Bulletin of Electrical Engineering and Informatics
Vol. 11, No. 5, October 2022, pp. 2756~2763
ISSN: 2302-9285, DOI: 10.11591/eei.v11i5.4083

A new modified grasshopper optimization algorithm

Abdulkareem Y. Abdalla[1], Turki Y. Abdalla[2]
[1]Computer Information Systems Department, College of Computer Science and IT, University of Basrah, Basrah, Iraq
[2]Department Computer Engineering, Engineering College, University of Basrah, Basrah, Iraq

Article Info

Article history:

Received May 15, 2022
Revised Jun 23, 2022
Accepted Jul 12, 2022

Keywords:

Exploitation
Exploration
Grasshopper
Meta-heuristic
Optimization

ABSTRACT

The grasshopper algorithm (GOA) is a recent algorithm. It is widely used in many applications and results in a good solution. The algorithm is simple and the accuracy in very high. The GOA has some limitations due to the use of linear comfort zone parameter that causes some difficulties in balancing between the exploration and exploitation which may lead to fall in a local optimum. In this paper a modification is made to improve the operation of GOA. A nonlinear function is developed to replace the linear comfort zone parameter. The benchmark of GOA authors is used for testing the performance improvement of the suggested modified GOA compared to the basic GOA. Results indicate that the MGOA outperforms original GOA, presenting a higher accuracy, faster convergence, and stronger stability. The proposed new modified GOA performs better than the original GOA.

Corresponding Author:

Abdulkareem. Y. Abdalla
Department Computer Information Systems, College of Computer Science and IT, University of Basrah
Basrah, Iraq
Email: abdulkareem.abdalla@uobasrah.edu.iq

1. INTRODUCTION

The grasshopper algorithm (GOA) is a new optimization algorithm developed in 2017 [1] to simulate the behavior of grasshoppers in nature. It has been used in different applications. GOA is formulated to simulate the behavior of grasshoppers. Different variants of GOA have been suggested. Binary GOA schemes are developed [2], [3]. In several works authors use chaotic function to enhanced GOA operation for different applications [4]-[12]. Results illustrated the good quality of the developed algorithms compared to GOA and other algorithms. An enhanced GOA scheme using a combination of levy flight with GOA (LGOA) is developed in [13]. The performance of LGOA was tested and compared with PSO and the basic GOA. Results illustrated the good quality of the developed algorithm compared to the basic GOA and PSO algorithm. Authors in [14] presented a dynamic GOA using dynamic weight and random jumping concept. It was illustrated that the suggested method has good properties. An adaptive GOA is developed to enhance the performance of GOA and used for optimization in different applications [15]-[17]. Results show good achievement for the proposed method. GOA is used in [18] for tuning the parameters of the Support Vector Machine. A dynamic GOA is developed and used for the unmanned aerial vehicles in [19]. The multiobjective GOA is used for solving the benchmark problems in [20].

An improved grasshopper using levy flight is presented in [21], [22]. Ewees *et al.* [23] an opposition-based learning approach to the GOA is presented. A fuzzy logic is used with GOA for electric vehicle charging sites in [24]. The new developed fuzzy GOA was tested on distribution networks and results show good performance. A GOA with fuzzy control scheme to track the maximum power point is proposed in [25]. GOA is utilized for tuning membership functions of fuzzy logic controller. It was illustrated that good performance was obtained.

In general the meta-heuristic algorithms use a random initial population for the considered problem, find solutions using the objective function. Algorithms enhance the solutions iteratively until satisfying the stopping condition and find the global optimum in different applications [26]-[29]. These algorithms have the following advantages: they are simple and easy to apply. Independent of derivation. All algorithms have one common feature in the searching process for the optimal solution: exploitation and exploration. The algorithm in the exploration, go through the global search space. While in the exploitation local search is performed.

The GOA work well and can find good solutions in a reasonable time. In general the GOA have linear comfort zone parameter which is weak in balancing exploitation and exploration. Also it may fall into local optimum. In [30], [31] authors presented different attempts for the improvement of GOA algorithm.

This paper proposed a new modified GOA. The major contributions of this work is to replace the linear comfort zone parameter by a nonlinear function which results in balancing exploitation and exploration and improve the achievement. This article is arranged as: The original GOA and the new MGOA are presented in sections 2 and 3 respectively. Results are shown in section 4. Section 5 contains the conclusions.

2. GRASSHOPPER ALGORITHM

The GOA is a new metaheuristic algorithm based on actions of long and sudden transitions of grasshoppers in groups. Searching for food sources is an important action of grasshoppers [1]. The searching process has two stages: exploration and exploitation. The far-reaching and sudden transitions of grasshoppers simulate the exploration, and the local transitions in searching for food locations simulate the second stage.

Pushing and pulling forces between the grasshoppers are described by a mathematical model. Pushing forces allow grasshoppers to go in the search area, whereas pulling forces help them in the exploitation of the expected areas. The convergence factor in GOA algorithm decrease the comfort zone to keep exploration and exploitation in balance. Finally, the current best solution was treated as a target to be improved. The mathematical model is as [1]:

$$\mathrm{x_i} = \mathrm{S_i} + \mathrm{G} + \mathrm{A_i} \tag{1}$$

where $\mathrm{S_i}$ is the social relation:

$$\mathrm{S_i} = \sum_{j=1,j\neq i}^{N} \mathrm{S}(\mathrm{d_{ij}})\, \hat{\mathrm{d}}_{ij} \tag{2}$$

where,
N: the total number of grasshoppers
d_{ij}: the space between grasshopper *i* and grasshopper *j*

$$d_{ij} = |x_j - x_i| \tag{3}$$

$$\hat{d}_{ij} = \frac{x_j - x_i}{d_{ij}} \tag{4}$$

$\mathrm{x_i}$: is the location of the grasshopper I, G in (1) is the force of gravity, defined as:

$$G = g\hat{e} \tag{5}$$

where g is a constant and e ̂is a unit vector in the direction of the center of earth and A is the direction of wind, defined as:

$$A = ue_w \tag{6}$$

u is a fixed factor and e_w is a unit vector in the wind direction
The model described in (1) could be redefined as:

$$x_i = \sum_{j=1,j\neq i}^{N} s\,(d_{ij})\, \hat{d}_{ij} - g\hat{e} + ue_w \tag{7}$$

where,

$$S(r) = fe^{r/l} - e^{-r} \tag{8}$$

where,

f: attraction constant
l: attractive length scale

The effects of wind and gravity are small when compared with the effects of relationships between grasshoppers, the model can be redefined as (9):

$$x_i = c\,(\sum_{j=1,j\neq i}^{N} c\,\frac{hb-sb}{2}\,s(\,d_{ij})\,\hat{d}_{ij}) + \widehat{H}d \tag{9}$$

where *hb* is the higher value and *sb* is the smaller value in the considered space, $\widehat{H}d$ is the current best solution, *c* is a comfort zone parameter which is described by:

$$c = c_l\ - iter\,\frac{c_l-c_s}{Maxiter} \tag{10}$$

where c_l is the larger value (c_l=1), c_s is the smaller value (c_s=0.0004), *iter* means the running iteration, and *Maxiter* is the highest number of iterations.

GOA is equipped with a convergence parameter that iteratively decreases the comfort zone to reconcile the exploration and exploitation and skip from local optima. It is an important coefficient and can be considered as a convergence parameter. The linear comfort zone parameter proposed in (9) may be not able to achieve the balancing between the exploration and exploitation. In order to enhance the achievement of the GOA a nonlinear comfort zone parameter is suggested to replace the linear parameter. The comfort zone parameter is suggested here to be as (c^m, m=2, 3,…). In the case of linear parameter, the value is linearly decreasing with iterations while in the modified algorithm here the value of the parameter is proportional to the square of the number of iterations (for m=2). Figure 1 shows values of the comfort zone parameter against iterations for different values of m. The value (m=1) means the parameter in the original algorithm, As can be seen the effect of changing m on the curve shape. They are giving different searching mechanisms.

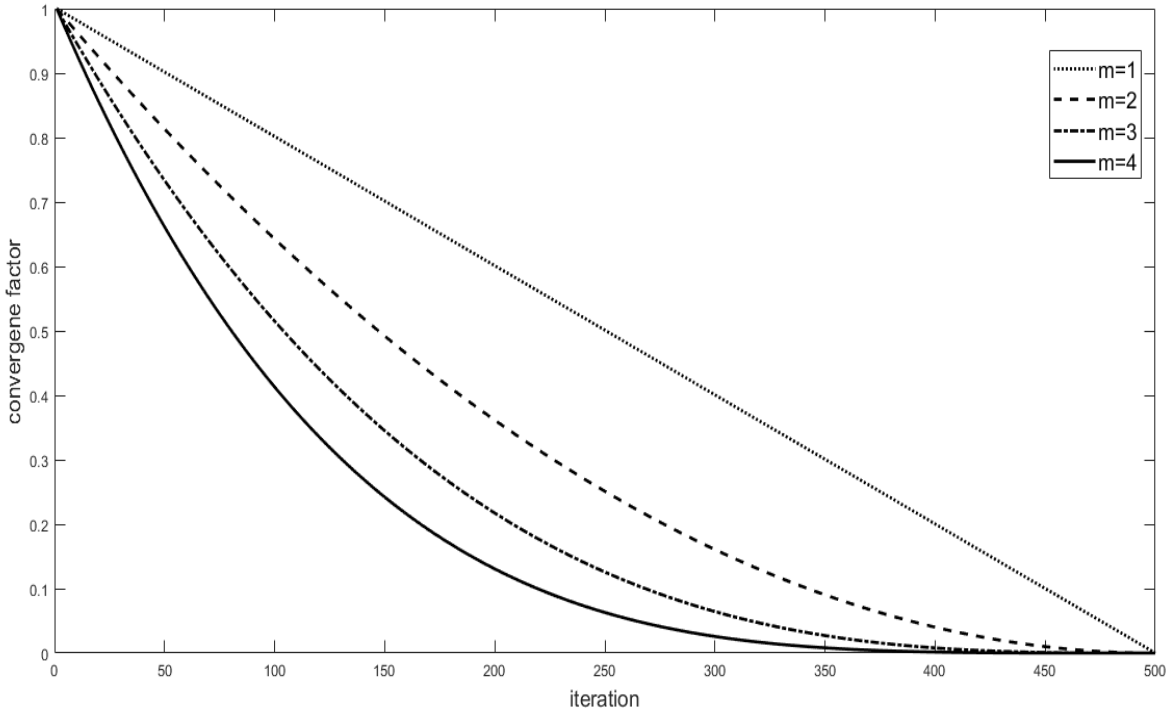

Figure 1. Comfort zone parameter for different values of m

3. MODIFIED GRASSHOPPER OPTIMIZATION ALGORITHM

In GOA a linear convergence parameter is used. The linear comfort zone parameter is not able to achieve the balancing between the exploration and exploitation In order to enhance the performance of the

GOA a nonlinear parameter is suggested to replace the linear parameter. In the modified algorithm the comfort zone parameter is chosen as:

$$c = \left[c_l - iter \frac{c_l - c_s}{Maxiter} \right]^m \quad (11)$$

where m=2, 3,…

Clearly the comfort zone parameter c directly affects the change amount in (9). If the parameter value is large the change amount is also large. The magnitude of change must be big in the exploration to keep going through the global optimum and skip from local optimum. The amount of change must be small in the exploitation to make the algorithm smoothly go in the global optimum. In original GOA, magnitude of the comfort zone parameter reduces linearly in the two stages. Also, the amount of change in (9) is fixed and cannot be arranged according to the requirements of the two stages, which reduces the efficiency of the algorithm. In the new modified algorithm the step size varies with the changing of iteration. Also the step size depends on the value of m. Each value of m give different step size and different pattern of variation as shown in Figure 1. The step size is large at the beginning points which enhance the exploration phase to ensure exploring the global optimum and avoid going to local optimum. Finally the step size decreases which increase the efficiency of exploitation by converging to the global optimum gradually. In (9) the outer c is used to reduce the transitions near the target. The inner c reduces the three zones. The initial population is generated using random numbers by the following equation:

$$x_i = x_{min} + rand[0,1](x_{max} - x_{min}) \quad (12)$$

where x_{min} and x_{max} are smaller and larger values of parameter x_i respectively.

Algorithm (The modified grasshopper optimization algorithm)

1: Select the values c_l, c_s
2: Select the size of population N and maximum number of iterations Maxiter
3: Find Random initial population xi(i = 1,..N) using (12)
4: Determine the fitness f (xi)
5: Let H=Best result
6: While stop is not satisfied do
7: Select the type of the nonlinear convergence parameter by setting the value of m
8: Update the convergence parameter c using equation (11)
9: Fori=1 to N do
10: Find the normalized distance in the range [1,4]
11:Use equation (12) to find position of current grasshopper
12: Keep the current grasshopper inside the boundaries
13: EndFor
14: Change H when the solution is better
15: iter=iter+1
16: Endwhile
17: keep the best solution as H

4. RESULTS AND DISCUSSIONS

In order to test the proposed modification. The benchmark used in the GOA original algorithm is adopted as a test environment. It consists of different kinds of functions such as unimodal and multimodal. The mathematical description and properties of these functions are illustrated in Table 1 and Table 2. Results of the modified GOA method have been compared to results of original GOA. All programs were written in MATLAB on a PC with the windows 10 and the RAM 8GB. In all test functions for the unimodal and multimodal 30 search agents over 500 iterations are used Population size is 100 for each function. In the experiment 10 attempts were performed using random initial values to find all results, the Mean fitness. standard deviation (SD), the best fitness and worst fitness. Results for the unimodal functions F1-F7 are presented in Table 3 (see in appendix). One can notice that the new MGOA show an excellent results in F1, F2, F4, F6 and F7 functions for m=2, 3 and 4. Results for m=4 are the best and indicate that the exploitation is improved in the modified algorithms. The main reason for the improved performance of MGOA is the replacement of the linear comfort zone by the proposed nonlinear scheme. Table 4 (see in appendix) lists the results of multimodal test functions F8-F13. The new MGOA works better in almost all test functions. Results illustrate that the new MGOA has good exploration properties.

Table 1. Unimodal functions

Function	Dim	Range	f_{mi}
$F1(x)=\sum_{i=1}^{n} x_i^2$	30	[-100-100]	0
$F2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30	[-100-100]	0
$F3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	30	[-100-100]	0
$F4(x)=Max_i\{\lvert x_i\rvert, \quad 1\le i\le n\}$	30	[-100-100]	0
$F5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	30	[-100-100]	0
$F6(x)=\sum_{i=1}^{n}(\lvert x_i+0.5\rvert)^2$	30	[-100-100]	0
$F7(x)=\sum_{i=1}^{n} i\,x_i^4+random[0,1)$	30	[-100-100]	0

Table 2. Multimodal functions

Function	Dim	Range	f_{min}
$F8(x)=\sum_{i=1}^{n}-x_i\sin(\sqrt{\lvert x_i\rvert})$	30	[-500-500]	316*d
$F9(x)=[x_i^2-10\cos(2\pi x_i)+10$	30	[-5.12- 5.12]	0
$F10(x)=-20\exp\left(-0.2\sqrt{(1/n\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\left(\frac{1}{n}\right)\sum_{i=1}^{n}cos2\,\pi x_i)\right)+20+e$	30	[-32-32]	0
$F11(x)=(\frac{1}{4000})\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(x_i/\sqrt{i})+1$	30	[-600-600]	0
$F12(x)=(\frac{\pi}{n})\{10\sin(\pi y_1)+\sum_{i=1}^{\pi n-1}(y_1-1)^2[1+10\,sin^2(y_{i+1})] + (y_n-1)^2\}(y_n-1)^2+\sum_{i=1}^{n}u(x_i,10.100.4) + \sum_{i=1}^{n}u(x_i,10.100.4)$; $y_i=1+\left(x_i+\frac{1}{4}\right)$; $u(x_i,a.k,m)=\begin{cases}k(x_i-a)^m\ x>a\\ o\ -a<x\ >a\\ k(-x_i-a)^m\ x<-a\end{cases}$	30	[-50-50]	0
$F13(x)=0.1\{sin^2(3\pi x_1)+] + \sum_{i=1}^{n}(x_1-1)^2\,[1+sin^2(3\pi x_1+1)] + (x_n-1)^2[1+sin^2(3\pi x_1+1)]\} + \sum_{i=1}^{n}u(x_x,5,100,4)$	30	[-50-50]	0

To show that results were not obtained by chance, the Wilcoxon ranksum test is used. If the probability (p) is lower than 0.05 then the difference between the two compared algorithms is significant. For this statistical test, all versions of the proposed algorithm in each test function is compared with the original algorithm. From Table 5 (see in appendix) it can be seen that the p-values between the original GOA and the proposed modified compared algorithms for all values of m in all test functions are less than 0.05. Clearly results illustrate the significant improvements in our modified versions of GOA.

In general the proposed modification results in good improvement for all values of m=2, 3 and 4 when compared with that of m=1 (original GOA algorithm). For more clarity, the best solutions were marked in bold. It can be seen that the performance of MGOA is better on almost all functions.

5. CONCLUSIONS

In this work, a novel version of GOA name MGOA was developed. A nonlinear comfort zone parameter is proposed to replace the linear parameter in the first GOA. The MGOA is evaluated using the functions presented by GOA researchers and compared with the original GOA. Results show that the new MGOA has an excellent exploitation and exploration capabilities and show that the new modified MGOA is effective and able to determine the optimal solutions in all functions. Experimental results indicate that the proposed MGOA outperforms original GOA, presenting a higher accuracy, faster convergence, and stronger stability. The future work will focus on two ideas. In the first different initial population generation strategies would be examined and in the second idea, further studies on the selection of the comfort zone parameter would be performed. Also we would find additional methods for enhancing the performance of GOA and use it in engineering applications.

APPENDIX

Table 3. Results of unimodal functions

Function	M	Mean	SD	Best	Worst
F1	1	2.1678e-06	6.6460e-06	3.4741e-07	6.1939e-06
	2	1.7792e-12	4.8005e-12	2.8956e-13	4.517e-12
	3	8.0000e-16	2.2295e-15	1.1104e-17	1.8334e-15
	4	5.0648e-21	2.0068e-20	2.2751e-22	1.6396e-20
F2	1	0.0242	0.1153	0.00032235	0.097916
	2	8.1921e-07	7.1790e-07	5.4949e-07	1.0825e-06
	3	2.1116e-08	5.9596e-08	1.707e-09	4.7223e-08
	4	4.1621e-10	2.3132e-09	1.8863e-12	1.8777e-09
F3	1	0.0589	0.0415	0.030205	0.0787536
	2	0.0082	0.0318	9.8936e-15	0.021164
	3	6.7911e-05	4.2910e-04	5.9138e-10	2.0549e-5
	4	5.4672e-07	5.1536e-06	5.5631 e-12	5.4358e-6
F4	1	0.0156	0.0632	0.0009637	0.054691
	2	2.2260e-06	6.0913e-06	3.1023e-7	4.5472e-6
	3	6.7911e-05	4.2910e-04	1.8964e-8	0.0003393
	4	1.3001e-07	1 8.0817e-07	6.0546e-10	6.4113e-7
F5	1	207.4406	778.2415	5.285	578.5194
	2	6.9545	4.5632	5.285	8.9299
	3	34.8973	105.5537	6.3543	71.2648
	4	7.5222	5.1641	4.1546	9.6059
F6	1	1.1800e-06	2.3602e-068	4.8463e-7	2.4919e-6
	2	2.9441e-12	4.7629e-12	1.8766e-12	5.2245e-12
	3	2.8883e-17	3.2975e-16	1.7654e-17	5.1127e-17
	4	3.7280e-21	6.5789e-21	2.7154e-21	5.9828e-21
F7	1	1.8255e-18	6.7801e-18	4.2132e-19	6.0694e-18
	2	2.8716e-30	2.7632e-30	1.128e-30	3.3738e-30
	3	2.6432e-39	1.0193e-38	3.3218e-44	6.5908e-39
	4	5.2546e-48	3.2254e-47	4.4555e-50	2.5652e-47

Table 4. Results of multimodal functions

Function	M	Mean	SD	best	Worst
F8	1	-6.9539e+03	1.6211e+03	-7646.987	-6327.604
	2	-8.2242e+03	1.1811e+03	-8640.5274	-7693.292
	3	-8.1603e+03	412.8873	-8302.5009	-8004.5934
	4	-6.1812e+03	1.4696e+04	-7819.1874	-7592.2661
F9	1	26.1436	32.0982	*9.9496*	*41.7881*
	2	19.8991	27.3206	10.9445	39.7983
	3	24.9734	37.7938	4.9748	51.7376
	4	16.4168	10.8309	8.9546	19.8991
F10	1	4.5754e-04	4.6372e-04	0.00026956	0.0006719
	2	6.5230e-07	6.0253e-07	3.6482e-7	9.4439e-7
	3	5.0896e-09	1.6358e-08	1.6549e-9	1.5182e-8
	4	1.4734e-11	1.4193e-11	3.5039e-12	4.1926e-11
F11	1	0.2551	0.1876	0.15587	0.31061
	2	0.2254	0.0644	0.19919	0.24359
	3	0.1783	0.2046	0.126105	0.30469
	4	0.1595	0.1737	0.076279	0.2044
F12	1	1.3859e-04	5.2703e-04	3.7083e-7	0.0003427
	2	1.0131e-06	4.1388e-06	3.459e-10	3.459e-6
	3	2.4014e-09	6.4920e-09	9.842e-12	4.6537e-9
	4	1.5215e-11	5.4786e-11	1.7518e-15	4. 8984e-11
F13	1	0.0044	0.0171	6.453 e-6	0.011068
	2	4.2959e-08	1.4298e-07	5.7973e-9	9.8332e-8
	3	1.0992e-09	3.8803e-09	1. 245e-10	3.1829e-9
	4	3.2598e-12	1.2862e-11	1.0399e-13	1.0122e-11

Table 5. Values of probability from Wilcoxon ranksum test

Function/m	2	3	4
F1	1.6494e-04	1.7265e-04	1.6494e-04
F2	1.6118e-04	1.6118e-04	1.7364e-04
F3	1.5748e-04	1.5748e-04	1.1203e-04
F4	1.5748e-04	1.6494e-04	1.6494e-04
F5	0.0107	0.1023	0.0206
F6	1.6494e-04	1.6494e-04	1.6494e-04
F7	1.4332e-04	1.4332e-04	1.5295e-04
F8	1.4851e-04	1.4851e-04	0.00196
F9	0.1612	0.8205	0.0409
F10	1.6780e-04	1.6780e-04	1.6494e-04
F11	0.0738	0.0108	0.0071
F12	0.0043	1.5748e-04	1.5748e-04
F13	1.5748e-04	1.5748e-04	1.5748e-04

REFRENCES

[1] S. Saremia, S. Mirjalili, and A. Lewis, "Grass hopper Optimisation Algorithm: Theory and application," *Advances in Engineering Software*, vol. 105, pp. 30-47, 2017, doi: 10.1016/j.advengsoft.2017.01.004.

[2] H. Pinto, A. Peña, M. Valenzuela, and A. Fernández, "A binary grasshopper algorithm applied to the knapsack problem," in *Artcial Intelligence and Algorithms in Intelligent Systems (Advances in Intelligent Systems and Computing)*, vol. 764, R. Silhavy, Ed. Cham, Switzerland: Springer, 2019, doi: 10.1007/978-3-319-91189-2_14.

[3] B. Crawford, R. Soto, A. Peña, and G. Astorga, "A binary grasshopper optimisation algorithm applied to the set covering problem," in *Cybernetics and Algorithms in Intelligent Systems (Advances in Intelligent Systems and Computing)*, vol. 765, pp. 1-12, R. Silhavy, Ed. Cham, Switzerland: Springer, 2018 , do: 10.1007/978-3-319-91192-2_1.

[4] A. Saxena, S. Shekhawat, and R. Kumar, "Application and development of enhanced chaotic grasshopper optimization algorithms," *Modelling and Simulation* in *Engineering*, vol. 2018, pp. 1-14, Dec. 2018, doi: 10.1155/2018/4945157.

[5] P. Zhang, W. Ma, Y. Dong, and B. D. Rouyendegh, "Multi-area economic dispatching using improved grasshopper optimization algorithm," *Evolving Systems*, vol. 12, pp. 837–847, 2021, doi: 10.1007/s12530-019-09320-6.

[6] G. Saravanan, D. S. Kumar, A. M. Ibrahim, and C. Karthikeyan, "Enhanced chaotic grasshopper optimization algorithm based PID controller for automatic voltage regulator system," *International Journal of Recent Technology and Engineering*, vol. 8, no. 4, pp. 348-354, 2019, doi: 10.35940/ijrte.D6874.118419.

[7] S. Arora and P. Anand, "Chaotic grasshopper optimization algorithm for global optimization," *Neural Computing and Applications*, vol. 31, no. 8, pp. 4385-4405, Aug. 2019, doi: 10.1007/s00521-018-3343-2.

[8] P. Suriya, S. Subramanian, S. Ganesan, and M. Abirami, "Generation and transmission expansion management using grasshopper optimization algorithm," *International Journal of Engineering Business Management*, vol. 11, Jan. 2019, doi: 10.1177/1847979018818320.

[9] A. Saxena, "A comprehensive study of chaos embedded bridging mechanisms and crossover operators for grasshopper optimisation algorithm," *Expert Systems with Applications*, vol. 132, pp. 166-188, Oct. 2019, doi: 10.1016/j.eswa.2019.04.043.

[10] Z. Xu *et al*., "Orthogonally-designed adapted grasshopper optimization: A comprehensive analysis," *Expert Systems with Applications*, vol. 150, no. 113282, Jul. 2020, doi: 10.1016/j.eswa.2020.113282.

[11] A. Saxena and R. Kumar, "Chaotic variants of grasshopper optimization algorithm and their application to protein structure prediction," in *Applied Nature-Inspired Computing: Algorithms and Case Studies (Springer Tracts in Nature-Inspired Computing)*, Singapore: Springer, pp. 151-175, 2020, doi: 10.1007/978-981-13-9263-4_7.

[12] S. Dwivedi, M. Vardhan, and S. Tripathi, "An effect of chaos grasshopper optimization algorithm for protection of network infrastructure," *Computer Networks*, vol. 176, Jul. 2020, no. 107251, doi: 10.1016/j.comnet.2020.107251.

[13] H. Zhang, Z. Gao, J. Zhang, and G. Yang, "Visual tracking with levy flight grasshopper optimization algorithm," in *Pattern Recognition and Computer Vision (Lecture Notes in Computer Science)*, vol. 11857, pp. 217-227, Z. Lin et al., Eds. Cham, Switzerland: Springer, 2019, doi: 10.1007/978-3-030-31654-9_19.

[14] R. Zhao, H. Ni, H. Feng, and X. Zhu, "A dynamic weight grasshopper optimization algorithm with random jumping," in *Advances in Computer Communication and Computational Sciences (Advances in Intelligent Systems and Computing)*, vol. 924, pp. 401-413, S. Bhatia, S. Tiwari, K. Mishra, and M. Trivedi, Eds. Singapore: Springer, 2019. doi: 10.1007/978-981-13-6861-5_35.

[15] A. Alhejji, M. E. Hussein, S. Kamel and S. Alyami, "Optimal Power Flow Solution With an Embedded Center-Node Unified Power Flow Controller Using an Adaptive Grasshopper Optimization Algorithm," in IEEE Access, vol. 8, pp. 119020-119037, 2020, doi: 10.1109/ACCESS.2020.2993762.

[16] K. Shankar, M. Elhoseny, E. D. Chelvi, S. K. Lakshmanaprabu and W. Wu, "An Efficient Optimal Key Based Chaos Function for Medical Image Security," in *IEEE Access*, vol. 6, pp. 77145-77154, 2018, doi: 10.1109/ACCESS.2018.2874026.

[17] M. Mansoor, A. F. Mirza, Q. Ling, and M. Y. Javed, "Novel grass hopper optimization based MPPT of PV systems for complex partial shading conditions," *Solar Energy*, vol. 198, pp. 499-518, Mar. 2020, doi: 10.1016/j.solener.2020.01.070.

[18] I. Aljarah, A. M. Al-Zoubi, H. Faris, M. A. Hassonah, S. Mirjalili, and H. Saadeh, "Simultaneous feature selection and support vector machine optimization using the grasshopper optimization algorithm," *Cognitive Computation*, vol. 10, no. 3, pp. 478–495, 2018, doi: 10.1007/s12559-017-9542-9.

[19] J. Wu *et al*., "Distributed trajectory optimization for multiple solar-powered UAVs target tracking in urban environment by adaptive grasshopper optimization algorithm," *Aerospace Science and Technology*, vol. 70, pp. 497-510, Nov. 2017, doi: 10.1016/j.ast.2017.08.037.

[20] B. Hekimoğlu and S. Ekinci, "Grasshopper optimization algorithm for automatic voltage regulator system," *2018 5th International Conference on Electrical and Electronic Engineering (ICEEE)*, 2018, pp. 152-156, doi: 10.1109/ICEEE2.2018.8391320.

[21] S. Z. Mirjalili, S. Mirjalili, S. Saremi, H. Faris, and I. Aljarah, "Grasshopper optimization algorithm for multi-objective optimization problems," *Applied Intelligence*, vol. 48, no. 4, pp. 805–820, 2018, doi: 10.1007/s10489-017-1019-8.

[22] J. Luo, H. Chen, Q. Zhang, Y. Xu, H. Huang, and X. Zhao, "An improved grasshopper optimization algorithm with application to financial stress prediction," *Applied Mathematical Modelling*, vol. 64, pp. 654–668, 2018, doi: 10.1016/j.apm.2018.07.044.

[23] A. A. Ewees, M. A. Elaziz, and E. H. Houssein, "Improved grasshopper optimization algorithm using opposition-based learning," *Expert Systems with Applications*, vol. 112, pp. 156–172, 2018, doi: 10.1016/j.eswa.2018.06.023.
[24] S. R. Gampa, K. Jasthi, P. Goli, D. Das, and R. C. Bansal, "Grasshopper optimization algorithm based two stage fuzzy multiobjective approach for optimum sizing and placement of distributed generations, shunt capacitors and electric vehicle charging stations," *Journal of Energy Storage*, vol. 27, Feb. 2020, p. 101117, doi: 10.1016/j.est.2019.101117.
[25] L. Bhukya and S. Nandiraju, "A novel photovoltaic maximum power point tracking technique based on grasshopper optimized fuzzy pproach," *International Journal of Hydrogen Energy*, vol. 45, no. 16, pp. 9416-9427, Mar. 2020, doi: 10.1016/j.ijhydene.2020.01.219.
[26] Z. T. Allawi and T. Y. Abdalla, "An optimal defuzzification method for interval type-2 fuzzy logic control scheme," *2015 Science* and Information Conference (SAI), 2015, pp. 619-627, doi: 10.1109/SAI.2015.7237207.
[27] A. Y. Abdalla and T. Y. Abdalla, "A modified artificial bee colony based fuzzy motion tracking scheme for mobile robot", Bulletin of Electrical Engineering and Informatics, vol. 11, no. 4, pp. 2160-2168, 2022, doi: 10.11591/eei.v11i4.3739.
[28] W. A. Shutnan and T. Y. Abdalla, "Artificial immune system based optimal fractional order PID control scheme for path tracking of robot manipulator," *2018 International Conference on Advance of Sustainable Engineering and its Application (ICASEA)*, 2018, pp. 19-24, doi: 10.1109/ICASEA.2018.8370949.
[29] A. K. A. Zahra and T. Y. Abdalla, "Design of Fuzzy Super Twisting Sliding Mode Control Scheme for Unknown Full Vehicle Active Suspension Systems Using an Artificial Bee Colony Optimization Algorithm," *Asian Jounal of control*, vol. 23, no. 4, pp 1966–1981, 2021, doi: 10.1002/asjc.2352.
[30] R. Zhao, H. Ni, H. Feng, Y. Song and X. Zhu, "An Improved Grasshopper Optimization Algorithm for Task Scheduling Problems," *International Journal of Innovative Computing, Information and Control*, vol. 15, no. 5, pp. 1967-1987, 2019, doi: 10.24507/ijicic.15.05.1967.
[31] H. Hu, P. Qin, and Z. Yang, "The improved grasshopper optimization algorithm and its applications," *Scientific Reports*, vol. 11, no. 23733, 2021, doi: 10.1038/s41598-021-03049-6.

BIOGRAPHIES OF AUTHORS

Abdulkareem Y. Abdalla is received the BSc. and the MSc. in computer science from Basra University in Iraq, and the PhD in artificial intelligence from the same university. He has worked at the same university as a lecturer and, assistant professor and as a head of the department for many years. His current areas of research are soft computing, optimization and intelligent system. He can be contacted at email: abdulkareem.abdalla@uobasrah.edu.iq.

Turki Y. Abdalla is received the BSc. degree and the MSc. in electrical engineering from Basra University in Iraq, and the PhD in computing and automation from the same university. He has worked at the same university as a lecturer and assistant professor for many years and currently he is a full professor in the same university. His current areas of research are robotics and optimization algorithms. He can be contacted at email: turki.abdalla@uobasrah.edu.iq.