Available online at http://jnrm.srbiau.ac.ir
Vol.1, No.4, Winter 2016
ISSN:1682-0169

Journal of New Researches in Mathematics

Science and Research Branch (IAU)

An Effective Algorithm in order to solve the Capacitated Clustering Problem

N. Mahmoodi Darani [a], P. Bassiri [b], M. Yousefikhoshbakht [*] [c]

(a) *Young Researchers and Elite club, Robatkarim Branch, Islamic Azad University, Robatkarim, Iran.*

(b) *Department of Mathematics, Payame Noor University, Tehran, Iran.*

(c) *Young Researchers & Elite Club, Hamedan Branch, Islamic Azad University, Hamedan, Iran.*

Received Autumn 2015, Accepted Winter 2016

Abstract

The capacitated clustering problem (CCP) is a data mining technique utilized to categorize a number of objects with known demands into k distinct clusters such that the capacity of each cluster is not violated, every object is allocated to exactly one cluster and sum of distances from all cluster centers to all other nodes is minimized. The CCP is an NP-hard combinatorial optimization problem. Therefore, practical large-scale instances of this problem cannot be solved by exact solution methodologies within acceptable computational time. Our interest was therefore focused on meta-heuristic solution approaches. For this reason, a modified imperialist competitive algorithm (MICA) is proposed for the CCP In this paper. The proposed MICA iterates steps between three basic phases, i.e., the random assignment phase to form clusters, the seed relocation phase to find a better median, and the local improvement phase to make a revision of the solution. The proposed algorithm is tested on several standard instances available from the literature. The computational results confirm the effectiveness of the presented algorithm and show that the proposed algorithm is competitive with other meta-heuristic algorithms for solving the CCP.

Keywords: Capacitated Clustering Problem, NP-hard Problems, Imperialist Competitive Algorithm, Swap Move, Insert Move.

---

*. Corresponding author: Email: khoshbakht@iauh.ac.ir & khoshbakht@iauh.ac.ir

(a) The authors would like to acknowledge the Young Researchers And Elite club, Robatkarim Branch, Islamic Azad University, Robatkarim, Iran for the financial support of this work.

1. Introduction

The Capacitated Clustering Problems (CCP) has been one of the most famous problems in operation research studied for many years. In this problem, a set of *n* node into *k* disjoint clusters with known uniform capacity is petitioned. During clustering the items with shortest assigning paths from centroids are grouped together. The summation of grouped items should not exceed the capacity of the cluster and each node is associated to only one cluster. In other words, the requesters are grouped based on their demands with optimal number of clusters and the minimum cost of each service delivery. Furthermore, most studies on the CCP were applied to facility location problems in several decades ago, but in recent years these studies have been often focusing on the development of solution algorithms clustering techniques used as essential tools to solve many practical problems. So, several variants of CCP have been considered in the recent literature such as a capacitated centered clustering problem (CCCP) [3], generalized assignment problem [4], and capacitated p-median problem (CPMP) [5]. Besides, clustering of related objects is of practical importance and can be found in diverse fields such as biology, economics, engineering, marketing, operations research, pattern recognition and statistics. The CCP is a very special case of the capacitated plant location problem with single source constraints, and many other combinatorial problems.

For example, Fisher and Jaikumar used the generalized assignment algorithm for vehicle routing ushered in the use of the CCP [5]. They used very simple heuristics to solve the generalized assignment problem which is a subset within the larger vehicle routing problem. The generalized assignment problem, however, requires the pre-specification of seed customers that serve as clustering points. In large problems, the choice of the seed customers can be extremely difficult. Algorithms for CCP, on the other hand, explicitly choose the seeds as part of the algorithm.

Mehrotra and Trick proposed a column generation with a specialized branching technique solved a maximum weighted cluster problem (MWCP) in the sub problem in 1988 [6]. Baldacci et al. Offered a new exact algorithm by modeling the capacity location problem as a set partitioning problem with cluster-feasibility constraints in 2002 [7]. A combination of the column generation and Lagrangean relaxation techniques was proposed by Lorena and Senne for solving the capacitated *p*-median problems [8]. Recently, Ceselli et al. Proposed a computational framework based on column generation and branch-and-price approaches for solving the capacitated network problems in 2009 [9]. Mulvey and Beck proposed classical sub-gradient heuristics and used randomly generated seeds as an initial solution for solving the CCP in 1984 [10] while in 1992 Koskosidis and Powell extended the work of [10] by an iterative algorithm [11]. This proposed algorithm was more effective than other heuristic algorithms and avoided the specification of seed nodes required by other algorithms. Besides, the iterative algorithms use self-correcting scheme in three phases including greedy assignment, seed relocation and local exchange. This iterative heuristic of is employed and compared with genetic algorithm (GA) for the following two reasons. First, the algorithm is based on the work of Fisher and Jaikumar [4] on the GAP and the work of Mulvey and Beck [10] on the CCP. Therefore, this algorithm can be a good representation of the typical heuristic algorithms. Second, the performance of the iterative heuristic is quite well, especially in the computing efficiency. It will be a good comparison to evaluate the performance of the GA. May [12] applied the iterative algorithm of Koskosidis and Powell with the geographical information system (GIS) to solve vehicle routing problems on an existing road network of Chung-Li City in the northern part of Taiwan.

Thangiah and Gubbi used a genetic sectoring method to find good clusters of customers for the vehicle routing problem with a "cluster-first route-second" problem solving strategy [13]. Once the clusters are identified by the genetic search, classical insertion and post-optimization procedures are applied to produce the chosen routes. The method used pseudo polar coordinate angles of the customers to divide the customers into clusters by planting a set of seed angles in the searching space and drawing a ray from the origin to each seed angle. Another genetic algorithm was also offered by How-Ming Shieh and May for solving CCP [14]. In this algorithm, binary coded strings are applied to represent the chromosome, which eliminates the occurrence of infeasible solution. Furthermore, the chromosome is separated into two parts for representing the nodes and the seeds of the clusters. Furthermore, a k-means algorithm which performs correct clustering without pre-assigning the exact number of clusters was introduced by Zalik [15]. On the other hand, Osman and Christofides [16] employed the hybrids of simulated annealing and tabu search methods to solve the CCP. Their test problems are published and available via the Internet, so their testing results may be used as the benchmarks to evaluate the performance of other algorithms. A new adaptive tabu search approach to solve the CCP is proposed by França et al. [17]. They used two neighborhood generation mechanisms of the local search heuristic including pairwise interchange and insertion. In this algorithm, an adaptive penalty function is used to handle the capacity constraint that improves the solution quality and convergence. Recently, the GRASP, tabu search and GRASP with Path Relinking are proposed by Deng and Bard (2011) [18], Gallego et al. (2013) [19], and Morán-Mirabal et al. (2013) [20] respectively.

The CCP belongs to non-deterministic polynomial time (NP-hard) problems [21]. So, it is hard to find the optimal solution for the larger the size of this problem. On the other hand, the ICA is a new and effective global search strategy that uses the social-political competition among empires as a source of inspiration. Since the ICA algorithm suffers from being trapped in local optima, this behavior has been introduced to overcome the local optima and speed up the convergence. Furthermore, the ICA has been considered by scientist and researchers nowadays and utilized for a lot of optimization problems. The results of using ICA show that this algorithm is competitive with other meta-heuristic algorithms. Therefore, in this paper, we propose a modified algorithm for solving the CCP based on an ICA (MGA). The experimental results on the standard benchmark problems compared with the other well-known algorithms, illustrate the robustness of the proposed algorithm.

The remainder of this paper is organized as follows: In Section 2, the proposed framework designed to solve the CPP is presented. In Section 3, computational experiments using real standard datasets is demonstrated, and shown the effectiveness by comparing the MGA to the existing methods. Finally, this paper concludes in Section 4.

2. The Proposed Algorithm

According to the complexity of the CCP, the exact optimization algorithms are unrealistic for large-scale problems and the heuristic algorithms are not able to escape from a sub-optimal point in most of the times. Moreover, the cost of delivery can be significantly reduced by using a software management without compromising the services provided to the requesters. On the other hand, clustering is a difficult combinatorial problem, and clustering algorithms can be hierarchical or partitioned. Hierarchical algorithm discovers successive clusters using previously established clusters, whereas partitioned algorithm defines all clusters at once. So, in this paper a hybrid meta-heuristic algorithm based on ICA is used to escape from the trap of local optima and finding better solutions. The proposed algorithm iterates steps between three basic phases, i.e., the random assignment phase to form clusters, the seed relocation phase to find a better median, and the local improvement phase to make a revision of the solution as shown in Figure 2. It is noted that, a set of seeds is necessary to initiate the randomly assignment phase. Inadequate initial set of seeds could generate lengthy clustering procedures and affect the quality of the final clustering solution. Therefore, efforts are needed to produce an adequate initial set of seeds.

In this section, at first the classic ICA is presented and then the MGA will be considered in more detail. The main contributions of the paper are as follows:

Presenting a modified ICA algorithm

Presenting a new algorithm called MGA for solving the CCP, which is equipped with diversification and intensification mechanisms

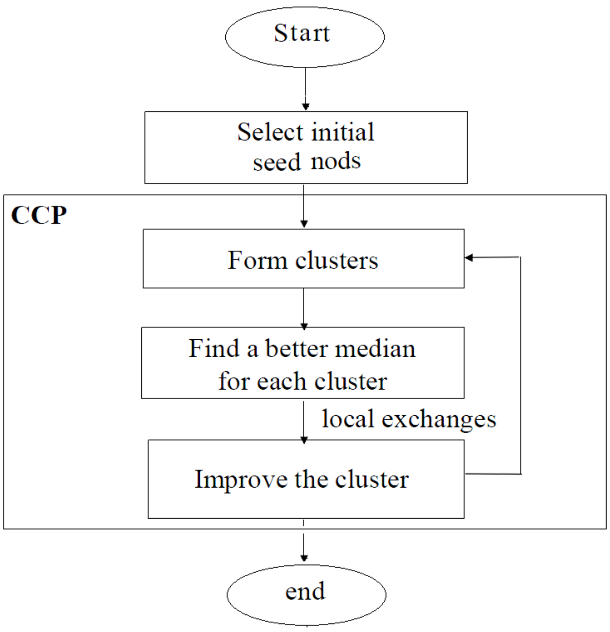

Figure 2. The levels of the propose algorithm

The MGA consists of two algorithms, including the ICA and some local search algorithms. It is noted that a construction algorithm like ICA produces a feasible solution while an improvement algorithm such as insert and swap exchanges can improve the solution produced by a constructive algorithm. In this section, the details of the proposed algorithm will be analyzed and its steps are described as follows.

At the first stage of the proposed algorithm, feasible solutions or primitive countries should be introduced in a way which is compatible with the construction of the mentioned problem. Therefore, an array shown in Figure 5 is used. In this array, the visited nodes are ordered from left to right in the first section and the center of each cluster is shown in the second section (where *n = 14* and *k = 3*). In this algorithm, only one kind of country is commonly employed for solving the CCP. In this technique, the *n* nodes are represented by a permutation of the integers from 1 to *n*. This permutation is partitioned into *k* cluster from 1 to *k*. In the example in Figure 5, the first cluster with center 13 contains nodes 1 and 9, the second cluster with center 6 contains nodes 10, 12, 11, 5, 4 and 2, and the third cluster with center 3 contains nodes 14, 7 and 8. When all of the clusters are obtained, the center of each cluster should be obtained. For this goal, all the nodes are tested as center and their function objective will be gotten.

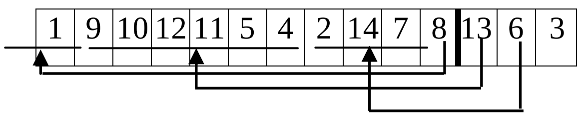

Figure 5. A Country in MGA

Therefore, a defined number of primitive solutions (*p*) must be randomly generated and imported in matrix D and the values of the objective function f_i for each $i = 1,..., p$ must be obtained. It is noted that using a random construction at this level leads to obtaining solutions which have an irregular construction in feasible space. Then, *m* countries that have better objective function are selected and are called empire countries. In this problem, we propose the formula (1) to randomly devote the number of colonies to each empire country. It should be noted that this formula leads to allocate more colonies are allocated to empires with better objective functions. The Int function used in the formula (1) is the floor function that causes the allocation of an integer number of colonies to each empire. If some countries do not belong to any empire due to the property of the Int function, they are allocated to the most powerful empire. After the empires are formed, each empire increases its quality using the imperialist countries which play local optima role. Here, we use an absorption function which includes randomization concept so that the results of combinations will not yield a very similar response. To achieve this goal, we have utilized a novel and innovative method.

$$k_j = \mathrm{Int}[(1 - \frac{(m-1)f_j}{\sum_{i=1}^{m} f_i}) \times (p-m)] \quad (1)$$

As an example, first, two possible responses [7 2 8 2 3 1 6 | 4 5] and [8 7 6 5 3 1 4 | 2 8] are considered as imperialist and colony respectively. In these countries, it is assumed that there are two clusters for everyone and [7 2 8] and [8 7 6 5] belong to the first cluster in both. Then, a random number between 1 and k (number of clusters) for each imperialist is selected and this cluster is replaced in the same cluster in the its colony. After that, extra nodes are eliminated. For example, in the above instance, if 1 is randomly selected, the obtained solution is [8 7 6 5 3 1 4 7 2 8| 2 8 4]. Then, the cluster with center 2 is deleted and the solution is [8 7 6 5 7 2 8| 2 4]. In this solution 7 and 2 are iterated two times and replaced with 1 and 3. So, the answer is [8 7 6 5 1 3 8| 2 4]. If this solution is not feasible based on its capacity, the swap and insert algorithms are considered until the feasible solution will be obtained. The absorption function is performed for all colonies in comparison to imperialist countries and the results and the values are replaced with the best results and the values obtained in the current iteration if the new results are better. At the next stage, p percent of countries experience a revolution. This causes variations in colonies in each empire and if possible their quality increases at each stage. The proposed methods for this stage are the insert move and, the swap move with variable probability. In the insert move, a candidate node is removed from its origin cluster and is inserted in the best position in another cluster. The replacement is done if the new results are better than the previous ones. On the swap move, two nodes from the two different clusters are randomly selected and exchanged with each other. It should be noted that a move which satisfies the problem's constraints is acceptable. Besides, the center is changed for each cluster in this algorithm. For example, if *k* is a center of the cluster *i*, the probability of selecting *j* as a new center is calculated by the formula (2). This formula leads to allocating a new node to the center of the cluster with lower cost in CCP. In this formula, c_{ij} is the distance between node *i* and node *j*.

$$k_j = \frac{1/c_{ij}}{\sum_{i=1}^{r} 1/c_{ij}} \qquad j = 1,\ldots,r \qquad (2)$$

Most successful meta-heuristic methods have paid attention to global search and search in the whole solution space as far as possible. As the algorithm proceeds, it moves to better solutions and the global search switches to a local search. We have factored in this issue too, and have represented the probability of insert and swap moves with $\alpha\, and\, \beta$ respectively so that $\alpha + \beta = 1$. Note that although swap move local search as a powerful global search algorithm is more used at the beginning of algorithm for global search, insert exchange is more applied at the end of the algorithm because these algorithms might lead to premature convergence to suboptimal regions. In other words, before the algorithm finishes a complete global search, it tends to adopt a local search and consequently relatively weak results are attained. Therefore, whenever the algorithm continues, the probability of α decreases and the probability of β increases. Adding this behavior to the imperialist algorithm revolution policy leads to creating the proper conditions for the algorithm to escape from local peaks. Thus, as mentioned before, the probability of using the insert and swap exchanges at the first step of the proposed algorithm are considered $\alpha = 0.30\, and\, \beta = 0.70$ and then during the steps of the proposed algorithm, they will be gradually converted to $\alpha = 0.70\, and\, \beta = 0.30$.

After the results and the values are calculated for all colonies, these countries might have a better objective function compared to their respective imperialists. Therefore, a colony with the best value in each empire is chosen and if it possesses a better objective function it replaces an imperialist country. In case there are a few colonies with the same objective function, one of them is chosen randomly and is compared with an imperialist country in the empire. From this stage up to the end of the algorithm, there will not be any change in objective functions of feasible values. Therefore, the best results and values of the objective function must be saved. For this purpose, two variables are chosen in order to save the best results and values until the current iteration. In each iteration after the imperialist countries were replaced, the best results and values for the imperialist countries are chosen as the best current results. Up to this stage in the algorithm, the purpose is conducting a general search to locate important areas for algorithm convergence. Now, important areas must be identified and the population must be converged toward them. After this stage, some of the initial population moves toward these areas. The power of empires is assessed at this stage. In imperialist competitions, more powerful empires must expand their territory through occupying other countries. In order to achieve this goal, the power of the empire is calculated using formula 2.

$$h_j = f_j + \lambda(s_j) \qquad j = 1,\dots,m \qquad (3)$$

In this formula h_j, s_j, and λ represent empire's total power, the average objective function of the colonies in each empire, and the [0 1] impact coefficient, which determines the relative power of a colony compared to an empire, respectively. A weaker empire loses its power by losing its weakest colony to the strongest empire. At this stage, the final condition is checked and if it is met, the algorithm ends. Otherwise, the algorithm is iterated by returning to absorption function step. To end the loop, two conditions must be met: the iteration of algorithm n times or the survival of just one empire. These conditions are checked at the end of each algorithm iteration. If any one of the conditions is met, the algorithm ends and the obtained results and values up to now are considered as the best values and results of the algorithm. The main steps of the proposed algorithm are summarized in the pseudo-code given in Figure 6.

Step 1: Generate some random solutions of the CCP and initialize the empires.

Step 2: Move the colonies toward their relevant imperialist by the proposed absorption function.

Step 3: Change p percent of colonies by the insert and swap moves (Revolution).

Step 4: If there is a colony in an empire which has better cost than the imperialist, exchange the positions of that colony and the imperialist.

Step 5: Compute the total cost of all empires by formula (3).

Step 6: Pick the weakest colony from the weakest empires and give it to the best empire (Imperialistic competition).

Step 7: Eliminate the powerless empires.

Step 8: If the quality of the best solution (s) is increased in this iteration, save the best so far solution.

Step 9: If stop conditions satisfied, stop, if not go to 2.

Figure 6. The process of MICA for solving the CCP

3. Computational Results

In this section, first, the parameter setting on a special benchmark is shown and then some numerical results of comparison between the proposed algorithm and some algorithms are given. These algorithms are applied and tested on several standard benchmarks with sizes ranging from 50 to 100 nodes. Because the proposed approach is a metaheuristic algorithm, the results are reported for ten independent runs in which the algorithm was executed until the stop condition of the algorithm is satisfied. The algorithms are coded by *C* language and implemented on a laptop at 2.6GHZ (4GB RAM).

The meta-heuristic approaches have many parameters that guide the search and consequently influence the method's performance. To obtain the best possible performance on a given problem, one should consider a task specific tuning of the parameter setting for the optimization method used. Determining the optimal parameter setting is an optimization task in itself, which is extremely computationally expensive. There are two common approaches for choosing parameter values [30]: *parameter tuning* and *parameter control*. The first approach selects the parameter settings before running the optimization method (and they remain fixed while performing the optimization). The second approach optimizes the method's parameters along with the problem's parameters. A detailed discussion and survey of parameter tuning methods is given by Eiben and Smit [30], who identify *sampling* methods as one type of parameter tuning methods. Sampling methods reduce the search effort by decreasing the number of investigat parameter settings as compared to the full factorial design. Two widely used sampling methods are Latin-squares and Taguchi orthogonal arrays (appropriate references are given by Eiben and Smit [30]).

In this paper, parameter tuning for the proposed optimization method was performed. So, A number of different alternative values were tested and the ones selected are those which yielded the best computational results concerning both the quality of the solution and the computational time needed to achieve this solution. In this algorithm, three important parameters including P, m and λ exist and the values of those directly or indirectly affect the performance of the algorithm. For each of the benchmark instances, ten different runs with the selected parameters were performed after the selection of the final parameters. In general, it is not easy to obtain the best combination of algorithm parameters, but a parameter setting procedure is necessary to reach the best balance between the quality of the solutions obtained. These parameters are selected in this paper after thorough testing. All of the parameter values have been determined on the P19 by the numerical experiments such that several alternative values for each parameter were tested while all the other values were held constant. It should be noted that only parameters which gave the best computational results concerning the quality of the solution were selected.

The ranges of the three parameters of the proposed algorithm are considered in Table 1 and the best solution is reported in Figure 7. In this figure, the horizontal axis shows the range of parameters and the vertical axis indicates the Gap of these algorithms. The Gap is computed by using formula (4) where $c(s^{**})$ is the best solution found by each algorithm for a given instance, and $c(s^{*})$ is the overall BKS for the same instance on the Web. A zero gap indicates that the best known solution of instance is found by the algorithm.

$$Gap = \frac{c(s^{**}) - c(s^{*})}{c(s^{*})} \times 100 \quad (4)$$

Table 1: Range of parameters of the proposed algorithm

Candidate Value	Description	Parameter
[100, 150, 200, 250, 300, 350, 400, 450, 500]	Number of primitive countries	p
Int [p/2, p/3, p/4. p/5, p/6, p/7, p/8,p/9,p/10]	Number of Imperialist countries	m
[0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5]	The impact coefficient, which determines the relative power of a colony compared to an empire	λ

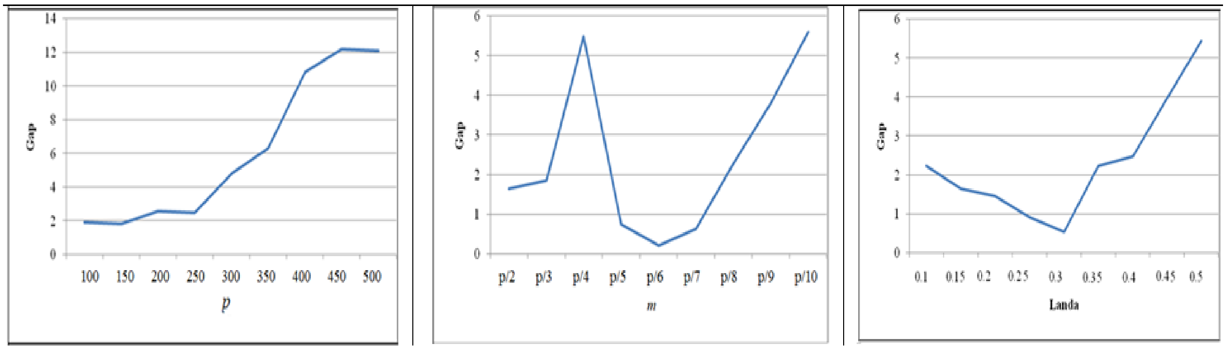

Figure 7. Parameters tuning of the proposed algorithm

As mentioned above, to determine the value of parameters several alternative values for each parameter were tested while all the others were held constant, and the ones that were selected gave the best computational results concerning the quality of the solution. The best solutions for this algorithm are P=150, m=p/6 and λ =0.3. Although the results confirm that our parameter setting worked well, it is also possible that better solutions could exist.

In this section, the experimental results of the proposed algorithm on a set of benchmark problems are shown and then the results obtained from calculations by the proposed algorithm are compared with other algorithms. Since there are few standard existing benchmarks for CCP especially in small size, this study generated some small test problems. All the examples are randomly located over a square with no route length restrictions. Therefore, a new set consisting of five small tests numbered from A1 to A5 with sizes ranging from 15 to 35 nodes including the depot were considered. Because the commercial linear programming software including ILOG and Cplex could find optimal solutions for the small-scale of the problems, like CCP and hence can be used to evaluate the accuracy of the proposed model. Therefore, we show the characteristics of instances and the results obtained by AIMMS and the proposed algorithm on the set of benchmark instances in Table 2. The information in this table is consisting of the number of nodes, the number of clusters, the solution costs obtained from AIMMS, the running time in seconds of AIMMS, the best solution costs of MGA and its computing time. Based on this table, AIMMS obtained the optimal solution only for instances A1, A2 and A3 and in the other instance automatically terminated before reaching to optimal solution. The results show that The MGA algorithm produced the optimal solutions for three out of the five problem instances in a reasonable time. Furthermore, this algorithm can obtain better solutions than AIMMS in A4 and A5 and obtain equal solutions with AIMMS for A1 and A2 and A3. Finally, the proposed algorithm in average improves the solution cost as much as 1.79% of these instances compared to AIMMS. For example, the Figure 8 shows the problem A1 and its optimal solution obtained by the proposed algorithm.

Table 2: Comparison results for MICA and the exact algorithm

Instance	n	m	AIMMS Cost	Time (Sec)	MICA Cost	Time (Sec)
A1	10	3	73.23	12	73.23	0.78
A2	20	5	84.32	78	84.32	1.24
A3	25	5	127.62	126	127.62	1.11
A4	30	7	147.32	323	142.62	1.26
A5	35	9	203.34	543	192.43	1.73

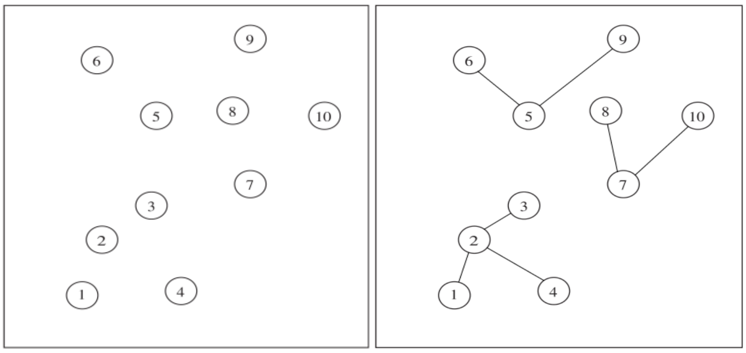

Figure 8. The optimal solution for A1

The set of standard benchmark problem instances contains between 50 and 100 nodes of the CCP problem, including P1, P2,…, P20. The first set contains 10 problem instances of size n=50 and p=5 (P1 to P10), and the second set contains 10 problem instances of size n=100 and p=10 (P11 to P20). All CCP problem instances are obtained from the standard library and because the proposed approach is a meta-heuristic algorithm, the results are reported for ten independent runs. Table 3 shows the results of the proposed algorithm for the CCP benchmark problem instances. In this table, Columns 2-6 show the problem size n, the number of clusters, the best value result of the PA (BVR), the worst value result of the PA (WVR), the average value result of the PA (AVR), Time of best value result of MICA and the best known solutions by other algorithms (BKS) over the ten runs for each problem.

Table 3. Comparison results for problem set

Instance	n	p	BVR	WVR	AVR	Time of BVR	BKS
P 1	50	5	713	713	713	2.34	713
P 2	50	5	740	740	740	2.21	740
P 3	50	5	751	751	751	3.12	751
P 4	50	5	651	651	651	2.13	651
P 5	50	5	664	664	664	2.98	664
P 6	50	5	778	778	778	3.14	778
P 7	50	5	787	787	787	3.12	787
P 8	50	5	820	820	820	2.02	820
P 9	50	5	715	715	715	2.84	715
P 10	50	5	829	834.12	832.1	3.82	829
P 11	100	10	1006	1010	1007	5.14	1006
P 12	100	10	966	971	969	5.25	966
P 13	100	10	1026	1026	1026	4.95	1026
P 14	100	10	982	984	983	6.23	982
P 15	100	10	1091	1095	1094	6.83	1091
P 16	100	10	954	961	955	5.99	954
P 17	100	10	1034	1044	1037	5.83	1034
P 18	100	10	1043	1043	1043	5.43	1043
P 19	100	10	1031	1039	1037	5.88	1031
P 20	100	10	1005	1005	1005	5.23	1005

It can be seen from Table 3, that the proposed algorithm has reached the best known solution in 20 out of the 20 instances. Also, in this Table the computational time needed (in seconds) for finding the best solution by the proposed algorithm is presented. The CPU time needed is significantly low and is very efficient. These results denote the efficiency of the proposed algorithm.

Generally, meta-heuristic solutions tend to be better than the heuristic solutions. Therefore, in Table 4, the efficiency and performance of the proposed algorithm are compared with the following meta-heuristic algorithms given in the literature for the CCP.

P, S and P+S algorithms →
proposed by Juan & Fernandez [31]
VNS algorithm →
proposed by Fleszar & Hindi [32]
FNS algorithm →
proposed by Kaveh et al. [33]
BKS →
the best know solutions published in the related literature

Besides, as mentioned before, n and p show the number of nodes and the number of clusters respectively. It should be noted that reported results for each instance is the best one over multiple runs. By comparing the results in this table, the proposed algorithm can reach 10 out of ten BKSs for all problems with size n=50 and P=5. Furthermore, %100 of the BKSs for instances with n=100 and P=10 are obtained by the proposed MICA algorithm. Although the PR, SS and PR+SS algorithms could not find the optimal solution in P8 and P10 with size n=50, P=5 in all 10 runs, the proposed algorithm is able to find the optimal solution to these problems. Thus, these algorithms have a weaker performance than the MICA in these instances. Furthermore, VNS and FNS find optimal solutions for all of P1 to P10 problems and they have zero Gap for all problems. Therefore, in the first instances set, the Gap of the results of the proposed algorithm from the BKS is zero percent.

Table 4. Comparison results for metaheuristic algorithms

Instance	N	p	Runs	P	S	P+S	VNS	FNS	MICA	BKS
P 1	50	5	10	713	713	713	713	713	713	713
P 2	50	5	10	740	740	740	740	740	740	740
P 3	50	5	10	751	751.2	751	751	751	751	751
P 4	50	5	10	651	651	651	651	651	651	651
P 5	50	5	10	664	664	664	664	664	664	664
P 6	50	5	10	778	778	778	778	778	778	778
P 7	50	5	10	787	787	787	787	787	787	787
P 8	50	5	10	821.1	820.9	820.9	820	820	820	820
P 9	50	5	10	715	715	715	715	715	715	715
P 10	50	5	10	831.4	831.7	831.4	829	829	829	829
P 11	100	10	10	1006.3	1006	1006	1006	1006	1006	1006
P 12	100	10	10	966	966	966	966	966	966	966
P 13	100	10	10	1026	1026	1026	1026	1026	1026	1026
P 14	100	10	10	983.7	984.2	983.7	982	982	982	982
P 15	100	10	10	1092.8	1093.2	1092.2	1091	1091.3	1091	1091
P 16	100	10	10	954.1	954	954	954	954	954	954
P 17	100	10	10	1034.5	1034	1034	1034	1035.4	1034	1034
P 18	100	10	10	1045	1043	1043.2	1043	1043	1043	1043
P 19	100	10	10	1032.1	1032.3	1032	1031	1031	1031	1031
P 20	100	10	10	1008.6	1006.5	1006	1005	1005	1005	1005

In the second set, there are 10 problem instances of size n=100 and p=10 (P11 to P20). In these instances, the Obtained results show that the proposed algorithm has better performance than PR, SS and PR+SS algorithms. In more details, the proposed algorithm is able to find the optimal solution in 10 runs, which is better than obtained solutions from PR, SS and PR+SS algorithms. On the other hand, Comparison of the FNS and the PA results show that, the proposed algorithm had almost equally resulted except in both P15 and P18 problems. In these problems, the proposed algorithm can obtain much better solutions than the NS. Finally, a comparison between the proposed algorithm and VNS shows that the quality of obtaining solutions of both algorithms are same. Due to each considered algorithms in this table was encoded in different softwares and different computers were used for performing computational experiments, so comparing their CPU times are not considered.

Finally, the proposed MICA has been tested on the large size of instances. This set categories in two groups which each one contains 10 instances and were taken from [7]. They have the same structure as the problems from [16] but are of bigger dimension. For set 1 (problems 21–30), n = 150 and p = 15 and for set 2 (problems 31–40), n = 200 and p = 20. All instances are considered planar and use Euclidean distances. Table 5 shows the results for problem sets 1 and 2. For these test problems, results for the approaches of Maniezzo et al. [34] (MMB), Baldacci et al. [7] (BHMM), Pirkul [35] (P) and Scheuerer and Wendolsky (SW) [36] are compared to the proposed algorithm. Furthermore, the CPU time in second for the proposed algorithm are shown in the seventh column of the table. Note that in our computations, the distances were rounded down to the nearest integer. Again, both combination strategies performed excellent.

Table 5. Comparison results for metaheuristics for the large problems

ID	BHMM	P	MMB	SW	MICA	Time of MICA	BKS
21	1290	1290	1289	1283	1283	7.87	1283
22	1291	1291	1291	1291	1291	7.24	1291
23	1220	1220	1219	1212	1212	8.34	1212
24	1236	1236	1235	1235	1235	8.65	1235
25	1188	1188	1188	1188	1188	8.93	1188
26	1227	1227	1227	1227	1227	8.51	1227
27	1269	1270	1269	1269	1269	7.81	1269
28	1180	1181	1180	1180	1180	7.95	1180
29	1260	1260	1259	1263	1261	9.73	1259
30	1243	1243	1242	1242	1242	9.24	1241
31	1446	-	1446	1446	1446	19.32	1446
32	1352	-	1351	1351	1351	19.82	1351
33	1390	-	1390	1390	1390	19.73	1390
34	1395	-	1394	1393	1391	19.19	1391
35	1401	-	1400	1398	1398	21.34	1398
36	1384	-	1383	1382	1382	21.83	1382
37	1399	-	1398	1385	1385	21.26	1385
38	1462	-	1461	1461	1459	20.53	1459
39	1427	-	1426	1426	1426	20.63	1426
40	1392	-	1392	1392	1392	20.78	1392

As it can be seen from this table, the best results by MICA are in bold. The experimental results show that the proposed method generally is better than the results of BHMM and P. For the instances 21, 23, 24, 29, 30, 32, 34, 35, 36, 37, 38 and 39, the algorithm of BHMM, and for the instances 21, 22, 24, 27, 28, 29 and 30, the algorithm of P cannot find the BKS shown in the literature, whereas the proposed method can find the BKS except for 31. When considered in details, the results of this comparison show that the proposed algorithm gains better solutions than the MMB in 8 out of 20 instances. Furthermore, the results indicate that although the SW produces solutions equal to the proposed algorithm for 17 instances, this algorithm cannot maintain this advantage in the other examples including 29, 34 and 38. So, Computational results of the MICA and GSAP show that these algorithms have a close competition, but the proposed algorithm produces better solutions more than SW. Overall, the proposed algorithm finds the optimal solution for 18 out of 20 problems published in the literature and gains nearly the BKS for instances 30 and 31. The average gap between the best solution of the proposed algorithm and the BKSs of the literature solutions is 0.012%. Therefore, the results indicate that the proposed algorithm is a competitive approach compared to the famous meta-heuristics.

Figure 9 shows the comparison between the gap values of the meta-heuristic algorithms for table 5. In this figure, the horizontal axis shows the names of the instances and the vertical axis shows the obtained Gap by each algorithm. As mentioned before, A zero gap indicates that the BKS is found by the algorithm.

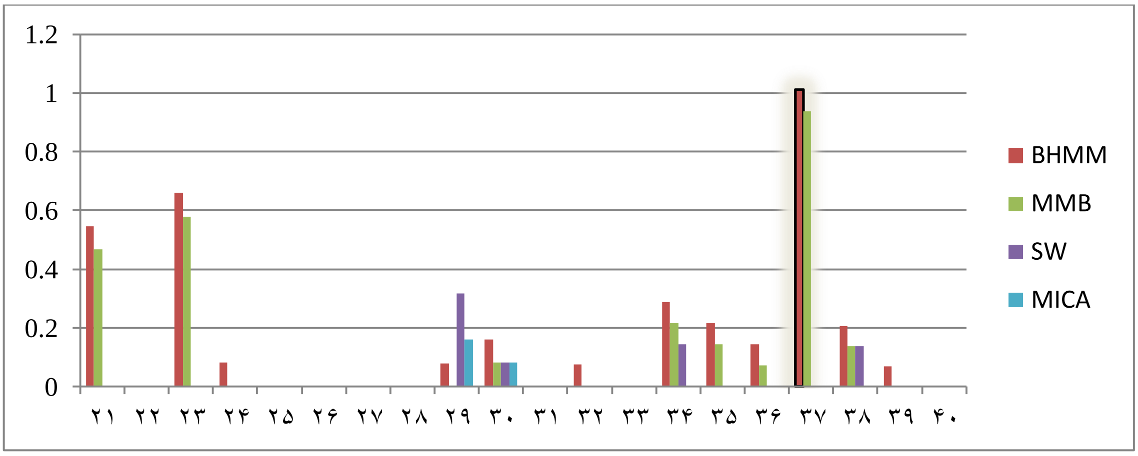

Figure 9. Comparison between results of metaheuristics algorithms

4. Conclusions

The CCP is a classic location problem with various applications in many practical situations. Applications of the CCP arise in a lot of problems for example in the context of vehicle routing [37], political districting [38] or sales force territories design [39]. So, a modified version of the ICA called MICA for solving the CCP was proposed in this paper The ICA is one of the new methods, which has great abilities to cope with different types of optimization problems. Several standard benchmark instances to evaluate the performance of the proposed algorithm are used and the experimental results show that the MICA confirms the effectiveness of the several approaches and finds optimal or nearly optimal solutions for allocating n objects into k clusters. It seems that the combination of the proposed algorithm with tabu search will yield better results for large problems of CCP. This will be more effective in improving available solutions and avoiding of local optimal solutions. Applying this method in other versions of the CCP, vehicle routing problem, School bus routing problem and the sequencing of jobs are suggested for future research studies.

References

[1] Baldacci, R., Hadjiconstantinou, E. and Mingozzi, A. "An exact algorithm for the capacitated vehicle routing problem based on a two commodity network flow formulation," Operations Research, 2004.

[2] Ralphs, T.K., "Parallel branch and cut for capacitated vehicle routing," Parallel Computing, 2003.

[3] Negreiros, M. and Palhano, A. "The capacitated centered clustering problem," Computers and Operations Research, 2006.

[4] Fisher, M. L. and Jaikumar, R. "A generalized assignment heuristic for vehicle routing," Networks, 11, 109-124, 1981.

[5] Chaves, A. A., de Assis Correa, F. and Lorena, L.A.N. "Clustering search heuristic for the capacitated p-median problem," Advances in Software Computing Series, 2007.

[6] Mehrotra, A. and Trick, M. A. "Cliques and clustering: A combinatorial approach," Operations Research Letters, 1998.

[7] Baldacci, R., Hadjiconstantinou, E., Maniezzo, V. and Mingozzi A., "A new method for solving capacitated location problems based on a set partitioning approach," Computers & Operations Research, 2002.

[8] Lorena, L.A. and Senne, E.L. "A column generation approach to capacitated p-median problems," Computers & Operations Research, 2004.

[9] Ceselli, A., Liberatore, F. and Righini, G. "A computational evaluation of a general branch-and-price framework for capacitated network location problems," Annals of Operations Research, 2009.

[10] Mulvey, J. and Beck, M.P. "Solving capacitated clustering problems," European Journal of Operational Research, 1984.

[11] Koskosidis, I. and Powell, W.B. "Clustering algorithms for consolidation of node orders into vehicle shipments," Transportation Research Part B, 1992.

[12] May, M. D., GIS-Aided Real-Time Vehicle Dispatching System for a Physical Distribution Center, Master Thesis, Department of Civil Engineering, National Central University, Chung-Li, Taiwan, 1994.

[13] Thangiah, S. R. and Gubbi, A.V. "Effect of genetic sectoring on vehicle routing problems with time windows," IEEE International Conference on Developing and Managine Intelligent System Projects, 1993.

[14] Shieh, H. M. and May, M. D. "Solving the capacitated clustering problem with genetic algorithms," Journal

of the Chinese Institute of Industrial Engineers, 2001.

[15] Zalik, K. R. "An efficient k-means clustering algorithm," Pattern Recognition Letters, 2008.

[16] Osman, I. H. and N. Christofides, "Capacitated clustering problems by hybrid simulated annealing and tabu search," International Transactions in Operational Research, 1, 317-336, 1994.

[17] Franca, P. M., Sosa, N. M. and Pureza, V. "An adaptive tabu search algorithm for the capacitated clustering problem," International Transactions in Operational Research, 2006.

[18] Deng, Y. and J.F. Bard, "A reactive GRASP with path relinking for the capacitated clustering,"Journal of Heuristics, vol. 17, pp. 119-152, 2011.

[19] Gallego, M., M. Laguna, R., Martí, and A. Duarte, "Tabu search with strategic oscillation for the maximally diverse grouping problem," Journal of the Operational Research Society, vol. 64, no. 5, pp. 724–734, 2013.

[20] Morán-Mirabal, L. F., J. L. González-Velarde, M. G. C. Resende, and R. M. A. Silva, "Randomized heuristics for handover minimization in mobility networks," Journal of Heuristics, vol. 19, pp. 845-880, 2013.

[21] Garey, M. R. and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.

[22] Sepehri Rad, H., Lucas, C., 2008, Application of imperialistic competition algorithm in recommender systems. In 13th international CSI computer conference (CSICC'08), Kish Island, Iran.

[23] Atashpaz-Gargari, E., Lucas, C., 2007, Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition, IEEE Congress on Evolutionary Computation (CEC 2007), 4661-4667.

[24] Atashpaz-Gargari, E., Hashemzadeh, F., Rajabioun, R. and Lucas, C., 2008, Colonial Competitive Algorithm, a novel approach for PID controller design in MIMO distillation column process, International Journal of Intelligent Computing and Cybernetics 1 (3), 337–355.

[25] Rajabioun, R., Atashpaz-Gargari, E., and Lucas, C., 2008, Colonial Competitive Algorithm as a Tool for Nash Equilibrium Point Achievement, Lecture notes in computer science 5073, 680-695.

[26] Biabangard-Oskouyi, A., Atashpaz-Gargari, E., Soltani, N., & Lucas, C., 2009, Application of imperialist competitive algorithm for materials property characterization from sharp indentation

test, International Journal of Engineering Simulation.

[27] Kaveh, A., Talatahari, S., 2010, Optimum design of skeletal structures using imperialist competitive algorithm, Computers and Structures 88, 1220–1229.

[28] Lucas, C., Nasiri-Gheidari, Z., Tootoonchian, F., 2010, Application of an imperialist competitive algorithm to the design of a linear induction motor, Energy Conversion and Management 51 (2010) 1407–1411.

[29] Shokrollahpour, E., Zandieh, M., Dorri., B., A novel imperialist competitive algorithm for bi-criteria scheduling of the assembly flowshop problem, International Journal of Production Research, 1–17, 2010.

[30] Eiben AE, Smit SK: Parameter tuning for configuring and analyzing evolutionary algorithms. Swarm and Evolutionary Computation, 1, 19-31, 2011.

[31] Juan, D. and Fernandez, E. “Hybrid scatter search and path relinking for the capacitated p-median problem”, European Journal of Operational Research, 2006.

[32] Fleszar, K. and Hindi, K.S. “An effective VNS for the capacitated p-median problem”, European Journal of Operational Research, 2008.

[33] Kaveh, P. Sabzevari Zadeh A. and Sahraeian, R. “Solving Capacitated P-median Problem by Hybrid K-means Clustering and FNS Algorithm,” International Journal of Innovation, Management and Technology, 2010.

[34] Maniezzo, V. Mingozzi, A. and Baldacci, R. “A bionomic approach to the capacitated p-median problem”, Journal of Heuristics, 4 (3), 263–280, 1998.

[35] Pirkul, H. “Efficient algorithms for the capacitated concentrator location problem”, Computers & Operations Research, 14 (3), 197–208, 1987.

[36] Scheuerer, S. and Wendolsky, A R. “Scatter search heuristic for the capacitated clustering problem”, European Journal of Operational Research, 169, 533–547, 2006.

[37] Koskosidis, Y.A. and Powell, W.B. Clustering algorithms for consolidation of customer orders into vehicle shipments,Transportation Research B, 26 (5), 365–379, 1992.

[38] Bozkaya, B., Erkut, E. and Laporte, G. “A tabu search heuristic and adaptive memory procedure for political districting”, European Journal of Operational Research, 144 (1), 12–26, 2003.

[39] Mulvey, J. M. and Beck, M.P. ‘Solving capacitated clustering problems”, European Journal of Operational Research, 18 (3), 339–348, 1984.