New generation of metaheuristics by inspiration from ancient

Sasan Harifi
Department of Computer Engineering, Karaj Branch
Islamic Azad University
Karaj, Iran
s.harifi@kiau.ac.ir

Madjid Khalilian
Department of Computer Engineering, Karaj Branch
Islamic Azad University
Karaj, Iran
khalilian@kiau.ac.ir

Javad Mohammadzadeh
Department of Computer Engineering, Karaj Branch
Islamic Azad University
Karaj, Iran
j.mohammadzadeh@kiau.ac.ir

Sadoullah Ebrahimnejad
Department of Industrial Engineering, Karaj Branch
Islamic Azad University
Karaj, Iran
ibrahimnejad@kiau.ac.ir

***Abstract*—Recently, the development of new metaheuristic algorithms has become very expansive. This expansion is especially evident in the category of nature-inspired algorithms. Nature is indeed the source of the solution in many problems, but the developed algorithms in this category used almost the same procedure for optimization. Before the development of nature-inspired algorithms, evolutionary-based algorithms were introduced. It seems that there is a need for some kind of change in this area. This change can be found in the new generation of algorithm development inspired by the ancient era. Ancient inspiration brings together all the positive aspects of nature and evolution. This paper discusses some applications of the ancient-inspired Giza Pyramids Construction (GPC) algorithm compared to the nature-inspired Emperor Penguins Colony (EPC) algorithm. Applications discussed in this paper include improving k-means clustering and optimizing the neuro-fuzzy system. Results from experiments show that the ancient-inspired GPC algorithm performed superior and more efficiently than algorithms inspired by other sources of inspiration.**

Keywords—metaheuristic; nature-inspired; ancient-inspired; clustering; k-means; neuro-fuzzy system;

I. Introduction

Metaheuristic algorithms are used as a soft computing method to solve complex problems. These algorithms use specific search methods and strategies to reach the optimal solution in the little time as possible [1]. Recently, the development of these algorithms has accelerated and they are used in almost every field from engineering to medicine.

So far, various categories for metaheuristic algorithms have been proposed by researchers. In general, these categories are nature-inspired, trajectory-based, evolutionary-based, and ancient-inspired. Of course, nature itself includes subcategories such as swarm, bio, physics/chemistry, human, and plant. The ancient-inspired category is a new category that has been recently introduced. This category has many of the good features found in other categories.

One of the efficient algorithms in the field of soft computing is the Emperor Penguins Colony (EPC) [2] algorithm, which is in the category of nature and subcategory of the swarm. This algorithm has been used in some applications. Some of its applications such as resource allocation [3], optimizing the neuro-fuzzy system [4], inventory control problem [5], and so on have already been reviewed.

In this paper, the only proposed ancient-inspired algorithm, namely the Giza Pyramids Construction (GPC) [6], has been used in two applications that are previously solved by the EPC algorithm. Applications discussed in this paper include improving k-means clustering and optimizing the neuro-fuzzy system. Also, two popular and state-of-the-art algorithms namely the Genetic Algorithm which is abbreviated to GA [7], and Particle Swarm Optimization which is abbreviated to PSO [8] have been compared with the EPC and GPC algorithms.

In the literature, in the application of k-means clustering Oliveira *et al* [9] improved this method though distributed scalable metaheuristics. They used two metaheuristics which were scalable. The scalable word means that their algorithm automatically searches the solution with the best clustering structure and an optimal number of clusters for scalable datasets. The first was enhanced k-means based on evolutionary operators. The second was applied evolutionary k-means to cluster in an independent way. Their method was compared with other clustering algorithms.

García *et al* [10] combined the k-means clustering with a special learning approach to carry out the process of binarization. For this purpose, they used Black Hole (BH) and Cuckoo Search (CS) metaheuristic algorithms. Gupta *et al* [11] improved k-means for the location-allocation problem using metaheuristics. They applied PSO, Ant Colony Optimization (ACO) and Bat Algorithm (BA) for this purpose. They concluded that with metaheuristics, about 20 to 25 percent improvement has been achieved.

In other studies, Kaur *et al* [12] proposed the combination of the Firefly Algorithm (FA) with the k-means method for using in the intrusion detection system. Their hybridization approach used clustering to create the training sample and used classification to measure the test set. They compared their method with other metaheuristics such as CS and BA. Tiacharoen [13] proposed an adaptive k-means method based on metaheuristic algorithms for image segmentation. For this purpose, the GA and PSO were applied. He claims that his method provides better output in comparison with other methods.

978-1-7281-8566-8/20/$31.00 ©2020 IEEE

To improve the k-means method Li *et al* [14] utilized the Improved Gravitational Search Algorithm (IGSA). They also compared their method with the traditional k-means, standard GSA, and PSO. Their simulation demonstrated that IGSA had preferable efficiency in terms of clustering modality. Shelokar *et al* [15] used ACO for clustering. They showed that the ACO approach had better than other metaheuristics such as GA, simulated annealing (SA), and Tabu Search (TS).

In the application of the neuro-fuzzy system, Zheng *et al* [16] utilized the DBBO that is Differential Biogeography-Based Optimization for tuning the parameter of a suggeno neuro-fuzzy system. Based on experiments, their method had an efficient performance. Taghavifar *et al* [17] used Differential Evolution (DE) to improve suggeno type neuro-fuzzy system. Their proposed method used to model wheel dynamics. Obo *et al* [18] operated Evolution Strategy (ES), Evolutionary Programming (EP), and GA for tuning the membership function parameters in the neuro-fuzzy system. Classification of human gestures was an application of their neuro-fuzzy system. Pandiarajan and Babulal [19] utilized a combination of Harmony Search (HS) metaheuristic algorithm with the fuzzy system to find the appropriate solutions in the power flow problem. Precup *et al* [20] used Gray Wolf Optimizer (GWO) for tuning parameters of a fuzzy control system. Chen *et al* [21] used three algorithms such as GA, PSO, and DE for training the Adaptive Neuro-Fuzzy Inference System (ANFIS). Karaboga and Kaya [22] reviewed many ANFIS training methods that have been studied by other researchers.

The rest of the paper is structured as follows: Section II introduces two algorithms namely EPC and GPC. Section III includes the applications and results. Section IV represents conclusions.

II. EPC AND GPC ALGORITHMS

A. *Description of the EPC algorithm*

In this subsection, the EPC algorithm previously published in [2] is briefly described. This algorithm modeled the group behavior of emperor penguins in Antarctica. Penguins formed the huddles to stay warm and work together to manage these huddles. After forming the huddle, the penguins coordinate spiral-like movements to reach the same amount of heat. The computational requirements of this algorithm include calculating the penguin's body surface area, calculating the penguin's body radiation, calculating the attraction, which is the amount of radiant heat from the body surface with a coefficient of attenuation of the penguin's body heat, and calculating the spiral-like movement. According to the reference, the body surface area is 0.56 m^2. Radiation is obtained through the following equation,

$$Q_{\text{penguin}} = A\varepsilon\sigma T_s^4 \tag{1}$$

where A is the surface area of the body, ε is the bird plumage emissivity and its value is 0.98. σ is the constant of Stefan-Boltzmann which its value is 5.6703×10^{-8} W/m^{2}K^4. Inside the mass, we have temperature that indicates by T and its value is 308.15 Kelvin based on references. The above equation must be combined with the calculation of attenuation of thermal photons to obtain the equation of attractiveness, so we have,

$$Q = A\varepsilon\sigma T_s^4 e^{-\mu x} \tag{2}$$

where x is the distance between two heat sources and μ is the attenuation coefficient. One of the important factors in determining the rate of convergence is this μ parameter. The value of μ must be considered as a positive value.

The attractiveness equation is placed in the equation of the coordinated spiral-like movement. In the original code of the algorithm, this equation is the logarithmic spiral type. However, other spirals are also used in [23]. Fig. 1 shows how the penguin moves and transitions.

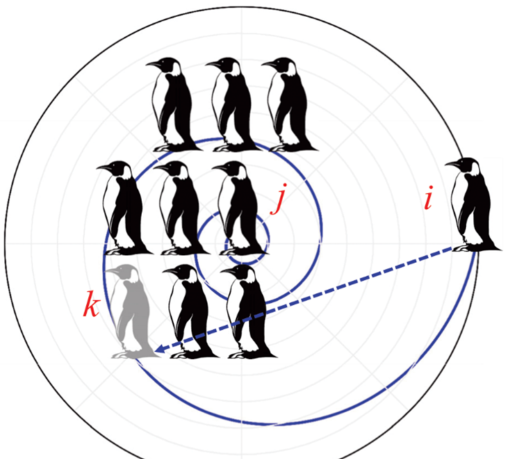

Fig. 1. Emperor penguin spiral-like movement schema [2]

Fig. 1 shows that assuming that the movement from the penguin is i to j, the presence of Q causes the movement to be incomplete and only a coefficient of complete movement to be performed, so the stopping point is the point k. The equation of spiral movement is as follows,

$$\begin{aligned} x_k &= ae^{b\frac{1}{b}\ln\{(1-Q)e^{b\alpha}+Qe^{b\beta}\}}\cos\left\{\frac{1}{b}\ln\{(1-Q)e^{b\alpha}+Qe^{b\beta}\}\right\} \\ y_k &= ae^{b\frac{1}{b}\ln\{(1-Q)e^{b\alpha}+Qe^{b\beta}\}}\sin\left\{\frac{1}{b}\ln\{(1-Q)e^{b\alpha}+Qe^{b\beta}\}\right\} \end{aligned} \tag{3}$$

where a and b are constants that are arbitrarily set. Their value is set between zero and one. Q is the equation of attractiveness. x and y are logarithmic spiral parameters. α and β are $tan^{-1}\frac{y_i}{x_i}$ and $tan^{-1}\frac{y_j}{x_j}$, respectively. Finally, the above equation is summed with a random vector to prevent the production of uniform solutions. So we have,

$$Eq. 3 + \varphi\epsilon_i \tag{4}$$

where φ is the mutation factor and ϵ is the random vector. After the spiral movement was done, the existence of $\varphi\epsilon_i$ helps the penguin to move a bit from its current position. Also, this equation helps the algorithm to have diversity. The EPC algorithm pseudo-code is described in Fig. 2.

```
STEP 1:
    generate initial population array of EPs (Colony Size);
    generate position of each EP;
    generate cost of each EP;
    calculate heat radiation (Eq. 1);
    determine initial attenuation coefficient ($\mu$);
    determine initial mutation coefficient ($\varphi$);
    STEP 2: for FirstIteration to MaxIteration do
        generate repeat copies of population array (as new_pop);
        STEP 3: for i=1 to n do (all n penguins)
            STEP 4: for j=1 to n do (all n penguins)
                if $cost_j < cost_i$ then
                    calculate attractiveness (Eq. 2);
                    calculate coordinated spiral movement (Eq. 3);
                    determine new position (Eq. 4);
                    evaluate new solutions of new_pop;
                end if
            END STEP 4
        END STEP 3
        merge population array with new_pop array;
        sort and find best solution;
        update heat radiation (decrease);
        update attenuation coefficient (decrease);
        update mutation coefficient (decrease);
    END STEP 2
END STEP 1
```

Fig. 2. Pseudo-code of the EPC algorithm

Decreasing algorithm parameters can be done by any method. These decreasing can be exponential or linear. In this paper, the exponential manner is used in all experiments and applications. Therefore, the value of the parameter is multiplied with a damping coefficient in each iteration.

B. Description of the GPC algorithm

In this subsection, the GPC algorithm previously published in [6] is briefly described.

This algorithm is inspired by ancient and introduces a new method of inspiration. The main idea of this algorithm is the method of building the pyramids in Egypt. These huge structures have gone far beyond the structures of their time. Due to various limitations in ancient times, a kind of optimization is seen in the construction of pyramids. Some workers are always trying to move the stone blocks to the installation place. The workers compete to become Pharaoh's special agent if they perform best. On the other hand, due to work pressure, if a worker cannot work well, it will be replaced by another. The computational requirements of this algorithm include calculating the amount of stone block movement, the amount of worker movements, estimating the new position of the block and the worker, and examining the probability of worker substitution.

The amount of motion of the stone block is calculated through the equations related to the ramp. See Fig. 3. According to figure, for a stone block, we have,

$$d = \frac{v_0^2}{2g(\sin\theta + \mu_k \cos\theta)} \tag{5}$$

where d is the displacement value of the block. θ is the angle of ramp that is usually considered between 8 and 14. g is the gravity. v_0 is the initial velocity of the stone block, which is considered a random number between zero and one. μ_k is the kinetic friction coefficient.

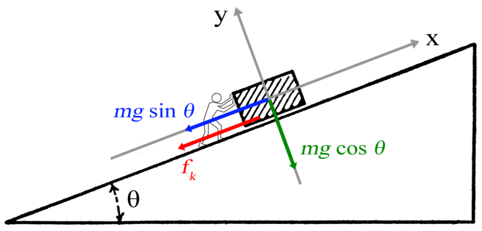

Fig. 3. The forces acting on the object [6]

The movement of the worker is done when moving the stone block to gain more control over the block. Friction is avoided for this movement. So we have,

$$x = \frac{v_0^2}{2g\sin\theta} \tag{6}$$

where x is the amount of worker movement when pushing the stone block. To specify the new situation of every worker and block, the two Eq. 5 and Eq. 6 are combined as follows,

$$\vec{p} = (\vec{p_i} + d) \times x\vec{\epsilon_i} \tag{7}$$

where $\vec{p_i}$ is the current situation, x is the worker displacement, d is the displacement value of the stone block, and $\vec{\epsilon_i}$ is a random vector. The existence of a random vector is to avoid uniformity of solutions. One of the interesting points in GPC is that according to the type of problem, the accumulative form of the above equation can be used to obtain more desirable solutions.

After determining the new position, the possibility of workers substitution is examined. Thus, if the primary solutions are $\Phi = (\varphi_1, \varphi_2, \dots, \varphi_n)$ and the generated solutions are $\Psi = (\psi_1, \psi_2, \dots, \psi_n)$, with a fifty percent probability, the solutions generated with the primary solutions are replaced. The result is $Z = (\zeta_1, \zeta_2, \dots, \zeta_n)$, so we have,

$$\zeta_k = \begin{cases} \psi_k, \text{if } rand[0,1] \le 0.5 \\ \varphi_k, \text{otherwise} \end{cases} \tag{8}$$

The GPC algorithm pseudo-code is described in Fig. 4.

```
STEP 1:
  generate initial population array of stones or workers;
  generate position and cost of stone block or worker;
  determine best worker as Pharaoh's agent;
  STEP 2: for FirstIteration to MaxIteration do
    STEP 3: for i=1 to n do (all n stone blocks or workers)
      calculate amount of stone block displacement (Eq. 5);
      calculate amount of worker movement (Eq. 6);
      estimate new position (Eq. 7);
      investigate possibility of substituting workers (Eq. 8);
      determine new position and new cost;
      if new_cost < Pharaoh's agent cost then
          set new_cost as Pharaoh's agent cost;
      end if
    END STEP 3
    Sort solutions for next iteration;
  END STEP 2
END STEP 1
```

Fig. 4. Pseudo-code of the GPC algorithm

III. Description of Applications and Experimental Results

This section describes the experiments, settings, parameters, and other required information. Also, in each of the subsections, explanations related to the application and its results are given. As mentioned earlier, in addition to the two EPC and GPC algorithms, the GA and PSO algorithms have also been used in the experiments. The parameters of each algorithm for each application are given in Table I. Some parameters for algorithms are set so that the algorithm performs best. It is not bad to mention this point that the crossover used for the GA is the arithmetic type and uses the Gaussian mutation. Also, the number of function evaluations (NFE) used as a stop condition in all algorithms. The value of calls for NFE is given in Table I.

TABLE I. Values of Parameters for All Algorithms

Alg	Parameters	Values in clustering	Values in neuro-fuzzy
GA	NFE calls	10000	15000
	Population size	20	30
	Percentage of crossover	0.8	0.7
	Percentage of mutation	0.3	0.5
PSO	NFE calls	10000	15000
	Swarm size	20	30
	Inertia weight	1	1
	Damping ratio of inertia weight	0.99	0.99
	Coefficient of Personal learning	2	1
	Coefficient of Global learning	2	2
EPC	NFE calls	10000	15000
	Colony size	20	30
	Damping ratio of heat radiation	0.98	0.98
	Coefficient of Attenuation	1	1
	Damping ratio of attenuation coefficient	0.98	0.98
	Coefficient of mutation	0.4	0.4
	Damping ratio of mutation coefficient	0.98	0.2
	Arbitrary value of a	0.2	0.2
	Arbitrary value of b	0.5	0.5
GPC	NFE calls	10000	15000
	Population size	20	30
	Gravity	9.8	9.8
	Ramp angle	14	10
	Initial velocity	rand(0, 1)	rand(0, 1)
	Least friction	1	1
	Most friction	10	10
	Probability of substitution	0.5	0.5

A. *Optimizing k-means clustering*

Before describing the application and the experiment results, it should be noted that the application of this subsection in [24] has been done by popular metaheuristic algorithms.

One of the heuristic method used for partitional clustering is the k-means [25]. In this method, selecting the best centroids is effective in receiving appropriate solutions. This clustering method is an optimization problem that metaheuristic algorithms can be used to solve it. The metaheuristic algorithm can help to select the best centroids. To do this, we need an appropriate objective function. One of the appropriate objective functions for this task is to calculate the within-cluster distance. Using this objective function, the total distance of every member from the centroids of the cluster is obtained. If this value reaches its minimum, it means that the clustering is done in the best way. So in general, we have,

$$\text{within cluster distance} = \sum_{i} \sum_{x \in c_i} d(x, m_i) \tag{9}$$

where x is the data, d is the distance that calculated as Euclidean space, and m is the center of the cluster. Also, c is i^{th} cluster. The metaheuristic algorithm tries to do the objective function minimization by examining the centroids and changing it, and since the number of clusters is already known, so the total objective function is equal to,

$$\text{obj func} = \sum_{j=1}^{k} \sum_{x \in c_j} d(x, m_j) = \sum_{i=1}^{n} \min_{1 \le j \le k} d(x_i, m_j) \tag{10}$$

Since the objective function is used as internal quality measurement, we need another quality measurement as external quality. This can be the error rate. The error rate is equal to,

$$\text{ER} = \frac{f}{n} \times 100 \tag{11}$$

where f is the number of objects that are misplaced, and n is the number of objects within dataset.

Experiments on this subsection were performed on seven real-world datasets. The main features of these seven datasets are listed in Table II.

TABLE II. The Datasets Characteristics for Clustering

Dataset	Attributes	Clusters	Instances
#1 Balance Scale (BS)	4	3	625
#2 Breast Cancer (BSW)	9	2	699
#3 CMC	9	3	1473
#4 Dermatology	34	6	366
#5 Glass Identification (GI)	9	6	214
#6 Haberman's Survival	3	2	306
#7 Iris	4	3	150

These mentioned datasets are available in [26]. The results of the experiments are also presented in Table III.

TABLE III. THE MEAN AND STANDARD DEVIATION OF WITHIN-CLUSTER DISTANCES AND ERROR RATE RECEIVED BY ALGORITHMS

Dataset	Criteria	Algorithms			
		GA	*PSO*	*EPC*	*GPC*
#1 BS	Mean StD ER (%)	1426 1.99 22.73	1424 0.92 21.09	1424 0.86 10.51	**1424** **0.42** **8.36**
#2 BCW	Mean StD ER (%)	3038 6.29 0.82	3028 0.09 0.72	**3028** **0.006** **0.36**	3028 0.01 0.38
#3 CMC	Mean StD ER (%)	5611 63.08 3.02	5534 3.95 3.56	5532 0.02 1.59	**5530** **0.008** **1.02**
#4 Der	Mean StD ER (%)	2644 47 8.98	2240 25 7.78	2195 40 3.99	**2184** **17** **3.24**
#5 GI	Mean StD ER (%)	290.03 15.84 36.79	244.82 11.56 34.70	**226.59** **13.31** **15.63**	228.41 14.23 17.35
#6 HaS	Mean StD ER (%)	2568 0.72 19.60	2567 0.29 22.03	2567 0.16 8.48	**2567** **0.09** **8.32**
#7 Iris	Mean StD ER (%)	99.24 6.10 7.42	97.46 4.39 8.13	96.65 0.0003 3.27	**96.61** **0.0001** **3.14**

B. Optimizing a neuro-fuzzy system

Before describing the application and the experiment results, it should be noted that the application of this subsection in [4] has been done by the EPC algorithm and other popular metaheuristic algorithms.

In the reference paper [4], ANFIS that is a neuro-fuzzy system is created. The steps of making this fuzzy system are as follows: first, an initial FIS is created through Fuzzy c-means (FCM). It is then learned by metaheuristic algorithms and finally evaluated. What the metaheuristic does is adjust the parameters of the membership function so that the evaluation criteria are minimized. The criteria of mean error, the standard deviation of the error, Mean Square Error (MSE), and Root Mean Square Error (RMSE) were used for evaluation.

In this case, according to Fig. 5, the error can be calculated by calculating the difference between the output of the system and the output of the model.

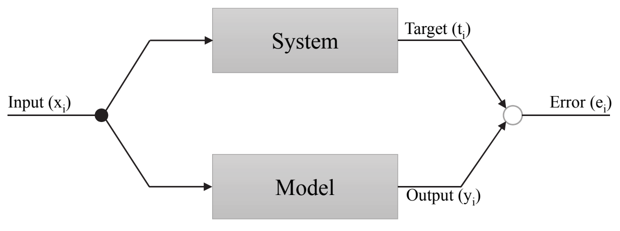

Fig. 5. Fuzzy system schema

The error e_i is achieved from the calculation of the difference between t_i and y_i namely we have $e_i = t_i - y_i$. Therefore, the MSE can be written for all possible instances as follows,

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} e_i^2 \tag{12}$$

MSE can be considered as a cost function or an objective function. The RMSE is also used for directly interpretable in terms of measurement units. We have,

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} e_i^2} \tag{13}$$

Experiments on this subsection were performed on seven real-world datasets. The main features of these seven datasets are listed in Table IV. These datasets used are available in [26].

TABLE IV. MAIN CHARACTERISTICS OF DATASETS FOR NEURO-FUZZY

Dataset	FCM-based intended clusters	Attributes	Number of Parameters (Dimensions)
#1 Abalone Shell Rings (ASR)	10	8	250
#2 Body Fat Percentage (BFP)	10	13	400
#3 Breast Cancer (BCW)	2	9	56
#4 Chemical Sensor (CS)	10	8	250
#5 House Pricing (HP)	10	13	400
#6 Iris	3	4	39
#7 Wine	3	13	120

TABLE V. THE RESULTS OF APPLYING METAHEURISTIC ALGORITHMS ON DATASETS FOR TRAINING PHASE

Dataset	Criteria	Algorithms			
		GA	*PSO*	*EPC*	*GPC*
#1 ASR	Mean StD MSE RMSE	-0.2408 2.1239 4.5654 2.1367	-9.90e-05 2.1451 4.5998 2.1447	0.0085 2.0702 **4.2842** **2.0698**	0.0092 2.1151 4.2898 2.0712
#2 BFP	Mean StD MSE RMSE	1.72e-14 4.1001 16.715 4.0884	5.61e-15 4.0613 16.4006 4.0498	0.2522 3.6472 13.2899 3.6455	-6.05e-15 3.5481 **13.0776** **3.6163**
#3 BCW	Mean StD MSE RMSE	-0.0191 0.1826 0.0336 0.1834	-0.0002 0.1677 0.0280 0.1675	-0.0052 0.1254 0.0157 0.1254	-0.0024 0.1623 **0.0130** **0.1141**
#4 CS	Mean StD MSE RMSE	0.1527 0.3253 5.4151 2.3270	1.61e-13 2.3856 5.6748 2.3822	-2.23e-14 2.2469 **5.0341** **2.2437**	1.61e-15 2.3112 5.0355 2.2440
#4 HP	Mean StD MSE RMSE	0.0105 4.1845 17.4607 4.1786	-0.0448 2.9421 8.6335 2.9383	0.1587 3.3601 11.2834 3.3591	0.0347 2.7386 **8.4558** **2.9079**
#6 Iris	Mean StD MSE RMSE	0.0019 0.0903 0.0080 0.0899	0.0004 0.0236 0.0005 0.0235	-0.0017 0.0183 0.0003 0.0183	-0.0007 0.0142 **0.0002** **0.0141**
#7 Wine	Mean StD MSE RMSE	0.0003 0.1661 0.0273 0.1654	0.0001 0.1355 0.0182 0.1350	0.0015 0.1030 0.0105 0.1026	0.0018 0.1010 **0.0100** **0.1004**

Table V demonstrated the results of the training phase obtained by the neuro-fuzzy system optimized with metaheuristic algorithms. It should be noted that 70% of the data were used for system training.

IV. Conclusions

In this paper, the Giza Pyramids Construction (GPC) algorithm, which is a novel algorithm inspired by ancient, was applied in two applications, including optimization of k-means clustering and optimization of the parameter in a neuro-fuzzy system. Both applications were previously solved by popular metaheuristic and state-of-the-art algorithms. For experiments, the GPC algorithm was compared with the GA, PSO, and EPC algorithms. For clustering application, mean and standard deviation from 30 independent runs were recorded. Clustering error rates were also measured. The results showed that the two algorithms namely GPC and EPC were much more successful than the GA and PSO algorithms. Also, the GPC algorithm performed slightly better than the EPC algorithm. In terms of the clustering error rate, the GPC algorithm has recorded fewer errors. The GPC algorithm performed better than other algorithms to optimize the neuro-fuzzy system parameters, which were tested with four criteria. In addition, the EPC algorithm was also successful and had a performance close to GPC. Overall, the results showed that the new approach based on ancient can somewhat revolutionize the development of metaheuristic algorithms. With the development of ancient-inspired algorithms, we can see more powerful algorithms than ever before. As future works, the GPC algorithm can be applied and evaluated in other issues, such as engineering problems.

References

[1] T. Weise, "Global optimization algorithms-theory and application," Self-Published Thomas Weise, 2009.

[2] S. Harifi, M. Khalilian, J. Mohammadzadeh, and S. Ebrahimnejad, "Emperor Penguins Colony: a new metaheuristic algorithm for optimization," Evolutionary Intelligence Vol. 12, pp. 211-226, 2019.

[3] S. Alghamdi, "Emperor based resource allocation for D2D communication and QoF based routing over cellular V2X in urban environment (ERA-D 2 Q)." Wireless Networks, Vol. 26, pp. 3419–3437, 2020.

[4] S. Harifi, M. Khalilian, J. Mohammadzadeh, and S. Ebrahimnejad, "Optimizing a Neuro-Fuzzy System based on nature inspired Emperor Penguins Colony optimization algorithm." IEEE Transactions on Fuzzy Systems. Vol. 28, pp. 1110-1124, 2020.

[5] S. Harifi, M. Khalilian, J. Mohammadzadeh, and S. Ebrahimnejad, "Optimization in solving inventory control problem using nature inspired Emperor Penguins Colony algorithm." Journal of Intelligent Manufacturing, pp. 1-15, 2020.

[6] S. Harifi, J. Mohammadzadeh, M. Khalilian, and S. Ebrahimnejad, "Giza Pyramids Construction: an ancient-inspired metaheuristic algorithm for optimization." Evolutionary Intelligence, pp. 1-19, 2020.

[7] JH. Holland, "Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence," MIT press, 1992.

[8] J. Kennedy, and R. Eberhart, "Particle swarm optimization," In Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4, pp. 1942-1948, 1995.

[9] GV. Oliveira, FP. Coutinho, RJGB. Campello, and MC. Naldi, "Improving k-means through distributed scalable metaheuristics," Neurocomputing, Vol. 246, pp. 45-57, 2017.

[10] J. García, B. Crawford, R. Soto, and G. Astorga, "A clustering algorithm applied to the binarization of swarm intelligence continuous metaheuristics," Swarm and evolutionary computation, Vol. 44, pp. 646-664, 2019.

[11] R. Gupta, SK. Muttoo, and SK. Pal, "Meta-Heuristic Algorithms to Improve Fuzzy C-Means and K-Means Clustering for Location Allocation of Telecenters Under E-Governance in Developing Nations," International Journal of Fuzzy Logic and Intelligent Systems, Vol. 19, pp. 290-298, 2019.

[12] A. Kaur, SK. Pal, and AP. Singh, "Hybridization of K-Means and Firefly Algorithm for intrusion detection system," International Journal of System Assurance Engineering and Management, Vol. 9, pp. 901-910, 2018.

[13] S. Tiacharoen, "Adaptive K-means image segmentation based on meta heuristic algorithm," In 2018 International Workshop on Advanced Image Technology (IWAIT), pp. 1-3, 2018.

[14] K. Li, J. Che, B. Wang, J. Zhang, F. Wang, and Z. Mi, "A meta-heuristic optimization based residential load pattern clustering approach using improved Gravitational Search Algorithm," In 2018 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), pp. 1-5, 2018.

[15] PS. Shelokar, VK. Jayaraman, and BD. Kulkarni, "An ant colony approach for clustering," Analytica Chimica Acta, Vol. 509, pp. 187-195, 2004.

[16] YJ. Zheng, HF. Ling, SY. Chen, and JY. Xue, "A hybrid neuro-fuzzy network based on differential biogeography-based optimization for online population classification in earthquakes," IEEE Transactions on Fuzzy Systems, Vol. 23, pp. 1070-1083, 2015.

[17] H. Taghavifar, A. Modarres Motlagh, A. Mardani, A. Hassanpour, A. Haji Hosseinloo, L. Taghavifar, and C. Wei, "Appraisal of Takagi–Sugeno type neuro-fuzzy network system with a modified differential evolution method to predict nonlinear wheel dynamics caused by road irregularities," Transport, Vol. 31, pp. 211-220, 2016.

[18] T. Obo, CK. Loo, M. Seera, and N. Kubota, "Hybrid evolutionary neuro-fuzzy approach based on mutual adaptation for human gesture recognition," Applied Soft Computing, Vol. 42, pp. 377-389, 2016.

[19] K. Pandiarajan, and CK. Babulal, "Fuzzy harmony search algorithm based optimal power flow for power system security enhancement," International Journal of Electrical Power & Energy Systems, Vol. 78, pp. 72-79, 2016.

[20] RE. Precup, RC. David, and EM. Petriu, "Grey wolf optimizer algorithm-based tuning of fuzzy control systems with reduced parametric sensitivity," IEEE Transactions on Industrial Electronics, Vol. 64, pp. 527-534, 2016.

[21] W. Chen, M. Panahi, and H. Pourghasemi, "Performance evaluation of GIS-based new ensemble data mining techniques of adaptive neuro-fuzzy inference system (ANFIS) with genetic algorithm (GA), differential evolution (DE), and particle swarm optimization (PSO) for landslide spatial modelling," Catena, Vol. 157, pp. 310-324, 2017.

[22] D. Karaboga, and E. Kaya, "Adaptive network based fuzzy inference system (ANFIS) training approaches: a comprehensive survey," Artificial Intelligence Review, Vol. 52, pp. 2263-2293, 2019.

[23] HG. Wahdan, HE. Abdelslam, THM Abou-El-Enien, and SS. Kassem, "Two-Modified Emperor Penguins Colony Optimization Algorithms," Revue d'Intelligence Artificielle, Vol. 34, pp. 151-160. 2020.

[24] S. Harifi, M. Khalilian, J. Mohammadzadeh, and S. Ebrahimnejad, "Using metaheuristic algorithms to improve k-means clustering: A comparative study," Revue d'Intelligence Artificielle, Vol. 34, pp. 297-305, 2020.

[25] S. Harifi, E. Byagowi, and M. Khalilian, "Comparative study of apache spark MLlib clustering algorithms," In International Conference on Data Mining and Big Data, pp. 61-73. Springer, Cham, 2017.

[26] D. Dua, and K. Taniskidou, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], Irvine, CA: University of California, School of Information and Computer Science, 2017.