Sādhanā Vol. 36, Part 3, June 2011, pp. 293–315.

A new hybrid imperialist competitive algorithm on data clustering

TAHER NIKNAM[1,*], ELAHE TAHERIAN FARD[2], SHERVIN EHRAMPOOSH[3] and ALIREZA ROUSTA[1,4]

[1]Marvdasht Branch, Islamic Azad University, Marvdasht, Iran, P.O. Box. 73711-13119
[2]Shiraz University, Shiraz, Iran, P.O. Box. 71345-1837
[3]Kerman Graduate University of Technology, Kerman, Iran, P.O. Box. 71315-115
[4]Department of Electrical and Electronic, Shiraz University of Technology, Shiraz, Iran, P.O. Box. 71555-313
e-mail: taher_nik@yahoo.com; etaherianfard@yahoo.com; s_ehrampoosh@yahoo.com

MS received 3 April 2010; revised 31 October 2010; accepted 5 December 2010

Abstract. Clustering is a process for partitioning datasets. This technique is very useful for optimum solution. *k*-means is one of the simplest and the most famous methods that is based on square error criterion. This algorithm depends on initial states and converges to local optima. Some recent researches show that *k*-means algorithm has been successfully applied to combinatorial optimization problems for clustering. In this paper, we purpose a novel algorithm that is based on combining two algorithms of clustering; *k*-means and Modify Imperialist Competitive Algorithm. It is named hybrid K-MICA. In addition, we use a method called modified expectation maximization (EM) to determine number of clusters. The experimented results show that the new method carries out better results than the ACO, PSO, Simulated Annealing (SA), Genetic Algorithm (GA), Tabu Search (TS), Honey Bee Mating Optimization (HBMO) and *k*-means.

Keywords. Modified imperialist competitive algorithm; simulated annealing; *k*-means; data clustering.

1. Introduction

Clustering is one of the unsupervised learning branches where a set of patterns, usually vectors in a multi-dimensional space, are grouped into clusters in such a way that patterns in the same cluster are similar in some sense and patterns in different clusters are dissimilar in the same sense. Cluster analysis is a difficult problem due to a variety of ways of measuring the similarity and dissimilarity concepts, which do not have a universal definition. Therefore, cluster seeking

*For correspondence

is experiment-oriented in the sense that clustering algorithms that can deal with all situations equally well are not yet available. Clustering is not the same as classification. In classification, input samples are labelled but in clustering, they have no initial tag. In fact, with using clustering methods, the same data is specified and implicitly labelled.

Data clustering algorithms can be divided into hierarchical or partitional. In this paper, we focus on the partitional clustering. There is a clustering method called k-means. It is the simplest and most commonly used algorithm employing a squared error criterion. In fact, it is a method of cluster analysis which aims to partition (N) observations into k clusters in which each observation belongs to the cluster with the nearest mean ($k < N$). The number of clusters is estimated by modified EM algorithm. It starts with a random initial partition and keeps reassigning the patterns to clusters based on the similarity between the pattern and the cluster centers until a convergence criterion is met. A major problem with this algorithm is that it is sensitive to the selection of the initial partition and do not guarantee the global optimum value; therefore, it may converge to a local minimum of the criterion function value if the initial partition is not properly chosen.

To overcome this drawback, many clustering algorithms based on evolutionary algorithms such as GA, TS and SA have been introduced. 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 & 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 & Murty (1999) have presented an approach called genetic k-means algorithm for clustering analysis. Mualik & 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 *et al* (2008) have proposed the HBMO algorithm to solve the clustering problem. A genetic algorithm that exchanges neighbouring centers for k-means clustering has presented by Laszlo & 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 & Wong 2002; Sung & 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 & 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 ICA to solve hard optimization problems, which not only have a better response but also converge very quickly in comparison with ordinary evolutionary methods.

Imperialist competitive algorithm (ICA) is one of the most powerful evolutionary algorithms (Atashpaz-Gargari & Lucas 2007a, 2007b; Rajabioun *et al* 2008a, 2008b; Atashpaz-Gargari *et al* 2008a, 2008b; Roshanaei *et al* 2008; Jasour *et al* 2008). It has been used extensively to solve different kinds of optimization problems. This method is based on socio-political process of imperialistic competition. ICA starts with an initial population. In this algorithm any individual of the population is called a country. Some of the best countries in the population are selected to be the imperialist states and all the other countries form the colonies of these imperialists. After dividing all colonies among imperialists and creating the initial empires, these colonies start moving toward their relevant imperialist country. This movement is a simple model of assimilation policy that was perused by some imperialist states. The movement of colonies toward their relevant imperialists along with competition among empires and also collapse mechanism will hopefully cause all the countries to converge to a state in which there exist just one empire in the world and all the other countries are its colonies. As a result, ICA could 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 we use the simulated annealing (SA) as a local search around the best solution found by MICA algorithm. This approach provides better opportunity of exploring for colonies. To use the benefits of k-means and ICA, 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 paper is organized as follows: In section 2, the cluster analysis problem is discussed. In sections 3 and 4, imperialist competitive algorithm and simulated annealing are described, respectively. In addition, in sections 5–7, modified MICA, the hybrid K-MICA and application of hybrid K-MICA in clustering are shown, respectively. In section 8, the feasibility of the hybrid K-MICA is demonstrated and compared with K-MICA, MICA-K, MICA, ICA, ACO, PSO, SA, GA, TS, HBMO and k-means for different data sets. Finally, section 9 includes a summary and the conclusion.

2. *k*-Means algorithm

The term 'k-means' was first used by MacQueen (1967), though the idea goes back to Hugo Steinhaus in 1956. Lloyd (1982) first proposed the standard algorithm in 1957 as a technique for pulse-code modulation, though it was not published until 1982. It is one of the most popular methods of clustering.

The goal that the algorithm prepends is to minimize the total cost function, which is a squared error function. Each cluster is identified by a centroid. The algorithm follows an iterative procedure. Initially, k cluster is created randomly. Next, the centroid of each group (cluster) is computed. After this, a new partition is built by associating each entry point to the cluster whose centroid is closest to it. Finally, the cluster centers are recalculated for the new clusters. The algorithm is executed until convergence is reached. The cost function is calculated as follows:

For each data vector, assign the vector to the cluster with the closest centroid vector, where the distance to the centroid is determined using

$$F(X, Y) = \sum_{i=1}^{N} \min \sum_{j=1}^{n} (x_{i,j} - y_{i,j})^2, \tag{1}$$

where X denotes the input data vector, Y denotes the centroid vector of cluster, n subscripts the number of features of each centroid vector and (N) the number of data input. Figure 1 shows its flowchart.

There are two problems in k-means as given below:

(i) It is unclear how to choose the initial centers of clusters.
(ii) Number of clusters need to be known in advance.

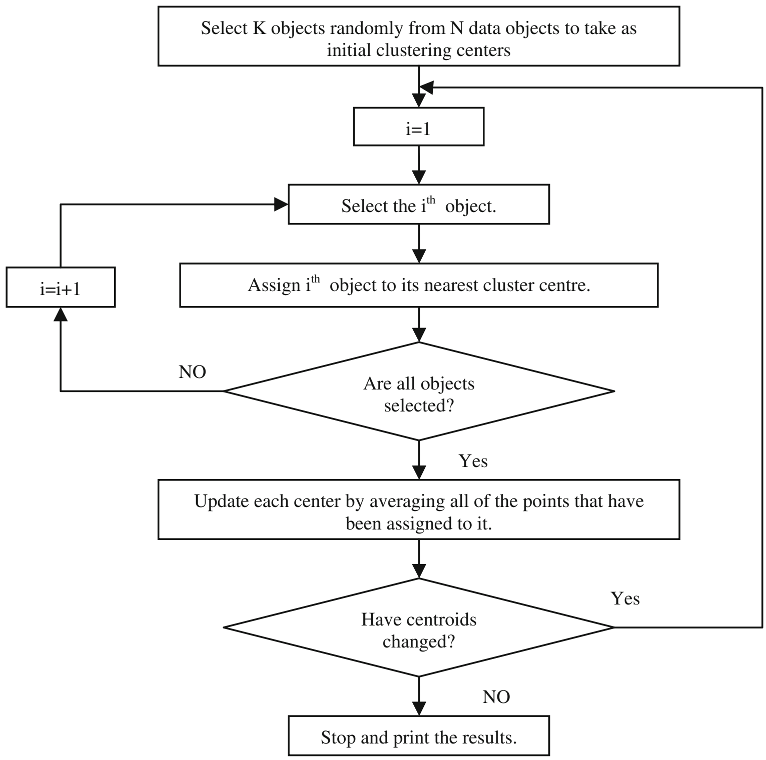

Figure 1. Flowchart of k-mean.

As k-means is an investigative method, there is no guarantee that it will converge to the global optimum, and the result may depend on the initial clusters. Because of the fact that the algorithm is usually very fast, it is common to run it several times with different starting conditions. However, reaching the convergence can be very slow. For solving this problem, ICA is employed to choose initial centers of clusters. The number of cluster centers k, can be known in advance. In practical data sets, it is impossible to determine what the best number of clusters is. Therefore, there are some different methods to find k (Safarinejadian *et al* 2010; Figueiredo & Jain 2002; Tibshirani *et al* 2001). We select one of the approaches proposed by Figueiredo & Jain (2002).

The EM (Expectation Maximization) algorithm is used in mathematical statistics in order to find estimates of the maximum likelihood parameters of probabilistic models, when the model depends on some hidden variables. Each iteration of the algorithm consists of two steps. In the E-step (expectation) the expected value of the likelihood function is calculated, and the latent variables are treated as observed. In the M-step (maximization), the maximum likelihood is estimated, thus increasing the expected likelihood that is calculated in the E-step. Then, this value is used for the E-step to the next iteration. This algorithm will continue until it converges to the answer.

To achieve this goal, the modified EM algorithm has been used (Figueiredo & Jain 2002). First, we initialize the modified algorithm with (N) clusters, which is above the numbers we knew to be presented. Then this algorithm is executed until convergence is reached. At this moment, the weight of each cluster represents the number of data entities associated to that cluster. Next, the

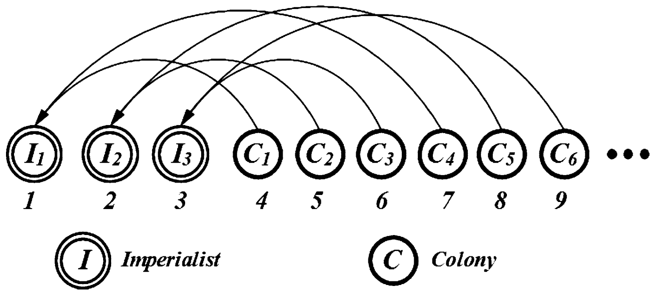

Figure 2. Divide colonies among imperialist.

cluster with zero weight will be removed. This producer will be gone on until finding the proper number of clusters. In this paper, the number of clusters is estimated by modified EM algorithm.

3. Original ICA

Imperialist competitive algorithm (ICA) is one of the most powerful evolutionary algorithms. It has been used extensively to solve different kinds of optimization problems. This method is based on socio-political process of imperialistic competition. ICA is started with an initial population. In this algorithm, each member of the population is called a country. Some of the best countries in the population are selected to be the imperialist states and all the other countries form the colonies of these imperialists. All the colonies of initial population are divided among the mentioned imperialists based on their cost function, respectively; for instance, we assume the number of imperialists is three. In this case, the forth to sixth countries will be considered as the first colony of the empires. After that, the seventh to the ninth countries will be selected respectively as the second colony of the empires. This action will be continued to the entire countries. This conception is illustrated in figure 2.

After all empires were formed, the competition between countries begins. First, the colonies in each of empires start moving toward their relevant imperialist state and change the place in the new position. This movement is shown in figure 3. In this model, a is a random variable with

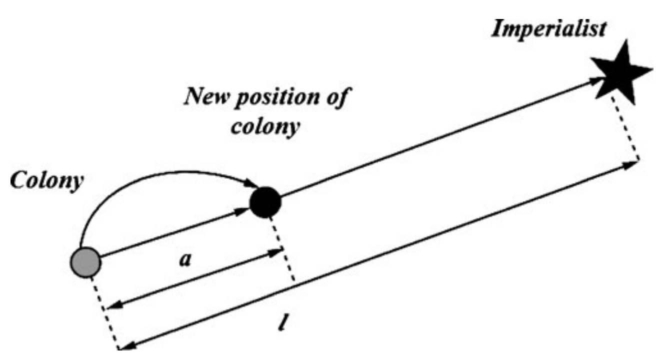

Figure 3. Moving colonies toward their related imperialist.

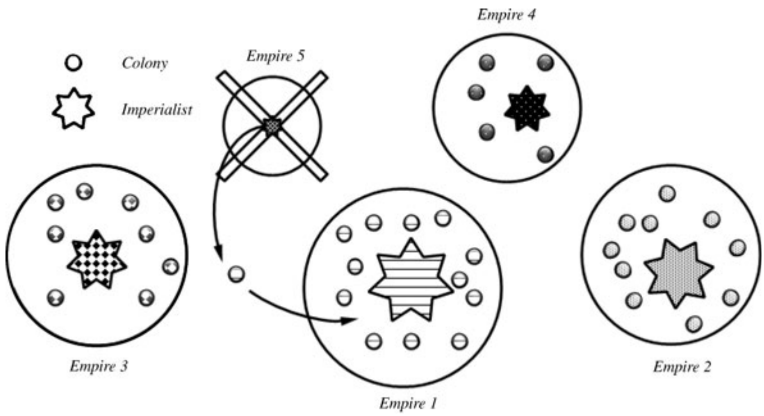

Figure 4. Elimination of the weakest empire.

uniform distribution and β is a number that is greater than one (in most of implementations, β is equal to 2). Also, l is supposed the distance between the colony and the imperialist.

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

During this movement, if the colony gets higher cost than its imperialist does, they will exchange their positions. After that, the algorithm will be continued with the new imperialist. The power of each empire is made up of imperialist cost function and colonies. Competition among empires is the most important part of ICA. It is based on empires power. In this case, the empire which is weaker than the others, loses its colonies until there will be no colony in that. The result of this action is extinction of the weakest empire. After that, its imperialist is considered as the best empire colony. This conception is depicted in figure 4.

The final level of imperialist rivalry is when there is only one empire in the world. This is the optimum point. ICA flowchart is expressed in figure 5.

4. Simulated annealing

Simulated annealing is a generalization of a Monte Carlo method for examining the equations of state and frozen states of n-body systems. The concept is based on the manner in which liquids freeze or metals recrystalise in the process of annealing. In an annealing process, a melt is initially at high temperature, disordered, and is slowly cooled so that the system at any time is approximately in thermodynamic equilibrium. As cooling proceeds, the system becomes more ordered then approaches a 'frozen' ground state at zero temperature. Hence, the process can be thought of as an adiabatic approach to the lowest energy state. If the initial temperature of the system is too low, the system may become quenched (Niknam *et al* 2008a).

The original Metropolis scheme was that an initial state of a thermodynamic system was chosen at energy E and temperature T. Holding T constant, the initial configuration is perturbed and the change in energy, ΔE, is computed. If the change in energy is negative, the new configuration is accepted. If the change in energy is positive, it is accepted with a probability factor $\exp(-\Delta E/T)$. This process is then repeated sufficient times to give good sampling statistics for

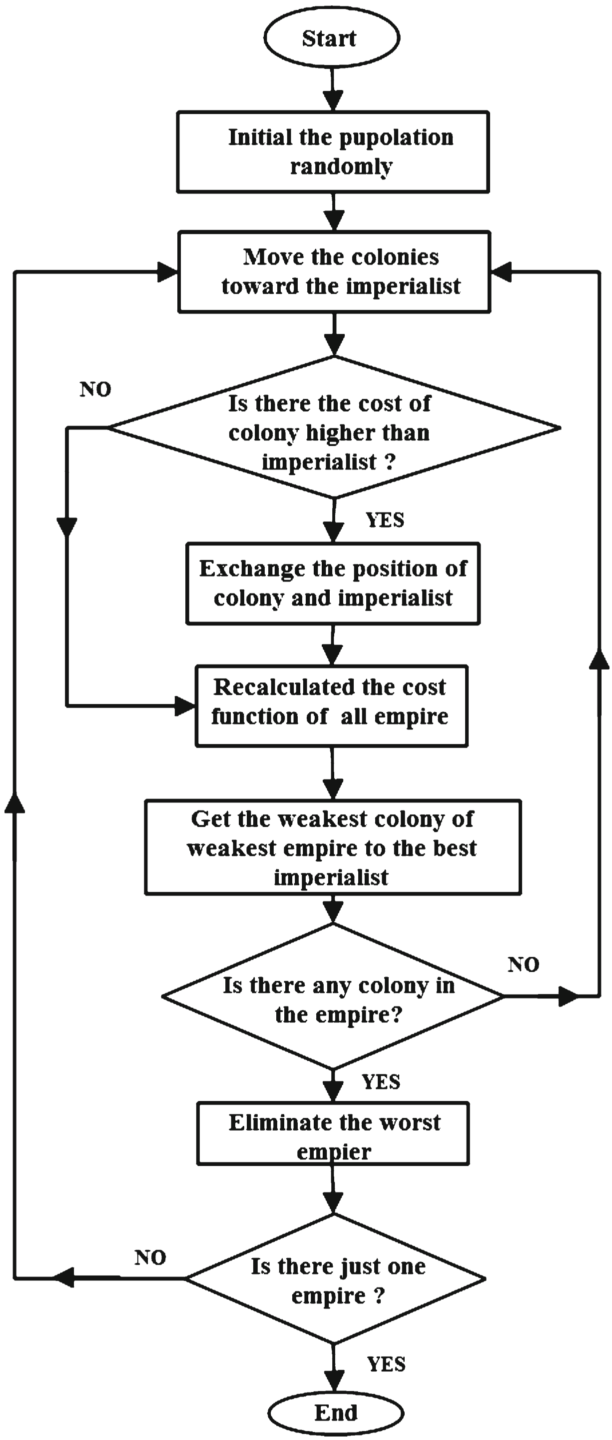

Figure 5. Imperialist competitive algorithm (ICA) flowchart.

the current temperature, and then the temperature is decremented and the entire process repeated until a frozen state is achieved at T = 0.

The general procedure for the SA algorithm can be summarized as follows (Bahmani Firouzi *et al* 2010):

Step 1: Select an initial solution X and an initial temperature T.
Step 2: Find another solution, namely X_{next}, by modifying the last answer X.
Step 3: Calculate the energy differential $\Delta E = f(X_{next}) - f(X)$.
Step 4: If $\Delta E < 0$ go to Step 9.
Step 5: Generate a random number, namely R, between 0 and 1.
Step 6: If $R < \exp(-\frac{\Delta E}{T})$ go to Step 9.
Step 7: Repeat Steps 2–6 for a number of optimization steps for the given temperature.
Step 8: If no new solution, X_{next} is accepted then go to Step 10.
Step 9: Decrease the temperature T, replace X with X_{next} and go to Step 2.
Step 10: Reheat the environment with setting T to a higher value.
Step 11: Repeat Steps 1 through 10 until no further improvement obtained.

5. Modified ICA

In order to improve the convergence velocity and accuracy of the ICA algorithm, this paper proposes a new mutation operator. Premature convergence may occur under different situations:

(i) The population has converged to local optimum of the objective function.
(ii) The population has lost its diversity.
(iii) The search algorithm has proceeded slowly or has not proceeded at all.

Mutation is a powerful strategy to diversify the ICA population and improve the ICA's performance in preventing premature convergence to local minima. The used mutation is described as follows:

Let X illustrate the position for each country in an N-dimensional problem. At first, 10 individuals are generated randomly according to the following equation:

$$\begin{aligned} &X_{mut,i} = X + \varepsilon \qquad i = 1, 2, 3, ..., 10 \\ &\varepsilon = \begin{bmatrix} \varepsilon_1 & \varepsilon_2 & ... & \varepsilon_n \end{bmatrix}_{1\times n} \\ &\varepsilon_i = rand(.). \end{aligned} \tag{3}$$

If the best solution among $X_{mut,i}$ is better than X, it replaces X. Also in the proposed algorithm, the SA algorithm has updated the best solution found by MICA, which is the best local search algorithm. In other words, in the hybrid algorithm, at each iteration, the best solution value is considered as an initial point of the SA algorithm.

6. Hybrid K-MICA

As it was said before, *k*-means is one of clustering algorithms, which can be implemented easily. However, it has some disadvantages. One of them is that its results depend on initial values. Another one is that it converges to local minimum. Recently, numerous ideas have been used to alleviate this drawback by using global optimization algorithms such as GA (Krishna & Murty 1999), TS (Ng & Wong 2002), PSO and hybrid K-PSO (Kao *et al* 2008), hybrid PSO-SA (Niknam *et al* 2008a, 2008b), hybrid PSO-ACO-K (Bahmani Firouzi *et al* 2010), HBMO (Fathian *et al* 2007), hybrid ACO-SA (Niknam *et al* 2008a, 2008b) and PSO-SA-K (Morales & Erazo 2009).The aim of this paper is using the combination method to solve these problems. As a result, we merge both *k*-means and MICA methods.

There are different methods for combination *k*-means with MICA. In the first case, *k*-means is used to generate the population and its output initializes MICA. The second type, MICA

```
Begin
        Generate an initial population randomly
        Estimate the number of k cluster
        Calculate the objective function for the initial population
        Sort the initial population based on their objective function values and select the imperialist states
        Divide colonies among imperialist
        Use k-means algorithm for each empire
          Do {
    Place one colony as 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 populations
                   Compare both new cost and select the best one
                   Replace j th colony with new one
                  }
               While (all colonies selected)
              Sort all colonies of j 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 colonies 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
          Run the SA algorithm while the best solution is considered as its initial point
                          If the output of SA is better than the best solution
                             Swap the best solution and the SA output
           }
          While (there is more than one empire)
          End
```

Figure 6. Pseudo code of hybrid K-MICA.

initializes the population and competition will be done; the last remaining empire will be given to *k*-means. Moreover, in the last one, population is generated with MICA, initial empires form, and then *k*-means is applied to improve the position of empire's colonies and imperialists. The

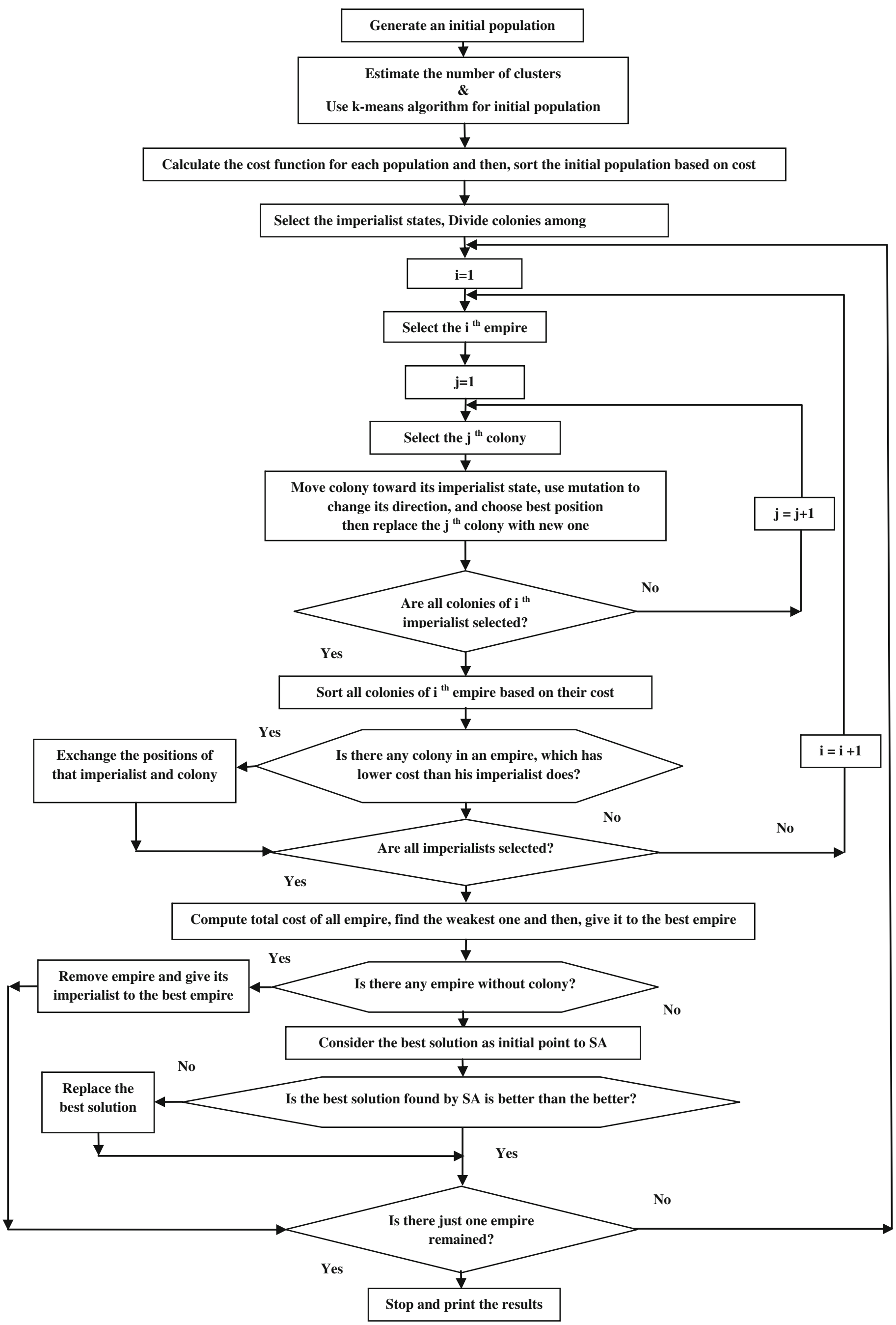

Figure 7. K-MICA flowchart.

result of this algorithm will be given to MICA, again. 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, the 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 converge 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 not to remove the powerless imperialist when it loses all possessions. This imperialist is one of the best answers and can be contributed in imperialistic competition as a weak colony or to be given to the powerful empire. Pseudo-code of hybrid K-MICA and flowchart of K-MICA are shown in figures 6 and 7, respectively.

7. Application of hybrid K-MICA on clustering

There are some input data, which would be clusters and the control variables that are referred to the cluster centers. In this part, the hybrid K-MICA application on clustering problem is shown in some steps as follows:

Step 1: Generate an initial population and then Estimate the number of k cluster

An initial population of input data is created randomly:

$$Population = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_{N_{pop}} \end{bmatrix} \tag{4}$$

$$\begin{aligned} &X_i = Country_i = [Center_1, Center_2, ..., Center_k] \;\; i = 1, 2, ..., N_{pop} \\ &Center_j = [x_1, x_2, ..., x_d] \;\; j = 1, 2, ..., k \\ &x_j^{\min} < x_j < x_j^{\max}, \end{aligned} \tag{5}$$

where X_i is one of the countries, $Center_j$ is j^{th} cluster center for i^{th} country, d is the dimension of each cluster center, N_{pop} is the number of population, and (k) is the number of clusters. $x_j^{\max}$ and $x_j^{\min}$ (each feature of center) are the maximum and minimum values of each point referring to the j^{th} cluster center. By using modified EM algorithm, k will be estimated.

Step 2: Calculate objective function value

The cost function is evaluated for each country as below:

$$F(X) = \sum_{m=1}^{N} \left(\sqrt{(Center_1 - Y_m)^2 + (Center_2 - Y_m)^2 + ... + (Center_K - Y_m)^2} \right). \tag{6}$$
$$(N = number\ of\ input\ data)$$

Step 3: Sort the initial population based on the cost function values

Initial population should be sorted ascending based on cost function. Select N_{col} of the sorted population as follows:

$$X\_sort = \begin{bmatrix} X_{s,1} & F_{s,1} \\ X_{s,2} & F_{s,2} \\ . & . \\ . & . \\ X_{s,N_{col}} & F_{s,N_{col}} \end{bmatrix}_{(N_{col})\times(n+1)} \tag{7}$$
$$N_{col} = (N_{pop} - N_{imp})$$
$$(n = number\ of\ parameters),$$

where, $X_{s,1}$ and $X_{s,N_{col}}$ are the vertices with the lowest and the highest function values, respectively. $F_{s,1}$ and $F_{s,N_{col}}$ represent the corresponding observed function values, respectively. N_{col} is the number of colonies. N_{imp} is the number of imperialists. N_{col_imp} is the number of colonies in each imperialist.

Step 4: Select the imperialist states

Countries with minimum cost function are chosen as the imperialist states and the remaining ones form the colonies of these imperialists.

Step 5: Divide colonies among imperialist

After selecting the imperialists, the remaining countries are assigned to the imperialists based on their cost function value, respectively.

In this method, after sorting the countries due to their cost function, each colony is divided into imperialists. These imperialists search the space independently and in different directions to evolve the solutions. The colonies are portioned as follows:

Matrix *X_sort* is partitioned into N_{imp} imperialists $Y_1, Y_2, ...Y_{N_{imp}}$, each containing N_{col} colonies, such that

$$Y_i = \begin{bmatrix} Y_{i,1} & F_{Y_{i,1}} \\ Y_{i,2} & F_{Y_{i,2}} \\ . & \\ . & \\ Y_{i,N_{col\_imp}} & F_{Y_{i,N_{col\_imp}}} \end{bmatrix}_{N_{col\_imp}\times(n+1)} , \quad i = 1, 2, ..., N_{imp} \tag{8}$$
$$Y_{i,j} = X_{s,(i+N_{imp}\times(j-1))}, \qquad j = 1, 2, .., N_{col\_imp}$$
$$F_{Y_{i,j}} = F_{Y_i,(i+N_{imp}\times(j-1))},$$

For example, for $N_{imp} = 3$, $X_{s,1}$ goes to imperialist 1 (Y_1), $X_{s,2}$ goes to imperialist 2 (Y_2), $X_{s,3}$ goes to imperialist 3 (Y_3), $X_{s,4}$ goes to imperialist 1 (Y_1), and so on.

Step 6: Use k-means algorithm for each empire

- Select an empire and choose one of its colonies, which represent initial group centroids.
- Assign each object (which has to be clustered) to the group that has the closest centroid.
- Recalculate the positions of the centroids.
- Go to step 3 and 4 until the centroids no longer move and these new centroids replace the old colony.
- Repeat step 1 to 6 for all empires.

Step 7: Move colonies toward their imperialist states
Step 8: Use mutation to change the direction of colonies as it is described in modified ICA clearly
Step 9: Check the cost of all colonies in each empire

As it is mentioned, during the colonies movement toward the imperialist, some of them may get better position to reach the imperialist. Indeed, it will be happened when colonies cost function is less than imperialists.

Step 10: Check total cost of each empire

It is possible to calculate each empire cost which is related to the power of both imperialist and its colonies. It is calculated as follows:

$$\begin{aligned} &T.C_n = F\,(imperialist_n) + \xi mean\,\{F\,(colonies\ of\ empire_n)\} \\ &(0 < \xi < 1)\,. \end{aligned} \tag{9}$$

$T.C_n$ is the total cost of n^{th} empire. In addition, ξ, as an attenuation coefficient, is used to reduce the effect of colonies cost ($0 < \xi < 1$).

Step 11: Do imperialistic competition

The probabilities assigned to the countries in the empire are inversely proportional to their cost. A country with the lowest cost has the greatest probability of being imperialist, while the country with the highest cost has the lowest probability of being imperialist. A random number determines which country is selected. This type of weighting is often referred to as Roulette wheel weighting. There are two techniques: rank weighting and cost weighting. We focused on Roulette wheel cost selection.

In cost weighting method, the probability of selection is calculated from the cost of the country rather than its rank in the population. Initially, a normalized cost is calculated for each country by subtracting the lowest cost of the discarded countries ($T.C_{N_{imp}+1}$) from the cost of all the countries in the empire:

$$T.C_n^{norm} = T.C_n - \min\cos t_{N_{imp}+1}\left\{T.C_{N_{imp}+1}\right\}. \tag{10}$$

Subtracting ($T.C_{N_{imp}+1}$) ensures all the costs are negative.

$$p_{p_n} = \left| \frac{T.C_n^{norm}}{\sum\limits_{n=1}^{N_{imp}} T.C_n^{norm}} \right| \tag{11}$$

where $T.C_n^{norm}$ is normalized total cost of the n^{th} empire and the possession probability of each empire is p_{p_n}. Next, it is necessary to calculate cumulative probability as:

$$\begin{aligned} C_{P_l} &= P_{P_l} \\ C_{p_n} &= \sum_{i=1}^{n} p_{p_l}. \end{aligned} \tag{12}$$

According to this equation cumulative probability for the last n is equal to one.

After that, a random number with uniform distribution generates and compares with all C_{Pn}. Each sector with higher probability will have more chance to be chosen. Therefore, the winner empire will be specified. This approach tends to weight the top country more when there is a large spread in the cost between the top and bottom country. The probabilities must be recalculated in each generation. 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

Powerless empires will collapse in the imperialistic competition and their colonies will be divided among other empires. In modelling collapse mechanism, different criteria can be defined for considering an empire powerless. In most of our implementation, when we assume an empire collapsed, we eliminate it as it loses all of its colonies.

Step 13: Run the SA algorithm

Simulated annealing will be run while the best solution is considered as its initial point. If the output of SA is better than the best solution, swap the best solution and the SA output.

Step 14: Check number of empire

If there is just one empire, stop it, otherwise return to step 7.

8. Experimental results

In this section, the scope is to compare 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. We use four real-life data sets *(Iris, Wine, Vowel and Contraceptive Method Choice (CMC))*, and a manufactured data set. They are described as follows:

8.1 *Iris data ($N = 150$, $d = 4$, $K = 3$)*

This data set includes 150 random samples of flowers from the iris species *setosa*, *versicolor*, and *virginica* collected by Anderson (1935). For each species there are 50 observations including sepal length, sepal width, petal length, and petal width in centimeters. This data set was used by Fisher (1936) in his initiation of the linear-discriminant-function technique (Bahmani Firouzi *et al* 2010).

8.2 *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. All attributes are continuous. There are no missing attribute values (Bahmani Firouzi *et al* 2010).

8.3 *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 (Bahmani Firouzi *et al* 2010).

8.4 *Vowel data set ($N = 871$, $d = 3$, $K = 6$)*

This data set consists of 871 patterns. There are six overlapping vowel classes and three input features (Bahmani Firouzi *et al* 2010).

8.5 *Manufactured data set*

This data set consists of 10000 patterns. Milling is one of the most versatile machining processes and it can be product of different shapes including flat surfaces, slots, and contours. Milling machine classification is based on design, operation, and/or purpose. In this part, five models of milling machines can be studied as follow (Huan Min Xu & Dong Bo Li 2008):

- Vertical Milling Machine (VMM)
- Horizontal Milling Machine (HMM)
- Single & Dual-Face modular Milling Machine (SDFMM)
- Plano-Milling Machine (PMM)
- Single & Double Column surface Milling Machine (SDCMM).

For these models, we select six attributes, each attribute in the set is denoted by its corresponding element:

- WT: Maximum weight of work piece (unit: kilogram or kg)
- PW: Power (unit: kilowatt or kw)
- SR: Surface roughness (unit: μm)
- FS: Flatness (unit: mm)
- SS: Maximum speed of spindle (unit: rpm)
- FT: Maximum feeds of table (unit: mm/min).

N_{pop}, N_{imp}, β, ξ, γ are the parameters which are needed for implementation of the hybrid K-MICA that are shown in table 1.

The results of 10 runs of the algorithm for iris data set with different parameters are depicted in the table 2. It illustrates this method is invariant about variable parameters.

The algorithms are implemented by using MATLAB 7.6 on a Pentium IV, 2.8 GHz, 2 GB RAM computer.

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; Niknam *et al* 2009; Bahmani Firouzi *et al* 2010), PSO (Kao *et al* 2008), SA (Sung & Jin 2000), GA (Krishna & Murty 1999), TS (Ng & Wong 2002), HBMO (Fathian *et al* 2008) and k-means (Niknam *et al* 2008b; Niknam *et al* 2009; Niknam & Amiri 2010; Sung & Jin 2000; Bahmani Firouzi *et al* 2010) has been given in tables 3, 4, 5 and 6 on four real-life data sets and table 7 on a manufactured data set for 100 random tails.

For Iris data (table 3) it is found that, the proposed clustering algorithm converges to the optimal value of 96.6554 in all times. The standard deviation of the fitness function for hybrid K-MICA clustering algorithm is zero. This means that proposed algorithm is very precise and reliable. In other words, it provides the optimum value and small standard deviation in comparison to those of other methods.

Table 1. Values of parameters of each of five algorithms.

Hybrid K-MICA		K-MICA and MICA-K		MICA		ICA	
Parameter	Value	Parameter	Value	Parameter	Value	Parameter	Value
N_{pop}	100	cN_{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 $\in [0.5\ 1]$	Final temperature	0.01	Evaporation rate	0.01		
Minimum speed	Randomly $\in [0\ 1]$	Number of Iterations detect steady stat	100	# iterations	1000		
Speed reduction	0.98	# iterations	30000				
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	$\omega_{\min}$	0.5		
# iterations	1000	# iterations	1000	$\omega_{\max}$	1		
				# iterations(7)	500		

The best value obtained by hybrid K-MICA clustering algorithm for Wine data (table 4) is 16292.65 while the other algorithms fail to attain this value in any of their runs. The results presented in table 5 show that the worst result obtained by the proposed algorithm is better than the best result found by other methods. Not only the best result but also the worst result of the proposed algorithm is less than the best results of other methods, which indicates the effectiveness of the proposed algorithm to solve the clustering problem.

The results obtained on the CMC data set show that hybrid K-MICA converges to the global optimum of 5,693.9198 while the best solutions of K-MICA, MICA-K, MICA, ICA, PSO,

Table 2. 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 3. 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 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	16292.65	16293.12	16293.81	0.4331
K-MICA	16293.85	16294.83	16296.5	0.924925
MICA-K	16293.6	16295	16296.8	0.992492
MICA	16293.9	16295.6	16296.94	1.002372
ICA	16295.24	16298.57	16304.61	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

Table 5. 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	5,693.9198	5,694.0012	5694.224	0.018423
K-MICA	5,695.8547	5,696.8659	5698.0194	1.111793
MICA-K	5,694.8723	5,695.5322	5,697.50	1.268275
MICA	5,699.2183	5,705.1485	5721.1779	7.397884
ICA	5,725.7062	5,736.3663	5752.9425	8.000562
PSO	5,700.9853	5,820.9647	5,923.2490	46.959690
SA	5,849.0380	5,893.4823	5,966.9470	50.867200
TS	5,885.0621	5,993.5942	5,999.8053	40.84568
GA	5,705.6301	5,756.5984	5,812.6480	50.3694
ACO	5,701.9230	5,819.1347	5,912.4300	45.634700
HBMO	5,699.2670	5,713.9800	5,725.3500	12.690000
K-means	5,842.20	5,893.60	5,934.43	47.16

Table 6. 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	148,967.2408	148,978.8291	148,999.3829	10.92318
K-MICA	149,279.9922	149,596.3367	149,955.0055	280.0121
MICA-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	2,847.08594
TS	149,468.268	162,108.5381	165,996.4280	2,846.23516
GA	149,513.735	159,153.498	165,991.6520	3,105.5445
ACO	149,395.602	159,458.1438	165,939.8260	3,485.3816
HBMO	149,201.632	161,431.0431	165,804.671	2,746.0416
K-means	149,422.26	159,242.89	161,236.81	916

Table 7. Results obtained by the algorithms for 100 different runs on manufactured data set.

Method	Function value			Standard deviation
	F_{best}	$F_{average}$	F_{worst}	
Hybrid K-MICA	15187. 5385	15187.6238	15187.954	0.104757
K-MICA	15191.0035	15880.054	16569.105	0.666306
MICA-K	15197.518	15211.856	15226.194	0.26355
MICA	15188.7359	15775.317	16363.096	4.102628
ICA	15259.5165	15551.466	158434155	2.118549

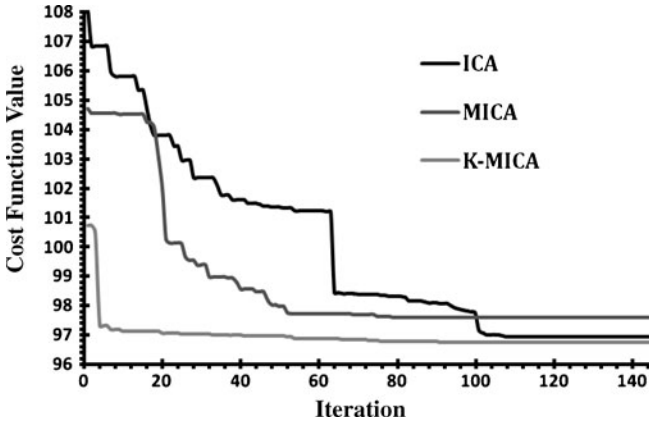

Figure 8. Convergence characteristic of ICA, MICA and K-MICA for the best solutions on Iris data.

SA, TS, GA, ACO, HBMO and k-means are 5695.8547, 5694.8723, 5699.2183, 5725.7062, 5700.9853, 5849.0380, 5885.0621, 5705.6301, 5701.9230, 5699.2670 and 5842.20. The standard deviation of the fitness function for this algorithm is 0.018423, which is significantly less than other methods. Table 6 shows the results obtained on Vowel data set. It can be seen from table 5 that the proposed technique provided better results than those obtained by other methods. Therefore, the proposed method is more robust and practically applicable to real systems.

Table 7 shows the results of algorithms on the manufactured data set. The optimum value obtained by the proposed algorithm is 15187.5385. Noticeably other algorithms fail to attain this value even once within 100 runs.

Figures 8–15 show the convergence characteristics of hybrid K-MICA, K-MICA, ICA-K, MICA, ICA, and k-means for the best solutions on Iris, Wine and Manufactured 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 100 iterations while K-MICA, MICA-K, MICA and ICA converge to the global optimum in about 95, 85, 80 and 110 iterations. The convergence characteristics of these algorithms for Wine data show that combining k-means with MICA converges

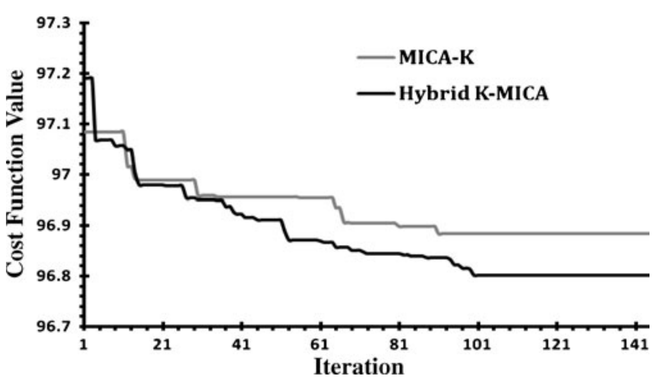

Figure 9. Convergence characteristic of MICA-K and hybrid K-MICA for the best solutions on Iris data.

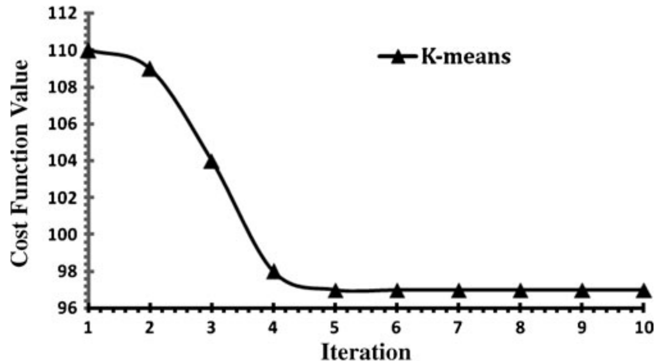

Figure 10. Convergence characteristic of k-means for the best solutions on Iris data.

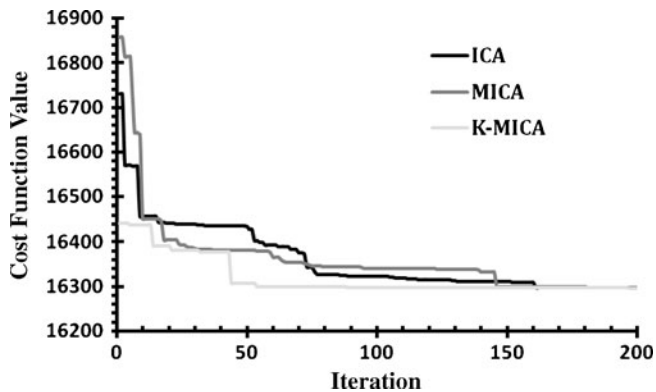

Figure 11. Convergence Characteristic of ICA, MICA and K-MICA for the best solutions on Wine data.

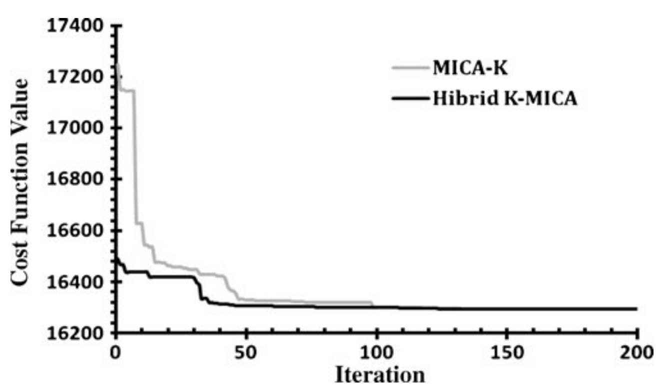

Figure 12. Convergence characteristic of MICA-K and hybrid K-MICA for the best solutions on Wine data.

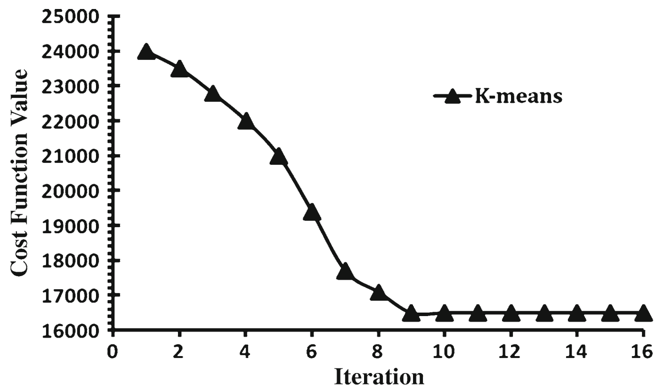

Figure 13. Convergence characteristic of *K*-means for the best solutions on Wine data.

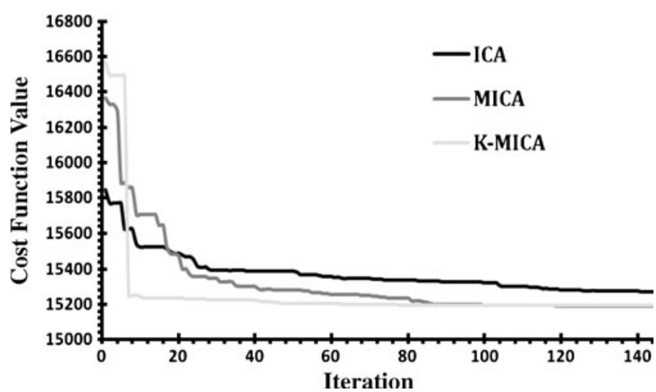

Figure 14. Convergence characteristic of ICA, MICA and K-MICA for the best solutions on a manufactured data.

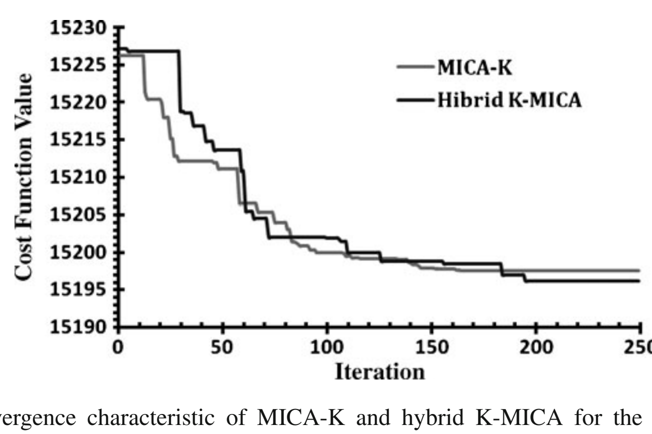

Figure 15. Convergence characteristic of MICA-K and hybrid K-MICA for the best solutions on a manufactured data.

to better results. While ICA and MICA converge to the global solution after 161 and 150 iterations, MICA-K, K-MICA and hybrid K-MICA converge after 140, 70 and 60 iterations. The convergence characteristics of these algorithms for Manufactured data set show that combining k-means with MICA converges to better results. While ICA and MICA converge to the global solution after 150 and 90 iterations, MICA-K, K-MICA and hybrid K-MICA converge after 170, 50 and 200 iterations. In all the data sets, while hybrid K-MICA needs more iterations than MICA-K and K-MICA algorithms, the best cost function will be achieved by it.

9. Conclusion

In this paper, a new hybrid evolutionary algorithm is proposed to find optimal or near optimal solutions for clustering problems of allocating N objects to k clusters. The proposed algorithm is based on a combination of ICA, SA and k-means algorithms called hybrid K-MICA. In this new algorithm, first, we use EM algorithm to find numbers of clusters and then apply k-means for each empire to select the best empires just before competition started by MICA. The algorithm is tested on several well-known real-life data sets. The experimental results indicate that the proposed optimization algorithm is at least comparable to the other algorithms in terms of function evaluations and standard deviations. In addition, the proposed algorithm is applicable when the number of clusters is not known *a priori*. The results illustrate that the proposed hybrid K-MICA optimization algorithm can be considered as a viable and an efficient heuristic.

Authors thank Dr. Behrooz Safarinejadian and Mrs. Shahinpour for their technical discussions.

References

Anderson E 1935 The irises of the Gaspe Peninsula. *Bulletin of the American Iris Society* 59: 2–5

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

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

Atashpaz-Gargari E, Hashemzadeh F, Lucas C 2008a Designing MIMO PID controller using colonial competitive algorithm. *Proceeding of IEEE CEC 2008, within IEEE WCCI 2008* 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. *Int. J. 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. *Int. J. Innovative Computing Information and Control*. 6(4): 1–10

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

Fathian M, B Amiri, Maroosi Ali 2008 A honey-bee mating approach on clustering. *The International Journal of Advanced Manufacturing Technology* 43(9–10): 809–821

Figueiredo M A T, Jain A K 2002 Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24(3): 381–396

Fisher R A 1936 The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 7: 179–188

Huan Min Xu, Dong Bo Li 2008 A clustering-based modeling scheme of the manufacturing resources for process planning. *The International Journal of Advanced Manufacturing Technology* 38(1–2): 154–162

Jasour A M, Atashpaz Gargari E, Lucas C 2008 Vehicle fuzzy controller design using imperialist competitive algorithm. *Second Iranian Joint Congress on Fuzzy and Intelligent Systems*. Tehran, Iran

Kao Y T, Zahar E, I. Kao 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 Cybernet, Part 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

Lloyd 1982 Least square quantization in PCM. *IEEE Transactions on Information Theory* 28(2): 129–137

MacQueen J B 1967 Some Methods for classification and Analysis of Multivariate Observations. *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability* 281–297

Morales A K, Erazo F R 2009 A search space reduction methodology for data mining in large data bases. *Engineering Application of Artificial Intelligence* 22(1): 92–100

Mualik U, Bandyopadhyay S 2000 Genetic algorithm-based clustering technique. *Journal of Pattern Recognition Letters* 33: 1455–1465

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

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 of Science A* 10(4):512–519

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

Rajabioun R, Hashemzadeh F, Atashpaz-Gargari E 2008a Colonial competitive algorithm: a novel approach for PID controller design in MIMO distillation column process. *Int. J. 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. *The International Federation of Automatic Control Congress. Seoul* Korea:9952–9957

Roshanaei M, Atashpaz-Gargari E, Lucas C 2008 Adaptive beamforming using colonial competitive algorithm. *2nd International Joint Conference on Computational Engineering*. Vancouver. Canada.

Safarinejadian B, Menhaj Mohammad B, Karrari Mehdi 2010 Distributed unsupervised gaussian mixture learning for density estimation in sensor networks. *IEEE Transactions on Instrumentation and Measurement* 59(9): 2250–2260

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 Letters* 33(5): 849–858

Tibshirani R, Walther G, Hastie T 2001 Estimating the number of clusters in a data set via the gap statistic. *J. Statistical Soc., Series B* 63(2):411–423

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