Merged Clustering Imperialists Algorithm (MCIA)

M. A. Soltani-Sarvestani, S. N. Mazloumi, and Hadi Seyedarabi

Abstract—**This paper introduces an improved evolutionary algorithm based on the imperialist competitive algorithm (ICA) called merged clustering imperialists algorithm (MCIA). Merged clustering imperialist algorithm is another version of clustering imperialist algorithm (CIA). The imperialist competitive algorithm that has recently been introduced is used in order to optimize problems and has shown its reliable performance. This novel optimization algorithm is inspired by socio-political process of imperialistic competition in the real world. In the ICA, there are two categories of countries: Imperialists and Colonies that each Imperialist observes its colonies by absorption policy. In the proposed algorithm, the changed ICA is used for clustering data. In the MCIA, according to the number of considered clusters, the Imperialists will be created, and they act like Imperialists in the ICA and absorb their colonies. In this process which colonies move toward Imperialists, the Imperialists borders and regions are specified and by this way whole region is clustered into the number of Imperialists. When the region and border of clusters are specified, with a Merge operator that is different in various clustering problems, the clusters' components merge to each other. Finally, the proposed algorithm is used for clustering an image and a desired result is obtained.**

Index Terms—**Clustering, imperialist competitive algorithm (ICA), repulsion policy, merged clustering imperialists algorithm (MCIA)**

I. Introduction

Evolutionary Algorithms (EA) [1], [2] inspire from nature and they have been used in many applications of network processor (NP) problems and in various fields of science. Some of the famous evolutionary algorithms proposed for optimization problems are: the genetic algorithm (GA), at first proposed by Holland, in 1962 [2]-[4], particle swarm optimization algorithm (PSO) first proposed by Kennedy and Eberhart [5]. In 2007, Atashpaz and Lucas proposed an algorithm as imperialist competitive algorithm (ICA) [6], [7]. This algorithm is inspired by a socio-human phenomenon. Since 2007, several attempts were performed in order to increase the efficiency of the ICA. Zhang *et al*. proposed an approach based on the concept of small probability perturbation to enhance the movement of colonies to imperialist [8]. Faez, Bahrami and Abdechiri introduced a new method using the chaos theory to adjust the angle of colonies movement toward the imperialist's positions [9]. In [10], they proposed another algorithm that applies the probability density function to adapt the angle of colonies movement towards imperialist's position dynamically.

In the imperialist competitive algorithm (ICA) there are two different types of countries, imperialists and colonies, which imperialists absorb their colonies. The imperialists use absorption policy to absorb colonies in different dimensions. In the real world, each country and each imperialist has a specific border which is usually clarified by some spots. In the proposed algorithm each imperialist is considered as the center of each cluster and absorbs colonies toward itself by absorption policy. When colonies move toward imperialists specify the border between them. In this approach the number of considered clusters, the imperialists will be created and colonies specify the borders spots between different clusters during their movement toward Imperialists. So a feature vector is considered and each colony by comparison with its feature vector with all imperialists' vectors recognizes that it is placed in which imperialist and clarifies the border between imperialists.

The rest of this paper is arranged in this order: Section II describes the brief description of imperialist competitive algorithm, Section III presents the proposed algorithm. In Section IV, the result will be analyzed and the performance of algorithms will be evaluated. Section V draws a conclusion from this paper.

Manuscript received November 14, 2011; revised December 13, 2011.

M. A. Soltani-Sarvestani and S. N. Mazloumi are with the Department of Computer Science, University Collage of Nabi Akram, Tabriz, Iran (e-mail: Soltani_mohammadamin@yahoo.com, negar.mazloumi@yahoo.com).

H. Seyedarabi is with the Faculty of Electrical and Computer Engineering, University of Tabriz, Iran (e-mail: seyedarabi@tabrizu.ac.ir).

II. Introduction of Imperialist Competitive Algorithm (ICA)

Imperialist competitive algorithm (ICA) is proposed by Atashpaz and Lucas for the first time, in 2007. ICA is a new evolutionary algorithm in the evolutionary computation (EC) field based on the human's socio-political evolution. The algorithm starts with an initial random population called countries then some of best countries in the population select to be the imperialists and the rest of them form the colonies of these imperialists. The colonies according to imperial power are divided between them. In an N_{var}-dimensional optimization problem, a country is a $1 \times N_{var}$ array. This array is defined as below:

$$country = [P_1, P_2, ..., P_{N\,\mathrm{var}}] \tag{1}$$

The cost of a country is found by evaluating the cost function f at the variables $(p_1, p_2, ..., p_{N_{\mathrm{var}}})$. Then

$$c_i = f(country_i) = f(P_1, P_2, ..., P_{N\,\mathrm{var}}) \tag{2}$$

The algorithm starts with N_{pop} initial countries and the N_{imp} of the most powerful countries which are chosen as imperialists. The remaining countries are colonies belong to imperialists in convenience with their powers. To distribute the colonies among imperialist proportionally, the normalized cost of an imperialist is defined as follow

$$C_n = c_n - \max_i \{c_i\} \quad (3)$$

where c_n is the cost of n^{th} imperialist and C_n is its normalized cost. Each imperialist with more cost value, will have less normalized cost value. By having the normalized cost, the normalized power of each imperialist is calculated as below and based on the colonies distributed among the imperialist countries.

$$P_n = \left| \frac{C_n}{\sum_{i=1}^{Nimp} C_i} \right| \quad (4)$$

P_n is the normalized power of an imperialist. On one hand, the normalized power of an imperialist is assessed by its colonies. Then the initial number of colonies of an empire will be

$$NC_n = rand\{p_n.(N_{col})\} \quad (5)$$

where NC_n is initial number of colonies of n^{th} empire and N_{col} is the number of all colonies. To distribute the colonies among imperialist, NC_n of the colonies is selected randomly and assigned to their imperialist. The imperialist countries absorb the colonies towards themselves using the absorption policy. The absorption policy makes the main core of this algorithm and causes the countries move towards to their minimum optima; this policy is shown in Fig.1. In the absorption policy, the colony moves towards the imperialist by x unit. The direction of movement is the vector from colony to imperialist, as shown in Fig.1. In this figure, the distance between the imperialist and colony is shown by d and x is a random variable with uniform distribution.

$$x \approx U(0, \beta \times d) \quad (6)$$

where β is greater than 1 and is near to 2. So, in [6] is mentioned that a proper choice can be β=2. In ICA algorithm, to search different points around the imperialist, a random amount of deviation is added to the direction of colony movement towards the imperialist. In fig.1, this deflection angle is shown as Θ, which is chosen randomly and with a uniform distribution.

$$\theta \approx U(-\gamma, \gamma) \quad (7)$$

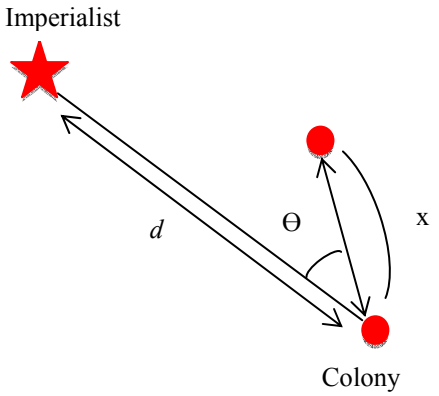


Fig. 1. Moving colonies toward their imperialist [6]

During moving toward the imperialist countries, a colony may reach to a better position, so the colony position changes according to position of the imperialist. The colony position changes according to position of the imperialist.

The imperialists absorb these colonies toward themselves with respect to their power that was described in (8). The total power of each imperialist is determined by the power of its both parts, the empire power plus percent's of its average colonies power.

$$TC_n = \cos t(imperialis\ t_n) + \xi.mean\{\cos t(coloniesof\ empire)\} \quad (8)$$

TC_n is the total cost of the n^{th} empire and ξ is a positive number which is considered to be less than one. In the ICA, the imperialistic competition has an important role. During the imperialistic competition, the weak empire will lose their power and their colonies. To model this competition, firstly the probability of possessing all the colonies is calculated by each empire with considering the total cost of empire.

$$NTC_n = \max_i \{TC_i\} - TC_n \quad (9)$$

where TC_n is the total cost of n^{th} empire and NTC_n is the normalized total cost of n^{th} empire. By having the normalized total cost, the possession probability of each empire is calculated as below:

$$P_{Pn} = \left| \frac{NTC_n}{\sum_{i=1}^{Nimp} NTC_i} \right| \quad (10)$$

After a while all the empires except the most powerful one, will be collapse and all the colonies will be under the control of this unique empire.

III. Merged Clustering Imperialists Algorithm (MCIA)

In this paper, a new algorithm called merged clustering imperialists algorithm (MCIA) is proposed which is originated from the imperialist competitive algorithm (ICA) and is used to cluster data. Another version of this algorithm which is named clustering imperialists algorithm (CIA) is introduced in the ICCEE 2011 [11]. In the ICA, there are two categories of countries, the colonial (or imperialist) and the colony, which the imperialists absorb their colonies by Absorption policy and in this way, the optimum will be found. In the proposed algorithm, the same process continues, but with a difference that colonies specify the different clusters' regions and borders during their movement toward imperialists. This approach specially is useful for continuous environment. As the number of generation increases in the CIA, the accuracy of obtained clusters increases, too.

A. Determining the Boundaries of Each Cluster

In the real world, every country and imperialist has a territory, which is using bordering around an imperialist; the territory of that imperialist can be clarified. Actually, the countries' borders can be considered as a kind of clustering and this fact is used in the proposed algorithm in order to cluster data. In this algorithm like ICA, a random initial

population is created and each individual is considered as a country. Then colonies and colonials specify among these countries. It is considerable that the number of imperialists should be equal to the number of clusters, so the number of clusters should be specified. It means that we should know the number of needed clusters previously. A feature vector like $p=(p_1, p_2,\cdots,p_n)$ is considered for any cluster, where P_i is i^{th} feature like color, intensity, distance and etc. A country which is closest to a cluster's feature vector is considered as imperialist of that cluster. In order to determine the likelihood between each country and different clusters, different distance criterion can be used like the equation (11) or (12).

$$likeness = \sqrt{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2} \quad (11)$$

$$likeness = |p_1 - a_1| + \ldots + |p_n - a_n| \quad (12)$$

p_i is i^{th} feature of a cluster and a_i is i^{th} feature of country a and *likeness* is likelihood rate. Each country that obtains less amount of *likeness* than the other countries, which is using (11) or (12), is selected as imperialist of that cluster. So after selecting imperialists, the rest countries are divided between them as their colonies. The colonies are divided between imperialists randomly and equally. Each colony defaults as (13) and has four properties.

$$Colony\ (Owner, current\_p, old\_p, Direction) \quad (13)$$

The *Owner* indicates the imperialists which colony belongs to, *cuurent_p* indicates the imperialist which colony is in its territory, *old_p* indicates the imperialist which the colony has been in its territory earlier and *Direction* indicates the type of colony movement toward imperialist. The main process of algorithm is controlled by these four properties. So when colonies and imperialists are specified the main process of algorithm will be started. In the main ICA, imperialists using absorption policy to absorb their colonies, but in the proposed algorithm in addition to absorption policy, repulsion policy is also defined. The repulsion policy is a strategy to disperse the colonies in the picture in order to specify clusters.

As it mentioned, each country has four parameters which is using *Direction* and *Owner* that specifies its movement direction. At first, at the start of the algorithm the value of *Direction* is one and this means that colony should get close to the imperialist. So each colony moves toward its imperialist until its distance is be less than a threshold. When the distance between colony and its imperialist is less than threshold, the value of *Direction* changes to -1 and it means that the colony should be get away from its Imperialist until reaches the end of area and then the value of *Direction* changes to 1 again. Fig.2 illustrates the colonies movement. As can be seen in fig. 2, when the value of direction is 1 the absorption policy is used and the colony gets close to its imperialist and when the value of direction is -1 the repulsion policy is used and the colony gets away from its imperialist.

When colonies move by Absorption and Repulsion policies, it uses two others of their parameters, *current_p* and *old_p* in order to specify clusters boundary. *Current_p* indicates the current imperialist which the Colony is located in at current time, and *old_p* indicate the imperialist which the Colony was placed in before. While these two parameters have been equal means that the Colony is in old imperialist, but if these two parameters be different it means that colony is located on a boundary regain between *Current_p* and *old_p* and this point recorded as a boundary point between these two Imperialists. Finally, by connecting boundary points of each cluster, the border of that cluster will be specified.

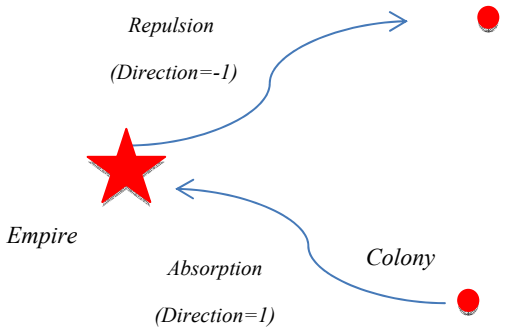


Fig. 2. The colonies movement

In order to determine clusters' regions, each colony puts a flag of the current imperialist in each iteration. Finally, by determining the point on the edges and in the clusters and connecting the proportional point together, the clusters will be specified.

B. Merge Operator

The proposed algorithm is dependent highly on the generation's number. So, when the time of clustering is considerable, the number of generation should be less than normal cases. To solve the time problem, the merge operator is defined. The merge operator is used to connect the components of clusters. When the low number of generation is considered, some of the clusters' borders and region points are specified and all data is not clustered. These borders and region points connect to each other by merge operator. The merge operator is different to various clustering problems and in this paper one of them is introduced in order to clustering pictures.

C. Median Merge Operator

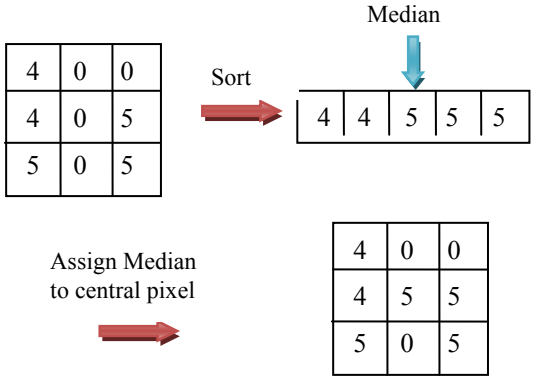


Fig. 3. The action of Median Merge operator

The merge operator which is used in this paper is named median merge operator. This operator is used for clustering a picture, and the similarity criteria is considered the rate of

Gray-level of pixels. First of all, a neighbourhood should be considered for each pixel. Maybe, in each neighbourhood there are some pixels that are not assigned to clusters. These pixels that have no clusters are named Zero-pixels. This operator is acted on Zero-pixels. A neighbourhood is considered for any Zero-pixel and after eliminating Zero-pixels, they will be sorted in an array. The median of this array is assigned to the centers of the pixels. Fig.3 illustrates this process for a 3×3 neighbourhood.

IV. Evaluation and Experimental Results

In this paper, we introduced a new algorithm based on the ICA called merged clustering imperialists algorithm (MCIA) and in order to test its performance, it is used for clustering image pixels. In these example similarity criteria is considered the rate of Gray-level of pixels and the goal is divided an image to 4 different clusters. So clustering performs as follow

$$\begin{aligned} &Cluster\ 1: \quad Intensity \in [0,63] \\ &Cluster\ 2: \quad Intensity \in [64,127] \\ &Cluster\ 3: \quad Intensity \in [128,191] \\ &Cluster\ 4: \quad Intensity \in [192,255] \end{aligned} \tag{14}$$

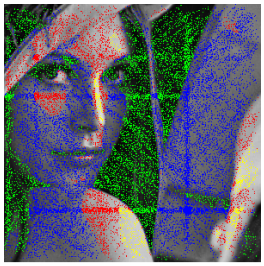
As it mentioned, first an initial random population will be created and each individual is considered as a country. Then Imperialists should be selected between these countries. According to (14), clusters' centers are respectively gray-levels: 31, 95,159 and 223. Based on (15), among all countries, each one which has more similarities with these clusters' center will be selected as the Imperialist of that cluster.

$$\begin{aligned} &Empire\ 1 = \min\left\{\sqrt{(31-a_i)^2}\right\} \\ &Empire\ 2 = \min\left\{\sqrt{(95-a_i)^2}\right\} \\ &Empire\ 3 = \min\left\{\sqrt{(159-a_i)^2}\right\} \\ &Empire\ 4 = \min\left\{\sqrt{(223-a_i)^2}\right\} \end{aligned} \quad ,i = 1,2,...,N_{cnt} \tag{15}$$

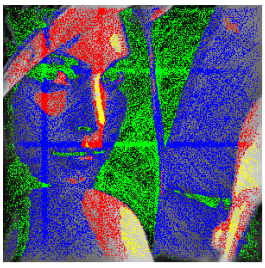
N_{cnt} is the number of countries. Then the algorithm will be started by these properties. Fig. 4 illustrates the results of applying the algorithm on an image with different generations. Fig. 4 (a) illustrates the main image. Fig. 4 (b) shows the result after 200 generation. Figures 4(c), 4(d), 4(e) and 4(f) respectively show the results after 500, 1000, 3000 and over 3000 generations. As can be seen in fig. 4(b), in the early generations the algorithm can specify an overview of each cluster. By continuing the algorithm to more generations, algorithm acts better and fitness will be increase. In the fig. 4 (e) clustering is nearly performed and there is only a little error, but as can be seen in fig. 4 (f), the image is clustered without any errors. Since the Merge operator has not been used, this algorithm is like CIA [11].

As can be seen in Fig. 4, in order to achieve a desirable result, the CIA should apply over 3000 generation, and it causes to pass a long time. In regards to cases that the time is not important, the CIA action is acceptable, but to cases that the time is important this approach doesn't work properly. To solve time problem, Merge operator is used. As can be seen in fig.4 (b) the carefulness of the CIA is up to 200 generation which is not exact. Fig. 5 illustrates the result of the MCIA and CIA up to 200 generations. As can be seen in fig. 5 after applying the CIA, all data are not clustered and there are only some separate components, but all clustered data are clustered exactly, without any errors. As can be seen in fig. 5 (b) by MCIA all data are clustered, but some of them are located in wrong cluster. The carefulness of CIA is more than MCIA, but the time of running of MCIA is less than CIA.

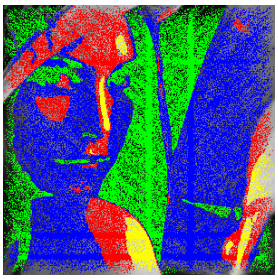
(a) Original Image



(b) The result up to 200 generation



(c) The result up to 500 generation

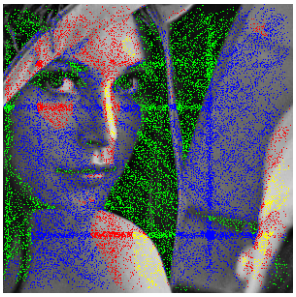


(d) The result up to 1000 generation


(e) The result up to 3000 generation


(f) The result after 3000 generation

Fig. 4. Results of applying the CIA [11] on an image with different number of generations


(a) CIA

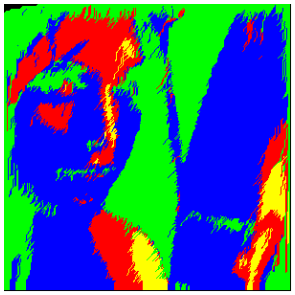

(b) MCIA

Fig. 5. The result of applying CIA and MCIA up to 200 generation

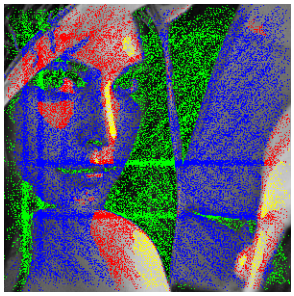
Fig. 6 illustrates the result of applying MCIA and CIA up to 400 generations. As can be seen in fig. 6 the CIA acts weakly in comparison with MCIA. The MICA can make a desirable result after 400 generations.

Fig.7 illustrates the result of applying CIA up to 3000 generations and MCIA after 800 and 1000 generations. As can be seen in fig. 7 the result of applying MICA up to 800 or 1000 generations is nearly like applying CIA up to 3000 generations. As can be seen in fig.7(c) there are some black points in the image. These black pixels are pixels which have not been clustered yet. As can be seen in fig 7(a) and 7(b) the MCIA has a problem to specify borders between clusters. So, both algorithms have some errors, but the time of applying MCIA is less than CIA.

TABLE I: THE ERRORS OF MCIA AND CIA WITH DIFFERENT GENERATION NUMBER

Number of Generation	Error in MCIA %	Error in CIA %
100	19.2122	85.5967
200	14.0189	76.4811
300	11.7200	67.9678
400	9.8811	61.0589
600	7.7700	50.96
800	6.1533	40.6367
1000	5.1600	34.7589
1500	4.3133	23.5289
2000	3.9022	16.4256
2500	3.3778	11.97
3000	3.1422	9.4722
5000	2.6222	4.9133
10000	2.2822	2.0789
20000	2.1878	1.3744

Table I illustrates the error of both algorithms with different generation number. As can be seen, in low generations, the error of MCIA is considerably less than CIA, but as the number of generation is increasing the error of CIA is less than MCIA.


(a) CIA


(b) MCIA

Fig. 6. The result of applying CIA and MCIA up to 400 generation


(a) the result of MCIA up to 800 generations


(b) the result of MCIA up to 1000 generations


(c) the result of CIA up to 3000 generations

Fig. 7. The result of applying CIA and MCIA

V. CONCLUSIONS

In this paper a new algorithm is introduced based on the imperialist competitive algorithm (ICA) which is called merged clustering imperialists algorithm (MCIA). In the ICA, there are two groups of countries: Imperialist and colony that imperialists absorb colonies using absorption policy. The main application of ICA is in optimization problems and this algorithm has shown its desired performance in optimization problems. In the MCIA, there are also two categories of countries like ICA. In the proposed algorithm, colonies move toward Imperialists and get away from them respectively using absorption and repulsion policies. The action of MCIA is like ICA, but instead of exploring the optimum, it search for clusters' borders and ultimately specifies the clusters' borders by connecting the border's points. In the proposed algorithm, by increasing the generation's number, clustering is performed more carefully. Eventually, it was observed that the merged clustering imperialists algorithm (MCIA) can be a proper solution in order to cluster data and has a desired performance in this field.

REFERENCES

[1] H. Sarimveis and A. Nikolakopoulos, "A Life Up Evolutionary Algorithm for Solving Nonlinear Constrained Optimization Problems," *Computer and Operation Research* vol. 32, no. 6, pp.1499-1514, 2005.

[2] H. M¨uhlenbein, M. Schomisch, and J. Born, "The Parallel Genetic Algorithm as Function Optimizer," in *Proc. 4th International Conf. on Genetic Algorithm,* University of California, San diego, 1991, pp. 270-278 .

[3] J. H. Holland, "ECHO: Explorations of Evolution in a Miniature World," In: J.D. Farmer and J. Doyne, editors, in *Proc. 2ND Conf. on Artificial Life,* 1990.

[4] M. Melanie, "An Introduction to Genetic Algorithms," Massachusett's: MIT Press, 1999.

[5] J. Kennedy and R.C. Eberhart, "Particle Swarm Optimization," in *Proc. of IEEE*, pp. 1942-1948, 1995.

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

[7] E. Atashpaz-Gargari, F. Hashemzadeh, R. Rajabioun, and C. Lucas, "Colonial Competitive Algorithm: A novel approach for PID controller design in MIMO distillation column process," *International Journal of Intelligent Computing and cybernetics (IJICC)* vol. 1 no. 3, pp. 337-355, 2008.

[8] Y. Zhang, Y. Wang, and C. Peng, "Improved Imperialist Competitive Algorithm for Constrained Optimization," *International Forum on Computer Science-Technology and Applications*, 2009.

[9] H. Bahrami, K.Feaz and M. Abdechiri, "Imperialist Competitive Algorithm using Chaos Theory for Optimization (CICA)," in *Proc. of The 12th International Conf. on Computer Modelling and Simulation*, 2010.

[10] H. Bahrami, K.Feaz and M. Abdechiri, "Adaptive Imperialist Competitive Algorithm (AICA)," in *Proc. of the 9th IEEE international Conf. on Cognitive Informatics (ICCI'10)*, 2010.

[11] S. N. Mazloumi, M. A. Soltani-Sarvestani, and Hadi seyedarabi, "Clustering Imperialist Algorithm (CIA)," in *Proc. of the 4th International Conference on Computer and Electrical Engineering (ICCEE 2011)*, 2011.

Mohammad-Amin Soltani-Sarvestani was borned in Shiraz, Iran in 1982. He received his bachelor degree from Shiraz Azad University, Iran, in 2005. After five years that he spent on work, he decided to continue his education in field of Computer science and Artificial Intelligence. He got his MSc degree in Artificial Intelligence from University College of Nabi Akram, Iran. His basic research is about Optimization.

Mohammad-Amin, after receive his bachelor degree, worked in many field of Computer engineering include Programming, Database and Animation. His current activities is studying about optimization and data minig.

Seyede Negar Mazloumi was borned in tonekabon, Iran in 1986. She received his bachelor degree from allameh mohandese nori University, Iran in 2009. She got his MSc degree in Artificial Intelligence from University College of Nabi Akram, Iran.

Hadi Seyedarabi received his B.S. degree from University of Tabriz, Iran, in 1993, the M.S. degree from K.N.T. University of technology, Tehran, Iran in 1996 and Ph.D. degree from University of Tabriz, Iran, in 2006 all in Electrical Engineering. He is currently an assistant professor of Faculty of Electrical and Computer Engineering in University of Tabriz, Tabriz, Iran. His research interests are image processing , computer vision, Human-Computer Interaction, facial expression recognition and facial animation.