2021 13th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC) | 978-1-6654-2836-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/IHMSC52134.2021.00017

Incorporation of Improved Differential Evolution into Hunger Games Search Algorithm

Sheng Li*, Jiayi Li†, Junyan Yi‡, Hang Yu*, Shi Wang*, and Shangce Gao†
*College of Computer Science and Technology, Taizhou University, Taizhou, 225300 China
†Faculty of Engineering, University of Toyama, Toyama, 930-8555 Japan,
‡Beijing University of Civil Engineering and Architecture, Beijing, 100044 China
Email: gaosc@eng.u-toyama.ac.jp

***Abstract*—Hunger games search (HGS) is a recently proposed meta-heuristic algorithm based on hunger-driven activities and animal behavior choices, and it has been proven to possess global exploration ability and can solve both constrained and unconstrained problems effectively. Differential evolution (DE) algorithm is a heuristic random search algorithm based on the evolution differences among different individuals, and it can exploit local regions within a neighborhood of an individual. To fully utilize characteristics of both algorithms, this paper for the first time proposes a hybrid algorithm based on HGS and DE. Furthermore, DE is also enhanced by a new ranking-based mechanism to generate more promising solutions. Experimental results on IEEE CEC2017 benchmark functions indicate the effectiveness of the hybrid algorithm, namely DEHGS, in comparison with some other state-of-the-art algorithms.**

***Keywords*—Computational intelligence, Hunger Games Search, Differential evolution, Optimization, Meta-heuristics**

I. Introduction

Heuristic algorithms have received great interests in the past decades [1], [2]. Various heuristic algorithms have emerged, and these algorithms can be roughly divided into two categories: evolutionary algorithms and swarm intelligence algorithms [3], [4]. Evolutionary algorithms usually mimic the evolution of organisms in nature. For example, genetic algorithms [5], which imitate Darwin's evolution and Mendel's genetic doctrine by computer, or differential evolution (DE) [6]–[9], which is based on simple differences among solutions. Swarm intelligence (SI) algorithms usually imitate the social behavior among organisms, and some representative SIs include particle swarm optimization (PSO) [10], [11], grey wolf optimizer (GWO) [12], [13], and hunger games search (HGS) [14].

In particular, HGS has drawn dramatically attention among the SI community due to its inherent specific metaphor and search mechanisms since its proposal in 2021. Yang et al. [14] proposed the HGS algorithm based on the phenomenon that hunger drives animals to find food. To be specific, hunger is what drives animals to move, learn and find food. HGS calculates the hunger level of each individual (i.e., solution for the problem) by each iteration. The exploration ability of an individual is proportional to its hunger level, so that individuals with high hunger level have more exploration ability and vice verse. The population is made to keep moving toward the region of the optimal solution. HGS is supported by an accurate and detailed mathematical derivation process and has shown its excellent global exploration capacity in various problems [14]. However, HGS still suffers from the problem of premature convergence and cannot find better solutions due to its weak local exploitation ability. In addition, the development and exploration of HGS is strongly affected by the hunger level of individuals in the population and the number of iterations. But, along with the number of iterations gradually approaches the predefined maximal number of iterations, the individuals in the population only perform simple operations near the optimal individuals, which significantly limits the performance of HGS.

In this paper, we propose a new algorithm based on HGS and the well-known DE. It can be expected that the newly proposed DEHGS can inherit search characteristics from both algorithms, i.e., the excellent exploration ability from HGS and the powerful exploitation capacity from DE. Additionally, to further improve the ability of DE, we also introduce a novel ranking-based solution generation mechanism. We generate new individuals by sorting all the individuals in the population according to their fitness in a descending order, and perform the crossover operation between the mutated high ranking individuals with the worse ones. Extensive experiments are conducted based on 30 IEEE CEC2017 benchmark functions. Convergence results show that the proposed DEHGS tends to maintain a better balance between exploitation and exploration. Solution accuracy together with the Wilcoxon statistical results indicate that DEHGS significantly outperforms its competitors, including the original HGS, a genetic learning particle swarm optimization [15], and another recently proposed meta-heuristic algorithm, i.e., wingsuit flying search (WFS) [16].

The contribution of this paper can be summarized as follows:

1) First and foremost, we for the first time proposed a new algorithm which incorporates the DE's local exploitation ability into the HGS's global exploration. It not only gives another evidence that well-performing algorithms should maintain a balance between exploration and exploitation, but also gives more insights into the search dynamics of such hybridization.

2) Secondly, we introduce a novel ranking-based generation mechanism in DE, which greatly improve its local search ability.

3) Last but not least, extensive experimental results also verify the superiority of the proposed DEHGS in comparison with its peers. Thus, the proposed algorithm also has great potential in solving other practical problems.

The remainder of this paper is organized as: Section II briefly introduces the principle of the basic HGS algorithm. In Section III, the newly proposed DEHGS algorithm is introduced in details. The experimental results and detailed analysis are presented in Section IV. Finally, we draw conclusions and point out some future work in Section V.

II. Hunger Games Search Algorithm

In HGS, there are two main methods for generating new individuals. One is to generate new individuals randomly around the original individual by relying only on the information of the original individual. The other is to generate new individuals based on the information of the best individual of this iteration. This is achieved by calculating the distance between the original individual and the best individual with the hunger value of the individual. The following equation is the search process for each iteration of HGS:

$$\vec{X_i}(t+1) = \begin{cases} \vec{X_i}(t) \cdot (1 + randn(1)), & r_1 < 0.03 \\ M_1 + M_2, & r_1 \geq 0.03, r_2 > E \\ M_1 - M_2, & r_1 \geq 0.03, r_2 \leq E \end{cases} \tag{1}$$

where t is the iteration number. $\vec{X_i}(t)$ represents the ith individual of the tth iteration in the population. $randn(1)$ is a random number satisfying normal distribution. r_n (n is the natural number) represents a random number which is in the range of $[0, 1]$. M_1 and M_2 represent two variables that arise when new individual is generated around the best individual, respectively. E is a variation used to control whether $M1$ and $M2$ are added or subtracted, and the formula for calculating E is as follows:

$$E = \frac{2}{e^{|F(i)-BF|} + e^{-|F(i)-BF|}} \tag{2}$$

where i is the individual number. $F(i)$ is the fitness of the ith individual. BF is the best fitness obtained in the current iteration process.

$$M_1 = \vec{W}_1 \cdot \vec{X}_b \tag{3}$$

$$M_2 = \vec{R} \cdot \vec{W}_2 \cdot \left| \vec{X}_b - \vec{X}(t) \right| \tag{4}$$

Eqs. (3)(4) are the calculation of M_1 and M_2. $\vec{X}_b$ represents the best individual in the current population. $\vec{W}_1$ and $\vec{W}_2$ are variables calculated based on the current individual hunger values. R is used to control the search range for generating new individuals and takes values in the range of $[-shrink, shrink]$. As the number of iterations increases R gradually decreases to 0. The variable $shrink$ is calculated as follows:

$$shrink = 2 \times (1 - \frac{FES}{MAXFES}) \tag{5}$$

The formula for $\vec{W}_1$ and $\vec{W}_2$ are shown below:

$$\vec{W}_1(i) = \begin{cases} hungry(i) \cdot \frac{N}{sum(hungry)} \times r_4, & r_3 < 0.03 \\ 1, & r_3 \geq 0.03 \end{cases} \tag{6}$$

$$\vec{W}_2(i) = \{ (1 - e^{-|hungry(i) - sum(hungry)|}) \times r_5 \times 2 \tag{7}$$

where N represents the number of individuals in the population. $hungry(i)$ represents the hunger value of each individual and the formula for $hungry(i)$ is shown below:

$$hungry(i+1) = \begin{cases} 0, & F(i) = BF \\ hungry(i) + H, & F(i) \neq BF \end{cases} \tag{8}$$

The formula for H is expressed as:

$$TH = \frac{F(i) - BF}{WF - BF} \times r_6 \times 2 \times (UB - LB) \tag{9}$$

$$H = \begin{cases} LH \times (1 + r), & TH < 100 \\ TH, & TH \geq 100 \end{cases} \tag{10}$$

where BF is the fitness of the best individual and WF is the fitness of the worst individual. UB and LB indicate the upper and lower bounds of the search space, respectively.

III. Incorporation of Improved Differential Evolution into Hunger Games Search Algorithm

In this section, we propose a new hybrid algorithm, namely DEHGS, which incorporates the search mechanisms of DE into the HGS algorithm. In addition, a ranking-based generation method is also proposed to further enhance the local exploitation of DE.

A. Ranking-based generation method

In life we can often see people who are more capable of guiding and helping people who are less capable. For example, students with good grades in a class tend to teach students with lower academic performance. Inspired by this phenomenon, we propose a novel ranking-based generation method in DE, which can be interpreted as follows:

Our basic idea is that, to further enhance the local exploitation ability of algorithms, the fitness of those solutions with low affinity should be largely improved. An promising way to realize this is to use information provided by good individuals in the population. That is solutions with low affinity should learn toward good ones. In the traditional DE algorithm, the variation operation is performed on all the individuals in the population, no matter their fitness. However, the mutated offspring solutions based on good and bad solutions are probably not better than its parents. In contrast, mutation between good individuals has a higher chance of producing better individuals. Therefore, in this paper, each individual in the population is ranked according to its fitness in a descending order. The first half of the individuals are regarded as good individuals, while the second half is treated as bad individuals. In addition, the numbers of these good individuals is denoted as $GoodPopIndexes$. The mutation is performed between

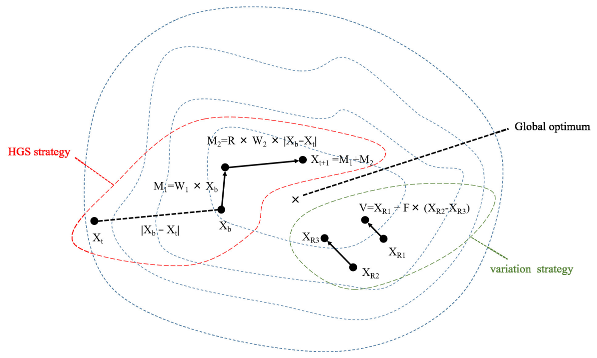

Fig. 1. This diagram shows the evolutionary steps of the DEHGS algorithm.

the good individuals, and crossover operator is implemented between the resulting mutated population with the poorer individuals in the second half.

The implementation of the ranking-based generation method can be mathematically represented as:

$$V = X_{R1} + F \times (X_{R2} - X_{R3}) \tag{11}$$

$$\vec{U}_i^{t+1} = \begin{cases} \vec{V}_i, & r_7 < CR \\ \vec{X}^t_{GoodPopIndexes(i)}, & r_7 \geq CR \end{cases} \tag{12}$$

where $R1$, $R2$, and $R3$ are randomly selected from a random permutation of $GoodPopIndexes$. $GoodPopIndexes(i)$ represents the ith index in $GoodPopIndexes$. $\vec{X}^t_i$ represents the ith individual in the tth generation population. $\vec{V}$ represents the individual in V corresponding to the ith individual number. F and CR represent the shrinkage factor and increasing crossover factor in the DE algorithm, respectively, which can affect the convergence speed and robustness of the algorithm. In this paper, we define $F = 0.3$ and $CR = 0.9$ [17].

B. DEHGS Algorithm

The evolutionary step diagram of the DEHGS algorithm is shown in Fig. 1. The HGS algorithm strategy is shown in the red circle on the left side of the figure, and the crossover variation strategy is shown in the green circle on the right side. The DEHGS algorithm alternately uses the HGS algorithm strategy and the crossover variation strategy to balance the exploration and exploitation of evolutionary algorithms. As a key issue of evolutionary algorithm, the balance between exploration and exploitation of the search has been widely studied [18]–[25]. Following these previous research, this work also aims to tackle this problem based on DE and HGS.

To be specific, when $rand(0, 1) < 0.5$, the HGS algorithm strategy is used for the population, otherwise the crossover variation strategy is used. In Fig. 1, $\vec{X}_{t+1}$ is the a new individual generated by the HGS algorithm, V is a new individual obtained by using the crossover variation strategy using the variation of good individuals, and V is crossed with the randomly selected individuals from the poorer population to obtain the final generated individual.

Pseudo code Algorithm 1 shows the procedures of the DEHGS algorithm.

Algorithm 1 Procedures of DEHGS

```
1: Initialize the parameters $N, T, D, SumHungry, F, CR$
2: Initialize the positions of Individuals $X_i (i = 1, 2, \ldots, N)$
3: while $FES \leq MAXFES$ do
4:   if $rand(0, 1) < 0.5$ then
5:     Calculate the fitness of all Individuals
6:     Update $BF, WF, X_b$
7:     Calculate the $Hungry, W_1, W_2$
8:     for $i = 1$ to $N$ do
9:       Calculate the $E$
10:      Update $R$
11:      Update $X_i$ by Eq. (1)
12:      $FES = FES + 1$
13:    end for
14:  else
15:    Initialize the $IndexSorted = sort(fitness)$
16:    $GoodPops = X_i(i = IndexSorted(1, 2, ..., N/2))$
17:    $BadPops = X_i(i = IndexSorted(1 + N/2, ..., N - 1, N))$
18:    Initialize the $R1, R2, R3$
19:    Initialize the new populations $U$
20:    $V = GoodPops_{R1} + F \times (GoodPops_{R2} - GoodPops_{R3})$
21:    Generate $j_{rand} = randint(1, D)$
22:    for $i = 1$ to $N/2$ do
23:      for $j = 1$ to $D$ do
24:        if $j = j_{rand}$ or $rand(0, 1) < CR$ then
25:          $U_i(j) = V_i(j)$
26:        else
27:          $U_i(j) = BadPops_i(j)$
28:        end if
29:      end for
30:    end for
31:    Calculate the fitness of $U$
32:    for $i = 1$ to $N/2$ do
33:      $n = IndexSorted(i)$
34:      $X_n = Selection(X_n, U_i)$
35:      $FES = FES + 1$
36:    end for
37:  end if
38: end while
```

IV. Experiment Results

DEHGS has been tested on 30 IEEE CEC2017 benchmark functions, including four types of problems: unimodal functions (F1-F3), simple multimodal functions (F4-F10), hybrid functions (F11-F20), and combined functions (F21-F30). Traditional HGS, genetic learning particle swarm optimizatio (GLPSO) [15] and wingsuit flying search algorithm (WFS) [16] algorithms are used to compare and evaluate the performance of DEHGS. Each algorithm is independently run 51 times, and the maximum function evaluating number $MAXFES$ is set to $10000D$, where D is the dimension size. The number of populations for all algorithms is initialized to 100.

The experimental results are summarized in Table I, where the best values are shown in bold. It records the mean and standard deviation of the obtained errors with the best-known results for all tested four algorithms. From this table, it is

TABLE I. EXPERIMENT RESULTS ON IEEE CEC2017.

	DEHGS		HGS		GLPSO		WFS	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	**5.86E+03**	6.02E+03	1.04E+04	7.72E+03	9.85E+04	4.74E+05	7.06E+08	3.32E+08
F2	1.04E+07	6.84E+07	**7.32E+02**	2.87E+03	3.43E+24	1.44E+25	3.64E+28	1.61E+29
F3	5.91E+02	9.26E+02	**3.85E+02**	6.59E+02	2.19E+04	5.15E+03	1.50E+04	4.39E+03
F4	9.31E+01	2.52E+01	**8.73E+01**	3.19E+01	2.91E+02	9.24E+01	2.45E+02	5.22E+01
F5	**8.86E+01**	2.42E+01	1.17E+02	3.55E+01	1.76E+02	1.92E+01	1.45E+02	2.98E+01
F6	**5.43E-01**	9.19E-01	5.98E-01	7.81E-01	5.09E+00	2.06E+00	2.55E+01	5.93E+00
F7	**1.43E+02**	3.81E+01	1.66E+02	3.42E+01	1.62E+02	5.41E+01	2.34E+02	3.75E+01
F8	**8.54E+01**	2.04E+01	1.09E+02	2.49E+01	1.53E+02	3.82E+01	1.35E+02	3.10E+01
F9	1.09E+03	8.64E+02	2.36E+03	9.32E+02	**1.41E+01**	9.25E+00	1.77E+03	1.13E+03
F10	**2.65E+03**	5.33E+02	2.75E+03	4.72E+02	6.54E+03	3.35E+02	4.60E+03	6.50E+02
F11	**7.88E+01**	3.46E+01	1.13E+02	4.20E+01	1.32E+02	6.01E+01	3.21E+02	6.94E+01
F12	**6.26E+05**	5.41E+05	1.02E+06	7.52E+05	7.84E+06	1.33E+07	9.80E+07	7.88E+07
F13	**1.89E+04**	2.01E+04	2.15E+04	2.14E+04	5.50E+04	2.30E+05	7.47E+05	8.14E+05
F14	4.87E+04	4.75E+04	4.66E+04	4.08E+04	**3.53E+04**	8.10E+04	7.67E+03	7.82E+03
F15	**1.06E+04**	1.11E+04	2.19E+04	1.57E+04	8.49E+03	8.31E+03	1.55E+05	1.58E+05
F16	**1.04E+03**	2.68E+02	1.10E+03	2.90E+02	1.36E+03	2.05E+02	9.72E+02	2.50E+02
F17	4.33E+02	2.03E+02	5.34E+02	2.06E+02	**2.78E+02**	1.65E+02	3.22E+02	1.07E+02
F18	**1.86E+05**	1.77E+05	2.59E+05	3.09E+05	6.95E+05	7.50E+05	2.16E+05	1.48E+05
F19	**1.31E+04**	1.63E+04	1.62E+04	1.88E+04	9.55E+03	1.39E+04	1.10E+06	1.13E+06
F20	4.49E+02	2.05E+02	5.40E+02	2.15E+02	**2.79E+02**	1.38E+02	3.92E+02	1.08E+02
F21	**2.88E+02**	1.87E+01	3.15E+02	2.69E+01	3.74E+02	2.34E+01	3.33E+02	2.85E+01
F22	1.25E+03	1.49E+03	2.70E+03	1.37E+03	**1.02E+02**	2.32E+00	3.03E+02	8.44E+01
F23	**4.55E+02**	2.39E+01	4.56E+02	2.11E+01	5.93E+02	2.10E+01	5.31E+02	3.99E+01
F24	**5.70E+02**	3.85E+01	5.93E+02	4.54E+01	6.56E+02	2.18E+01	5.79E+02	3.66E+01
F25	3.91E+02	1.23E+01	**3.90E+02**	9.86E+00	4.33E+02	2.13E+01	5.23E+02	3.62E+01
F26	**1.98E+03**	5.70E+02	2.22E+03	5.05E+02	2.94E+03	9.36E+02	2.54E+03	7.02E+02
F27	5.21E+02	1.54E+01	**5.21E+02**	1.41E+01	6.67E+02	2.15E+01	6.19E+02	3.09E+01
F28	**4.18E+02**	3.16E+01	4.22E+02	4.04E+01	5.46E+02	7.72E+01	6.33E+02	7.87E+01
F29	**8.05E+02**	2.07E+02	8.73E+02	1.71E+02	8.68E+02	1.78E+02	1.01E+03	1.44E+02
F30	**8.17E+03**	3.94E+03	8.60E+04	1.30E+05	9.17E+04	1.54E+05	6.90E+06	5.63E+06
W/T/L	-/-/-		16/11/3		19/6/5		24/4/2	

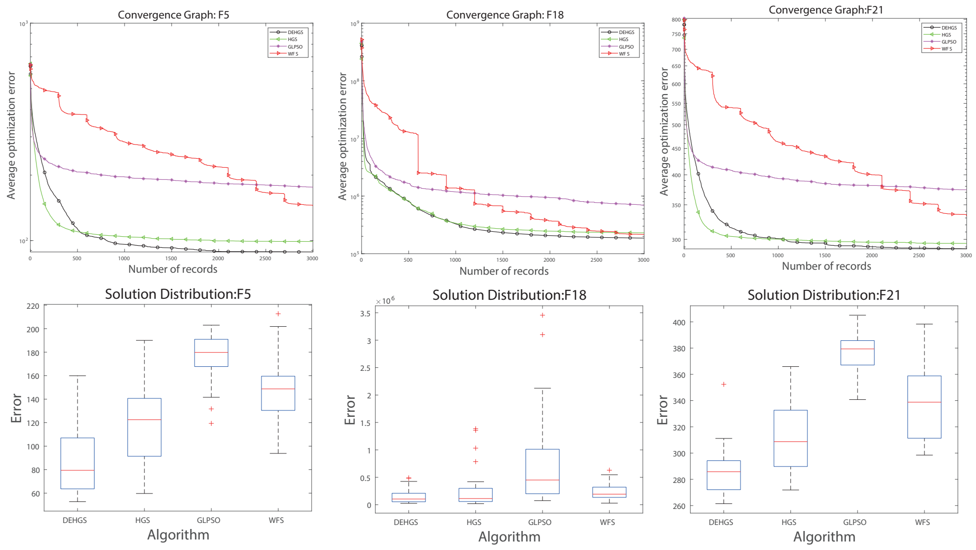

Fig. 2. Convergence graphs and Boxplot graphs on F5, F18, and F21 of all tested algorithms.

clear that the proposed DEHGS outperforms its competitors on 20 out of 30 functions. Additionally, we also use Wilcoxon statistical test with a significance level of $\alpha = 0.05$ to detect the difference among algorithms. The W/T/L value in Table I represents the DEHGS algorithm performs significantly better/tied/worse than its counterpart, respectively. From it, we can conclude that DEHGS significantly outperforms its peers in terms of solution accuracy.

Furthermore, Fig. 2 shows the convergence graphs and box-and-whisker graphs obtained by tested algorithms. Three typical functions F5, F18, and F21 are illuminating to exhibit the convergence properties of each algorithm. It is apparent that DEHGS has a fast convergence speed in comparison with the others, and it can usually find better solutions than the other algorithms. In addition, the box-and-whisker graphs also verify the robustness of the proposed algorithm.

V. CONCLUSIONS

In this paper, we study the HGS algorithm based on hunger search mechanism and DE algorithm based on the evolution of differences among solutions. From the investigation, we found that HGS and DE can possess global exploration and local exploitation abilities, respectively. By realizing this, we innovatively propose a hybrid DEHGS by incorporating a ranking-based generation method enhanced DE into HGS. By doing so, the exploration and exploitation of the algorithm is well balanced. Experimental results based on 30 widely used IEEE CEC2017 functions verify the superiority of the proposed method in terms of accuracy, convergence speed, and robustness in comparison with some other state-of-the-art ones.

This research opens the door to the following researches: 1) DEHGS can be further improved by using some other sophisticated mechanisms, such as population structure [26], [27], memetic computing [28], and parameter self-adaptive [29]. 2) The performance of DEHGS should be verified on real-world problems, e.g., Internet of things [30], [31], protein structure prediction [32], neural networks learning [33], etc.

Acknowledgment

This research was partially by Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Nos. 19KJB520057, 19KJB520058 and 19KJB520059) and Scientific Research Project of Beijing Municipal Education Commission (No. KM202010016011).

References

[1] X.-S. Yang, "Nature-inspired optimization algorithms: Challenges and open problems," *Journal of Computational Science*, vol. 46, p. 101104, 2020.

[2] Y. Wang, S. Gao, M. Zhou, and Y. Yu, "A multi-layered gravitational search algorithm for function optimization and real-world problems," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 1, pp. 94–109, 2021.

[3] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Computers & Industrial Engineering*, vol. 137, p. 106040, 2019.

[4] Y. Wang, Y. Yu, S. Cao, X. Zhang, and S. Gao, "A review of applications of artificial intelligent algorithms in wind farms," *Artificial Intelligence Review*, vol. 53, no. 5, pp. 3447–3500, 2020.

[5] D. Whitley, "A genetic algorithm tutorial," *Statistics and computing*, vol. 4, no. 2, pp. 65–85, 1994.

[6] S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, and M. Zhou, "Chaotic local search-based differential evolution algorithms for optimization," *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 51, no. 6, pp. 3954–3967, 2021.

[7] Z. Xu, S. Gao, H. Yang, and Z. Lei, "SCJADE: Yet another state-of-the-art differential evolution algorithm," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 16, no. 4, pp. 644–646, 2021.

[8] S. Gao, K. Wang, S. Tao, T. Jin, H. Dai, and J. Cheng, "A state-of-the-art differential evolution algorithm for parameter estimation of solar photovoltaic models," *Energy Conversion and Management*, vol. 230, p. 113784, 2021.

[9] J. Sun, S. Gao, H. Dai, J. Cheng, M. Zhou, and J. Wang, "Bi-objective elite differential evolution for multivalued logic networks," *IEEE Transactions on Cybernetics*, vol. 50, no. 1, pp. 233–246, 2020.

[10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[11] S. Song, J. Ji, X. Chen, S. Gao, Z. Tang, and Y. Todo, "Adoption of an improved PSO to explore a compound multi-objective energy function in protein structure prediction," *Applied Soft Computing*, vol. 11, pp. 539–551, 2018.

[12] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[13] Z. Wang, H. Yang, Z. Wang, Y. Todo, Z. Tang, and S. Gao, "A novel spherical search based grey wolf optimizer for optimization problems," in *2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS)*. IEEE, 2020, pp. 38–43.

[14] Y. Yang, H. Chen, A. A. Heidari, and A. H. Gandomi, "Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts," *Expert Systems with Applications*, vol. 177, p. 114864, 2021.

[15] Y.-J. Gong, J.-J. Li, Y. Zhou, Y. Li, H. S.-H. Chung, Y.-H. Shi, and J. Zhang, "Genetic learning particle swarm optimization," *IEEE Transactions on Cybernetics*, vol. 46, no. 10, pp. 2277–2290, 2015.

[16] N. Covic and B. Lacevic, "Wingsuit flying search—a novel global optimization algorithm," *IEEE Access*, vol. 8, pp. 53 883–53 900, 2020.

[17] Y. Yu, S. Gao, Y. Wang, and Y. Todo, "Global optimum-based search differential evolution," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 2, pp. 379–394, 2018.

[18] B. Morales-Castañeda, D. Zaldivar, E. Cuevas, F. Fausto, and A. Rodríguez, "A better balance in metaheuristic algorithms: Does it exist?" *Swarm and Evolutionary Computation*, vol. 54, p. 100671, 2020.

[19] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM computing surveys (CSUR)*, vol. 45, no. 3, pp. 1–33, 2013.

[20] Z. Li, H. Yang, Z. Zhang, Y. Todo, and S. Gao, "Spherical evolution enhanced with salp swarm algorithm," in *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 2020, pp. 62–66.

[21] Z. Tang, K. Wang, J. Shi, S. Tao, Y. Todo, and S. Gao, "A novel optimization algorithm inherited from gravitational and spherical search dynamics," in *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 2020, pp. 91–96.

[22] J. Shi, J. Yu, C. Lee, Y. Todo, and S. Gao, "A hybrid spherical search and whale optimization algorithm," in *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 2020, pp. 44–49.

[23] P. Cai, H. Yang, Y. Zhang, Y. Todo, Z. Tang, and S. Gao, "A sine cosine algorithm enhanced spherical evolution for continuous optimization problems," in *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 2020, pp. 1–6.

[24] L. Du, Y. Zhang, S. Sato, Y. Todo, Z. Tang, and S. Gao, "Differential evolution-based wingsuit flying search for optimization," in *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 2020, pp. 7–12.

[25] Q. Li, Y. Yu, Z. Wang, Y. Todo, and S. Gao, "A novel brain storm optimization algorithm driven by sine-cosine search mechanism," in *2020 12th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 1. IEEE, 2020, pp. 3–8.

[26] Y. Wang, S. Gao, Y. Yu, Z. Cai, and Z. Wang, "A gravitational search algorithm with hierarchy and distributed framework," *Knowledge-Based Systems*, vol. 218, p. 106877, 2021.

[27] Y. Wang, Y. Yu, S. Gao, H. Pan, and G. Yang, "A hierarchical gravitational search algorithm with an effective gravitational constant," *Swarm and Evolutionary Computation*, vol. 46, pp. 118–139, 2019.

[28] Y. Yu, S. Gao, S. Cheng, Y. Wang, S. Song, and F. Yuan, "CBSO: a memetic brain storm optimization with chaotic local search," *Memetic Computing*, vol. 10, no. 4, pp. 353–367, 2017.

[29] Z. Lei, S. Gao, S. Gupta, J. Cheng, and G. Yang, "An aggregative learning gravitational search algorithm with self-adaptive gravitational constants," *Expert Systems with Applications*, vol. 152, p. 113396, 2020.

[30] J. Cheng, G. Yuan, M. Zhou, S. Gao, C. Liu, H. Duan, and Q. Zeng, "Accessibility analysis and modeling for IoV in an urban scene," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4246–4256, 2020.

[31] J. Cheng, G. Yuan, M. Zhou, S. Gao, C. Liu, and H. Duan, "A fluid mechanics-based data flow model to estimate VANET capacity," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 6, pp. 2603–2614, 2019.

[32] S. Gao, S. Song, J. Cheng, Y. Todo, and M. Zhou, "Incorporation of solvent effect into multi-objective evolutionary algorithm for improved protein structure prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 15, no. 4, pp. 1365–1378, 2018.

[33] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neural model with effective learning algorithms for classification, approximation, and prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 2, pp. 601–614, 2019.