A Clustering Algorithm using Cellular Learning Automata based Evolutionary Algorithm

R. RASTEGAR, M. RAHMATI, Member, IEEE, M. R. MEYBODI
Computer Eng. Department, Amirkabir University, Tehran, Iran
{rrastegar, rahmati, meybodi}@ce.aut.ac.ir

Abstract -- **In this paper, a new clustering algorithm based on CLA-EC is proposed. The CLA-EC is a model obtained by combining the concepts of cellular learning automata and evolutionary algorithms. The CLA-EC is used to search for cluster centers in such a way that minimizes the squared-error criterion. The simulation results indicate that the proposed algorithm produces clusters with acceptable quality with respect to squared-error criteria and provides a performance that is significantly superior to that of the K-means algorithm.**

Index Terms -- **Clustering, Cellular Learning Automata, CLA-EC, Optimization.**

I. INTRODUCTION

Clustering is an important unsupervised classification method used in identifying some inherent structure present in a set of data. The purpose of clustering is to group data into subsets that have useful meaning in the context of a particular problem [1]. Various clustering methods have been developed which may be classified into the following categories: hierarchical clustering, learning network clustering, mixture model clustering, objective-function-based clustering, and partition clustering, etc [5][16]. The clustering problem can be stated as finding the clusters such that the between-group scatter is maximized and within-group scatter is minimized. Many heuristic techniques for clustering exist in the literatures, which address the global minimization of squared-error criterion function, Genetic Algorithms (GA) [1][3][4][7] and Simulated Annealing (SA) [6] are two of these techniques.

The Cellular Learning Automata (CLA), which is introduced for the first time in [11], is a mathematical model for modeling dynamical complex systems that consists of large number of simple components. The simple components of CLA, which have learning capabilities, act together to solve a particular problem. This model has been applied to several problems such as image processing [10], channel assignment in cellular mobile system [2], function optimization [13], modeling of rumor diffusion [8], VLSI Placement [19], and modeling commerce networks [9]. In [13], CLA and evolutionary computing are combined to obtain a new model called CLA-EC for optimization problems. This model is capable of performing search in complex, large and multimodal landscape. In this paper, a new clustering algorithm based on CLA-EC is proposed. The CLA-EC is used to search for cluster centers in such a way that minimizes the squared-error criterion. Due to parallel nature of CLA-EC model, the proposed algorithm is appropriate for clustering large data set. In order to demonstrate the effectiveness of the proposed CLA-EC-clustering, 6 different two-dimensional data sets and IRIS data set are considered. Our experimental results of clustering indicate that the CLA-EC based clustering algorithm provides a performance that is significantly superior to that of the K-means algorithm.

The rest of the paper is organized as follows. Section 0II briefly presents the clustering problem. Section III gives a brief review of learning automata, cellular learning automata and CLA-EC model. The proposed clustering algorithm is described in section IV. Section V presents the simulation results for different data sets and the last section (VI) is the conclusion.

II. CLUSTERING

In a clustering problem, a data set, in N-dimensional Euclidean space, $S=\{x_1,\ldots,x_M\}$ is given, where $x_i \in R^N$ and M is the number of data. Considering K clusters, represented by $C_1,\ldots,C_K$,the clusters should satisfy the following conditions,

$$C_i \neq \phi \ , \ i = 1,\ldots,K \ , \ \bigcup_{i=1}^{K} C_i = S \ ,$$

$$C_i \cap C_j = \phi \ , \ i \neq j \ , \ i, j = 1,\ldots,K \ .$$

Among various clustering methods, the K-means method is more attractive than others in practical applications [15]. The K-means clustering algorithm is one of the well-known clustering methods, which is based on an iterative hillclimbing algorithm. One of the most important disadvantages of this algorithm is that it is very sensitive to the initial configuration and may be trapped in a local minimum [15]. Therefore, several approximate methods such as Simulated Annealing (SA) [6] and Genetic Algorithm (GA) [1][3][4][7] have been developed to solve the above problem.

III. THE CLA-EC MODEL

Learning Automata: Learning Automata are adaptive decision-making devices operating on unknown random environments. The Learning Automaton has a finite set of actions and each action has a certain probability (unknown for the automaton) of getting rewarded by the environment of the automaton. The aim is to learn choosing the optimal action (i.e. the action with the highest probability of being rewarded) through repeated interaction on the system. If the learning algorithm is chosen properly, then the iterative process of interacting on the environment can be made to result in selection of the optimal action. Figure 1 illustrates

how a stochastic automaton works in feedback connection with a random environment. Learning Automata can be classified into two main families: fixed structure learning automata and variable structure learning automata (VSLA). In the following the variable structure learning automata is described.

A VSLA is a quintuple $< \alpha, \beta, p, T(\alpha,\beta,p) >$, where α, β, p are an action set with s actions, an environment response set and the probability set p containing s probabilities, each being the probability of performing every action in the current internal automaton state, respectively. Function T is the reinforcement algorithm, which modifies the action probability vector p with respect to the performed action and the received response. Let a VSLA operate in an environment with $\beta=\{0,1\}$. Let $n \in N$ be the set of nonnegative integers that represent instance of iterations. A general linear schema for updating action probabilities can be represented as follows. Let action i be performed at instance n. If $\beta(n)=0$ *(reward)* ,

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)]$$

$$p_j(n+1) = (1-a)\,p_j(n) \qquad \forall j \quad j \neq i$$

If $\beta(n)=1$*(penalty)* ,

$$p_i(n+1) = (1-b)\,p_i(n)$$

$$p_j(n+1) = (b/s-1) + (1-b)\,p_j(n) \quad \forall j \quad j \neq i$$

Where a and b are reward and penalty parameters. When $a=b$, the automaton is called L_{RP}. If $b=0$ the automaton is called L_{RI} and if $0<b<<a<1$, the automaton is called $L_{R\varepsilon P}$. For more Information about learning automata the reader may refer to [12] [17].

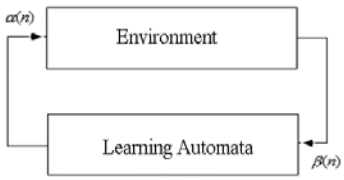


Fig. 1. The interaction between learning automata and environment

Cellular Learning Automata: The Cellular Learning Automata (CLA) [11][20] is a mathematical model for dynamical complex systems that consists of large number of simple components. The simple components, which have learning capabilities, act together to produce complicated behavioral patterns. A CLA is a cellular automata in which learning automaton (or multiple learning automaton) is assigned to its every cell. The learning automaton residing in a particular cell determines its state (action) on the basis of its action probability vector. There is a rule that CLA operate under it. The rule of CLA and the actions selected by the neighboring LAs of any particular LA determines the reinforcement signal to that LA (multiple LA). In CLA, the neighboring LAs of any particular LA constitute its local environment. The local environment is non-stationary because it varies as action probability vector of neighboring LAs vary. The operation of cellular learning automata can be described as follows: At the first step, the internal state of every cell is specified. The internal state of every cell is determined on the basis of action probability vectors of learning automaton (automata) residing in that cell. The initial value may be chosen on the basis of past experience or at random. In the second step, the rule of cellular learning automata determines the reinforcement signal to each learning automaton (automata) residing in that cell. Finally, each learning automaton updates its action probability vector on the basis of the supplied reinforcement signal and the chosen action. This process continues until the desired result is obtained.

The model of CLA-EC (Cellular Learning Automata based Evolutionary Computing) [13]: The CLA-EC model [13] is obtained by combining cellular learning automata and evolutionary computing. This model is capable of performing search in complex, large and multimodal landscape. In CLA-EC, similar to other evolutionary algorithms, the parameters of the search space are encoded in the form of genomes. Each genome has two components, model genome and string genome. Model genome is a set of learning automata. The set of actions selected by this set of learning automata determines the second component of genome (string genome). Based on a local rule, a reinforcement signal vector is generated and given to the set of learning automata. According to the learning algorithm, each learning automaton in the set of learning automata updates its internal state according to a learning algorithm. Then each learning automata in a cell chooses one of its actions using its probability vector. The set of actions chosen by the set of automata residing in a cell determines a candidate string genome that may replace the current string genome. The fitness of this string genome is then compared with the fitness of the string genome residing in that cell. If the fitness of the generated genome is better than the quality of the sting genome of the cell, the generated string genome becomes the string genome of that cell. The process of generating string genome by the cells of the CLA-EC is repeated until a termination condition is satisfied. In order to have an effective algorithm, the designer of the algorithm must be careful about determining a suitable genome representation, fitness function for the problem at hand, the parameters of CLA such as the number of cells (population size), the topology, and the type of the learning automata for each cell. Assume f be a real function that is to be minimized.

$$\min\{f(\xi) \,|\, \xi \in B^m\}$$

where B^m is $\{0,1\}^m$. In order to use CLA-EC for optimization of function f, first a set of learning automata will be associated to each cell of CLA-EC. The number of learning automata associated to a cell of CLA-EC is the number of bits in the string genome representing points of the search space of f. Each automaton has two actions: 0 and 1. The CLA-EC iterates the following steps until the termination condition is met.

Step1: every automaton in cell i chooses one of its actions using its action probability vector.

Step 2: cell i generates a new string genome, η^i, by combining the actions chosen by the set of learning automata of cell i. The newly generated string genome is obtained by concatenating the actions of the automata (0 or 1) associated to that cell.

Step 3: Every cell i computes the fitness value of string genome η^i; if the fitness of this string genome is better than the one in the cell, then the new string genome η^i becomes the string genome of that cell. That is

$$\xi_{n+1}^{i} = \begin{cases} \xi_n^i & f(\xi_n^i) \le f(\eta_{n+1}^i) \\ \eta_{n+1}^i & f(\xi_n^i) > f(\eta_{n+1}^i) \end{cases}$$

where ξ_n^i and η_n^i present the string genome and the new string genome of cell *i* at instance *n*.

Step 4: Se cells of the neighboring cells of the cell *i* are selected. This selection is based on the fitness value of the neighboring cells according to truncation strategy [18].

Step 5: Based on the selected neighboring cells a reinforcement vector is generated. This vector becomes the input to the set of learning automata associated to the cell. Let *Ne(i)* be set selected neighbors of cell *i*. Define,

$$N_{i,j}(k) = \sum_{l \in Ne(i)} \delta(\xi_n^{l,j} = k),$$

where,

$$\delta(\exp) = \begin{cases} 1 & \text{if exp is true} \\ 0 & otherwise \end{cases}$$

$\beta^{i,j}$, the reinforcement signal given to learning automaton *j* of cell *i,* is computed as follows,

$$\beta_n^{i,j} = \begin{cases} u(N_{i,j}(1) - N_{i,j}(0)) & if \ \xi_t^{i,j} = 0 \\ u(N_{i,j}(0) - N_{i,j}(1)) & if \ \xi_t^{i,j} = 1 \end{cases}$$

where *u*(.) is a step function. The overall operation of CLA-EC is summarized in the algorithm of figure 2.

```
While not done do
   For each cell i in CLA do in parallel
      Generate a new string genome;
      Evaluate the new string genome;
      If f(new string genome) < f(old string genome) then
            Accept the new string genome
      End if
      Select Se cells from neighbors of cell i ;
      Generate the reinforcement signal vector;
      Update internal state LAs of cell i
   End parallel for
End while
```

Fig. 2. Pseudocode of CLA-EC

IV. A CLA-EC BASED CLUSTERING ALGORITHM

We propose to use the CLA-EC model to determine the *K* cluster centers of the data set in R^N; thereby clustering the set of *M* points of $S=\{x_1,\ldots,x_M\}$. The sum of the squared distances of the points from their respective cluster centers is taken as the clustering metric, and denotes it by *f*. The aim is to search the cluster centers in such a way that function *f* be minimized. The proposed algorithm consists of three phases: preprocessing phase, the CLA-EC phase and the clustering phase.

A. Preprocessing Phase

The purpose of preprocessing phase is to reduce the size of the search space on which CLA-EC will operate. To reduce the size of the search space, at first the largest and the smallest values of each dimension of data set is found as follows:

$$\min_j = \min_{1 \le i \le M} \{\mathbf{x}_{i,j}\}, \ \max_j = \max_{1 \le i \le M} \{\mathbf{x}_{i,j}\}$$

$$\Delta_j = \max_j - \min_j$$

where $x_{i,j}$ is the *j*th components of x_i. Second, a new search space where we denote it by R' which is $R' = [0,\Delta_1] \times \ldots \times [0,\Delta_N]$ is defined. where × is Cartesian product sign.

B. The CLA-EC Phase

In the CLA-EC phase, clusters are optimized with respect to the squared error criterion. The characteristics of the applied CLA-EC are as follows.

String genome representation: Each string genome is represented by a binary string consisting of $M \times N$ parts where each part is a representation of an encoded real number. Let $\lambda'_{i,j}$ be $(i \times N + j)^{th}$ part of string genome where *j* is the dimension of the center of cluster *i* in R'. If binary representation of $\lambda'_{i,j}$ has w_{ij} bits then in a *N*-dimensional space with *K* clusters, the length of a string genome is $m=\Sigma\Sigma w_{ij}$.

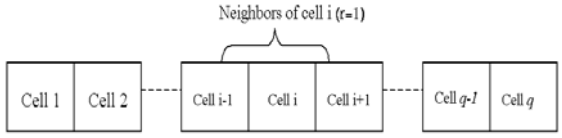
Fitness function (total sum of distances): To compute the fitness value of ξ, at first, we compute $\lambda'_{i,j}$ by decoding ξ, and set $\lambda_{i,j}$ to be $(\lambda'_{i,j}+min_j)$. The fitness value of genome is computed as follows:

$$f(\xi) = f(\boldsymbol{\lambda}_1,\ldots,\boldsymbol{\lambda}_K) = \sum_{i=1}^{M} (A^*_i)^2,$$

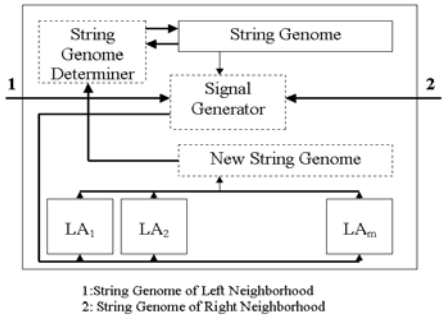
where,

$$A_{i,j} = \left\| \mathbf{x}_i - \boldsymbol{\lambda}_j \right\| \text{ And } A^*_i = \min_{1 \le j \le K} A_{i,j}$$

Parameters of CLA: A one dimensional CLA with wrap around connection and with the neighborhood shown in figure 3a is used. The neighbors of cell *i* are cell *i-1* and cell *i+1*. The architecture of each cell is shown in figure 3b. Each cell is equipped with *m* learning automata. The string genome determiner compares the new string genome with the string genome residing in the cell. The string with the higher quality replaces the string genome of the cell. Depending on the neighboring string genomes and the string genome of the cell, a reinforcement signal will be generated by the signal generator.



(a)



(b)

Fig. 3. The topology of the CLA-EC used in this paper.

Termination Criteria: CLA-EC stops after a pre-specified number of iterations. The best string genome

found in the last iteration is the solution to the clustering problem. For the experimentations that follow the maximum number of iteration is set to 200.

C. The Clustering Phase

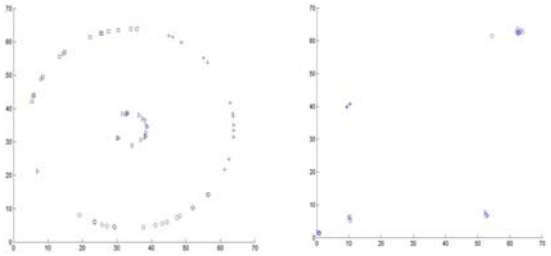
In this phase, the clusters are created using their centers, which are encoded in the best string genome reported by the pervious phase. This is done by assigning each point x_i, $i=1...M$, to one of the clusters C_k with center λ_k such that,

$$C_k = \arg\min_{1\le j\le K} A_{i,j},$$

where

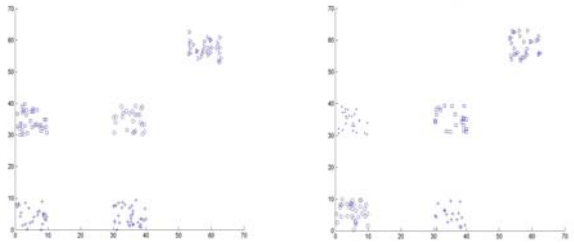
$$A_{i,j} = \left\| \mathbf{x}_i - \boldsymbol{\lambda}_j \right\|$$

All ties are resolved arbitrary.



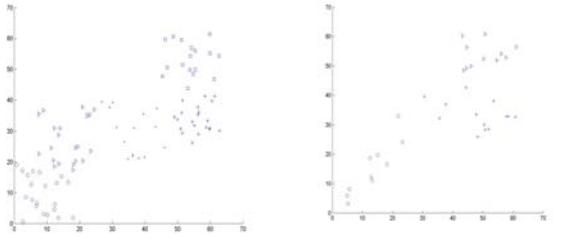
(a) (b)

Fig. 4. (a) Data 1 (b) Data 2.



(a) (b)

Fig. 5. (a) Data 3 (b) Data 4.



(a) (b)

Fig. 6. (a) Data 5 (b) Data 6.

V. SIMULATION RESULTS

Several simulations are conducted in order to evaluate the effectiveness of the proposed method. The results are then compared with the results obtained for K-means algorithm. Simulations are conducted for seven different data sets, which we call them Data 1, Data 2, Data 3, Data 4, Data 5, Data 6 and IRIS Data set. The characteristics of these data sets are given below.

Data 1: a two-dimensional data set with 4 clusters and 59 points as shown in figure 4a.

Data 2: a two-dimensional data set with 4 clusters and 25 points as shown in figure 4b.

Data 3: a two-dimensional data set with 4 clusters and 170 points as shown in figure 5a.

Data 4: a two-dimensional data set with 5 clusters and 128 points as shown in figure 5b.

Data 5: a two-dimensional data set with 5 clusters and 100 points as shown in figure 6a.

Data 6: a two-dimensional data set with 3 clusters and 35 points as shown in figure 6b.

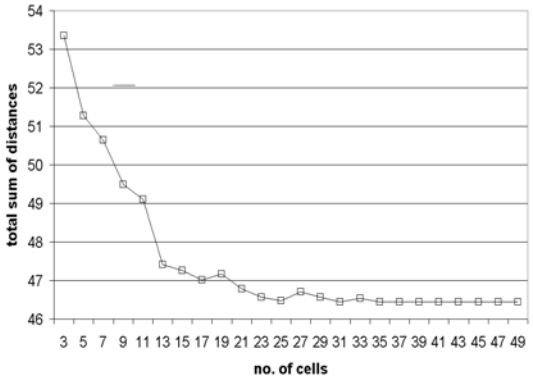
Iris data: This data set represents different categories of irises having four feature values. The four feature values represent the sepal length, sepal width, petal length and the petal width in centimeter. It has 3 clusters with 150 points.

For the sake of convenience in presentation, we use *CLA-EC*(*automata(a,b), se ,q)* to refer to the CLA-EC algorithm with *q* cells, the number of selected cell *Se* and when using learning automata *automata* with reward parameter *a* and penalty parameter *b.*

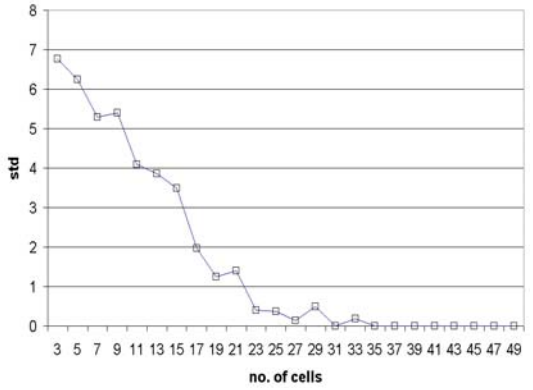
Experiment I: In this experiment we study the effectiveness (quality of clusters found) of the proposed clustering algorithm with respect to CLA-EC parameters such as penalty and reward parameters, the number of cells, and the number of selected cells. Tables 1 of [21] shows the results of the CLA-EC-clustering algorithm for data set Data 1 for different values of the parameters of the CLA-EC such as the type of the learning automata, the penalty and the reward parameters of the learning automata, the number of selected cells and the number of cells. For each simulation maximum number of iterations of CLA-EC is taken to be 200. It is clear that as the mean and the standard deviation decrease the quality of the clustering becomes better. By careful inspection of the results reported in Table 1 [21] it is found that as the number of cells increases, the mean and the standard deviation of the result decreases. Also, it has been found that, better results are obtained when each automaton uses L_{RP} or $L_{R\epsilon P}$ learning algorithm and when *Se* is set to 1. Figure 7 shows the effect of the number of cells in CLA-EC(L_{RP}(0.01,0.01),1,-) on clustering IRIS data set. It is shown that as the number of cells increases the quality of clustering becomes better. Figure 8 shows the fitness of the best genome (solid line) and the mean of the fitness of genomes (dashed line) for each iteration when using CLA-EC(L_{RP}(0.01,0.01),1,5) for Data 1 and Data 4.

Experiment 2: In this experimentation we compare the results of the proposed algorithm with that of K-means algorithm. For this experimentation CLA-EC has 5 cells, each automaton uses L_{RP} learning algorithm with *a=b=0.1, Se* is *1* and the maximum number of iterations is set to be 200. The results of 50 simulations for Data 2 and Data 3 are shown in figure 9. For Data 1 it is found that the CLA-EC-clustering algorithm provides the optimal value of 9502.44 in 28% of the runs whereas K-means algorithm attains this value in 8% of the runs. Both algorithms get trapped at a local minimum for the other runs. For Data 2, CLA-EC-clustering attains the best value of 239.10 in all the runs. K-means, on the other hand, attains this value for 28% of the runs, while in other runs it gets stuck at some of its local minima (such as 3433.77, 3497.16 and 5551.52). For Data 3,

Data 4, Data 5, Data 6, and IRIS data set the CLA-EC-clustering attains the best values of 15545.09, 1873.71, 5525.34, 3000.43, and 46.44 in 30%, 100%, 10%, 40%, %30 of the runs, respectively. The best values attained by the K-means algorithm for these data sets are 15545.09, 1873.71, 5515.34, 3000.43, and 46.44 in 20%, 30%, 2%, 30%, and %24 of runs, respectively. Table 2 shows the summary of results of this experiment. By careful inspection of the results it is found that the CLA-EC(L_{RP}(0.1,0.1),1, 5) performs better than the K-means method for Data 1, Data 2, Data 3, Data 4, Data 4, Data5, Data 6, and IRIS data set.



(a)



(b)

Fig. 7. Effect of the number of cells on the quality of clustering for IRIS data set when using CLA-EC(L_{RP}(0.01,0.01),1, -) -- (a) shows mean and (b) shows the standard deviation over 50 runs.

VI. CONCLUSION

In this paper, a new clustering algorithm based on CLA-EC, called CLA-EC-clustering, was developed. The CLA-EC finds the cluster centers, in such a way that the squared-error criterion be minimized. In order to demonstrate the effectiveness of the CLA-EC-clustering, 6 different two-dimensional data sets and IRIS data set were considered. The results of simulations showed that the CLA-EC-clustering algorithm provides a performance that is significantly superior to that of the K-means algorithm. Due to the parallel nature of CLA-EC, the proposed algorithm is very suitable for clustering large data set.

REFERENCES

[1] Bandyopadhyay, S., and Maulik, U., "An Evolutionary Technique based on K-means Algorithm for Optimal Clustering in R^N ", Information Sciences, No. 146, PP. 221-237, 2002.

[2] Beigy, H., and Meybodi, M. R., "A Self-Organizing Channel Assignment Algorithm: A Cellular Leaning Automata Approach", Vol. 2690 of Springer-Verlag Lecture Notes in Computer Science, PP. 119-126, Springer-Verlag, 2003.

[3] Bhuyan, J. N., Raghavan, V. V., and Elayavalli, V. K., "Genetic Algorithm for Clustering with an Ordered Representation", International Conference on Genetic Algorithms'91, PP. 408-415, 1991.

[4] Garai, G., and Ghaudhuri, B. B., "A Novel Genetic Algorithm for Automatic Clustering", Pattern Recognition Letters, No. 25, PP. 173-1187, 2004.

[5] Jain, A. K., Duin, R. P. W., and Mao, J., "Statistical Pattern Recognition: A Review", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol 22, PP. 4-37, 2000.

[6] Klein, R., and Dubes, R., "Experiments in Projection and Clustering by Simulated Annealing", Pattern Recognition, Vol 22, PP. 213-220, 1989.

[7] Maulik, U., and Bandyopadhyay, S., "Genetic Algorithm-Based Clustering Technique", Pattern Recognition, No. 33, PP. 1455-1465, 2000.

[8] Meybodi, M. R., and Taherkhani, M., "Application of Cellular Learning Automata to Modeling of Rumor Diffusion", in Proceedings of 9th Conference on Electrical Engineering, Power and Water institute of Technology, Tehran, Iran, PP. 102-110, May 2001.

[9] Meybodi, M. R., and Khojaste, M. R., "Application of Cellular Learning Automata in Modeling of Commerce Networks", in Proceedings of 6th Annual International Computer Society of Iran Computer Conference CSICC2001, Isfahan, Iran, PP. 284-295, 2001.

[10] Meybodi, M. R., and, Kharazmi, M. R., "Image Restoration Using Cellular Learning Automata", in Proceedings of the Second Iranian Conference on Machine Vision, Image Processing and Applications, Tehran, Iran, PP. 261-270, 2003.

[11] Meybodi, M. R., Beygi, H., and Taherkhani, M., "Cellular Learning Automata", in Proceedings of 6th Annual International Computer Society of Iran Computer Conference CSICC2001, Isfahan, Iran, PP. 153-163, 2001.

[12] Narendra, K. S., and Thathachar, M. A. L., *Learning Automata: An Introduction*, Printice-Hall Inc, 1989.

[13] Rastegar, R., and Meybodi, M. R., "A New Evolutionary Computing Model based on Cellular Learning Automata", Accepted in IEEE CIS 2004, Singapore, 2004.

[14] Saheb Zamani, M., Mehdipour, M., and Meybodi, M. R., "Implementation of Cellular Learning Automata on Reconfigurable Computing Systems", IEEE CCGEI 2003 Conference, Montreal, Canada, May 2003.

[15] Selim, S. Z., and Ismail, M. A., "K-means-type Algorithm: Generalized Convergence Theorem and Characterization of Local Optimality", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol 6, PP 81-87, 1984.

[16] Yang, M. S., and Wu, K. L., "A Similarity-Based Robust Clustering Method", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 26, No. 4, April 2004.

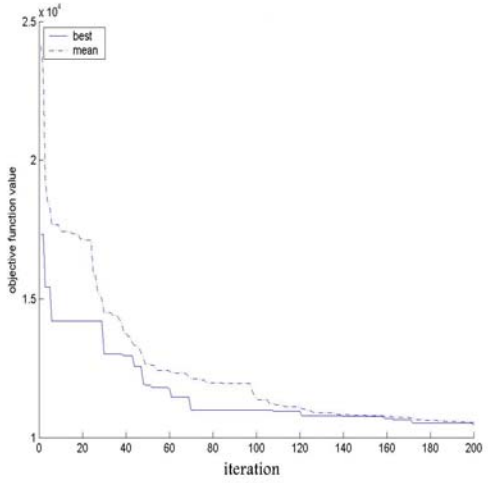
[17] Thathachar, M. A. L., Sastry, P. S., "Varieties of Learning Automata: An Overview", IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 32, No. 6, PP. 711-722, 2002.

[18] Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, New York, 1989.

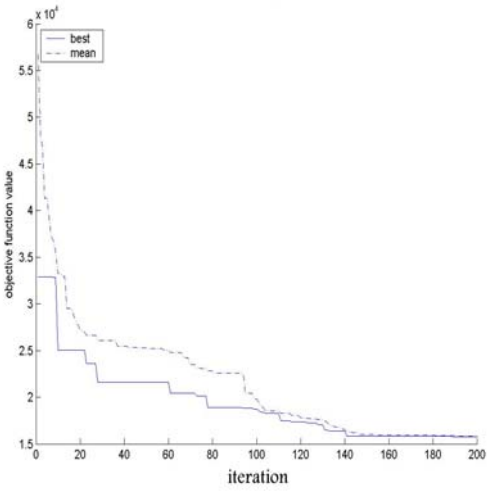
[19] Meybodi, M. R., and Mehdipour, F., "VLSI Placement Using Cellular Learning Automata", in Proceedings of 8th Annual International Computer Society of Iran Computer Conference CSICC2001, Mashhad, Iran, PP. 195-203, 2003.

[20] Beigy, H. and M. R. Meybodi, "A Mathematical Framework for Cellular Learning Automata", Advances in Complex Systems, to appear.

[21] Rastegar, R., Rahmati, M., and Meybodi, M. R., "A New Clustering Algorithm based On CLA-EC", Technical Report, Computer Eng. Department, Amirkabir University, 2004.

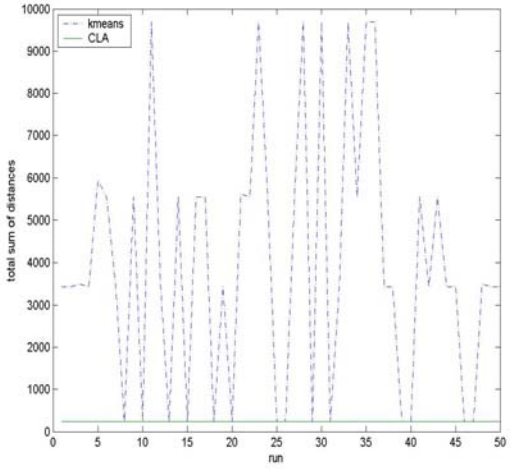


(a)

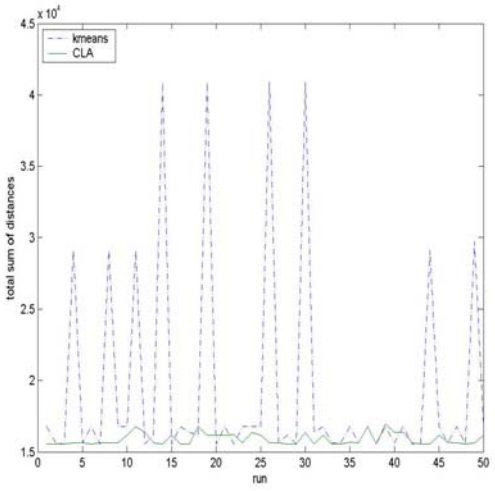


(a)

Fig. 8. The fitness of the best genome (solid line) and the mean of the fitness of genomes (dashed line) for each iteration when using CLA-EC(L_{RP}(0.01,0.01),1,5) for a) Data 1 b) Data 4



(a)



(b)

Fig. 9. Comparison of the K-means and the CLA-EC(L_{RP}(0.1,0.1),1, 5) (a) shows the total sum of distances obtained for K-means and CLA-EC(L_{RP}(0.1,0.1),1, 5) over 50 different runs for Data 2- (b) shows the total sum of distances obtained for K-means and CLA-EC(L_{RP}(0.1,0.1),1, 5) over 50 different runs for Data 3. The solid line is for the CLA-EC(L_{RP}(0.1,0.1),1, 5) and the dashed line is for the K-means algorithm.

TABLE 1. The results of the CLA-EC(L_{RP}(0.1,0.1),1, 5) algorithm (*maximum 200 iterations)* and the K-means algorithm for Data 1,2,3,4,5,6, IRIS - Columns 'Mean' and 'Std' show the mean and standard deviation over 50 runs.

Data Set	(CLA-EC-clustering) Mean	(CLA-EC-clustering) Std	(Kmeans) Mean	(Kmeans) Std
1	10067.92	656.10	10373.33	694.78
2	239.10	0	3980.59	3073.99
3	15889.02	413.71	19457.38	7526.98
4	1873.71	0	8473.58	5424.68
5	5683.50	229.75	5920.9	817.05
6	3078.0	118.17	3206.67	469.77
Iris	51.27	6.25	52.96	8.37