Engineering Applications of Artificial Intelligence 24 (2011) 306–317

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

An efficient hybrid algorithm based on modified imperialist competitive algorithm and K-means for data clustering

Taher Niknam [a,*], Elahe Taherian Fard [b], Narges Pourjafarian [b], Alireza Rousta [a]

[a] Electronic and Electrical Department, Shiraz University of Technology, Modars Blvd., P.O. 71555-313, Shiraz, Iran
[b] Shiraz University, Shiraz, Iran

ARTICLE INFO

Article history:
Received 9 January 2010
Received in revised form
21 June 2010
Accepted 5 October 2010
Available online 12 November 2010

Keywords:
Imperialist competitive algorithm (ICA)
Data clustering
K-means clustering
Hybrid evolutionary algorithm

ABSTRACT

Clustering techniques have received attention in many fields of study such as engineering, medicine, biology and data mining. The aim of clustering is to collect data points. The K-means algorithm is one of the most common techniques used for clustering. However, the results of K-means depend on the initial state and converge to local optima. In order to overcome local optima obstacles, a lot of studies have been done in clustering. This paper presents an efficient hybrid evolutionary optimization algorithm based on combining Modify Imperialist Competitive Algorithm (MICA) and K-means (K), which is called K-MICA, for optimum clustering N objects into K clusters. The new Hybrid K-ICA algorithm is tested on several data sets and its performance is compared with those of MICA, ACO, PSO, Simulated Annealing (SA), Genetic Algorithm (GA), Tabu Search (TS), Honey Bee Mating Optimization (HBMO) and K-means. The simulation results show that the proposed evolutionary optimization algorithm is robust and suitable for handling data clustering.

1. Introduction

Cluster analysis is a data analysis tool used for grouping data with similar characteristics. It has been used in data mining tasks such as unsupervised classification and data summation. The basic objective in cluster analysis is to discover natural groupings of objects. Cluster analysis techniques have been used in many areas such as qualitative interpretation and data compression, process monitoring, local model development, analysis of chemical compounds for combinatorial chemistry, discovering of clusters in DNA dinucleotides, classification of coals, manufacturing and production (process optimization and troubleshooting), medicine (several diagnostic information stored by hospital management systems), nuclear science, financial investment (stock indexes and prices, interest rates, credit card data, fraud detection), radar scanning and research and development planning, telecommunication network (calling patterns and fault management systems) (Kao et al., 2008; Cao and Cios, 2008; Zalik, 2008; Krishna and Murty, 1999; Mualik and Bandyopadhyay, 2000; Fathian and Amiri, 2007; Laszlo and Mukherjee, 2007; Shelokar et al., 2004; Ng and Wong, 2002; Sung and Jin, 2000; Niknam et al., 2008a, 2008b, 2009; Niknam and Amiri, 2010; Bahmani Firouzi et al., 2010). Generally cluster analysis algorithms have been utilized where the huge data are stored. Data clustering algorithms can be either hierarchical or partitional. In this paper we focus on the partitional clustering and in particular, a popular partitional clustering method called K-means clustering. The K-mean clustering algorithm is one of the most efficient clustering algorithms (Kao et al., 2008; Cao and Cios, 2008; Zalik, 2008; Krishna and Murty, 1999; Mualik and Bandyopadhyay, 2000; Fathian and Amiri, 2007; Laszlo and Mukherjee, 2007; Shelokar et al., 2004; Ng and Wong, 2002; Sung and Jin, 2000; Niknam et al., 2008a, 2008b, 2009; Niknam and Amiri, 2010; Bahmani Firouzi et al., 2010). It starts by initializing the K cluster centers. The input vectors (data points) are assigned to one of the existing clusters according to the square of the Euclidean distance from the clusters. The mean (centroid) of each cluster is then computed in order to update the cluster center. This update occurs as a result of the change in the membership of each cluster. The process of reassigning the input vectors and the update of the cluster centers are repeated until the values of the cluster centers do not change. However, the K-means algorithm suffers from several drawbacks. The objective function of the K-means is not convex and it may contain many local minima. Consequently, in the process of minimizing the objective function, there exists a possibility of getting stuck at local minima. The outputs of the K-means algorithm, therefore, heavily depend on the initial choice of the cluster centers. To overcome this drawback, many clustering algorithms based on evolutionary algorithms such as GA, TS and SA have been introduced (Kao et al., 2008; Cao and Cios, 2008; Zalik, 2008; Krishna and Murty, 1999; Mualik and Bandyopadhyay, 2000; Fathian and Amiri, 2007; Laszlo and Mukherjee, 2007; Shelokar et al., 2004; Ng and Wong, 2002;

* Corresponding author. Tel.: +98 711 7264121; fax: +98 711 7353502.
E-mail addresses: niknam@sutech.ac.ir, taher_nik@yahoo.com (T. Niknam).

0952-1976/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2010.10.001

Sung and Jin, 2000; Niknam et al., 2008a, 2008b, 2009; Niknam and Amiri, 2010; Bahmani Firouzi et al., 2010). For instance, Kao et al. (2008) have proposed a hybrid technique based on combining the K-means algorithm, Nelder-Mead simplex search, and PSO for cluster analysis. Cao and Cios (2008) have presented a hybrid algorithm according to the combination of GA, K-means and logarithmic regression expectation maximization. Zalik (2008) has introduced a K-means algorithm that performs correct clustering without pre-assigning the exact number of clusters. Krishna and Murty (1999) have presented an approach called genetic K-means algorithm for clustering analysis. Mualik and Bandyopadhyay (2000) have proposed a genetic algorithm based method to solve the clustering problem and experiment on synthetic and real life data sets to evaluate the performance. It defines a basic mutation operator specific to clustering called distance-based mutation. Fathian and Amiri (2007) have proposed the HBMO algorithm to solve the clustering problem. A genetic algorithm that exchanges neighboring centers for K-means clustering has presented by Laszlo and Mukherjee (2007). Shelokar et al. (2004) have introduced an evolutionary algorithm based on ACO algorithm for clustering problem. Ng and Sung have proposed an approach based on TS for cluster analysis (Ng and Wong, 2002; Sung and Jin, 2000). Niknam et al. (2008a, 2008b) have presented a hybrid evolutionary optimization algorithm based on the combination of ACO and SA to solve the clustering problem. Niknam et al. (2009) have presented a hybrid evolutionary algorithm based on PSO and SA to find optimal cluster centers. Niknam and Amiri (2010) have proposed a hybrid algorithm based on a fuzzy adaptive PSO, ACO and K-means for cluster analysis. Bahmani Firouzi et al. (2010) have introduced a hybrid evolutionary algorithm based on combining PSO, SA and K-means to find optimal solution.

However, most of evolutionary methods such as GA, TS, etc, are typically very slow to find optimum solution. Recently researchers have presented new evolutionary methods such as ACO, PSO and MICA to solve hard optimization problems which not only have a better response but also converge very quickly in comparison with ordinary evolutionary methods (Kao et al., 2008; Cao and Cios, 2008; Zalik, 2008; Krishna and Murty, 1999; Mualik and Bandyopadhyay, 2000; Fathian and Amiri, 2007; Laszlo and Mukherjee, 2007; Shelokar et al., 2004; Ng and Wong, 2002; Sung and Jin, 2000; Niknam et al., 2008a, 2008b, 2009; Niknam and Amiri, 2010; Bahmani Firouzi et al., 2010; Morales and Erazo, 2009; Sakai and Imiya, 2009; Twellmann et al., 2008). MICA is a novel algorithm which has ability in dealing with different types of optimization problems. The basic idea in this algorithm is to divide countries into two types: imperialist states and colonies (Rajabioun et al., 2008a, 2008b; Atashpaz-Gargari and Lucas, 2007a, 2007b; Atashpaz-Gargari et al., 2008a, 2008b; Roshanaei et al., 2008; Jasour et al., 2008. Imperialistic competition and assimilation policy cause the colonies to converge to optimum position. In this approach, initial countries are generated by using evolutionary algorithm and MICA's movement rule is applied to collapse weak empires and they are taken over by powerful ones. MICA should be taken into account as a powerful technique. Nevertheless, it may be trapped in local optima especially when numbers of imperialists increase. To alleviate this drawback, mutation can help to divert the movement of colonies toward their relevant imperialist into new positions also a chaotic local search (CLS) is used to get rid of the local optima. This approach provides better opportunity of exploring for colonies. To use the benefits of K-means and MICA, and reduce their disadvantages a novel hybrid evolutionary optimization method, called Hybrid K-MICA is presented in this paper, for optimum clustering N objects into K clusters. This hybrid algorithm not only has a better response but also converges more quickly than ordinary evolutionary algorithms. In this method, after generating initial countries, K-means is applied to improve the position of colonies.

The main contributions of the paper are as follows:
(i) present a modified ICA algorithm, (ii) combine the modified ICA algorithm with a CLS algorithm and (iii) present a new algorithm for cluster analysis.

The paper is organized as following: In Section 2, the cluster analysis problem is discussed. In Section 3 Imperialist Competitive Algorithm is introduced. In Sections 4–6, modified MICA, the Hybrid K-MICA and application of Hybrid K-MICA in clustering are shown, respectively. In Section 7, the feasibility of the Hybrid K-MICA is demonstrated and compared with K-MICA, MICA-K, ICA, MICA, ACO, PSO, SA, GA, TS, HBMO and K-means for different data sets. Finally, Section 8 includes summary and the conclusion.

2. Cluster analysis problem

K-means is one of the simplest unsupervised learning algorithms. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume K clusters) fixed as a priori. The main idea is to determine K centroids. These centroids should be placed in a cunning way as different locations cause different results. So, the best choice is to place them as much as possible far away from each other. The next step is to take each point referring to a given data set and associate it to the nearest centroid. When no point is left out, the first step is completed and an early grouping is done. At this point we need to recalculate K new centroids as centers of the clusters resulting from the previous step. After we have these K new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop, we may notice that the K centroids change their location step by step until no more changes are done. In other words, centroids do not move any further. Finally, the goal of this algorithm is to minimize an *objective function*, which in this case is a squared error function (Morales and Erazo, 2009; Sakai and Imiya, 2009; Twellmann et al., 2008).

The objective function has been calculated as follows:

$$cost(X) = \sum_{i=1}^{N} \min\{\|Y_i - X_j\|\}, \quad j = 1,2,3,\ldots,K \tag{1}$$

where $\|Y_i - X_j\|$ is a chosen distance measurement between a data input Y_i and the cluster center X_j. N and K are the number of input data and the number of cluster centers, respectively. Fig. 1 shows its flowchart.

The algorithm is composed of the following steps:

- Place K points into the space represented by the objects that are clustered. These points represent initial group centroids.
- Assign each object to the group that has the closest centroid.
- When all objects have been assigned, recalculate the positions of the K centroids.
- Repeat Steps 2 and 3 until the centroids are immobilized.

3. Original ICA

ICA is a population-based stochastic search algorithm. It has been introduced by Atashpaz and Lucas, recently (Rajabioun et al., 2008a, 2008b; Atashpaz-Gargari and Lucas, 2007a, 2007b; Atashpaz-Gargari et al., 2008a, 2008b; Roshanaei et al., 2008; Jasour et al., 2008). Since then, it is used to solve some kinds of optimization problem (Rajabioun et al., 2008a, 2008b; Atashpaz-Gargari and Lucas, 2007a, 2007b; Atashpaz-Gargari et al., 2008a, 2008b; Roshanaei et al., 2008; Jasour et al., 2008). The algorithm is inspired by imperialistic competition. It attempts to present the

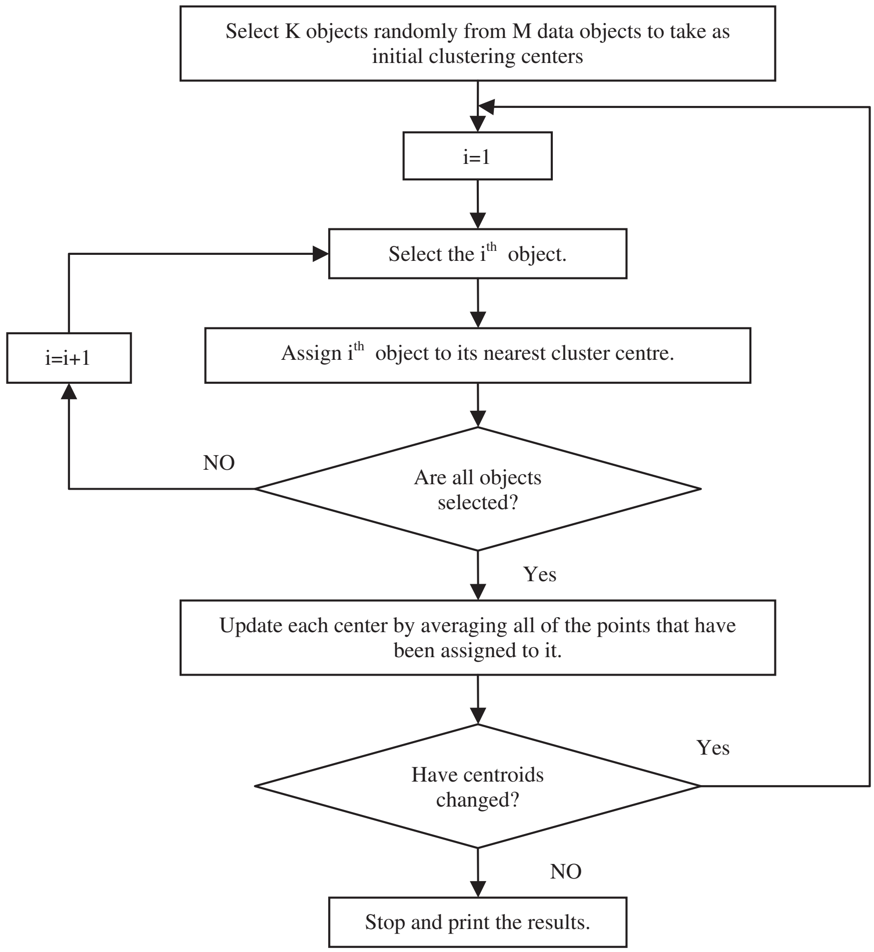

Fig. 1. Flowchart of K-means.

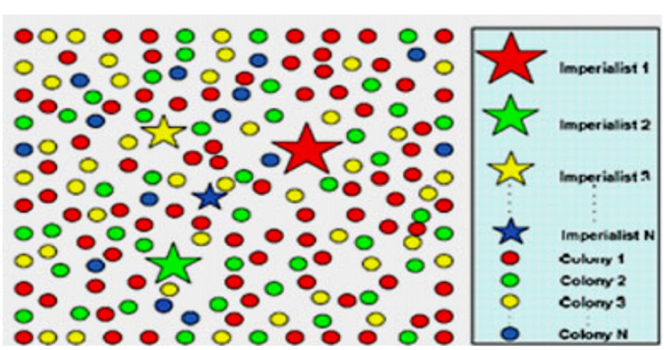

Fig. 2. Generating the initial empires.

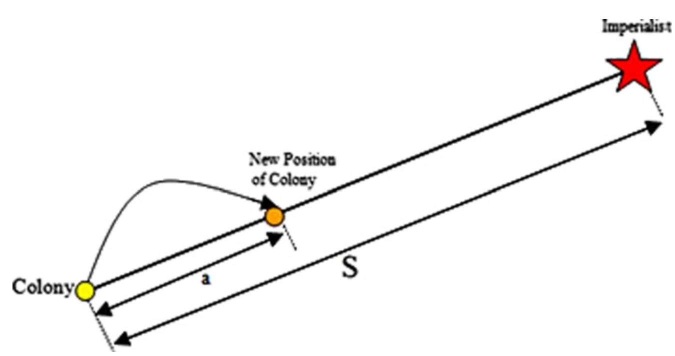

Fig. 3. Moving colonies toward their related imperialist.

social policy of imperialisms to control more countries and use their sources when colonies are dominated by some rules. If one empire loses its power, the rest of them will compete to take its possession. In ICA, this process is simulated by individuals that are known as countries.

This algorithm starts with a randomly initial population and objective function which is computed for them. The most powerful countries are selected as imperialists and the others are colonies of these imperialists .Then the competition between imperialists take place to get more colonies .The best imperialist has more chance to possess more colonies. Then one imperialist with its colonies makes an empire. Fig. 2 shows the initial populations of each empire (Rajabioun et al., 2008a, 2008b; Atashpaz-Gargari and Lucas, 2007a, 2007b; Atashpaz-Gargari et al., 2008a, 2008b; Roshanaei et al., 2008; Jasour et al., 2008). If the empire is bigger, its colonies are greater and the weaker ones are less. In this figure Imperialist 1 is the most powerful and has the greatest number of colonies.

After dividing colonies between imperialists, these colonies approach their related imperialist countries. Fig. 3 represents this movement. Based on this concept each colony moves toward the imperialist by a units and reaches its new position. Where a is a random variable with uniform (or any proper) distribution, β, a number greater than 1, causes colonies move toward their imperialists from different direction and S is the distance between colony and imperialist

$$a \sim U(0,\beta \times S) \tag{2}$$

If after this movement one of the colonies possess more power than its relevant imperialist, they will exchange their positions. To begin the competition between empires, total objective function of each empire should be calculated. It depends on objective function of both an imperialist and its colonies. Then the competition starts, the weakest empire loses its possession and powerful ones try to gain it. The empire that has lost all its colonies will collapse. At last the most powerful empire will take the possession of other empires and wins the competition.

To apply the ICA for clustering, the following steps have to be taken (Rajabioun et al., 2008a):

Step 1: The initial population for each empire should be generated randomly.
Step 2: Move the colonies toward their relevant imperialist.
Step 3: Exchange the position of a colony and the imperialist if its cost is lower.
Step 4: Compute the objective function of all empires.
Step 5: Pick the weakest colony and give it to one of the best empires.
Step 6: Eliminate the powerless empires.
Step 7: If there is just one empire, stop, if not go to 2.

The last Imperialist is the solution of the problem.

4. Modified ICA

In order to improve the convergence velocity and accuracy of the ICA, this article recommends a modified imperialist competitive algorithm (MICA). Premature convergence may occur under different situations: the population converges to local optima of the objective function or the search algorithm proceeds slowly or does not proceed at all. Mutation is a powerful strategy which diversifies the ICA population and improves the ICA's performance on preventing premature convergence to local minima. In this article a new mutation operator is used.

During the assimilation policy each colony (X) moves toward its relevant imperialist by a unit, where the initial distance between them is S. The new position of each colony would beX^{t+1}_{move} (t is the number of iteration):

$$a \sim U(0,\beta \times S)$$
$$X^{t+1}_{move,j} = X^t_j + a \tag{3}$$

X^t_j and $X^{t+1}_{move,j}$ are jth colony of each empire. After this movement for each colony X, a mutant colony X^{t+1}_{mut} is generated as follows (Atashpaz-Gargari and Lucas, 2007a):

$$X^{t+1}_{mut,j} = X^t_{m1} + rand(.) \times (X^t_{m2} - X^t_{m3}) \tag{4}$$

$$X_{mut,j} = [x_{mut,1}, x_{mut,2}, \ldots, x_{mut,b}]_{1\times b}, \quad b = K \times d$$

Then the selected colony would be:

$$X^{t+1}_{new,j} = [x_{new,1}, x_{new,2}, \ldots, x_{new,b}]_{1\times b} \tag{5}$$

$$x_{new,z} = \begin{cases} x_{mut,z} & \text{if } rand(.) < \gamma \\ x_z & \text{otherwise} \end{cases}, \quad z = 1,2,\ldots,b$$

where $rand(.)$ is a random number between 0 and 1, γ is a number less than 1.$m1$, $m2$, $m3$ are three individuals which are selected from initial colonies randomly. In order to cover the entire colonies uniformly, it is better to select them as $m1 \neq m2 \neq m3 \neq j$. K is the number of clusters and the dimension of each cluster center will be d.

To choose the best colony between $X_{move,\,j}$ and $X_{new,\,j}$ to replace jth colony (X_j), objective function is used

$$X^{t+1}_j = \begin{cases} X^{t+1}_{move,j} & \text{if } cost(X^{t+1}_{move,j}) \le cost(X^{t+1}_{new,j}) \\ X^{t+1}_{new,j} & \text{otherwise} \end{cases} \tag{6}$$

5. Hybrid K-MICA

As mentioned before, K-means is used for its easiness and simplicity for applications. However, it has some drawbacks. First, its result may depend on initial values. Also, it may converge to local minimum. Recently, numerous ideas have been used to alleviate this drawback by using global optimization algorithms such as GA (Krishna and Murty, 1999), TS (Ng and Wong, 2002), PSO (Kao et al., 2008), hybrid PSO-SA (Niknam et al., 2009), hybrid PSO-ACO-K (Bahmani Firouzi et al., 2010), HBMO (Fathian and Amiri, 2007), ACO (Shelokar et al., 2004), hybrid ACO-SA (Niknam et al., 2008a, 2008b), PSO-SA-K (Morales and Erazo, 2009) and hybrid K-PSO (Kao et al., 2008). In this article a new algorithm has been presented which mix both algorithms K-means and MICA.

There are different methods for combination K-means with MICA. In the first case K-means is used to generate the population, its output initializes MICA. The second type, MICA initializes the population and competition will be done, the last remaining empire will be given to K-means. And the last one, population is generated with MICA and initial empires form, then K-means is applied to improve the position of empire's colonies and imperialists, the result of this algorithm, again, will be given to MICA. Although the results of these procedures are better than the MICA, however the convergence speed and accuracy of the last one is the best.

The final algorithm is called Hybrid K-MICA. According to original ICA, first primary population generates and then empires with their possessions take place. Applying K-means to each empire causes us to improve the initial population of colonies. It makes the hybrid algorithm converges more quickly and prevents it from falling into local optima. The outputs of K-means form the initial empires of modified ICA.

To improve the income of algorithm, it is better, that the powerless imperialist would not be removed when it loses all possessions. This imperialist is one of the best answers and that can be contributed in imperialistic competition as a weak colony or to be given to the powerful empire. Flowchart and pseudo-code of these proposed algorithms are shown in Figs. 4 and 5, respectively.

6. Application of hybrid K-MICA on clustering

In this section, the application of Hybrid K-MICA on clustering problem is presented. To approach, the following steps should be taken and repeated. There are some inputs data which would be cluster and the control variables are the cluster centers.

Step 1: Generate an initial population

An initial population of input data is generated by chaos initialization as follows:

$$Population = \begin{bmatrix} X_1 \\ X_2 \\ \ldots \\ X_{N_{intial}} \end{bmatrix}$$

$$X_i = Country_i = [Center_1, Center_2, \ldots, Center_K] \quad i = 1,2,\ldots,N_{initial} \tag{7}$$
$$Center_j = [x_1, x_2, \ldots, x_d] \qquad j = 1,2,\ldots,K$$

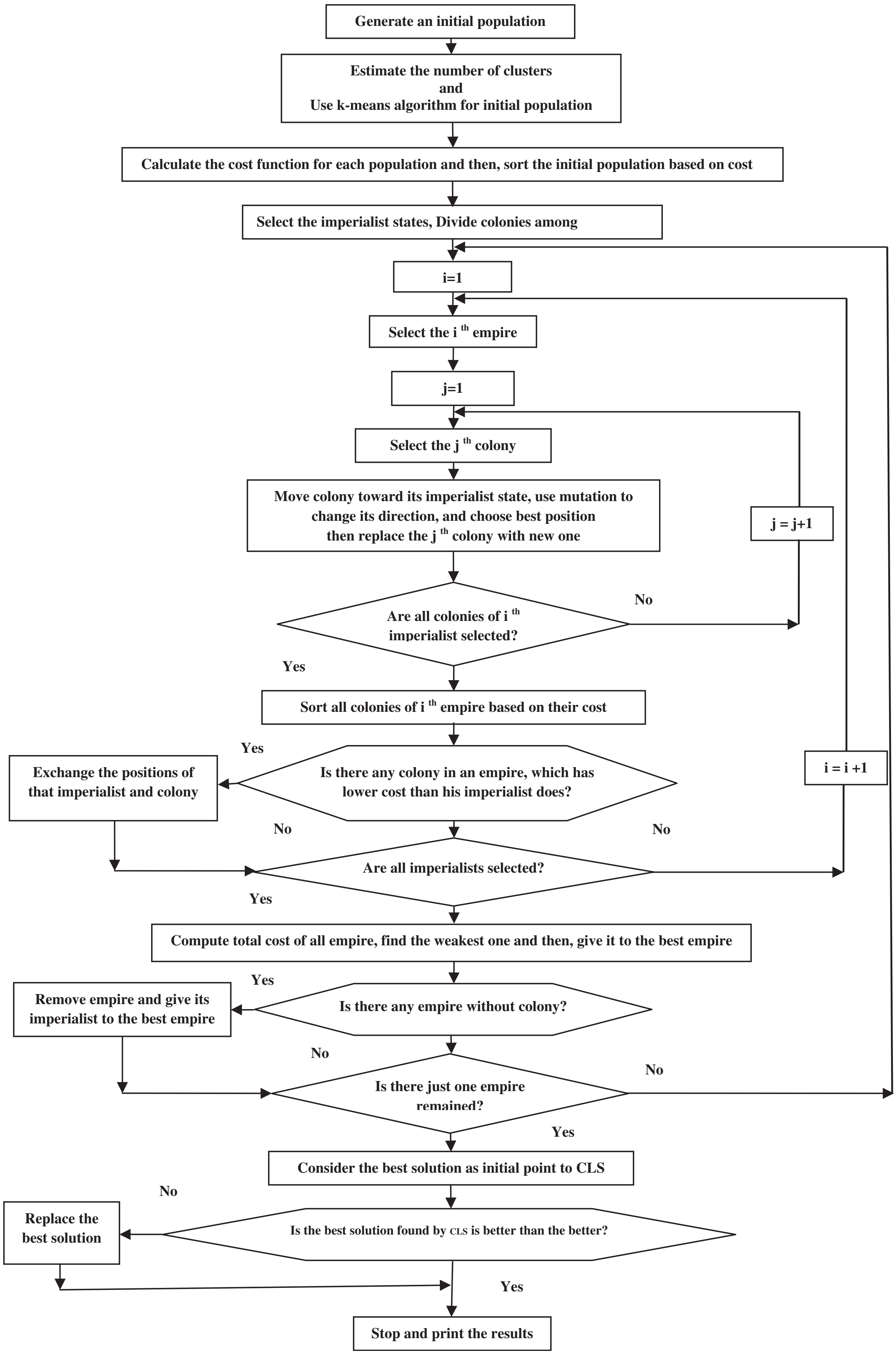

Fig. 4. Flowchart of K-MICA.

```
Begin
    Generate an initial population randomly
    Calculate the objective function for the initial population
    Sort the initial population based on their objective function values
    Select the imperialist states
    Divide colonies among imperialist
    Use K-means algorithm for each empire
     do{
        Place one colony as an initial K centroids object that are clustered.
        do{
           Assign each object to the group that has the closest centroid
           Recalculate the positions of the K centroids
           }
       while (the centroids no longer move)
       }
    while (all colonies selected)
    do{
       do{
          Select the i^th empire
          do{
             Select the j^th colony
             Move the colony toward its imperialist state
             Use mutation to change the direction of colony
             Calculate the objective function value for the two new population
             Compare both new cost and select the best one
             Replace j^th colony with new one
             }
        while (all colonies selected)
        Sort all colonies of i^th empire based on their cost functions
        Check cost of all colonies in each empire
        if there is a colony which has a lower cost than its imperialist
             exchange the position of the colony and the imperialist
        end if
        Update the position of the i^th empire
       }
      while (all empires selected)
        calculate total cost of empires
        find the weakest empire
       Give one of its colony to the winner empire
       Check the number of colony in each empire
       If there is an empire without colony
           Remove empire and give its imperialist to the best empire
       End if
    Search around the global solution by CLS
    One country selected randomly is replaced with the best solution among them

       }
      While (there is more than one empire)
      End
```

Fig. 5. Pseudo code for the Hybrid K-MICA.

$$H = k*d$$

$$X_0 = [x_0^1, x_0^2, \ldots, x_0^H]$$

$$x_0^j = rand(.) \times (x_{max}^j - x_{min}^j) + x_{min}^j, \quad j = 1,2,\ldots,H$$

$$X_i = [x_i^1, x_i^2, \ldots, x_i^H], \quad i = 2,\ldots,N_{intial}$$

$$x_i^j = 4 \times x_{i-1}^j \times (1 - x_{i-1}^j), \quad j = 1,2,\ldots,H$$

$$x_j^{min} < x_j < x_j^{max}$$

where $Center^j$ is jth cluster center for ith country. X_i is one of the countries. $N_{initial}$ is the number of population and d is the dimension of each cluster center. x_j^{max} and x_j^{min} (each feature of center) are the maximum and minimum value of each point referring to the jth cluster center which are in order. K is the number of clusters. H is the number of state variables. X_o is an initial solution.

Step 2: Calculate objective function value

Suppose that we have N sample feature vectors. The objective function is evaluated for each country as follows:

Step 2-1: $i=1$ and $Objec=0$.

Step 2-2: select the ith sample.

Step 2-3: calculate the distances between the ith sample and $Center_j$ ($j=1,\ldots,K$).

Step 2-4: add the value of $Objec$ with the minimum distance calculated in Step 2-3 ($Objec = Object + \min(|Center_i - Y_m|, \quad i=1, 2,\ldots,K)$).

Step 2-5: if all samples have been selected, go to the next step, otherwise $i=i+1$ and return to step 2-2.

Step 2-6: $Cost(X) = Objec$.

The objective function is calculated mathematically as below:

$$Cos(X) = \sum_{m=1}^{N} \min(|Center_i - Y_m|, \quad i = 1,2,\ldots,K)$$
$$(N = \text{number of input data}) \tag{8}$$

Step 3: Sort the initial population based on the objective function values.

The initial population is ascended based on the value of their objective function.

Step 4: Select the imperialist states.

Countries that have the minimum objective function are selected as the imperialist states and the remaining ones form the colonies of these imperialists.

Step 5: Divide colonies among imperialist.

Based on power of each imperialist the colonies are divided among them. The power of each imperialist calculated as followings:

$$C_n = \min_i\{cost_i\} - cost_n \tag{9}$$

$$p_n = \left| \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \right| \tag{10}$$

$$C_n^{norm} = round\{p_n(N_{col})\} \tag{11}$$

In above equations, $cost_n$ is the cost of nth imperialist and C_n the normalized cost of each one. The normalized power of each imperialist introduced as p_n, then the initial number of colonies for each empire will be C_n^{norm} where N_{col} and N_{imp} are number of all colonies and imperialists.

Step 6: Use K-means algorithm for each empire.

Step 7: Move colonies toward their imperialist states as described in Section 3.

Step 8: Use mutation to change the direction of colonies. It is mentioned in modified ICA.

Step 9: Check the cost of all colonies in each empire.

During the previous steps cost of each colony might have changed .Check the cost of all colonies of an empire if there is one that have a lower cost than its relevant imperialist, exchange their position.

Step 10: Check total cost of each empire.

Cost of each empire depends on power of both imperialist and its colonies. It is calculated as follows:

$$TC_n = cost(imperialist_n) + \xi \ mean\{cost(colonies\ of\ empire_n)\} \tag{12}$$

TC_n is the total cost of nth empire, ξ is an attenuation coefficient between 0 and 1 to reduce the effect of colonies cost.

Step 11: Do imperialistic competition

All empires according their power (total cost), try to get colonies of weakest empire.

$$TC_n^{norm} = TC_n - \max_j\{TC_j\} \tag{13}$$

$$P_{P_n} = \left| \frac{TC_n^{norm}}{\sum_{j=1}^{N_{imp}} TC_j^{norm}} \right| \tag{14}$$

where TC_n^{norm} is normalized total cost of nth empire and the possession probability of each empire is P_{P_n}

The roulette wheel can be used for stochastic selection of the winner empire which will dominate the weakest colony of weakest empire.

To perform Roulette Wheel algorithm, it is necessary to calculate cumulative probability as follows:

$$C_{P_n} = \sum_{i=1}^{n} P_{P_i}$$

According to this equation cumulative probability for $n=1$ is equal to its probability, while for the last n it corresponds to one. Then a random number with uniform distribution generates and compares with all C_{P_n}. Each sector with higher probability will have more chance to be chosen. Therefore the winner empire will specify.

As it is mentioned to use Roulette Wheel algorithm, computing cumulative distribution function is essential. To reduce this time consuming step an approach has been presented as below:

$$P = [P_{P_1}, P_{P_2}, \ldots, P_{P_{N_{imp}}}] \tag{15}$$

$$R = [r_1, r_2, \ldots, r_{N_{imp}}], \quad r_1, r_2, \ldots, r_{N_{imp}} \sim U(0,1) \tag{16}$$

$$D = P - R = [D_1, D_2, \ldots, D_{N_{imp}}] = [P_{P_1} - r_1, P_{P_2} - r_2, \ldots, P_{P_{N_{imp}}} - r_{N_{imp}}] \tag{17}$$

P is the vector of possession probability of all empires and R is a vector with uniformly distributed random numbers. Maximum index in D shows Winner Empire that gets the colony.

After realizing the winner empire, the weakest colony of the weakest empire will be given to the winner one. Then we should subtract one of the populations of this weak empire and add one to the winner's population.

Step 12: Remove weakest empire.

If there is any empire without colony, eliminate it. Replace one of the weakest colonies of best empire (low cost) with this imperialist.

Step 13: Apply chaotic local search (CLS) to search around the global solution

ICA has gained much attention and widespread applications in different fields. However, it often converges to local optima. In order to avoid this shortcoming, a CLS algorithm is used to search around the global solution in the paper. CLS is based on the logistic equation as follows:

$$Cx_i = [cx_i^1, cx_i^2, \ldots, cx_i^H]_{1\times H}, \quad i = 0,1,2,\ldots,N_{choas} \tag{18}$$

$$cx_{i+1}^j = 4 \times cx_i^j \times (1 - cx_i^j), \quad j = 1,2,\ldots,H$$

$$cx_0^j = rand(.)$$

$$cx_i^j \int [0,1], \quad cx_0^j \notin \{0.25.0.5,0.75\}$$

In the CLS, the best solution is considered as an initial solution (X_{cls}^0) for CLS. X_{cls}^0 is scaled into (0,1) according to the following equation:

$$X_{cls}^0 = [x_{cls,0}^1, x_{cls,0}^2, \ldots, x_{cls,0}^H]_{1\times H}$$

$$Cx = [cx_0^1, cx_0^2, \ldots, cx_0^H] \tag{19}$$

$$cx_0^j = \frac{x_{cls,0}^j - x_{min}^j}{x_{max}^j - x_{min}^j}, \quad j = 1,2,\ldots,H$$

The chaos population for CLS is generated as follows:

$$X_{cls}^i = [x_{cls,i}^1, x_{cls,i}^2, \ldots, x_{cls,i}^H]_{1\times H}, \quad i = 1,2,\ldots,N_{choas} \tag{20}$$

$$x_{cls,i}^j = cx_{i-1}^j \times (x_{max}^j - x_{min}^j) + x_{min}^j, \quad j = 1,2,\ldots,H$$

where cx_i^j indicates the jth chaotic variable, N_{choas} is the number of individuals for CLS. *rand(.)* is a random number between zero and one.

The objective function is evaluated for all individuals of CLS. One country selected randomly is replaced with the best solution among them.

Step 14: Check number of empire.

If there is just one empire remained, stop. Else go to step 7.

7. Experimental results

The experimental results comparing the Hybrid K-MICA clustering algorithm with several typical stochastic algorithms including K-MICA, MICA-K, MICA, ICA, ACO, PSO, SA, GA, TS, HBMO and K-means are presented. They are used for four real-life data sets (*Iris, Wine, Vowel and Contraceptive Method Choice* (*CMC*)), which are described as follows:

Iris data ($N=150$, $d=4$, $K=3$): The Iris flower data set is a multivariate data set introduced by Sir Ronald Aylmer Fisher

(1936) as an example of discriminant analysis. The data set consists of 50 samples from each of three species of Iris flowers (*Iris setosa*, *Iris virginica* and *Iris versicolor*). For each species there are 50 observations including sepal length, sepal width, petal length, and petal width in centimeters. Based on the combination of the four features, Fisher developed a linear discriminant model to distinguish the species from each other. It is used as a typical test for many other classification techniques. It has three classes (with some overlap between classes 2 and 3). Therefore, the value of K is chosen to be 3 for this data (Bahmani Firouzi et al., 2010).

Wine data ($N=178, d=13, K=3$)*:* This is the wine data set, which is also taken from MCI laboratory. These data are the results of a chemical analysis of wines grown in the same region in Italy extracted from three different cultivars. There are 178 instances with 13 numeric attributes in wine data set. 106 instances are for training, 36 for validation, and 36 for testing. All attributes are continuous. There are three classes corresponding to three different cultivars. The attributes are alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines and proline. There are no missing attribute values (Bahmani Firouzi et al., 2010).

Contraceptive Method Choice ($N=1473$, $d=10$, $K=3$): This data set is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey. The samples were married women who either were not pregnant or did not know if they were at the time of interview. The problem is to predict the current contraceptive method choice (no use of long-term methods, or short-term methods) of a woman based on her demographic and socio-economic characteristics. The attributes are wife's age, wife's education, husband's education, number of children ever born,

Table 1
Simulation results of Hybrid K-MICA algorithm parameters for Iris data set.

Case	N_{pop}	N_{imp}	β	ξ	γ	Best solution	Worst solution	Average solution
1	150	12	20	0.5	0.8	96.6554	96.6554	96.6554
2	150	8	20	0.1	0.7	96.6554	96.6554	96.6554
3	120	4	15	0.05	0.6	96. 6554	96.6554	96.6554
4	120	12	15	0.5	0.7	96. 6554	96.6554	96.6554
5	100	8	10	0.1	0.6	96. 6554	96.6554	96.6554
6	100	4	10	0.05	0.5	96. 6554	96.6554	96.6554
7	80	8	5	0.5	0.5	96. 6554	96.6554	96.6554
8	80	4	5	0.1	0.4	96. 6554	96.6554	96.6554
9	30	8	1	0.05	0.4	96. 9672	97.1304	96.9948
10	30	4	1	0.5	0.3	96. 8634	96.9716	96.9141

Table 2
Results obtained by the algorithms for 100 different runs on Iris data.

Method	Function value			Standard deviation
	F_{best}	$F_{average}$	F_{worst}	
Hybrid K-MICA	96.6554	96.6554	96.6554	0
K-MICA	96.6564	96.68344	96.7588	0.0359928
MICA-K	96.6556	96.66691	96.7071	0.018515
MICA	96.6562	96.6664	96.6919	0.0114455
ICA	96.6997	96.8466	97.0059	0.1114908
PSO	96.8942	97.2328	97.8973	0.347168
SA	97.4573	99.957	102.01	2.018
TS	97.365977	97.868008	98.569485	0.53
GA	113.986503	125.197025	139.778272	14.563
ACO	97.100777	97.171546	97.808466	0.367
HBMO	96.752047	96.95316	97.757625	0.531
K-means	97.333	106.05	120.45	14.6311

Table 3
Centers obtained by the algorithms for the best results on Iris data.

Iris data		
Center1	Center2	Center3
5.934321	6.733392	5.012193
2.797787	3.067797	3.403116
4.417873	5.630093	1.471585
1.417208	2.106860	0.235411

Table 5
Centers obtained by the algorithms for the best results on Wine data.

Wine data		
Center1	Center2	Center3
12.81	12.31	13.65
2.52	2.58	1.72
2.42	2.3	2.38
19.69	21.21	16.82
99.02	92.56	104.94
2.13	2.12	2.73
1.85	1.58	3.03
0.395	0.39	0.23
1.51	1.29	1.87
5.59	4.4	5.5
0.865	0.984	1.07
2.39	2.53	3.16
686.94	463.53	1137.31

Table 4
Results obtained by the algorithms for 100 different runs on Wine data.

Method	Function value			Standard deviation
	F_{best}	$F_{average}$	F_{worst}	
Hybrid K-MICA	16,292.65	16,293.06	16,293.23	0.27013
K-MICA	16,293.85	16,294.83	16,296.5	0.924925
MICA-K	16,293.6	16,295	16,296.8	0.992492
MICA	16,293.9	16,295.6	16,296.94	1.002372
ICA	16,295.24	16,298.57	16,304.61	2.934509
PSO	16,345.9670	16,417.4725	16,562.3180	85.4974
SA	16,473.4825	17,521.094	18,083.251	753.084
TS	16,666.22699	16,785.45928	16,837.53567	52.073
GA	16,530.53381	16,530.53381	16,530.53381	0
ACO	16,530.53381	16,530.53381	16,530.53381	0
HBMO	16,357.28438	16,357.28438	16,357.28438	0
K-means	16,555.68	18,061	18,563.12	793.213

wife's religion, wife's now working, husband's occupation, standard-of-living index, media exposure, and contraceptive method used (Bahmani Firouzi et al., 2010).

Table 6
Results obtained by the algorithms for 100 different runs on CMC data.

Method	Function value			Standard deviation
	F_{best}	$F_{average}$	F_{worst}	
Hybrid K-MICA	5693.9198	5693.9623	5694.023	0.05354
K-ICA	5695.8547	5696.8659	5698.0194	1.111793
ICA-K	5694.8723	5695.5322	5697.50	1.268275
MICA	5699.2183	5705.1485	5721.1779	7.397884
ICA	5725.7062	5736.3663	5752.9425	8.000562
PSO	5700.9853	5820.9647	5923.2490	46.959690
SA	5849.0380	5893.4823	5966.9470	50.867200
TS	5885.0621	5993.5942	5999.8053	40.84568
GA	5705.6301	5756.5984	5812.6480	50.3694
ACO	5701.9230	5819.1347	5912.4300	45.634700
HBMO	5699.2670	5713.9800	5725.3500	12.690000
K-means	5842.20	5893.60	5934.43	47.16

Table 7
Centers obtained by the algorithms for the best results on CMC data.

CMC data		
Center1	Center2	Center3
43.6378435	24.4160009	33.5003583
2.9838119	3.0425546	3.1323624
3.4332324	3.5100282	3.5530743
4.5987409	1.7875114	3.6536758
0.8162801	0.9331797	0.8228042
0.7878479	0.7892882	0.6935348
1.8379755	2.2944430	2.1022457
3.4236738	2.9751945	3.2830577
0.08381778	0.0399678	0.0632146
1.6483970	1.9993621	2.1176835

Vowel data set (N=871, d=3, K=6). This data set consists of 871 Indian Telugu vowel sounds. There are six overlapping vowel classes and three input features. These data sets cover examples of data of low, medium and high dimensions. This consists of a three dimensional array including speaker, vowel, input. The speakers are indexed by integers 0–89. The vowels are indexed by integers 0–10. For each utterance, there are ten floating-point input values, with array indices 0–9. All entries are integers (Bahmani Firouzi et al., 2010).

The parameters required for implementation of the Hybrid K-MICA algorithm areβ, ξ, α, N_{pop}, N_{imp}. The results of 10 runs of the algorithm for iris data set with different parameters are depicted in the Table 1. It illustrates that this hybrid algorithm have a little dependency on variation of the parameters.

The algorithms are implemented by using Matlab 7.6 on a Pentium IV, 2.8 GHz, 1 GB RAM computer.

Tables 2, 4, 6 and 8, present a comparison among the results of Hybrid K-MICA, K-MICA, MICA-K, MICA, ICA, ACO (Shelokar et al., 2004; Niknam et al., 2008a, 2008b, 2009; Niknam and Amiri, 2010; Bahmani Firouzi et al., 2010), PSO (Kao et al., 2008), SA (Niknam et al., 2008a), GA (Krishna and Murty, 1999), TS(Ng and Wong, 2002), HBMO (Fathian and Amiri, 2007) and K-means (Niknam et al., 2008a, 2008b, 2009; Niknam and Amiri, 2010; Bahmani Firouzi et al., 2010; Morales and Erazo, 2009) for 100 random tails on four real-life data sets. Centers related to the best results for Hybrid K-MICA on these data sets are shown in Tables 3, 5, 7 and 9.

The simulation results given in Tables 2–9 show that Hybrid K-MICA is very precise and reliable. In other words, it provides the optimum value and small standard deviation in comparison to those of other methods. The results obtained on the iris data set show that Hybrid K-MICA converges to the global optimum of 96.6554 each time while the best solutions of K-MICA, MICA-K, MICA, ICA, PSO, SA, TS, GA, ACO, HBMO and K-means are 96.6564, 96.6556, 96.6562, 96.6997, 96.8942, 97.4573, 97.3659, 113.9865, 97.1007, 96.7520 and 97.3330. The standard deviation of the fitness function for this algorithm is zero, which is significantly less than

Table 8
Results obtained by the algorithms for 100 different runs on Vowel data.

Method	Function value			Standard deviation
	F_{best}	$F_{average}$	F_{worst}	
Hybrid K-MICA	148,967.24	149,100.35	149,116.72	8.583
K-ICA	149,279.9922	149,596.3367	149,955.0055	280.0121
ICA-K	149,244.6165	149,737.0533	150,837.4077	638.8130
MICA	149,332.1800	150,204.1368	150,982.4586	733.0634
ICA	150,991.6147	151,547.0511	152,735.1651	704.0907
PSO	148,976.0152	151,999.8251	158,121.1834	28,813.4692
SA	149,370.4700	161,566.2810	165,986.4200	2847.08594
TS	149,468.268	162,108.5381	165,996.4280	2846.23516
GA	149,513.735	159,153.498	165,991.6520	3105.5445
ACO	149,395.602	159,458.1438	165,939.8260	3485.3816
HBMO	149,201.632	161,431.0431	165,804.671	2746.0416
K-means	149,422.26	159,242.89	161,236.81	916

Table 9
Centers obtained by the algorithms for the best results on different data sets.

Vowel data					
Center1	Center2	Center3	Center4	Center5	Center6
375.4492841016	623.7187082763	439.2441852984	357.2621132835	407.8919016085	506.9882882437
2149.4029499135	1309.5914890562	987.6881301247	2291.4363895068	1018.0520762932	1839.6594262991
2678.4436693818	2333.4546455726	2665.4737179485	2977.3931145386	2317.8268935542	2556.1992947450

other methods. Table 4 indicates the results gained on wine data set. It brings us into the conclusion that although the basis of Hybrid K-MICA, K-MICA and MICA-K are the same but this new hybrid algorithm converges to the best global solution and has a better average and worst solution in comparison with those two methods. For CMC data set, the standard deviation of K-MICA, MICA-K, MICA, ICA, PSO, SA, TS, GA, ACO, HBMO and K-means are 1.111793, 1.268275, 7.397884, 8.000562, 46.959690, 50.867200, 40.84568, 50.3694, 45.634700, 12.690 and 47.16 while this is 0.05354 for Hybrid K-MICA which shows much difference compared to others. For the vowel data set, the best global solution, the average and worst solutions and standard deviation of the Hybrid K-MICA algorithm are 148,967.24, 149,100.35, 149,116.72 and 8.583 which are much smaller than those of other algorithms.

Comparing results of ICA and MICA depict the effect of applying mutation. For instance the consequences of these two algorithms on Iris data set are 96.6562, 96.6664, 96.6919 and 0.0114455 for MICA and 96.6997, 96.8466, 97.0059 and 0.1114908 for ICA which show better results on the best, average and the worst solutions and standard deviation, respectively. The results of Hybrid K-MICA, K-MICA and MICA-K in comparison to MICA display the advantages of using K-means. For vowel data set, the best global solutions for Hybrid K-MICA, K-MICA, MICA-K and MICA are 148,967.2408, 149,279.9922, 149,244.6165 and 149,332.1800 that appear meaningful variation. Algorithmic parameters for all algorithms are illustrated in Table 10.

Figs. 6–17 show the convergence characteristics of Hybrid K-MICA, K-MICA, MICA-K, MICA, ICA, and K-means for the best solutions on Iris and wine data set.

Simulation results of the figures show that K-means algorithm converges faster than the other algorithms but converges prematurely to a local optimum. For the iris data set Hybrid K-MICA converges to the global optimum after 49 iterations while K-MICA, MICA-K, MICA and ICA converge to the global optimum in

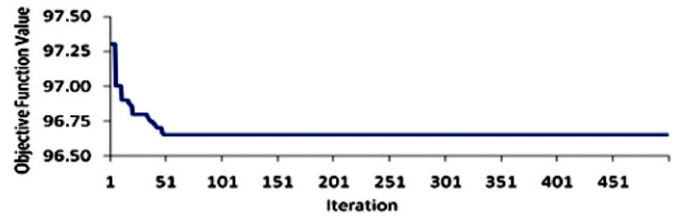

Fig. 6. Convergence characteristic of Hybrid K-MICA for the best solutions on Iris data.

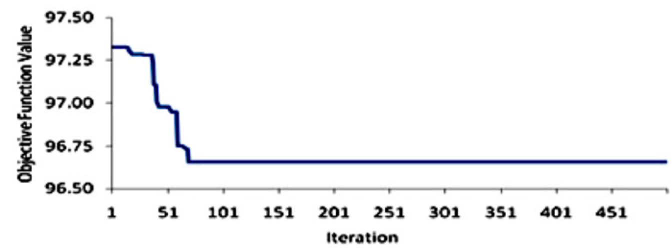

Fig. 7. Convergence characteristic of K-MICA for the best solutions on Iris data.

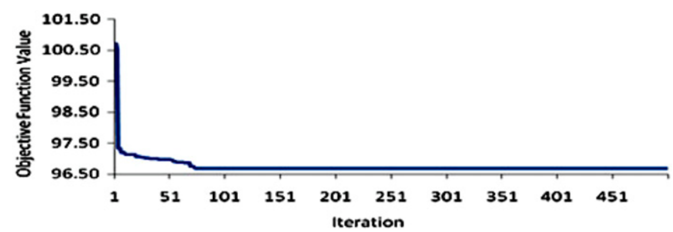

Fig. 8. Convergence characteristic of MICA-K for the best solutions on Iris data.

Table 10
Values of parameters of each of five algorithms.

K-MICA		MICA-K		MICA		ICA	
Parameter	Value	Parameter	Value	Parameter	Value	Parameter	Value
N_{pop}	100	N_{pop}	100	N_{pop}	100	N_{pop}	100
N_{imp}	6	N_{imp}	6	N_{imp}	8	N_{imp}	8
β	5	β	5	β	5	β	5
ξ	0.05	ξ	0.05	ξ	0.05	ξ	0.05
γ	0.7	γ	0.7	γ	0.7	γ	0.7
# iterations	500	# iterations	500	# iterations	500	# iterations	500

HBMO		SA		ACO	
Parameter	Value	Parameter	Value	Parameter	Value
# queens	1	Probability threshold	0.98	# ants	50
# drones	150	Initial temperature	5	Probability threshold for maximum trail	0.98
Capacity of spermatheca	50	Temperature multiplier	0.98	Local search probability	0.01
Maximum speed	Randomly ∈ [0.5 1]	Final temperature	0.01	Evaporation rate	0.01
Minimum speed	Randomly ∈ [01]	Number of Iterations detect steady stat	100	# iterations	1000
Speed reduction	0.98	# iterations	30,000		
Crossover	1.5				
# iterations	1000				

GA		TS		PSO	
Parameter	Value	Parameter	Value	Parameter	Value
Population	50	Tabu list size	25	# Swarm	$10 \times K \times d$
Crossover	0.8	Number of trial solutions	40	$C_1 = C_2$	2
Mutation rate	0.001	Probability threshold	0.98	ω_{min}	0.5
# iterations	1000	# iterations	1000	ω_{max}	1
				# iterations (7)	500

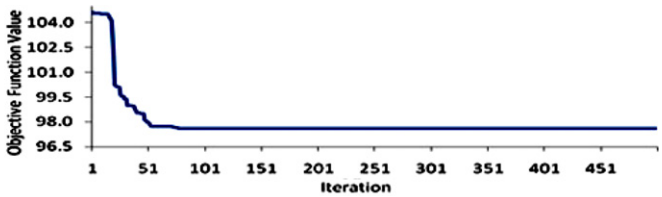

Fig. 9. Convergence characteristic of MICA for the best solutions on Iris data.

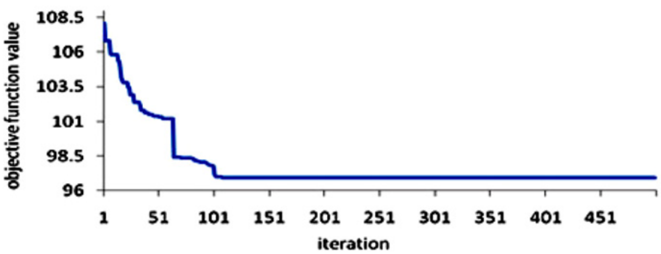

Fig. 10. Convergence characteristic of ICA for the best solutions on Iris data.

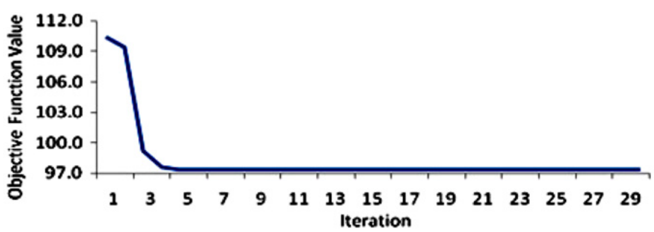

Fig. 11. Convergence characteristic of K-means for the best solutions on Iris data.

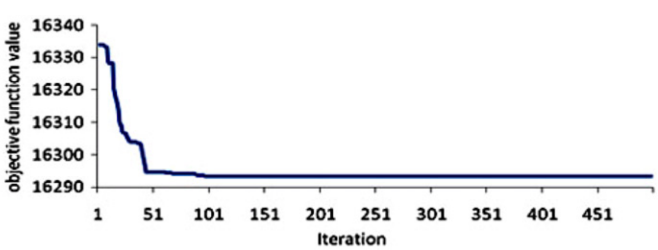

Fig. 12. Convergence characteristic of Hybrid K-MICA for the best solutions on Wine data.

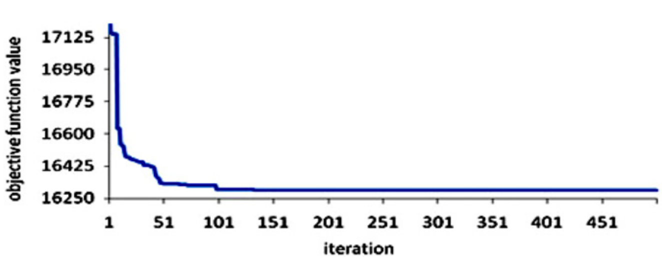

Fig. 13. Convergence characteristic of K-MICA for the best solutions on Wine data.

about 69, 74, 78 and 109 iterations. The convergence characteristics of these algorithms for wine data show that combining K-means with MICA converges to better results. While ICA and MICA converge to the global solution after 161 and 146 iterations, MICA-K, K-MICA and Hybrid K-MICA converge after 132, 141 and 97iterations.

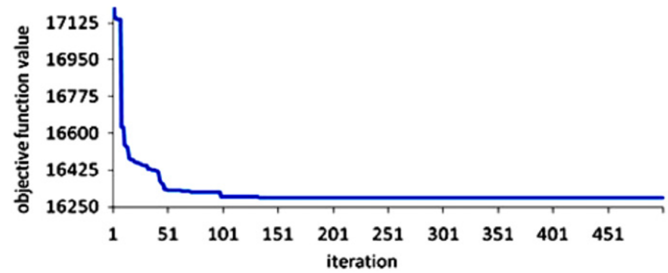

Fig. 14. Convergence characteristic of MICA-K for the best solutions on Wine data.

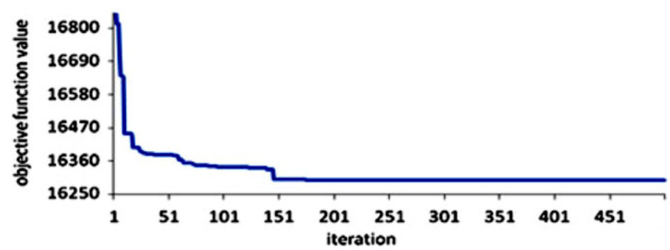

Fig. 15. Convergence characteristic of MICA for the best solutions on Wine data.

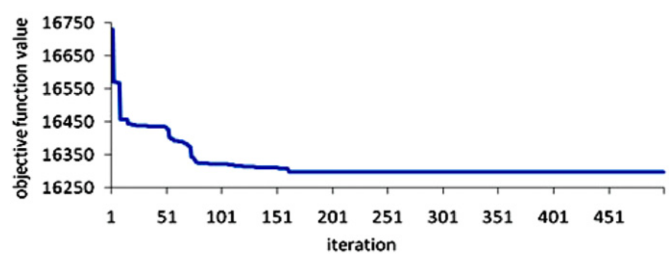

Fig. 16. Convergence characteristic of ICA for the best solutions on Wine data.

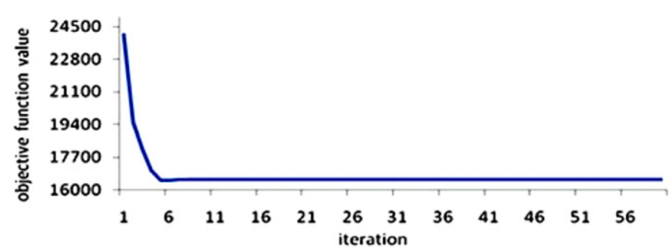

Fig. 17. Convergence characteristic of K-means for the best solutions on Wine data.

8. Conclusion

The imperialist competitive algorithm is a new method, which has great abilities to cope with different types of optimization problems. However, it is still in its infancy and intensive studies are needed to improve its performance. In this paper, a novel hybrid methodology called Hybrid K-MICA is introduced and debated in detail. Hybrid K-MICA is a combination of two powerful optimization algorithms; K-means and Imperialist Competitive Algorithm. In this new algorithm, we use K-means for each empire to select the best empires just before competition starts by MICA. It has been shown that this combination can produce better empires and causes the most-fit imperialist last. Hybrid K-MICA algorithm has been developed in this paper to solve clustering problems. The results illustrate that the proposed Hybrid K-MICA optimization algorithm can be considered as a viable and an efficient heuristic method to find optimal or nearly optimal solutions for clustering problems of allocating *N* objects to *K* clusters.

The experimental results indicate that the proposed optimization algorithm is comparable to other algorithms in terms of best, worst and average solutions and standard deviation. The convergence of the proposed hybrid algorithm to the global optimum solution is better than that of other evolutionary algorithms. Regardless of robustness and efficiency of Hybrid K-MICA algorithm, it is applicable when the number of clusters is known as a priori.

References

Atashpaz-Gargari, E., Lucas, C., 2007b. Designing an optimal PID controller using Colonial Competitive Algorithm. In: Proceedings of the First Iranian Joint Congress on Fuzzy and Intelligent Systems, Mashhad, Iran.

Atashpaz-Gargari, E., Lucas, C., 2007a. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In: Proceedings of the IEEE Congress on Evolutionary Computation, Singapore, pp. 4661–4667.

Atashpaz-Gargari, E., Hashemzadeh, F., Lucas, C., 2008a. Designing MIMO PIID controller using colonial competitive algorithm: applied to distillation column process. In: Proceeding of IEEE CEC 2008, within IEEE WCCI 2008, Hong Kong, pp. 1929–1934.

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

Bahmani Firouzi, B., Sha Sadeghi, M., Niknam, T., 2010. A new hybrid algorithm based on PSO, SA, and K-means for cluster analysis. International Journal of Innovative Computing Information and Control 6 (4), 1–10.

Cao, D., Cios, Krzysztof J., 2008. GAKREM: a novel hybrid clustering algorithm. Information Sciences 178, 4205–4227.

Fathian, M., Amiri, B., 2007. A honey-bee mating approach on clustering. The International Journal of Advanced Manufacturing Technology 38 (7-8), 809–821.

Jasour, A.M., Atashpaz Gargari, E., Lucas, C., vehicle fuzzy controller design using imperialist competitive algorithm. In: Proceedings of the Second First Iranian Joint Congress on Fuzzy and Intelligent Systems, Tehran, Iran, 2008.

Kao, Y.T., Zahara, E., Kao, I.W., 2008. A hybridized approach to data clustering. Expert Systems with Applications 34 (3), 1754–1762.

Krishna, K., Murty, M., 1999. Genetic k-means Algorithm. IEEE Transactions on Systems, Man and Cybernetics B Cybernet 29, 433–439.

Laszlo, M., Mukherjee, S., 2007. A genetic algorithm that exchanges neighboring centers for k-means clustering. Pattern Recognition Letters 28 (16), 2359–2366.

Morales, A.K., Erazo, F.R., 2009. A search space reduction methodology for data mining in large databases. Engineering Applications of Artificial Intelligence 22 (1), 57–65.

Mualik, U., Bandyopadhyay, S., 2000. Genetic algorithm-based clustering technique. Pattern Recognition 33, 1455–1465.

Ng, M.K., Wong, J.C., 2002. Clustering categorical data sets using tabu search techniques. Pattern Recognition 35 (12), 2783–2790.

Niknam, T., Amiri, B., 2010. An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis. Applied Soft Computing 10 (1), 183–197.

Niknam, T., Olamaie, J., Amiri, B., 2008a. A hybrid evolutionary algorithm based on ACO and SA for cluster analysis. Journal of Applied Science 8 (15), 2695–2702.

Niknam, T., Bahmani Firouzi, B., Nayeripour, M., 2008b. An efficient hybrid evolutionary algorithm for cluster analysis. World Applied Sciences Journal 4 (2), 300–307.

Niknam, T., Amiri, B., Olamaie, J., Arefi, A., 2009. An efficient hybrid evolutionary optimization algorithm based on PSO and SA for clustering. Journal of Zhejiang University Science 10 (4), 512–519.

Rajabioun, R., Hashemzadeh, F., Atashpaz-Gargari, E., 2008a. 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.

Rajabioun, R., Hashemzadeh, F., Atashpaz-Gargari, E., Mesgari, B., Rajaiee Salmasi, F., 2008b. Identification of a MIMO evaporator and its decentralized PID controller tuning using colonial competitive algorithm. In: Proceedings of the 17th World Congress, The International Federation of Automatic Control, Seoul, Korea, July 6–11, pp. 9952–9957.

Roshanaei, M., Atashpaz-Gargari, E., Lucas, C., 2008. Adaptive beamforming using colonial competitive algorithm. In: Proceedings of the Second International Joint Conference on Computational Engineering, Vancouver, Canada.

Sakai, T., Imiya, A., 2009. Unsupervised cluster discovery using statistics in scale space. Engineering Applications of Artificial Intelligence 22 (1), 92–100.

Shelokar, P.S., Jayaraman, V.K., Kulkarni., B.D., 2004. An ant colony approach for clustering. Analytica Chimica Acta 509 (2), 187–195.

Sung, C.S., Jin, H.W., 2000. A tabu-search-based heuristic for clustering. Pattern Recognition 33 (5), 849–858.

Twellmann, T., Baese, A.M., Lange, O., Foo, S., Nattkemper, T.W., 2008. Model-free visualization of suspicious lesions in breast MRI based on supervised and unsupervised learning. Engineering Applications of Artificial Intelligence 21 (2), 129–140.

Zalik, K.R., 2008. An efficient k-means clustering algorithm. Pattern Recognition Letters 29, 1385–1391.