Available online at www.sciencedirect.com
ScienceDirect
Mathematics and Computers in Simulation 178 (2020) 1–15
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Original articles

Human urbanization algorithm: A novel metaheuristic approach

Hadi Ghasemian[a], Fahimeh Ghasemian[b,*], Hamed Vahdat-Nejad[a]

[a] *Department of Computer and Information Technology Engineering, University of Birjand, Birjand, Iran*
[b] *Department of Computer Engineering, Faculty of Engineering, Shahid Bahonar University of Kerman, Kerman, Iran*

Received 19 December 2019; received in revised form 19 May 2020; accepted 21 May 2020
Available online 2 June 2020

Abstract

In the recent decade, the application of metaheuristic algorithms for optimization and solving different problems is increased and various nature-inspired algorithms are designed to improve the quality and convergence speed of getting the answer. In this essay, a novel metaheuristic search for optimization inspired by human behavior for urbanization and improving life situations is proposed. This algorithm uses different strategies including combined searching in an open and limited scope and population management and concentration of agents in the search process. The evaluation results using a wide range of different test functions show the prevalence of the proposed algorithm compared with other potent algorithms in most search spaces.

Keywords: Optimization; Metaheuristic; Urbanization; Human behavior

1. Introduction

Recently optimization is one of the most noticed topics among researches. Optimization is defined as the couple (S, f), where S is the answer space and $f : S \longrightarrow R$ is a function for optimization. $s \in S$ is an answer if its assessment results to the best value over all the search space that satisfies the below constraint:

$$\forall s \in Sf(s^*) \leq f(s) \tag{1}$$

Traditional optimization methods which are based on the gradient are time-consuming to solve complicated optimization problems [9]. So, researchers have proposed meta-heuristic algorithms that are able to find approximate solutions for optimization problems in an acceptable time.

Metaheuristic algorithms are now becoming powerful methods for solving optimization problems [35] and new variants are continually being suggested. The behavior of alive creatures is one of the sources for developing metaheuristic algorithms. These algorithms which are called swarm intelligence algorithms include ant colony algorithm [11], particle swarm optimization [12], fish swarm algorithms [20], artificial bee colony algorithm [18], firefly algorithm [34,38], cuckoo search algorithm [15], bat algorithm [39], fruit fly algorithm [23], Social Spider Optimization [17] and Locust Search Algorithm [10].

Some special capabilities are necessary for an efficient metaheuristic algorithm [37]:

* Corresponding author.
E-mail addresses: hadi.ghasemian@yahoo.com (H. Ghasemian), ghasemianfahime@gmail.com (F. Ghasemian), vahdatnejad@birjand.ac.ir (H. Vahdat-Nejad).

https://doi.org/10.1016/j.matcom.2020.05.023

- Coverage of the most important parts of the search space where the global optimum is located.
- Escaping any local optimum.
- Generating new solutions that are more likely better than the existing solutions.

To satisfy these conditions, an algorithm should establish a good balance between two important parts of any metaheuristic algorithms: intensification and diversification. "Diversification means to generate diverse solutions to explore the search space on the global scale, while intensification means to focus on the search in a local region by exploiting the information that a current good solution is found in this region" [37].

Generally, the goal of all metaheuristic algorithms is creating a balance between search phases including disclosing new areas with better quality (diversification or exploration) and concentration on discovered areas (intensification or exploitation) while their difference is the mechanism for creating this balance [30]. Every imbalance between these two phases in different search stages causes trapping in local minimums. At the first of search process, diversification is very important and then the algorithm gradually aggregates the agents and directs the search on favorable areas for extraction.

There are some strategies for improving the management of diversification and intensification. Population management is a strategy that controls the number of search agents during the algorithm and is an important factor to improve the performance of metaheuristic algorithms. The algorithm proposed by sorensen [29] uses the population management for improving the performance. Combining metaheuristic algorithms is another strategy. No metaheuristic algorithm is perfect and cannot expect that one metaheuristic algorithm gives the best answer for all optimization problems. Using a hybrid two-stage strategy for more effective search can improve the answer's quality and convergence [7]. However, the selection and combination of metaheuristic algorithms for a special problem are difficult [5]. Therefore, this paper designs a new optimization algorithm to find optimal solutions by human behavior for urbanization and improving life situations. This algorithm uses a hybrid method with different strategies to control the diversification and intensification. The two-stage behavior of the human in urbanization (adventuring on the earth to select suitable places for urbanization) and citizenship in city boundaries, makes the search a two-stage hybrid process. Besides, based on factors like population change and migration, the algorithm manages the population change and concentration. To test the effectiveness of the algorithm, the algorithm will optimize a total of 20 test functions and optimization results obtained by this algorithm will be compared with the results of the PSO [12], ICA [2] and GSA [24] algorithm. The obtained results show the prevalence of the proposed algorithm compared with other potent algorithms in most search spaces.

The rest of this article is organized as follows: the previous works are reviewed in Section 2; human behavior in urbanization is explained in Section 3; the proposed algorithm is discussed in Section 4; experiments are made to compare the algorithms in Section 5; the article is concluded in Section 6.

2. Related works

Optimization methods are divided into two groups: mathematical programming and metaheuristic algorithms. Mathematical programming includes integer programming, convey programming and linear programming [30]. The benefit of the first group is in finding the best solution to the problem. However, increase in search space results in exponential growth in the computation time. Also, these kinds of algorithms need complete information about the problem such as gradient information [25]. To efficiently explore the search space to find approximate solutions, metaheuristic algorithms are proposed. These algorithms are not problem-specific and use strategies to guide the search process for finding near-optimal solutions.

Based on different viewpoints, there are several ways to classify metaheuristic algorithms. Some of the most important classifications are summarized in the following [6].

2.1. Nature-inspired and non-nature inspired

This classification is based on the origins of the algorithm. Algorithms like Particle Swarm Optimization (PSO) [12], Ants Colony Optimization (ACO) [11], Artificial Bee Colony Algorithm (ABC) [18], Genetic Algorithm (GA) [16], Biogeography-Based Optimization Algorithm (BBO) [28], Bats Algorithms (BA) based on bats behavior [35], Cuckoo Search (CS) mimics behavior of the cuckoo [36], Firefly algorithm (FF) inspired by the flashing behavior of luminous insects [33] and Social Spider Optimization [17] are nature-inspired and Tabu Search and Iterated Local Search are non-nature inspired. Many recent hybrid algorithms do not place in either class (or belong to both at the same time).

2.2. Population-based and single point search

This classification is based on the number of solutions used at the same time to solve the problem. Population-based algorithms evolve a population of search points to find a solution while a single point just considers one point in the search space and evolves it. Tabu Search and Iterated Local Search are single point and Genetic and Ant Colony Algorithms are population-based.

2.3. Memory usage and memory-less methods

This is an important feature to classify metaheuristic algorithms which is based on the use of the search history. The use of memory is considered as one of the vital elements of a powerful metaheuristic. There are different ways of using memory which keep tracks of recent solutions, decisions and visited search space.

2.4. Anthropological algorithms

Due to the thought power and intelligence, human has a special place among all alive creatures. So, it can be expected that the algorithm which is based on human behavior can outperform other algorithms [27]. These algorithms which mimick human behavior to find the answer are called anthropological or Artificial Human Optimization (AHO) Field algorithms. Artificial Humans solve the optimization problems by imitating real Humans in the search space. Each Artificial Human corresponds to a point in the solution space [14].

These algorithms can be created from scratch or by adding into other existing Optimization Algorithms [13]. Mustafa [13] proposed six novel AHO algorithms by modifying existing PSO algorithms with AHO Field Concepts. These algorithms are titled “Human Safety Particle Swarm Optimization (HuSaPSO)”, “Human Kindness Particle Swarm Optimization (HKPSO)”, “Human Relaxation Particle Swarm Optimization (HRPSO)”, “Multiple Strategy Human Particle Swarm Optimization (MSHPSO)”, “Human Thinking Particle Swarm Optimization (HTPSO)” and “Human Disease Particle Swarm Optimization (HDPSO)”. BrainStorm Optimization (BSOA) [27] based on research about human brain, Cultural Algorithm (CA) [26] based on human social perfection, Oriented Search Algorithm (OSA) [40], Imperialist Competitive Algorithm (ICA) [2] based on imperialism model and BRAin Drain Optimization (BRADO) algorithm [3] and HBBO [1] are instances of other anthropological algorithms.

The algorithm proposed in this article is a nature-inspired, population-based algorithm inspired by the behavior of the most intelligent creature, humans in urbanization. In this algorithm, the searching process is hybrid and in two phases. This hybrid behavior causes algorithm to achieve a better answer in different problems.

3. Human behavior in urbanization

For the study of urbanization, the history of humans should be explored as there is a direct relation between urban and human history. According to the studies [32], four main factors affect human urbanization including adventure, migration, population change and concentration.

3.1. Adventure

The present human or the Homo sapiens is a two-footed creature of the Hominidae family that has the thought and feeling of power. Indeed, he has an up warded body with free arms caused he can walk and carry things for long distances. This physical and mental feature lets the human to explore and adventure suitable places for life. According to studies on found fossils DNA, the first human lived in Africa about 200,000 years ago [31] (Fig. 1.a), while in 2011 his population was grown to 7 milliards and inhabited all over the earth except Antarctica.

The human motivation for leaving Africa was not studied well, but some reasons such as population increase and weather have been mentioned. This distribution happened during a long period. In the first effort in about 125,000 years ago, a group of human spread across South, West and North-West Africa [21]. Also, some of them received to North Africa and after that to Central Asia bypassing green Sahara (Fig. 1.b). This was the first exit from Africa. But this group was driven into extinction by a global freeze up which turned this area and north Africa into the extreme desert about 90,000 years ago [8]. Approximately 5000 years after this freeze up, another group

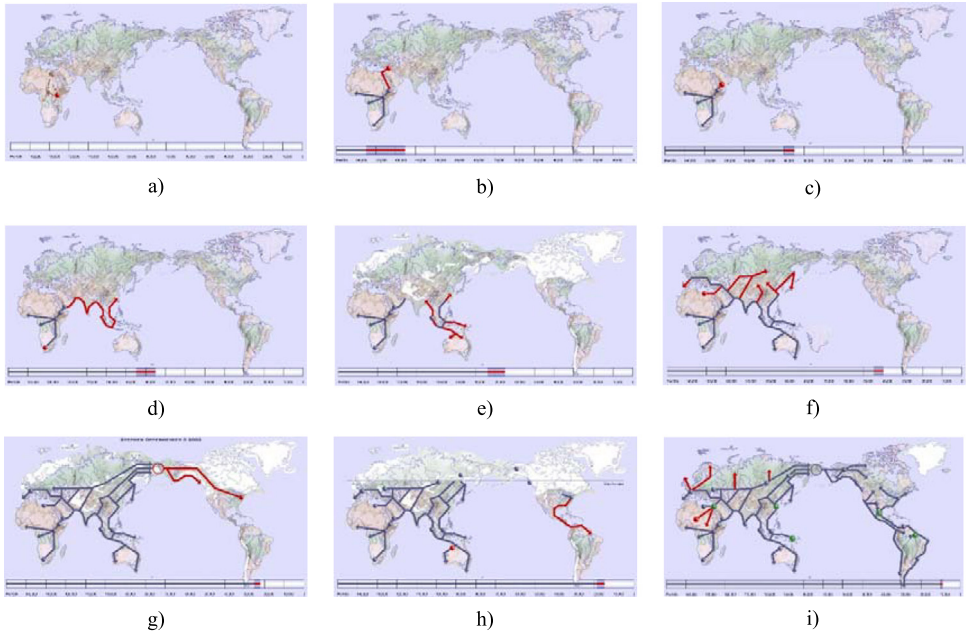


Fig. 1. Human distribution path over the decades.

entered Asia bypassing the Red Sea (Fig. 1.c) and distributed there [22] (Fig. 1.d). All none African human is originated from this group.

In all the motions for exploring better areas, two factors are involved: movement based on the previous experiences and random motions. In this paper, this motion for exploring new areas based on these two factors is called adventuring.

3.2. Migration

Indeed, the determinant that results in the concentration of population and meritocracy in different cities is migration. Migration is the change of life place from a city to another city by assessing its facilities. The cities hegemony criteria are different from time to time. In the past, the weather was the dominant factor but recently according to the last researches by Mr. Black [4] these factors include economic, political, cultural and environmental factors. People tend to migrate to cities where they experience a better life situation.

3.3. Population change and concentration

The population in cities with appropriate living conditions grows rapidly. Cities that have poor living conditions are less likely to grow because of various problems such as illness and famine. So, the population is concentrated in cities with better conditions.

4. Human urbanization algorithm

This section introduces the Human Urbanization Search algorithm (HUS), which is a population-based meta-heuristic method. The HUS algorithm is inspired by human behavior during his/her life on the earth. The proposed algorithm is a population-based searching method and concepts like adventure and motion for finding new places, migration from city to another for an easier and better life situation, city extent and population accumulation are used for designing this algorithm. In this algorithm, seekers are human and search space is earth where seekers search for suitable spots for building cities. Its population comprises several vectors that represent the center of cities. Each center represents a possible solution in the search space. The HUS algorithm searches for the optimal solution over

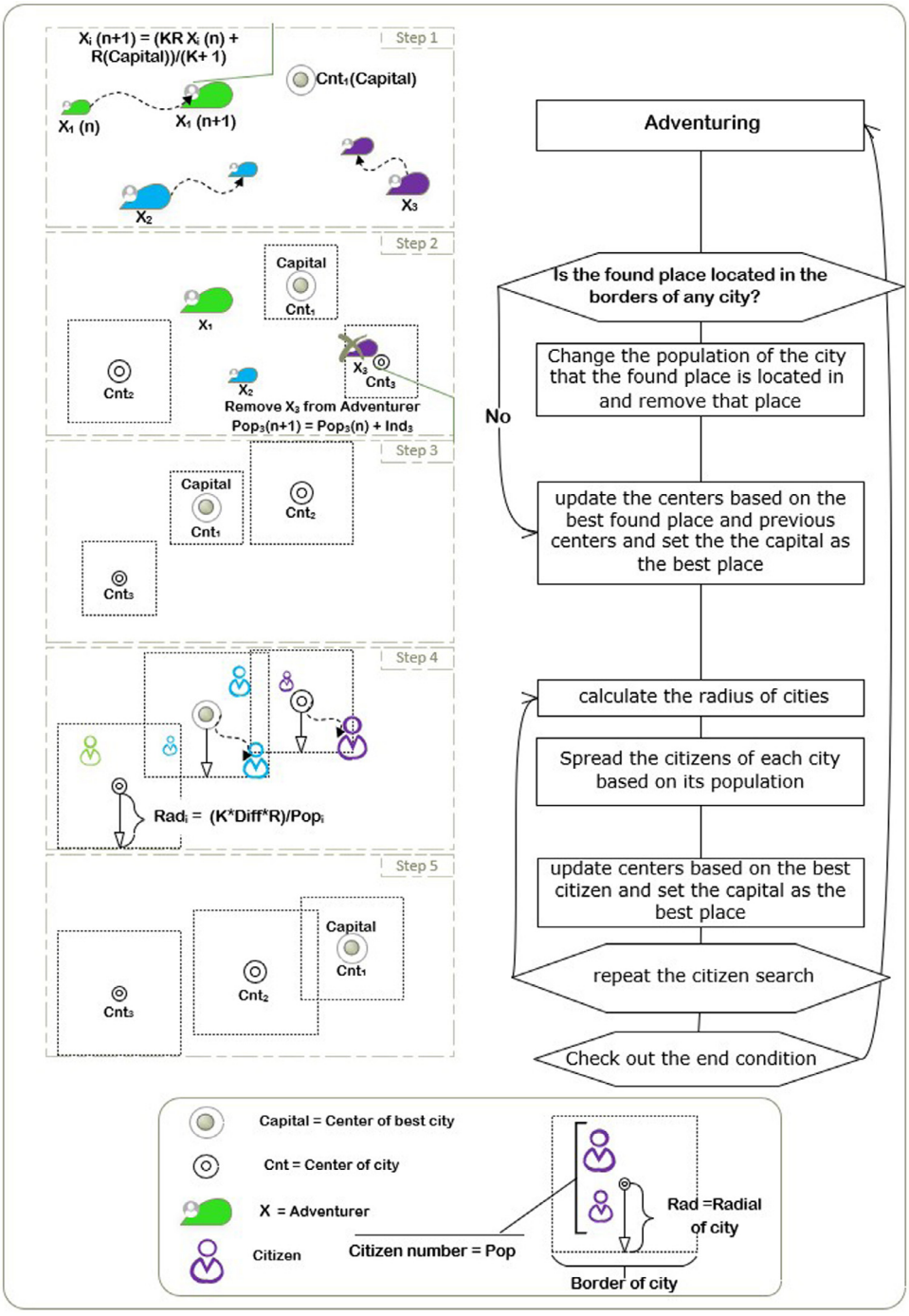


Fig. 2. Flow chart and scenario chart of HUS.

several iterations. In each iteration, the center of cities is updated using the past experiences of adventurers and randomized factors.

There are two types of seekers: adventurers and citizens. Adventurers search all the solution space and citizens are limited to the border of their cities. Each city is described in terms of five parameters as shown in Fig. 2: number, center, radius, population and population change index. At each iteration, the city centers will be the best

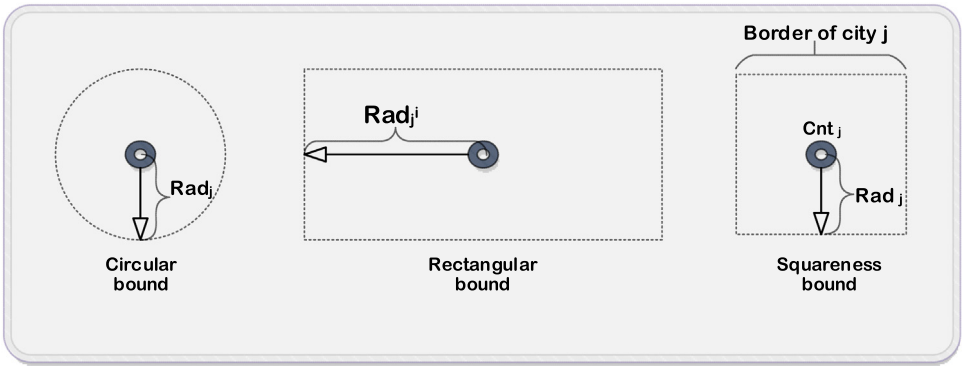


Fig. 3. Different models for specifying city boundaries.

present city centers and points that adventurers found. Also, the radius of each city is updated based on the city center fitness, its population, and randomized factors. This radius determines the border of the city where its citizens are located. Next, citizens are placed in the city boundaries considering the population of each city. Then, the best point in every city is selected as its center.

4.1. HUS algorithm formulation

- Step1: Initially adventurers search for suitable places to amend city centers. Adventurers are represented as a matrix (1). (in all formula M is the number of adventurers and N is the dimension of the problem answer).

$$X_i(t) = (x_i^1, x_i^2, \ldots, x_i^N), \quad i = 1, 2, \ldots, M \tag{2}$$

where M is the number of adventurers, N is the dimension of the problem answer, $X_i(t)$ represents adventurer i in iteration t and x_i^j shows feature j of adventurer i.
In every iteration, adventurers are updated using below equation:

$$X_i(t+1) = \frac{K * R * X_i(t) + \bar{R} * xpr}{K+1}, \quad \bar{R} = 1 - R \tag{3}$$

Where R is a random number in $[0, 1]$ and xpr is the current capital which reflects the past experience of adventurers. In addition, K is a parameter of the algorithm for controlling diversification and intensification of the adventurers.

- Step 2: In these stages, the points that adventurers found are evaluated and the population of cities is updated. First, the points found by adventurers are sorted from the best to the worst using the fitness function. Next, each point is examined to check if it is located on the border of the cities. If an adventurer was located in the boundaries of a city, that adventurer is removed; the population of that city is changed according to the population index and the process is continued by starting from the next adventurer. For example, the first adventure is checked that is located in the first city boundaries or not. If it was not, the second city is checked and this process continues to last city. If the adventurer X_i (X_i has the i_{th} rank among adventurers) is in the boundaries of city j, the population of city j (Pop_j) in stage t are updated according to below equation:

$$Pop_j(t+1) = Pop_j(t) + Ind_j \tag{4}$$

where Ind_j is the index of city j.
The city boundaries can be identified in the form of circle, square and rectangle (Fig. 3). In circular boundaries, adventure X_i is located in city j, if its distance from the city center (Cnt_j) were smaller than the city radius (Rad_j) (4).

$$\sqrt{(x_i^1 - Cnt_j^1)^2 + (x_i^2 - Cnt_j^2)^2 + \cdots + (x_i^N - Cnt_j^N)^2} \leq Rad_j \tag{5}$$

Indeed, in squareness boundaries, the difference between adventurer's vector and vector of city center must be smaller than the city radius (5).

$$\left|Cnt_j^k - x_i^k\right| \leq Rad_j, \quad k = 1, 2, \ldots, N \tag{6}$$

In squareness and circular boundaries, radius for every city is a $Rad_{(1*1)}$ matrix but in rectangular boundaries, it is a $Rad_{(1*n)}$ matrix. Therefore, in this model, an adventurer is in the city boundaries if the difference between adventurer's vector and city center vector be smaller than the radius matrix of the city (6).

$$\left|Cnt_j^k - x_i^k\right| \leq Rad_j^k, \quad k = 1, 2, \ldots, N \tag{7}$$

In summary, the squareness model is easier to use compared with two other models and needs a lower memory in comparison with the rectangular model. More exact specifications and better management on search space boundaries are the advantages of a rectangular model. In this article, the squareness model is utilized in the implementation of the algorithm.

- Step 3: In this stage, the center of new cities is selected based on the present city centers and new adventurers' points. The set of existing city centers and adventurers' points are sorted regarding the fitness function and the first M points have opted as new city centers. In addition, the center of the first city is the capital.
- Step 4: In this stage the radius of each city is calculated:

$$Rad_j = \frac{K * Diff * R}{Pop_j} \tag{8}$$

Where Rad_j is the radius of city j, R is a random number and Pop_j represents the population of city j. Diff is a factor that is calculated based on the quality of the city center compared to the capital using the below equation. The main aim of including this factor is to direct the searching process on the boundaries of the favorable city. Citizens of better cities are more centralized.

$$Diff = \arctan\left(\left|\frac{(F(Capital) - F(Cnt_j))}{F(Capital)}\right|\right) + 0.5 \tag{9}$$

Moreover, K in iteration i is calculated using Eq. (10). The main propose of influencing it in the calculation is to create intermediate intensification and diversification.

$$K = \frac{(50 * (cos(i * s) + 1))}{i^{Rimpk}}, s = \frac{(500 * \Pi)}{(2 * Itr)} \tag{10}$$

Where Itr represents the number of iterations. $Rimpk$ is another parameter of the algorithm and is used for creating a balance between diversification and intensification in searching the cities' boundaries. Diversification would be more drastic for higher values. If the search space fluctuates erratically, lower values for instance 1 are recommended. For flat spaces, higher values are better. In this paper, value 1 is chosen as default.

The main role of influencing this factor in radius calculation is managing diversification and intensification intermediately and generally in citizen search. It creates a wave behavior in the city radius (Fig. 4). This behavior helps the algorithm to have intermediate diversification and intensification. It causes that the algorithm concentrates in favorable search areas. Also, it helps the algorithm to get rid of local minimums by increasing the city boundaries. This factor gradually decreases regarding the iteration number which results in the decline of the city boundaries.

Next, local search is done in more appropriate places. The population of each city is an indicator of the importance of that city. In the other words, the city that has more population is searched more. Here the default value for the city population is computed using below formula:

$$Pop_k = M - k, \quad k = 1, 2, \ldots, M \tag{11}$$

next, population are spread randomly based on the population of each city. The place of each citizen is computed using below equation:

$$Citizen_j{}^i = Cnt_j + (R^i * Rad_j), \quad i = 1, 2, \ldots, Pop_j \quad j = 1, 2, \ldots, M \tag{12}$$

where $Citizen_j{}^i$ is the place of citizen i of city j, R^i is a random number from [−1,1].

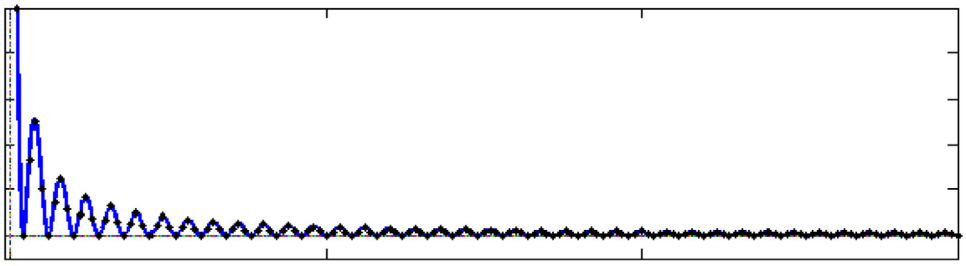


Fig. 4. Waving behavior of the city radius caused by the factor k.

Table 1
The parameter setting of different algorithms.

Algorithm	Parameters	
HUS1	k = 2 Ripmk = 1	Pop = [10,9,8,7,6,5,4,3,2,1], Ind = [0,0,0,0,0,0,0,0,0,0]
3–5 HUS2		Pop = [1,1,1,1,1,1,1,1,1,1], Ind = [1,1,1,0,0,0,0,0,0,0]
3–5 HUS3		Pop = [50,40,30,2,1,1,1,1,1,1], Ind = [−1,−1,−1,0,0,0,0,0,0,0]
3–5 HUS4		Pop = [1,1,1,1,1,1,1,1,1,1], Ind = [0,0,0,0,0,0,0,1,1,1]
3–5 HUS5		Pop = [1,1,1,1,1,1,1,1,1,1], Ind = [0,0,0,1,1,1,1,0,0,0]
PSO (Biological)	$K_1 = 1, K_2 = 2, \omega = 0.7$	
ICA (Anthropology)	$\xi = 0.1, \beta = 2$	
GSA (Inanimate)	$G_0 = 100, \alpha = 20$	

- Step 5: In this stage, the best citizen of every city is chosen as the city center. Next, all centers are sorted. Finally, the capital is updated. The capital is the best area that is founded so far. So, among all city centers, the best center is chosen as the capital.
  Steps 4 and 5 can be repeated frequently at each iteration. By default, these steps are repeated two times.

At the end of each iteration, the end condition is examined. If it is not satisfied, steps 1 to 5 are repeated.

5. Evaluation

There are a lot of functions for evaluating metaheuristic algorithms [19]. In the following, the most common functions for evaluating metaheuristic algorithms are described. These functions are divided into three groups: unimodal, free-dimension multimodal and fixed-dimension multimodal to evaluate the power of the algorithm in escaping from local minimums.

The experiment was simulated in Matlab Software.[1] For Comparison of HUS, some potent and well-known algorithms from different categories including GSA [24] of inanimate algorithm, PSO [12] of biological algorithms and ICA [2] of anthropology algorithms have been considered. The parameters of these algorithms are shown in Table 1.

5.1. Experiments

In the first experiment, we compare our proposed algorithm with PSO, GSA, and ICA. Each algorithm is run 100 times. Also, the iteration number is considered 500 and the dimension of the problems is 30 except in cases that the dimension is fixed.

Five different versions of HUS are considered (HUS1, HUS2, . . . , HUS5). They differ in the initial population and population index matrix as described in Table 1. In all versions, the number of agents is set to 120 (10 adventurers and 55 citizens who are placed two times).

[1] The code of HUS algorithm is available at: https://github.com/hadighasemian/HUS.

Table 2
Result of different versions of HUS, PSO, GSA and ICA.

Test function	Data type	HUS1	HUS2	HUS3	HUS4	HUS5	PSO	ICA	GSA
f1: Sphere model (Unimodal)	Min	7.75E-105	2.25E-107	2.97E-104	5.10E-104	2.91E-105	2.391699	6.10E-14	1.57E-17
	Avg	3.15E-96	1.64E-97	2.95E-92	1.28E-93	1.34E-94	8.780612	1.36E-12	3.78E-17
	Max	1.86E-94	1.33E-95	2.95E-90	1.28E-91	1.17E-92	30.50547	1.37E-11	8.90E-17
F2: Schwefel's problem 2.22 (Unimodal)	Min	4.86E-53	1.79E-53	9.20E-54	4.79E-53	4.79E-53	0.356698	5.05E-09	2.10E-08
	Avg	2.35E-49	3.98E-49	1.45E-49	1.65E-49	1.06E-49	0.684814	3.71E-08	0.002075
	Max	7.58E-48	2.30E-47	6.94E-48	1.29E-47	2.23E-48	1.599991	3.17E-07	0.140517
F3: Schwefel's problem 1.2 (Unimodal)	Min	1.07E-101	2.18E-102	9.64E-103	3.73E-102	5.79E-102	291.1555	6.738184	185.9886
	Avg	2.38E-89	1.89E-86	3.14E-83	6.50E-91	1.68E-86	640.3395	33.31009	557.7976
	Max	2.31E-87	1.34E-84	3.14E-81	3.99E-89	1.67E-84	1041.174	99.52812	1122.997
F4: Schwefel's problem 2.21 (Unimodal)	Min	1.09E-52	8.59E-53	8.98E-53	2.05E-52	3.90E-52	3.662661	1.645924	0.042214
	Avg	2.52E-49	3.25E-49	3.67E-49	5.84E-49	6.34E-49	5.145924	4.747301	3.537165
	Max	5.37E-48	6.68E-48	1.12E-47	9.88E-48	1.62E-47	6.986798	11.12944	8.209443
F5: Generalized Rosenbrock's (Unimodal)	Min	26.64544	26.7921	28.54392	28.20504	26.66372	95.2217	0.905558	24.82733
	Avg	27.05049	28.45148	28.75454	28.46436	27.13153	419.1642	48.95652	36.52124
	Max	27.36396	28.79507	28.83299	28.80986	27.43015	2744.457	467.8727	239.9645
f6: Step (Unimodal)	Min	0	0	0	0	0	5	0	0
	Avg	0	0	0	0	0	14.79	0	0.81
	Max	0	0	0	0	0	32	0	20
F7: Quartic with Noise (Unimodal)	Min	0.000168	0.000201	5.85E-05	0.000169	7.22E-05	0.010494	0.010504	0.008455
	Avg	0.003913	0.027181	0.004063	0.004642	0.004203	0.020374	0.028205	0.029199
	Max	0.012034	0.119266	0.013728	0.022944	0.013925	0.034075	0.05689	0.08198
F8: Generalized Rastrigin's (Multimodal)	Min	0	0	0	0	0	20.3661	0.000216	6.964713
	Avg	0.371114	1.449467	0	0	0	43.43187	3.493911	16.94415
	Max	16.91498	19.89919	0	0	0	75.12498	7.95974	26.86388
F9: Ackley's (Multimodal)	Min	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	0.436722	1.33E-07	2.97E-09
	Avg	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	1.685789	8.46E-07	4.62E-09
	Max	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	2.932477	8.72E-06	6.66E-09
F10: Generalized Griewank (Multimodal)	Min	0	0	0	0	0	1.015338	1.40E-13	9.908773
	Avg	0	0	0	0	0	1.079234	0.016819	16.97536
	Max	0	0	0	0	0	1.261514	0.09822	26.26186
F11: Generalized Penalized_2D (Multimodal)	Min	9.34E-08	0.006237	0.000968	0.000149	1.30E-07	0.004392	1.24E-15	2.13E-19
	Avg	0.00373	0.014924	0.006743	0.001891	0.002203	0.0587	6.19E-12	0.628077
	Max	0.110274	0.036687	0.324295	0.005806	0.107182	0.316035	7.85E-11	2.649708
F12: Generalized Penalized_3D (Multimodal)	Min	0.059153	0.597272	0.278142	0.195275	0.12725	0.220847	1.42E-14	2.82E-18
	Avg	0.796061	1.303298	2.517837	2.23636	1.357501	0.840977	5.08E-11	2.207319
	Max	2.966074	2.974197	2.980953	2.976651	2.966074	2.044891	1.41E-09	15.00675
F13: Kowalik's (Multimodal, dim = 4)	Min	2.75E-09	3.87E-09	1.02E-06	8.40E-07	2.92E-11	5.50E-10	1.26E-08	0.000187
	Avg	0.000305	0.000427	0.000735	0.000463	0.000403	0.000158	0.000454	0.00318
	Max	0.001554	0.007641	0.009007	0.005359	0.005626	0.001287	0.002555	0.00783

(*continued on next page*)

The absolute difference between the optimal solution and the best, worst and average solution is reported for all executions of the algorithms. According to these result (Table 2), a remarkable improvement can be seen for f_1 to f_7(unimodal functions). The three achieved solutions are significantly better for f_1, f_2, f_3 and f_4.

Table 2 (*continued*).

Test function	Data type	HUS1	HUS2	HUS3	HUS4	HUS5	PSO	ICA	GSA
F14: Six-hump Camel-Back (Multimodal, dim = 2)	Min	4.65E-08	4.65E-08	4.65E-08	4.65E-08	4.65E-08	4.65E-08	4.65E-08	4.65E-08
	Avg	4.65E-08	4.65E-08	5.00E-08	4.78E-08	4.65E-08	4.65E-08	4.65E-08	4.65E-08
	Max	4.67E-08	4.65E-08	1.83E-07	1.75E-07	4.66E-08	4.65E-08	4.65E-08	4.65E-08
F15: Branin (Multimodal, dim = 2)	Min	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113
	Avg	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113
	Max	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113	0.000113
F16: Hartman's Family (Multimodal, dim = 3)	Min	0.002782	0.002782	0.002082	0.002781	0.002782	0.002782	0.002782	0.002782
	Avg	0.002782	0.002782	0.002707	0.002782	0.002782	0.002782	0.002782	0.002782
	Max	0.002782	0.002782	0.002782	0.002782	0.002782	0.002782	0.002782	0.002782
F17: Hartman's Family (Multimodal, dim = 6)	Min	0.001995	0.001995	0.000989	0.00055	0.001995	0.001995	0.001995	0.001995
	Avg	0.001995	0.004293	0.001863	0.001965	0.001995	0.057149	0.02038	0.003166
	Max	0.001995	0.116915	0.001995	0.001995	0.001995	0.116898	0.116898	0.119105
F18: Shekel's Family (Multimodal, dim = 5)	Min	3.22E-07	3.21E-07	2.15E-05	8.74E-05	3.26E-07	3.21E-07	3.21E-07	3.21E-07
	Avg	2.547634	1.272222	4.638763	4.741162	2.802078	3.74304	2.462141	3.526652
	Max	5.098002	5.098002	5.098003	5.098002	5.098002	7.522728	7.522728	7.47034
F19: Shekel's Family (Multimodal, dim = 7)	Min	0.000139	0.000141	9.26E-06	9.81E-06	0.00014	0.000141	0.000141	0.000141
	Avg	2.284351	1.540252	4.517897	4.411195	2.497773	2.77915	2.424461	0.120204
	Max	5.315128	5.315128	5.315129	5.315128	5.315128	8.565207	7.650866	7.636903
F20: Shekel's Family (Multimodal, dim = 10)	Min	0.000109	0.00011	5.92E-06	7.94E-06	0.000109	0.00011	0.00011	0.00011
	Avg	2.16225	1.404228	4.272254	4.271314	2.75757	2.714419	2.595087	0.00011
	Max	5.407819	5.407819	5.40782	5.407819	5.407819	8.67682	8.114566	0.00011

Although the performance of HUS is not significantly better for free-dimension and multimodal functions but in most cases, it exhibits better or equal performance. Results of the HUS algorithm for optimizing fixed-dimension multimodal functions do not have remarkable differences with other algorithms, but in most cases, the answers are equal or better. For example, results of f_{17} have a significant improvement on maximum; in minimum and average it is better and equal with other algorithms respectively. For function f_{15}, all results are equal. Also, for f_{18}, f_{19}, f_{20} functions, HUS achieves a better answer in the maximum and equal answer for minimum, while on average GSA receives the better results.

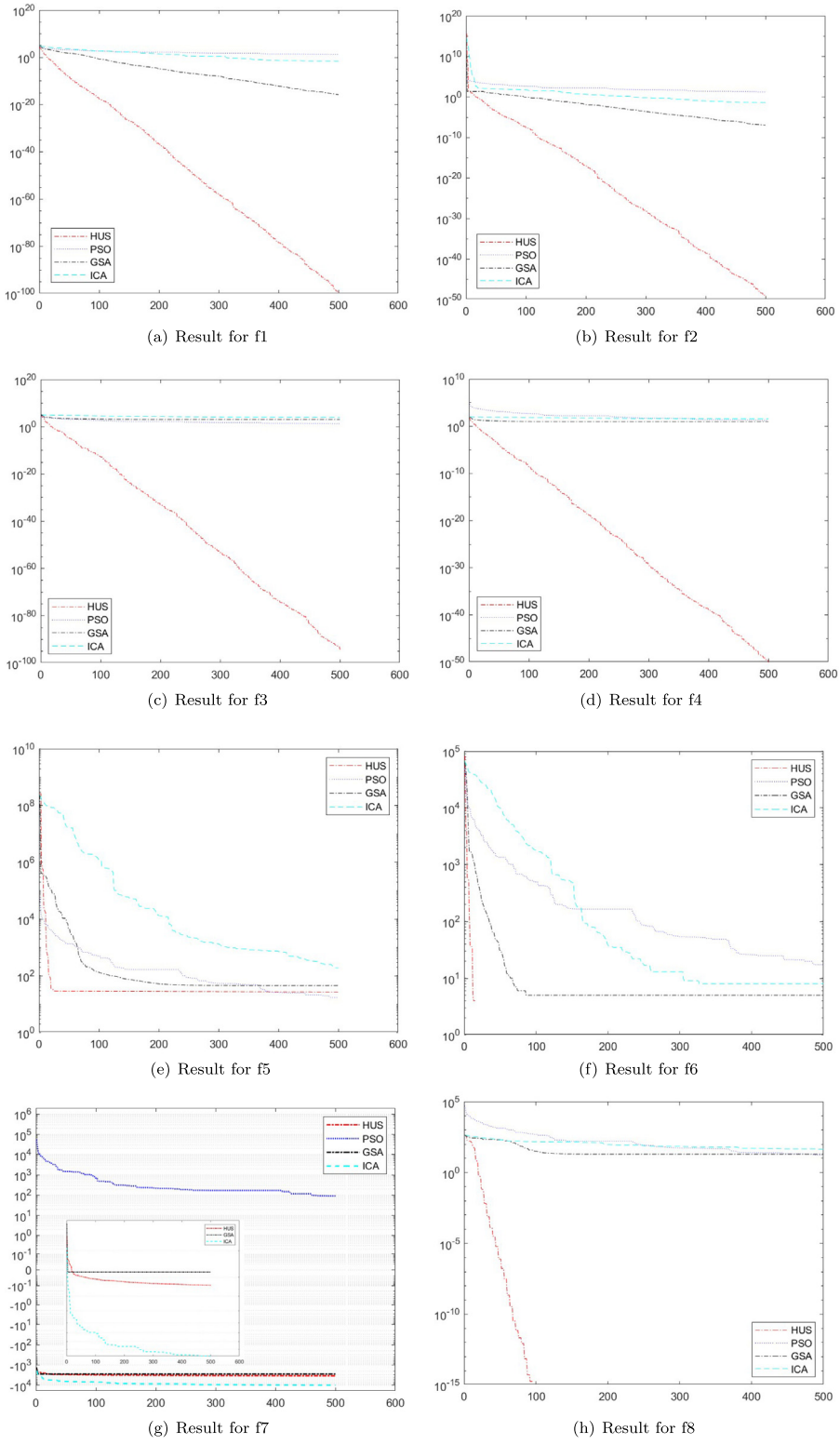
It can be seen in Fig. 5.a to .j that the behavior of HUS is descending and by increasing the iteration number, the answer improves while other algorithms have been leveled off. We can conclude that the convergence of HUS is great for unimodal functions. For multimodal functions, it is able to achieve comparable results with other algorithms.

Moreover, the Wilcoxon-test is also carried out, the result is presented in Table 3. Note that the symbols "(+)", "(−)", and "(=)" denote that HUS is significantly better than, significantly worse than, and almost the same as the corresponding competitors respectively, in terms of Wilcoxon-test results. We can see that HUS yields the most promising performance.

6. Conclusion

In this article, an attempt is made to propose a new metaheuristic algorithm named HUS based on human behavior. There are a lot of inspiring resources to design a new metaheuristic algorithm, but human beings are the most intelligent creature in the world. So, it can be expected that the algorithm which is based on human behavior can outperform other algorithms [27]. The HUS algorithm is inspired by human behavior in building, improving and destroying unbans. The search process is done in open and limited spaces. Although, diversification and intensification are the duty of both phases these two spaces cover each other's defects by a hybrid behavior.

The proposed HUS algorithm is tested over a set of 20 functions that are rich in type (unimodal, free-dimension multimodal, fixed-dimension multimodal). The results are compared with those of ICA, PSO, and GSA; the results showed that HUS is highly competitive compared with the other algorithms. For unimodal functions, HUS outperforms the other algorithms in most cases. Also, it exhibits better or equal performance for most multimodal functions.



(a) Result for f1

(b) Result for f2

(c) Result for f3

(d) Result for f4

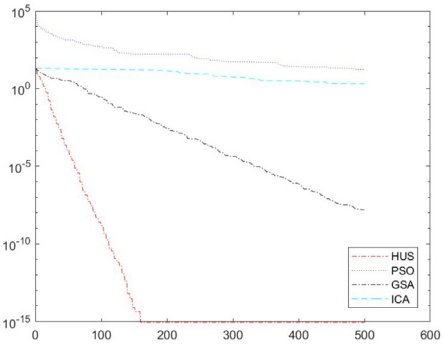
(e) Result for f5

(f) Result for f6

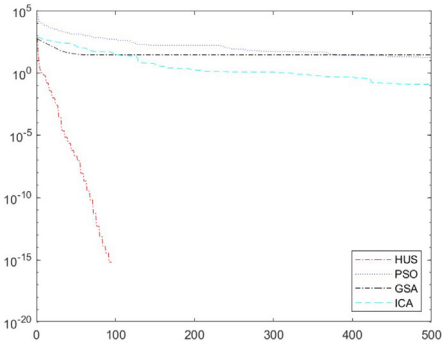
(g) Result for f7

(h) Result for f8

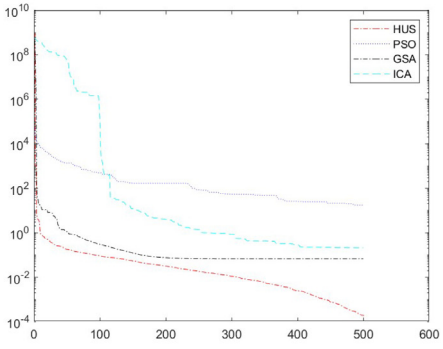
Fig. 5. Comparison of different algorithms.



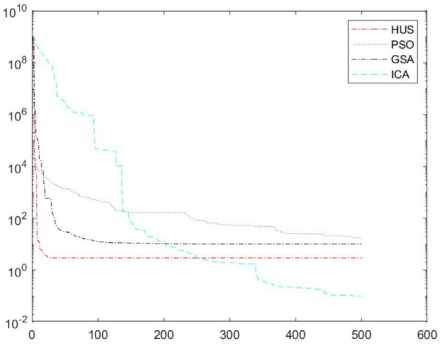
(a) Result for f9



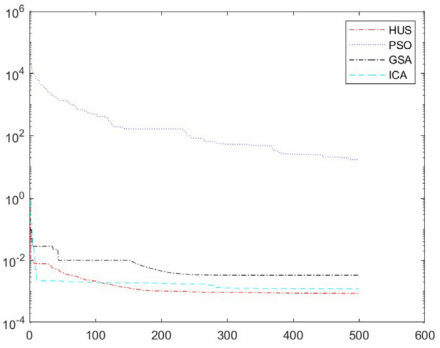
(b) Result for f10



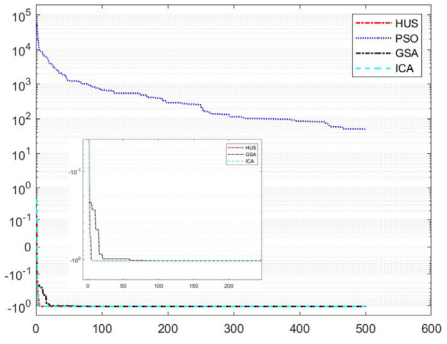
(c) Result for f11



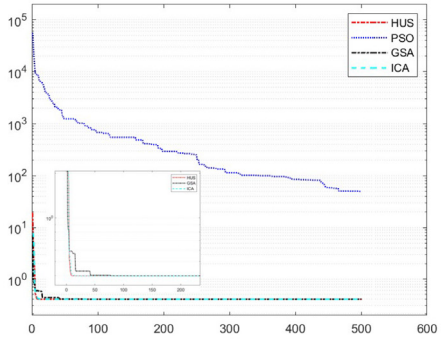
(d) Result for f12



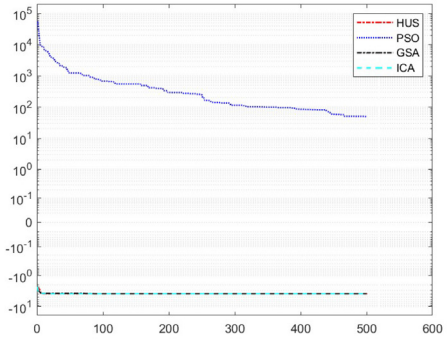
(e) Result for f13



(f) Result for f14

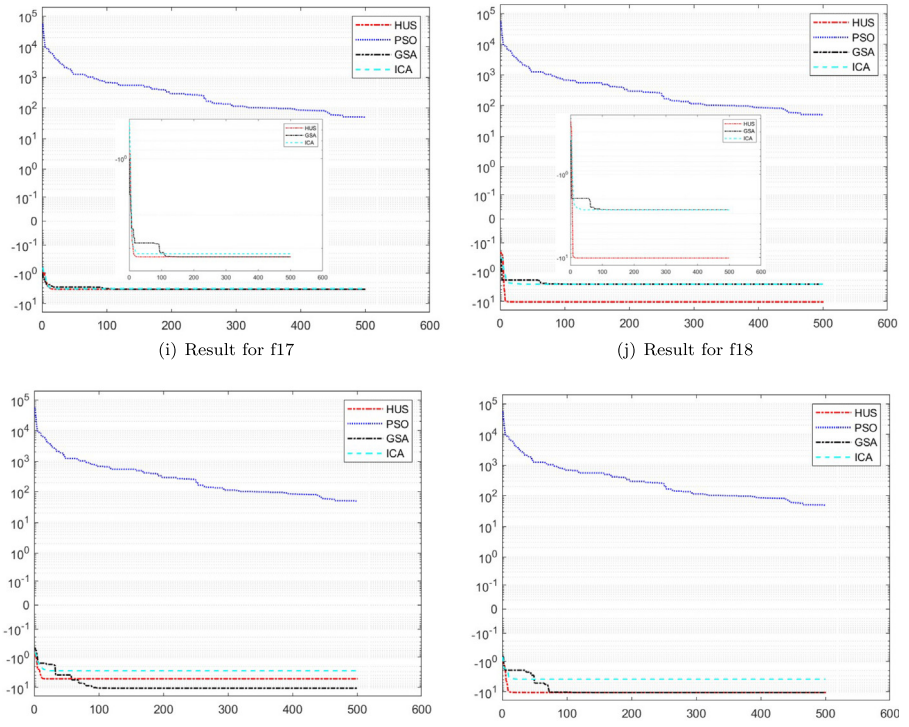


(g) Result for f15



(h) Result for f16

Fig. 5. (*continued*).



(i) Result for f17 (j) Result for f18

(k) Result for f19 (l) Result for f20

Fig. 5. (*continued*).

Table 3
P-values for Wilcoxon test.

Test function	HUS2	ICA	PSO	GSA
f1: Sphere model(Unimodal)	$1.42E-97 \pm 5.76E-97$	0.086708±0.222937 (+)	5.40E-90±5.30E-89 (+)	0.20009±1.914417 (+)
F2: Schwefel's problem 2.22 (Unimodal)	1.63E-49±8.47E-49	0.054239±0.040426 (+)	9.50E-47±6.37E-46 (+)	0.131029±0.457598 (+)
F3: Schwefel's problem 1.2 (Unimodal)	4.61E-88±4.59E-87	9077.635±3400.368 (+)	1.45E-85±1.09E-84 (+)	984.1945±32.6022 (+)
F4: Schwefel's problem 2.21 (Unimodal)	2.18E-48±1.49E-47	40.91633±8.502343 (+)	5.74E-46±4.14E-45 (+)	7.142638±2.273025 (+)
F5: Generalized Rosenbrock's (Unimodal)	27.06115±0.108545	416.2004±601.6016 (+)	28.42479±0.028674 (+)	67.18578±64.78403 (+)
f6: Step (Unimodal)	0±0	2.13±1.732955 (+)	0±0 (=)	12.3±26.91719 (+)
F7: Quartic with Noise (Unimodal)	0.521380452±0.265028	0.260536±0.096493 (−)	0.520663352±0.291221 (=)	0.086704±0.041916 (−)

(*continued on next page*)

Table 3 (*continued*).

Test function	HUS2	ICA	PSO	GSA
F8: Generalized Rastrigin's (Multimodal)	0.288489944±1.223146	52.48502±13.91196 (+)	0.3397989±1.149454 (=)	28.84388±7.870661 (+)
F9: Ackley's (Multimodal)	8.88E-16±0	5.135103±5.41786 (+)	0±0 (−)	8.44E-08±7.26E-07 (+)
F10: Generalized Griewank (Multimodal)	0±0	0.174329±0.126742 (+)	8.88E-16±0 (+)	29.54933±7.556295 (+)
F11: Generalized Penalized_2D (Multimodal)	0.000509437±0.004317	0.095409±0.17276 (+)	0±0 (−)	1.773052±1.055497 (+)
F12: Generalized Penalized_3D (Multimodal)	0.932524±0.911522	0.095219±0.095896 (−)	0.2119639±0.066452 (−)	8.367204±5.768816 (+)
F13: Kowalik's (Multimodal, dim = 4)	0.000777327±0.00102	0.002711±0.004 (+)	2.379602±0.517524 (+)	0.004865±0.003395 (+)
F14: Six-hump Camel-Back (Multimodal, dim = 2)	−1.0316±0	−1.0316±0 (=)	9.7586783±3.616424 (+)	−1.0316±0 (=)
F15: Branin (Multimodal, dim = 2)	0.39789±0	0.398256±0.001782 (=)	0.000326936±4.86E-05 (−)	0.39789±0 (=)
F16: Hartman's Family (Multimodal, dim = 3)	−3.8628±0	−3.76231±0.261272 (+)	−1.0316±0 (+)	−3.8628±0 (=)
F17: Hartman's Family (Multimodal, dim = 6)	−3.322±0	−3.26136±0.059738 (+)	0.39789±0 (+)	−3.31961±0.016807 (=)
F18: Shekel's Family (Multimodal, dim = 5)	−8.166682±2.4967	−5.84407±3.313197 (+)	3±0 (+)	−6.01755±3.624743 (+)
F19: Shekel's Family (Multimodal, dim = 7)	−8.012759±2.655835	−4.69467±3.053067 (+)	−3.8628±0 (+)	−9.98066±1.599416 (−)
F20: Shekel's Family (Multimodal, dim = 10)	−8.969238±2.46387	−4.63696±2.929732 (+)	−3.322±0 (+)	−10.2238±1.418965 (−)

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The authors received no financial support for the research, authorship, and/or publication of this article.

References

[1] S.A. Ahmadi, Human behavior-based optimization: a novel metaheuristic approach to solve complex optimization problems, Neural Comput. Appl. 28 (1) (2017) 233–244.
[2] E. Atashpaz-Gargari, C. Lucas, Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, in: Evolutionary Computation, 2007. CEC 2007. IEEE Congress on, IEEE, 2007, pp. 4661–4667.
[3] J. Basiri, F. Taghiyareh, A. Ghorbani, Collaborative team formation using brain drain optimization: a practical and effective solution, World Wide Web 20 (6) (2017) 1385–1407.
[4] R. Black, W.N. Adger, N.W. Arnell, S. Dercon, A. Geddes, D. Thomas, The effect of environmental change on human migration, Global Environ. Change 21 (2011) S3–S11.
[5] C. Blum, J. Puchinger, G.R. Raidl, A. Roli, Hybrid metaheuristics in combinatorial optimization: A survey, Appl. Soft Comput. 11 (6) (2011) 4135–4151.
[6] C. Blum, A. Roli, Metaheuristics in combinatorial optimization: Overview and conceptual comparison, ACM Comput. Surv. 35 (3) (2003) 268–308.
[7] C. Blum, A. Roli, Hybrid metaheuristics: an introduction, in: Hybrid Metaheuristics, Springer, 2008, pp. 1–30.
[8] S.L. Carto, A.J. Weaver, R. Hetherington, Y. Lam, E.C. Wiebe, Out of africa and into an ice age: on the role of global climate change in the late pleistocene migration of early modern humans out of africa, J. Hum. Evol. 56 (2) (2009) 139–151.
[9] J. Chen, H. Cai, W. Wang, A new metaheuristic algorithm: car tracking optimization algorithm, Soft Comput. 22 (12) (2018) 3857–3878.
[10] E. Cuevas, M.A.D. Cortés, D.A.O. Navarro, Optimization based on the behavior of locust swarms, in: Advances of Evolutionary Computation: Methods and Operators, Springer, 2016, pp. 101–120.
[11] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. B 26 (1) (1996) 29–41.
[12] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on, IEEE, 1995, pp. 39–43.
[13] S. Gajawada, H. Mustafa, Novel artificial human optimization field algorithms-the beginning, 2019, arXiv preprint arXiv:1903.12011.
[14] S. Gajawada, H. Mustafa, Ten artificial human optimization algorithms, Trans. Mach. Learn. Artif. Intell. 7 (3) (2019) 01–16.
[15] A.H. Gandomi, X.S. Yang, A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Eng. Comput. 29 (1) (2013) 17–35.
[16] J.H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence, MIT Press, 1992.
[17] J. James, V.O. Li, A social spider algorithm for global optimization, Appl. Soft Comput. 30 (2015) 614–627.
[18] D. Karaboga, Artificial bee colony algorithm, Scholarpedia 5 (3) (2010) 6915.
[19] M. Molga, C. Smutnicki, Test functions for optimization needs 101, 2005.
[20] M. Neshat, G. Sepidnam, M. Sargolzaei, A.N. Toosi, Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications, Artif. Intell. Rev. 42 (4) (2014) 965–997.
[21] S. Oppenheimer, Out of Eden: The Peopling of the World, Robinson, 2012.
[22] S. Oppenheimer, Out-of-africa, the peopling of continents and islands: tracing uniparental gene trees across the map, Philos. Trans. R. Soc. B 367 (1590) (2012) 770–784.
[23] W.T. Pan, A new fruit fly optimization algorithm: taking the financial distress model as an example, Knowl.-Based Syst. 26 (2012) 69–74.
[24] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, Gsa: a gravitational search algorithm, Inf. Sci. 179 (2009) 2232–2248.
[25] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, Gsa: a gravitational search algorithm, Inf. Sci. 179 (13) (2009) 2232–2248.
[26] R.G. Reynolds, An introduction to cultural algorithms, in: Proceedings of the Third Annual Conference on Evolutionary Programming, World Scientific, 1994, pp. 131–139.
[27] Y. Shi, Brain storm optimization algorithm, in: International Conference in Swarm Intelligence, Springer, 2011, pp. 303–309.
[28] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[29] K. Sorensen, M. Sevaux, Mal pm: memetic algorithms with population management, Comput. Oper. Res. 33 (5) (2006) 1214–1225.
[30] E.G. Talbi, Metaheuristics: From Design to Implementation, Vol. 74, John Wiley & Sons, 2009.
[31] E. Trinkaus, Early modern humans, Annu. Rev. Anthropol. 34 (2005) 207–230.
[32] M.J. White, International Handbook of Migration and Population Distribution, Springer, 2016.
[33] X.S. Yang, Firefly algorithms for multimodal optimization, in: International Symposium on Stochastic Algorithms, Springer, 2009, pp. 169–178.
[34] X.S. Yang, Firefly algorithm, stochastic test functions and design optimisation, Int. J. Bio-Inspired Comput. 2 (2) (2010) 78–84.
[35] X.S. Yang, A new metaheuristic bat-inspired algorithm, in: Nature Inspired Cooperative Strategies for Optimization, NICSO 2010, Springer, 2010, pp. 65–74.
[36] X.S. Yang, S. Deb, Cuckoo search via lévy flights, in: Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, IEEE, 2009, pp. 210–214.
[37] X.S. Yang, S. Deb, S. Fong, Metaheuristic algorithms: optimal balance of intensification and diversification, Appl. Math. Inf. Sci. 8 (3) (2014) 977.
[38] X.S. Yang, X. He, Firefly algorithm: recent advances and applications, Int. J. Swarm Intell. 1 (1) (2013) 36–50.
[39] X.S. Yang, A. Hossein Gandomi, Bat algorithm: a novel approach for global engineering optimization, Eng. Comput. 29 (5) (2012) 464–483.
[40] X. Zhang, W. Chen, C. Dai, Application of oriented search algorithm in reactive power optimization of power system, in: Electric Utility Deregulation and Restructuring and Power Technologies, 2008. DRPT 2008. Third International Conference on, IEEE, 2008, pp. 2856–2861.