetic algorithms in search, optimization and machine learning addison-wesley, 1989*, Addison-Wesley, Publishing Company, Reading, Mass., 1989.

[15] Goldberg, D.E., and Holland, J.H., "Genetic algorithms and machine learning", *Machine learning*, 3 (2) (1988) 95-99.

[16] Holland, J.H., *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, The University of Michigan Press, Ann Arbor, 1975.

[17] Hwang, Y.H., and Lee, Y.H., "Uncapacitated single allocation p-hub maximal covering problem", *Computers & Industrial Engineering*, 63 (2) (2012) 382-389.

[18] Kara, B., and Tansel, B., "The single-assignment hub covering problem: Models and linearizations", *Journal of the Operational Research Society*, 51 (1) (2003) 59-64.

[19] Kratica, J., Stanimirović, Z., Tošić, D., and Filipović, V., "Two Genetic Algorithms for Solving the Uncapacitated Single Allocation p-Hub Median Problem", *European Journal of Operational Research*, 182 (2007) 15-28.

[20] Marić, M., Stanimirović, Z., and Stanojević, P., "An efficient memetic algorithm for the uncapacitated single allocation hub location problem", *Soft Computing*, 17 (3) (2013) 445-466.

[21] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin, Heidelberg, 1996.

[22] O'Kelly, M.E., "A quadratic integer program for the location of interacting hub facilities", *European Journal of Operational Research*, 32 (3) (1987) 393-404.

[23] Peiró, J., Corberán, Á., and Martí, R., "Grasp for the uncapacitated r-allocation p-hub median problem", *Computers & Operations Research*, 43 (2014) 50-60.

[24] Peker, M., and Kara, B.Y., "The p-hub maximal covering problem and extensions for gradual decay functions", *Omega*, 54 (2015) 158–172.

[25] Stanimirović, Z., "A genetic algorithm approach for the capacitated single allocation p-hub median problem", *Computing and Informatics*, 29 (1) (2010) 117-132.
[26] Stanojević, P., Marić, M., and Stanimirović, Z., "A hybridization of an evolutionary algorithm and a parallel branch and bound for solving the capacitated single allocation hub location problem", *Applied Soft Computing*, 33 (2015) 24-36.
[27] Topcuoglu, H., Corut, F., Ermis, M., and Yilmaz, G., "Solving the uncapacitated hub location problem using genetic algorithms", *Computers & Operations Research*, 32 (4) (2005) 967-984.
[28] Wagner, B., "Model formulations for hub covering problems", *Journal of the Operational Research Society*, 59 (7) (2008) 932-938.
[29] Weng, K., Yang, C., and Ma, Y., "Two artificial intelligence heuristics in solving multiple allocation hub maximal covering problem", In *Intelligent Computing*, Springer, Berlin, Heidelberg, 2006.
[30] Yaman, H., "Allocation strategies in hub networks", *European Journal of Operational Research*, 211 (3) (2011) 442-451.