International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 3, June 2021, pp. 2293∼2306
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i3.pp2293-2306

An artificial immune system algorithm for solving the uncapacitated single allocation p-Hub median problem

Fatima Zahraa Grine[1], Oulaid Kamach[2], Abdelhakim Khatab[3], Naoufal Sefiani [4]
[1,2]Innovative Technologies Laboratory, National School of Applied Sciences, Tangier, Morocco
[3]Laboratory of Computer Engineering, Production and Maintenance (LGIPM), Lorraine University, Metz, France
[4]Faculty of Sciences and Technologies, Tangier, Morocco

Article Info

Article history:

Received Feb 29, 2020
Revised Sep 19, 2020
Accepted Oct 6, 2020

Keywords:

Artificial immune system
Clonal selection algorithm
Evolutionary computations
Hub location problem
Metaheuristic
p-Hub median problem

ABSTRACT

The present paper deals with a variant of hub location problems (HLP): the uncapacitated single allocation p-Hub median problem (USApHMP). This problem consists to jointly locate hub facilities and to allocate demand nodes to these selected facilities. The objective function is to minimize the routing of demands between any origin and destination pair of nodes. This problem is known to be NP-hard. Based on the artificial immune systems (AIS) framework, this paper develops a new approach to efficiently solve the USApHMP. The proposed approach is in the form of a clonal selection algorithm (CSA) that uses appropriate encoding schemes of solutions and maintains their feasibility. Comprehensive experiments and comparison of the proposed approach with other existing heuristics are conducted on benchmark from civil aeronautics board, Australian post, PlanetLab and Urand data sets. The results obtained allow to demonstrate the validity and the effectiveness of our approach. In terms of solution quality, the results obtained outperform the best-known solutions in the literature.

Corresponding Author:

Fatima Zahraa Grine
Innovative Technologies Laboratory
National School of Applied Sciences
Tangier, Morocco
Email: fz.grine@uae.ac.ma

1. INTRODUCTION

In a distribution network, hubs refer to facilities that allow consolidation, sorting, switching and connecting of flows between origins and destinations [1]. The design of a hubs network (i.e., distribution network containing hubs) aims to reduce the cost resulting from connecting many origin-destination pairs of nodes. Indeed, using hub facilities to consolidate flows brings the advantages of economies of scale. Hub networks are widely exploited in many industrial areas including telecommunication, air passenger and freight transportation, postal delivery and public transportation network [1, 2]. In transportation networks, for example, decision makers are usually faced with the commodity flows problem and need to develop efficient and competitive logistic strategies to manage such flows. It has been shown that logistic platforms (hubs) can indeed offer a very interesting competitive potential to cover the commodity flows issue. Furthermore, in addition to their ability to reduce logistic costs, logistic platforms can be designed by considering sustainable criteria.

Dealing with hub location problems (HLP), two main decisions are addressed, namely the location of hubs and the allocation of the demand nodes to the selected hubs. The selection of hubs is performed so

that to determine the optimal routes of flow between origin-destination pairs. Since the pioneered work by O' Kelly [3], hub location problem attracted many researchers resulting in several hub location optimization models appeared in the literature.

In the research literature, the classification of HLP can generally made according to four criteria [2, 4]: the type of space for locating hub-nodes, the assignment rules, the number of hubs and the capacity of hubs. Regarding the first criterion (i.e., type of space to locate hub-nodes) it is also referred to as network domain which can either be discrete or continuous. In the discrete domain, hubs can be located only at a set of finite points, while in the continuous domain, hubs can be located arbitrarily in a plan or sphere [4]. The assignment rules criterion refers to two types of nodes allocation, namely single and multiple allocation. In the former type of allocation, non-hub nodes as demand points are assigned to only one hub, while in the later non-hub nodes as demand points can be assigned to more than one hub [5]. According to the third criterion, the number of hubs in network can either be a decision variable or rather defined exogenously and denoted as p. In the latter case one speaks about p-HLP. In terms of the fourth criterion (i.e. hub capacity), the capacity of a hub can be of two types limited, in which case one deal with capacitated HLP, and unlimited and therefore deal with the uncapacitated HLP. Objective functions usually consist, for example, on minimizing the total cost, minimizing the maximum of travelling time. The reader can refer to some surveys provided in [1, 2, 4, 6]. More recent works have studied more complex models within HLP by relaxing some classical assumptions while incorporating additional features to meet real-life applications' requirements, see for example [7–9].

Sadeghi *et al.* [10] considered the p-Hub covering location problem where reliability of the transportation network is accounted for by assuming that road (i.e., link) capacities are subject to stochastic degradations. The road capacity reliability is then defined as the probability that the maximum network capacity is greater than or equal to the total incoming flow to the network. A stochastic optimisation model is proposed where the objective is to minimize the total transportation cost. The resulting model is solved by using differential evolution (DE) algorithm.

In [8], the authors, motivated by the application of delivery within the same day in the city, deal with the problem of joint single allocation p-Hub median location and routing problem, and pick-up and delivery. The proposed approach extends the uncapacitated single allocation p-Hub median problem to include vehicle routing with simultaneous pick-up and delivery. Several mixed integer programming formulations are provided where the objective is to minimize the cost of transferring the flow between hubs and routing the flow in the network. Two heuristic approaches based on multi start simulated annealing and ant colony system to solve these problems.

Recently, Alumur *et. al.* [7] incorporate the service time constraints and congestion in hub location problems. Service time is reflected by the travel time on the network connections and the time flows processing time at the hubs level. Congestion is introduced to model the delay caused by the increase in processing times at the hubs level. The authors developed mixed-integer linear programming (MILP) formulations for the single and multiple allocation strategies, and for the direct shipments case within the multiple allocation option.

In a more recent paper, Taherkhani and Alumur [9] considered the HLP from a profit point of view. The profit is defined as the total revenue minus total cost. The total cost is induced by the transportation cost, the installation cost of hubs, in addition to the cost of operating hub links. By considering different allocation strategies, four mathematical models are formulated to determine the location of hubs, designing the hub networks, and routing the demand.

The present paper deals with solution methods dedicated to the uncapacitated single allocation p-Hub median problem (USApHMP) where hubs are assumed to have unlimited capacities and each spoke is assigned to one and only one hub. Furthermore, the number of hubs, denoted as p, to locate is assumed to be known (exogeneous parameter). The objective of the USApHMP consists to jointly select appropriate hub facilities and to allocate demand nodes to these selected facilities so that to minimize the transportation costs of the overall flow in a network.

In the literature, there are exact and approximate solution methods used to solve the USApHMP. In [3], a quadratic formulation is proposed for the USApHMP problem. Later on, two integer linear formulations were proposed by [11, 12]. To solve the resulting optimization model in [3], a full enumeration method is used combined with two assignment heuristics based on the nearest and the second nearest hub. In [13] Klincewiczhas proposed an exchange heuristic based on local neighborhood search which was proved to be more effective than the method developed by [3]. Since USApHMP is NP-hard [14], exact method are valid only for low-sized problems. Therefore, developing efficient solution methods is crucial to deal with large-sized

hub location problems. In [11], Klincewicz developed two heuristics based on tabu search and on a greedy randomized adaptive search procedure (GRASP) where nodes are assumed to be assigned to their nearest hub.

A new variant of tabu search algorithm, called TABUHUB, is proposed by Skorin-Kapov and [12]. This approach considers equally the locational and the allocational part of the hub location problem. It also consists of a single exchange heuristic for the location of hubs, and for the reallocations of non-hub nodes. Ernst and Krishnamoorthy [15] developed a heuristic method based on simulated annealing (SA) to solve the USApHMP. They have developed an LP-based branch and bound solution method by using the SA upper bound. Pérez *et al.* [16] have presented an algorithm based on a hybridization combining the variable neighborhood search (VNS) together with the path relinking (PR) heuristics. Their VNS-PR hybrid heuristic outperforms the TABUHUB-based approach [12] and SA-based approach [15] in both solution quality and computational times.

In Kratica *et al.* [17] two genetic algorithm GA-based approaches are proposed. New encoding schemes and modified genetic operators are used to maintain the feasibility of individuals. The proposed approach exploited the idea of frozen bits to increase the diversity of genetic algorithm. To improve the computational time, the authors used the caching technique to avoid the fitness calculation for individuals that reappear during the GA run. Experiments conducted on both civil aeronautics board (CAB) and Australia post (AP) benchmark data sets, demonstrated to be robustness and efficiency of the approach in solving the USApHMP with up to 200 nodes and 20 hubs.

Urošević *et al.* [18] proposed a new general variable neighborhood search algorithm for solving the USApHMP. The authors provided three neighborhoods and structures to efficiently computing the network's total flow. In addition, the authors proposed a new nested strategy in designing a deterministic variable neighborhood descent local search.

The more recent solution approach of the USApHMP appeared in Benaini *et al.* [19]. the authors implemented a parallel genetic algorithm on graphic processing unit (GPU) for solving the problem. Their approach improved some best-known solutions in cost and in computing times.

The present paper develops a new approach to efficiently solve the USApHMP. The approach is based on the artificial immune systems (AIS) framework which is a new class of computational intelligence inspired by the biologic immune system [20]. As point out in [21], AIS is one of the most effective methods to be used while dealing with optimization problems. It is a sort of population based metaheuristics which is known for its ability to reach global optimum and avoid local optimums [22]. The most appropriate AIS algorithms for optimization problems are clonal selection algorithms (CSA) which are based on clonal selection theory [23]. CSA starts by generating a set of candidate solutions in the form of an initial population, and improves the features of the population by applying evolutionary operators until a stopping criterion is reached. Its computational cost is reduced when compared to other evolutionary algorithms [22].

This paper is an extension to the work presented in [24]. In our previous work we provided a summary about theoretical framework of the artificial immune systems and a detailed review of the clonal selection algorithm. In this study we provide numerical results to the proposed approach and compare it with existing methods in the literature.

Accordingly, this paper develops a CSA-based method to solve the USApHMP, which is an adaptation of the optimization version of the CLONALG algorithm [22]. To the best of our knowledge, this heuristic has never been used before in solving the USApHMP. Comprehensive experiments and comparison of the proposed approach with other existing heuristics are conducted on benchmark from civil aeronautics board and Australian post data sets. Roughly speaking, the proposed approach is compared with: The simulated annealing method (SA), proposed by Ernst and Krishnamoorthy [15], the hybrid variable neighborhood search VNS and path relinking PR heuristics presented by [16], Genetic algorithm GAHUB2 proposed by Kratica *et al.* [17], and the general variable neighborhood search (GVNS) proposed by [18]. The results obtained allow to demonstrate the validity and the effectiveness of our approach. In terms of solution quality, the results obtained outperform the best-known solutions in the literature.

The rest of the paper is structured around 5 sections. Section 2 provides the mathematical formulation of USApHMP used in this work. A brief overview of artificial immune systems and clonal selection algorithms is given in section 3. Section 4 describes the proposed approach. Computational results on different problem instances from the literature are reported in section 5. Conclusions and future research works are summarized in section 6.

2. NOTATION AND MAIN WORKING ASSUMPTIONS

This section presents the notation system used and also states the main working assumptions made in this paper.

2.1. Notation

$\mathcal{G}$	Hub network graph model
$\mathcal{N}$	Set of nodes of a hub network graph $\mathcal{G}$
$\mathcal{A}$	Set of arcs of a hub network graph $\mathcal{G}$
N	Number of nodes in hub network graph $\mathcal{G}$
p	Number of hubs allocated in a hub network
(i,j)	Arcs linking two nodes i and j of $\mathcal{G}$
w_{ij}	Flow routed in the arc (i,j)
d_{ij}	Distance between nodes i and j
χ	Collection cost per flow and per distance units incurred between a spoke (non-hub) and hub nodes
α	Transfer cost per flow and per distance units defined between hubs
δ	Distribution cost per flow and per distance units defined between a hub and a spoke destination nodes

2.2. Assumptions

- The number of hubs to locate is predetermined (p).
- Hubs are assumed to have unlimited capacities.
- Each origin-destination node is assigned to a single hub.
- Direct link between nodes is not allowed.
- The network is complete.

3. PROBLEM DEFINITION AND MATHEMATICAL FORMULATION

Let us consider a hub network modelled as a complete graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of arcs. A note $i \in \mathcal{N}$ can either be a hub or a spoke (non-hub nodes). To simplify the notations, nodes are considered as numbered from 1 to cardinality N of $\mathcal{N}$. An arc $(i,j) \in \mathcal{A}$ $(i = 1, \cdots, N; j = 1, \cdots, N)$ is a link between nodes i and j, respectively, the origin and destination. Each arc $(i,j) \in \mathcal{A}$ is characterized by the amount w_{ij} of flow to be routed, and a distance d_{ij} between nodes i and j $(d_{ii} = 0)$. Three cost rates are considered, namely the collection cost χ defined between origin and hub nodes, the transfer cost α defined between hubs, in addition to the distribution cost δ defined between hub and destination nodes.

The uncapacitated single allocation p-hub median problem (USApHMP) considered in the present work is a variant of the HLP where hubs are assumed to have unlimited capacities and each spoke is assigned to one and only one hub. Furthermore, the number p of hubs to locate is assumed to be known (exogenous parameter). The objective of the USApHMP consists to jointly select appropriate hub facilities and to allocate demand nodes to these selected facilities so that to minimize the transportation costs incurred from collection, transfer and distribution of the overall flow in the network. To model this optimization problem, we consider a binary decision variable z_{ij} defined as (1):

$$z_{ij} = \begin{cases} 1, & \text{if node } i \text{ is allocated to} \\ & \text{a hub located at node } j, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Accordingly, the objective function is computed as the sum of collection, transfer and distribution costs as [18]:

$$\sum_i \sum_j \sum_k \sum_l w_{ij}(\chi.d_{ij}.z_{ik} + \alpha.d_{kl}.z_{ik}.z_{jl} + \delta.d_{jl}.z_{jl}) \tag{2}$$

The resulting USApHMP optimization problem can be then formulated as:

3.1. Model USApHMP

Minimize

$$\sum_i \sum_j \sum_k \sum_l w_{ij}(\chi.d_{ij}.z_{ik} + \alpha.d_{kl}.z_{ik}.z_{jl} + \delta.d_{jl}.z_{jl}) \tag{3}$$

Subject to:

$$\sum_i z_{ii} = p \tag{4}$$

$$\sum_k z_{ik} = 1, \quad \forall i \tag{5}$$

$$z_{ik} \leq z_{kk}, \quad \forall i, k \tag{6}$$

$$z_{ij} \in \{0, 1\}, \quad \forall i, j, \tag{7}$$

In the above optimization model, the objective function in (3) minimizes the overall transportation cost. Constraints in (4) ensures that exactly p hubs are located. Constraints given by (5) ensure that all nodes are hubs or are assigned to a single hub. Finally, constraints in (6) ensures that no node is assigned to a location unless a hub is opened at that site, which also mean that node to node connection is not allowed in the network.

4. SOLUTION METHOD

The optimization model USApHMP formulated by (3)-(7), is NP-hard. An exhaustive examination of all possible solutions is therefore not realistic due to the computational time limitation. Meta-heuristic algorithms are well known approaches for their efficiency and effectiveness to search, in a reasonable amount of time, the optimal or a near optimal solution of complex combinational problems. Based on the artificial immune systems (AIS) framework, this paper develops a new approach to efficiently solve the USApHMP (3)-(7). AIS are computational intelligence algorithms inspired by the natural biological immune systems. The four known AIS-based algorithms are immune network, negative/positive selection and clonal selection algorithms. They are all recognized for their powerful adaptive learning and robustness, in addition to their valuable properties to avoid premature convergence and improve local search [23]. They have indeed been successfully applied to a large range of domains including classification/clustering, anomaly detection and pattern recognition [25, 26].

To solve the USApHMP problem, the present paper develops a procedure based on clonal selection algorithm (CSA). CSA is inspired by the features of clonal selection theories of immune systems and appeared first in the work by de Castro and Von Zuben [27]. Since then, it has been demonstrated to be an efficient method to solve multi-modal and combinatorial optimization problems. The CSA is constructed around 7 steps and processes: Population initialization, affinity evaluation, selection and cloning (proliferation), hypermutation, receptor editing, convergence and termination condition. Algorithm 1 describes the pseudo code of our algorithm.

The analogy between the immune system and the algorithm is presented in Table 1. In this Table, an antigen refers to the value of the objective function, an antibody represents a candidate feasible solution, and the affinity value refers to the performance measure of an antibody. The affinity represents indeed the degree of matching between the antibody and the antigen. The detailed design (solution encoding and operators' definitions, initial population selection, $\cdots$) of the CSA developed here will be discussed as shown in Table 1.

Algorithm 1 Pseudo code of the proposed algorithm

Parameters:
MaxIt: Maximum Number of Generations
Psize: Population size
P: Population (Set of antibodies)
Nsel: Number of Abs (Individuals) to select for cloning
N: Number of nodes in the network (bit string length of antibody)
d: Number of random antibodies to replace at the end of each generation (d antibodies with lowest affinity)
Idv: Individuals (Antibodies)
Nc: Number of clones generated by each selected antibody
$\beta : Multiplying\ factor(user\ parameter)$
Proposed Algorithm:
Input: P, $Nsel$, N, Nc, d, β
Output: Pf
$P \leftarrow rand(Psize, N)$
while $Number\ of\ Generation\ <\ MaxIt$ **do**
  for i=1 to Psize **do**
    $Calculate\ objective\ function\ value\ for\ each\ individual\ (Pi)$
  end for
  $Pselect \leftarrow Select(P, Nsel)$ using SUS selection method
  for i=1 to Nsel **do**
    $Pclones \leftarrow Clone(Pi, \beta)$
  end for
  for i=1 to Pclones **do**
    $Interchanging\ mutation(Pi)$
    $Calculate\ objective\ function\ value\ for\ each\ individual(Pi)$
  end for
  $P \leftarrow Select(P, Pclones, Psize)$
  $Prand \leftarrow rand(d, N)$
  $Replace(d, Prand)$
end while
Return P
Report the best antibody as the final solution

Table 1. Analogy between natural immune system principles and our developed system

Natural immune system	Artificial immune system applied in USApHMP
Antigen	The value of the objective function to be optimized
Antibody	Candidate feasible solution to the problem
Affinity	Degree of matching between an antibody and the antigen
Proliferation of antibody	Creation of new antibodies by using of clonal operators

4.1. Solution representation (encoding), and initial population generation

In order to apply the CSA to the stated optimization problem, a specific representation scheme to encode antibodies is required. In the present work, individual solution (antibody) is represented by a vector **s** of integers whose length is given by the number N of nodes in the complete graph $\mathcal{G}$. The i^{th} element of the vector solution **s** corresponds to the hub number to which node i is assigned to. To illustrate, let us consider the solution **s** :

$$\mathbf{s} = [1\ 4\ 7\ 4\ 7\ 4\ 7\ 7\ 1\ 4\ 1].$$

Accordingly, the corresponding hub network is composed of a total of $N = 11$ nodes among which $p = 3$ nodes are defined as hubs. The three pre-selected hubs are numbered as 1, 4, and 7. From this representation, one may observe that each node of the network is linked to the appropriate hub. For example, nodes 2, 6 and

10 are all linked to hub 4, while nodes 3, 5 and 8 are assigned to hub 7. It is worth noticing that each hub is assigned to itself.

To generate the initial population, a procedure is developed on the basis of flow quantities emanating from and destined to a given node. Roughly speaking, the procedure considers the total amount O_i and D_i of flow originating from and destined to a node i ($i \in \mathcal{N}$), respectively. These two quantities are evaluated as: $O_i = \sum_j w_{ij}$ and $D_i = \sum_j w_{ji}$. The resulting total amount F_i of flows at each node i of the network is computed as $F_i = O_i + D_i$. The overall nodes are then arranged in a decreasing order of total flows. From the resulting ordered list, the initial population is then generated in two steps as summarized in the pseudo code of algorithm 2. From this algorithm,the first step defines the 80% of candidate solutions of the population. For this part of the population, for each individual we choose in a randomly way p hubs from nodes that represent the highest total amount of flows in the network. To do this, we sort nodes in a list (flow-list) in decreasing order of total amount of flows, and we select p hubs from the nodes that are in the range $1...3/4n$ of the flow-list. This strategy promotes the nodes that has higher amount of flow to have a higher probability of being selected as hub. The second step construct the remaining 20% of the population by choosing p hubs randomly from the entire set of nodes. Once the selection of hubs is achieved, we proceed to the allocation of the remaining spokes (non-hub nodes) to the established hubs. For each non-hub node i, the located hubs are sorted in ascending order of distance from i, then i is allocated to the nearest hub. Each established hub is allocated to itself, following the assumptions of the problem.

Algorithm 2 Population Initialization

Input : N,p.
Output: Set of feasible solutions.
Calculate the total amount of flow originated and destined to each node: Total-Flow$_k = O_k + D_k$.
Sort the nodes in decreasing order of Total-Flowk values in a list, flow-list.
Hub-Set= N$_k | N_k = flow - list[i], \forall i, 1 \leq i \leq \frac{3}{4} \times N$.
for $i = 1\ to\ (\frac{4}{5} \times pop_{size})$ **do**
    Select p hubs from Hub-Set, randomly.
    Assign each spoke to the nearest hub based on distance values.
end for
for $(\frac{4}{5} \times pop_{size}) + 1\ to\ pop_{size}$ **do**
    Select p hubs from all nodes randomly.
    Assign each spoke to the nearest hub based on distance values.
end for

4.2. Affinity determination

Let us recall that the affinity value of a given candidate solution measures the degree of matching between antibodies and antigen. In the present work, the affinity of a solution is inverse proportional to the solution fitness. We use the global cost given by the objective function of the optimization model USApHMP (3) to evaluate the fitness of each solution. Each solution in the population has a chance of surviving to the next generation with respect to its affinity level. The affinity value is computed for each solution to conduct the selection and cloning process.

4.3. Selection and cloning

Selection process is a stochastic procedure, based on the affinity/fitness value of each solution in the population, used to select the solutions that will move on to the subsequent generation. The stochastic universal sampling (SUS) is employed in this implementation of clonal selection algorithm (algorithm 3), it was developed as a single-phase sampling algorithm with minimum spread and zero bias. This selection method is a development of roulette wheel selection strategy (RWS); it uses N equally spaced pointers instead of a single selection pointer (N is the required number of selection), see algorithm 3. In the population, each individual gets a reproduction probability which depends on his own affinity value and the total affinity of all other individuals. The principle of this method consists to map the individuals to a line, such that each individual gets a portion of the line which is proportional to its affinity value. Then, a single random number (pointer1) is generated in the range $[0, 1/N]$ and N pointers are generated starting with pointer1 and

spaced by $1/N$. The N individuals whose affinity spans the positions of the pointers are selected see Figure 1. In Figure 1, the selected individuals consist of individuals 1, 2, 3, and 6. Unlike the RWS, this selection method gives weaker individuals in the population a chance to be selected.

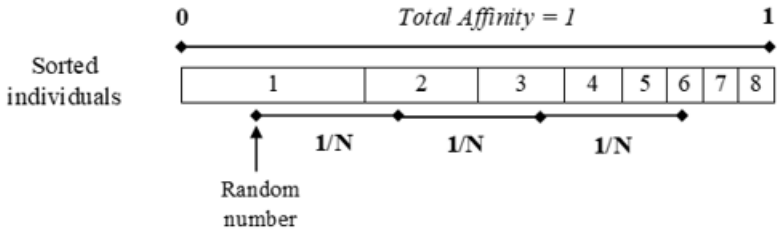

Figure 1. Stochastic universal sampling

Algorithm 3 Stochastic Universal Sampling

Input : The population P, Number of antibodies to select N
Output : The population after selection
$sum \leftarrow 0$
$pointer \leftarrow random[0, 1/N]$
for i=1 to N **do**
  $sum \leftarrow sum + p_i$ (p_i *is the probability of selection of antibody* Ab_i)
  while $sum > pointer$ **do**
    $Ab'_i \leftarrow Ab_i$
    $pointer \leftarrow pointer + 1$
  end while
end for
return $Ab'_1, ..., Ab'_N$

The antibodies that are selected from the initial population will be cloned directly proportional to their affinity values. The number of clones generated from each selected antibody depends on the selection probability of the antibody. The number of clones of a selected antibody i is hereafter denoted as NC_i and computed as:

$$NC_i = \left\lfloor \frac{\beta N_s}{r_i} \right\rfloor, \tag{8}$$

where β is a multiplying factor (here β is set to $\beta = 2$), r_i is the rank of the selected antibody i, and N_s is the total number of selected antibodies. The function $\lfloor x \rfloor$ gives the value of the integer nearest to x. From (8) results a proliferation phenomena according to which for an antibody with high affinity value (i.e. with low rank) corresponds a high number of clones. Consequently, the selection and cloning process gives to solutions with low costs (high affinity value) a high chance of being selected and also to be cloned with a high number of clones.

4.4. Hypermutation operator

Once the cloning process is achieved, the hypermutation operation is carried out on clones. In the present paper, a two-phase hypermutation method is adopted. This method consists first an interchanging mutation followed by a scramble mutation. This later is performed only when no affinity improvement is obtained in the interchanging mutation. To illustrate, let us consider an arbitrary solution vector **s** from the clone population. The interchanging mutation consists to choose randomly two elements, say $\mathbf{s}(i)$ and $\mathbf{s}(j)$ of the solution vector **s**, and then to interchange their respective hubs' values. If the affinity value of the resulting new solution vector $\mathbf{s}'$ is better than that of the original solution **s**, then the antibody **s** is replaced by the new one

$\mathbf{s}'$. In case of no affinity value improvement, a scramble mutation is then performed on the original antibody $\mathbf{s}$. The scramble mutation consists to chose randomly a subset of elements and then to scramble or shuffle their respective hubs' values. Once again, if the affinity value of the resulting new solution improves that of the original solution, then the original antibody is replaced by the new one. If the affinity of the solution is not improved by both hypermutation phases, the original antibody string is conserved as is. Once the hypermutation process is accomplished, antibodies are sorted with respect to their respective affinity values.

4.5. Receptor editing operator

After the hypermutation process, new antibodies are introduced to ensure population diversity. Accordingly, a fraction of antibodies with the worst affinity values in the population are eliminated and replaced by randomly generated newcomers. The purpose of receptor editing operator consists to avoid pre-mature solution convergence by escaping from local optima.

4.6. Convergence and termination

When the maximum number of generations is reached, the convergence is thought to be attained and the corresponding solution can be considered as the best one, and the algorithm stops. In the contrary case, the search process is restarted from step 2 and the improvement process continues.

5. RESULTS AND DISCUSSION

In this section, computational results of the proposed CSA heuristic are provided and compared to the best results appeared in the literature. The clonal selection algorithm was implemented in MATLAB (2016a) and experiments are performed on a 2.00 GHz Intel Core IV PC, with 8 GB of RAM.

To compare our approach to the existing works, all experiments are conducted on four different sets of benchmark instances commonly used in the literature related to the HLP [4, 14]. The first is the well-known CAB data set provided by the US civil aeronautics board. The CAB data set considers 100 cities in the US and provides Euclidean distances $d_{ij}$and the amount of flows w_{ij} routed between each pair of cities (i, j). Dealing with CAB data set, three instances with $N \in 10, 20, 25$ nodes and up to $p = 4$ hubs are selected. For these instances, the collection and distribution costs rates are equally estimated to $\chi = \delta = 1$ while the transfer cost rate α takes values between 0.2 and 1 ($\alpha \in [0.2, 1]$). The second data set is established within the Australian postal (AP) delivery system where 200 postal districts in the metro Sydney area are considered. Transportation costs are proportional to the Euclidian distances d_{ij} between two postal districts i and j with a postal flow w_{ij}. Seven instances are also selected where the number N of nodes is such that $N \in 10, 20, 25, 40, 50, 100, 200$ nodes and up to $p = 20$ hubs. In the AP data set, collection, transfer and distribution costs rates are, respectively, estimated to $\chi = 3$, $\alpha = 0.75$ and $\delta = 2$. The third data set is PlanetLab instances which is a real world node-to-node delay data for performing internet measurements [18]. In these networks, $\alpha = \chi = \delta = 1$ and the distance matrix does not respect the triangle inequality due to missing links. The flow w_{ij} is equal to 0 if i = j and 1 otherwise. The fourth data set concerns the very large size instances with 1000 nodes of the Urand dataset which was generated by [18]. In these benchmarks, the nodes coordinates were randomly generated from 0 to 105 and the flow matrix was randomly generated.

The comparison of our approach against the existing works is made on the basis of both CPU time and solution quality criteria. Solutions quality comparisons is obtained by measuring the gap defined (in percent) as the absolute value of the ratio:

$$\text{Gap} = \frac{\text{Cost function obtained} - \text{Best exiting cost function}}{\text{Best exiting cost function}} \tag{9}$$

The Standard deviation of the average gap is also calculated:

$$\sigma = \sqrt{\frac{1}{20}\sum_{i=1}^{20}(Gap_i - aGap)^2} \tag{10}$$

In all experiments conducted hereafter, each test instance was launched 20 times and the elimination ratio is set at 30% in the Receptor Editing process. The search process is altered whenever the maximum

number of generations is reached. In what follows, this stoppage criterion is set at 500 and 1000 for small and large problem instances, respectively.

5.1. Experiments set #1: Case of small to moderate size problems

By using the optimal solution obtained from CPLEX solver as a reference, we will show that our heuristic based on CSA also provides optimal solutions within a low computation time. For varying values of the parameters α and that of the number p of hubs, the results obtained for CAB instances are reported in Tables 2 and 3. For varying values of the number p of hubs, the results obtained for AP instances are given in Tables 4 and 5. In these tables, $\mathbb{E}$ (CPU time) and $\mathbb{E}$ (Gap), refer, respectively, to the average CSA computation time, and the average gap value of the 20 runs performed.

From the results reported in Tables 2, 3, 4, and 5, one may observe that our heuristic method reaches all previously known optimal solutions for small to moderate instances. From these table, one may also observe that the computation time is very low and the proposed approach is capable of providing an optimal solution in less than 1 second for instances with $N = 50$ nodes. Moreover, the robustness of our method is confirmed as indicated by the low values of standard deviation. Now that the proposed approach has been validated against previously published works for small to moderate problem sizes, numerical experiments will be run for large size problems.

Table 2. Results of experiment #1: Case of CAB instances with $N = 20$

N	p	α	CPLEX	CSA	$\mathbb{E}$(CPU time)	$\mathbb{E}$(Gap)	σ
20	2	0.2	979.087		0.093	0.000	0.000
		0.4	1042.566		0.092	0.000	0.000
		0.6	1106.044	opt	0.096	0.000	0.000
		0.8	1169.523		0.092	0.000	0.000
20	3	1.0	1210.077		0.097	0.000	0.000
		0.2	724.538		0.151	0.000	0.000
		0.4	847.767		0.153	0.000	0.000
		0.6	970.996	opt	0.154	0.000	0.000
		0.8	1091.050		0.158	0.000	0.000
		1.0	1156.072		0.170	0.000	0.000
20	4	0.2	577.621		0.168	0.027	0.019
		0.4	727.099		0.177	0.000	0.000
		0.6	869.157	opt	0.180	0.000	0.000
		0.8	1008.492		0.200	0.000	0.000
		1.0	1111.015		0.222	0.000	0.000

Table 3. Results of experiment #1: Case of CAB instances with $N = 25$

N	p	α	CPLEX	CSA	$\mathbb{E}$(CPU time)	$\mathbb{E}$(Gap)	σ
25	2	0.2	1000.907		0.098	0.000	0.000
		0.4	1101.629		0.110	0.000	0.000
		0.6	1201.206	opt	0.119	0.000	0.000
		0.8	1294.085		0.123	0.000	0.000
		1.0	1359.190		0.130	0.088	0.047
25	3	0.2	767.349		0.188	0.000	0.000
		0.4	901.699		0.199	0.000	0.000
		0.6	1033.565	opt	0.201	0.000	0.000
		0.8	1158.831		0.220	0.000	0.000
		1.0	1256.630		0.227	0.000	0.000
25	4	0.2	629.634		0.230	0.000	0.000
		0.4	787.515		0.233	0.000	0.000
		0.6	939.206	opt	0.232	0.000	0.000
		0.8	1087.662		0.240	0.098	0.156
		1.0	1211.232		0.248	0.112	0.208

Table 4. Results of experiment #1: Case of AP instances with $N \in \{10, 20, 25\}$

N	p	CPLEX	CSA	$\mathbb{E}$(CPU time)	$\mathbb{E}$(Gap)	σ
10	2	167493.06	opt	0.044	0.000	0.000
	3	136008.13	opt	0.089	0.000	0.000
	4	112396.07	opt	0.130	0.000	0.000
	5	91105.37	opt	0.166	0.033	0.098
20	2	172816.69	opt	0.098	0.000	0.000
	3	151533.08	opt	0.139	0.022	0.068
	4	135624.88	opt	0.142	0.000	0.000
	5	123130.09	opt	0.199	0.020	0.125
25	2	175541.98	opt	0.120	0.000	0.000
	3	155256.32	opt	0.190	0.000	0.000
	4	139197.17	opt	0.210	0.000	0.000
	5	123574.29	opt	0.289	0.000	0.000

Table 5. Results of experiment #1: Case of AP instances with $N \in \{40, 50\}$

N	p	CPLEX	CSA	$\mathbb{E}$(CPU time)	$\mathbb{E}$(Gap)	σ
40	2	177471.67	opt	0.199	0.116	0.215
	3	158830.54	opt	0.299	0.000	0.000
	4	143968.88	opt	0.333	0.000	0.000
	5	134264.97	opt	0.457	0.000	0.000
50	2	178484.29	opt	0.366	0.000	0.000
	3	158569.93	opt	0.405	0.000	0.000
	4	143378.05	opt	0.499	0.000	0.000
	5	132366.95	opt	0.699	0.202	0.117

5.2. Experiments set #2: Case of large size problems

In this set of experiments, the proposed approach is applied in large size problems. We will also show that our heuristic based on CSA provides fast, good and robust solutions. The performance of the proposed heuristic will be compared to 6 existing heuristic methods: Simulated annealing method (SA) [15], the hybrid Variable neighbourhood search (VNS) and path re-linking (PR) heuristics [16], Genetic algorithm (GAHUB2) in [17], the general variable neighborhood search (GVNS) [18] and the parallel genetic algorithm on GPU (GPU) [19]. In the present experiments, instances are selected from the AP data set where the number of the nodes $N \in 100, 200$, as shown in Tables 6 and 7 and from PlanetLab data set with up to 414 nodes, see Tables 8 and 9.

Table 6. Results of experiments #2: Solution quality criteria (case of AP instances)

N	p	SA	VNS-PR	GAHUB2	GVSN	GPU	CSA
100	5	136969.44	136929.44	136929.44	136929.44	136929.44	**129020.22**
	10	106469.57	106469.57	106469.57	106469.57	106469.57	106469.57
	15	90605.10	90605.10	90533.523	90533.523	90533.523	**90126.122**
	20	80682.71	80682.71	80270.962	80270.962	80270.962	80270.962
200	5	140409.41	140318.36	140175.645	140062	140062	**139976.521**
	10	111088.33	110955.27	110147.657	110147.657	110147.657	110147.657
	15	95460.54	95111.06	94496.406	94459.201	94459.201	94459.201
	20	85560.39	85439.92	85129.343	84955.328	84955.328	84955.328

Table 7. Results of experiments #2: Average computation time $\mathbb{E}$ (CPU time) criteria (case of AP instances)

N	p	SA	VNS-PR	GAHUB2	GVSN	GPU	CSA
100	5	80.55	0.48	22.108	0.077	1.310	3.245
	10	161.86	0.91	40.733	0.666	1.310	3.245
	15	279.79	1.32	57.453	3.219	1.49	4.77
	20	522.70	1.92	79.200	3.572	1.63	4.98
200	5	399.73	0.86	169.733	5.157	3.602	6.435
	10	776.58	1.87	259.142	5.600	3.722	6.675
	15	1105.29	2.11	313.017	17.656	3.783	7.74
	20	1555.90	3.20	374.751	12.979	3.841	7.85

Table 8. Results of experiments #2: Solution quality criteria (case of PlanetLab instances)

Instance	N	p	GVSN	GPU	CSA
01-2005	127	12	2927946	2904434	2904434
02-2005	321	19	18579238	18329984	**18278594**
03-2005	324	18	20569390	20284132	20284132
04-2005	70	9	739954	730810	**720945**
05-2005	374	20	25696352	25583240	25583240
06-2005	365	20	22214156	22191592	22191592
07-2005	380	20	30984986	30782956	30782956
08-2005	402	21	30878576	30636170	**30387655**
09-2005	419	21	32959078	32649752	32649752
10-2005	414	21	32836162	32687796	32687796
11-2005	407	21	27787880	27644374	27644374
12-2005	414	21	28462348	28213748	28213748

Table 9. Results of experiments #2: Average computation time $\mathbb{E}$ (CPU time) criteria (case of PlanetLab instances)

Instance	N	p	GVSN	GPU	CSA
01-2005	127	12	148.9	0.5	1.87
02-2005	321	19	462.7	6.9	8.98
03-2005	324	18	543.8	7.5	8.65
04-2005	70	9	0.6	0.3	1.12
05-2005	374	20	622.6	8.3	12.54
06-2005	365	20	581.7	7.9	11.42
07-2005	380	20	546.6	8.5	12.54
08-2005	402	21	637.6	8.7	12.88
09-2005	419	21	684.9	9.3	13.23
10-2005	414	21	731.9	9.1	13.77
11-2005	407	21	588.3	9.2	12.75
12-2005	414	21	680.3	9.1	13.14

5.3. Experiments set #3: Case of very large size problems

We also compare our heuristic method with the GVNS and GPU on very large instances from Urand data set with 1000 nodes generated by [18], as shown in Tables 10 and 11.

Table 10. Results of experiments #3: Solution quality criteria

N	p	GVSN	GPU	CSA
1000	2	198071412.53	8184986.50	**7889215.78**
	3	169450816.35	7024184.00	7024184.00
	4	150733606.87	6184749.01	6184749.01
	5	142450250.26	5860994.06	**5018955.03**
	10	114220373.07	4752317.00	4752317.00
	20	92883250.98	3928617.48	3928617.48

Table 11. Results of experiments #3: Average computation time $\mathbb{E}$ (CPU time) criteria

N	p	GVSN	GPU	CSA
1000	2	1.7245	9.321	11.122
	3	8.1550	9.785	11.896
	4	2.2240	10.431	13.125
	5	58.6070	10.89	13.895
	10	187.8385	13.7	18.231
	20	403.4280	17.923	25.875

By analyzing the above results, we conclude that our CSA method is more robust than other related methods for large sized problems, see the number of times it reached the best solution. In regards to small sized problems, the results of our approach are similar to those existed in the literature, but when the problem size increases our proposed heuristic enhances the results of GVNS and GPU in terms of solutions quality. Moreover, we could improve the best-known solution for eight instances in total.

6. CONCLUSION

In this paper we develop a clonal selection algorithm for solving the Uncapacitated single allocation p-Hub median problem. The clonal selection algorithm mimics the clonal selection principles in improving the affinity of individuals through generations in order to achieve the near optimal solution. The computational results state that our CSA method is more robust than other related methods for large sized problems, see the number of times it reached the best solution. In regards to small sized problems, the results of our approach are similar to those existed in the literature, but when the problem size increases our proposed heuristic enhances the results of GVNS in terms of solutions quality. Moreover, we could improve the best-known solution for six instances in total. The main contributions of this paper can be summarized as: i) To our best knowledge, CSA is applied to the USApHMP for the first time; ii) A new different selection method "Stochastic universal sampling "is used which is a development of the roulette wheel selection strategy (RWS) usually used in the literature; iii) An appropriate encoding scheme is used, which keep the feasibility of solutions through generations; and iv) Quality solution was enhanced for six instances.

In the future, we will concentrate on extending our heuristic method on an other more complex hub location problem model, which captures real life circumstances more effectively. The model will take into account the perishability nature of transported products in the network, i.e., the products have to be routed to their customers through the hub network in a limited time.

REFERENCES

[1] M. Campbell, J. F. Ernst, and A. T., Krishnamoorthy, "Hub location problems," Springer-Berlin, 2002.

[2] I. Contreras, "Hub Location Problems," *Springer International Publishing*, 2015.

[3] M. O'Kelly, "A quadratic integer program for the location of interacting hub facilities," Euroean *Journal of Operational Research*, vol. 32, no. 3, pp. 339-404, 1987.

[4] R. Z. Farahani, M. Hekmatfar, A. B. Arabani, and E. Nikbakhsh, "Hub location problems: A review of models, classification, solution techniques, and applications," *Computers and Industrial Engineering*, vol. 64, no. 4, pp. 1096-1109, 2013.

[5] F. H. Boukani, B. F. Moghaddam, and M. S. Pishvaee, "Robust optimization approach to capacitated single and multiple allocation hub location problems," *Computational and Applied Mathematics* vol. 35, no. 1, pp. 45-60, 2016.

[6] S. Alumur and B. Y. Kara, "Network hub location problems: The state of the art," Euroean *Journal of Operational Research*, vol. 190, no. 1, pp. 1-21, 2008.

[7] S. A. Alumur, S. Nickel, B. Rohrbeck, and F. S. da gama, "Modeling congestion and service time in hub location problems," *Applied Mathematical Modelling*, vol. 55, pp. 13–32, 2018.

[8] Z. Kartal, S. Hasgul, and A. T. Ernst, "Single allocation p-hub median location and routing problem with simultaneous pick-up and delivery," *Transportation Ressearch: Part E*, vol. 108, pp. 141-159, 2017.

[9] G. Taherkhani and S. A. Alumur, "Profit maximizing hub location problems," *Omega*, vol. 86, pp. 1-15, 2019.

[10] M. Sadeghi, F. Jolai, and Y. Rahimi, "A new stochastic approach for a reliable p-hub covering location problem," *Computers and Industrial Engineering*, vol. 90, pp. 371-380, 2015.

[11] J. G. Klincewicz, "Avoiding local optima in thep-hub location problem using tabu search and grasp," *Annals of Operations Research*, vol. 40, no. 1, pp. 283–302, 1992.

[12] D. Skorin-Kapov and J. Skorin-Kapov, "On tabu search for the location of interacting hub facilities," European *journal of Operational Research*, vol. 73, no. 3, pp. 2502-509, 1994.

[13] J. G. Klincewicz, "Heuristics for the p-hub location problem," European *journal of Operational Research*, vol. 53, no. 1, pp. 25-37, 1991.

[14] S. Abdinnour-helm and M. A Venkataramanan, "Solution approaches to hub location problems," *Annals of Operations Research*, vol. 78, pp. 31–50, 1998.

[15] A. T. Ernst and M. Krishnamoorthy, "Efficient algorithm for the uncapacitated single allocation p-hub median problem," *Location Science*, vol. 4, no. 3, pp. 139-154, 1996.

[16] M. P. Perez, F. A. Rodrıguez, and J. M. Moreno-Vega, "A hybrid vns-path relinking for the p–hub median problem," *IMA Journal of Management Mathematics*, vol. 18, no. 2, pp. 157-171, 2007.

[17] J. Kratica, Z. Stanimirovic, D. Tosic, and V. Filipovic, "Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem," European *journal of Operational Research*, vol. 182,

no. 1, pp. 15-28, 2007.
[18] A. Ilic, D. Urosevic, J. Brimberg, and N. Mladenovic, "A general variable neighborhood search for solving the uncapacitated single allocation p-hub median problem," European *journal of Operational Research*, vol. 286, no. 2, pp. 289-300, 2010.
[19] A. Benaini, A. Berrajaa, J. Boukachour, and M. Oudani, "Solving the uncapacitated single allocation p-hub median problem on gpu," *Bioinspired Heuristics for Optimization, Studies in Computational Intelligence, N. A. Talbi EG., Ed. Springer, Cham*, 2019, pp. 27-42.
[20] J. Daudi, "An overview of application of artificial immune system in swarm robotic systems," *Advances in Robotics and Automation*, vol. 04, no. 1, pp. 11–18, 2015.
[21] L. N. de Castro and J. Timmis, "Artificial Immune Systems: A New Computational Intelligence Approach," *Springer-Verlag, New York*, 2002.
[22] L. N. de Castro and F. J. V. Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation*, vol. 06, no. 3, p. 239–251, 2002.
[23] C. Zheng-yi, Y. Xue-yang, L. Ya-lun, and Z. Si-feng, "Throughput optimization in cognitive wireless network based on clone selection algorithm," *Computers and Electrical Engineering*, vol. 52, pp. 328-336, 2016.
[24] F. Z. Grine, O. Kamach, and N. Sefiani, "A new efficient metaheuristic for solving the uncapacitated single allocation p-hub median problem," *International Colloquium on Logistics and Supply Chain Management (LOGISTIQUA)*, 2018, pp. 69-74.
[25] Z. Jieqiong, C. Yunfang, and W. Zhang, "A survey of artificial immune applications," *Artificial Intelligence Review*, vol. 34, no. 1, pp. 19-34, 2010.
[26] M. Y. Y. El-Sharkh, "Clonal selection algorithm for power generators maintenance scheduling," *International Journal of Electrical Power and Energy Systems*, vol. 57, pp. 73-78, 2014.
[27] L. N. D. Castro and F. V. Zuben, "The clonal selection algorithm with engineering applications," *Workshop on Artificial Immune Systems and Their Applications, Las Vegas, USA*, 2000, pp. 36-37.