Proceedings of the 2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design

A Comprehensive Learning Moth-Flame Optimization with Low Discrepancy Sequence

Fuqing Zhao
School of Computer and Communication
Lanzhou University of Technology
Lanzhou, China
fzhao2000@hotmail.com

Qiaoyun Wang
School of Computer and Communication
Lanzhou University of Technology
Lanzhou, China
1309643164@qq.com

Hui Zhang
School of Computer and Communication
Lanzhou University of Technology
Lanzhou, China
1601199793@qq.com

***Abstract*—The moth-flame optimization (MFO) algorithm is extensively employed to attain the global optimization of a problem. The original MFO algorithm has drawbacks of low population diversity, slow convergence speed, and falling into local optimum easily. An improved moth-flame optimization algorithm (CLMFOLDS) based on comprehensive learning (CL) mechanism and low discrepancy sequence (LDS) is presented in this paper to solve the problem. A random population with uniform distribution is generated in the search space by using LDS. The information of the entire population is learned to gain the position of the new moth, which is called the CL strategy. External storage is designed to save the suboptimal solution throughout the iteration. The elimination mechanism is employed to strike out the poor solution in the population to enhance the global search capability of the algorithm. The CLMFOLDS is assessed in the CEC 2017 benchmark problem. Experimental results illustrate that the CLMFOLDS algorithm is superior to state-of-the-art algorithms.**

Keywords—Moth-flame optimization, Elimination mechanism, Low discrepancy sequence, Comprehensive learning strategy

I. Introduction

The optimization algorithm, which is divided into traditional optimization algorithms and intelligent optimization algorithms, is widely utilized by researchers to acquire the global optimum of a problem. Mathematical methods are utilized by traditional optimization algorithms to find the optimization, such as quadratic programming. The specific expression of function, which is hardly achieved in real life, is required in advance in traditional optimization algorithms. The expression of a concrete function is not required by the intelligent optimization algorithm. The intelligent optimization algorithm, which is divided into evolutionary algorithms (EAs) [1] and swarm intelligence (SI) [2] algorithms, only performs the sampling procedure. The iterative update of sampling points is designed to catch the optimal solution to a problem. EAs, which include genetic algorithm (GA) [3] and so on, are inspired by the evolution theory of Darwin. The optimal solution is finally found by preserving the optimization of each iteration. SI algorithms, which do not know the expression of the function early, are inspired by the social behavior of an animal in nature. SI algorithms include particle swarm optimization (PSO) [4], [5], [6], ant colony algorithm (ACO) [7], artificial bee colony algorithm (ABC) [8], [9], gray wolf algorithm (GWO) [10], and whale optimization algorithm (WOA) [11].

This work was financially supported by the National Natural Science Foundation of China under grant 62063021. It was also supported by the Key talent project of Gansu Province (ZZ2021G50700016), the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (21YF5WA086), Lanzhou Science Bureau project (2018-rc-98), and Project of Gansu Natural Science Foundation (21JR7RA204), respectively.

MFO [12] is inspired by the special navigation mechanism of a moth, which is called transverse orientation. The moth moves at a stationary degree to the moon at night. The moth flies in a straight line due to transverse orientation. However, moths are easily disturbed by artificial light sources in real life, which causes that transverse orientation to be ineffective. This conduct causes the moth to make a fatal spiral around the flame (artificial light). The moth is considered as the candidate solution and the flame is regarded as the optimization. The moth spirals towards the flame, which means that the problem wins the global optimization. Reachers are attracted by the MFO algorithm in recent years due to its simple and easy structure. MFO algorithm [13] is employed in the long-term capacity planning of micro-grid equipment. MFO [14] is employed to detect early faults in rolling bearings. MFO [15] is utilized for the disease detection of tomatoes. MFO [16] is designed to predict the complexity of protein. Variants are put forward to overcome the shortcoming of the original MFO algorithm in recent years. MFO algorithm tends to sink into local optima when dealing with high-dimensional and multi-mode problems. BFGSOLMFO [17] is the addition of Orthogonal Learning (OL) and Broyden-Fletcher-Goldfarb-Shanno (BFGS) to the MFO algorithm. The proposed algorithm alleviates the tendency of the original MFO algorithm falling into the local optimum on high-dimensional and multi-mode problems. OL is employed to generate preferable candidate solutions and guide the whole population towards the potential solution region. BFGS, which is employed to search for latent global optimization after position updating, enhances the population diversity. Gauss mutation and chaos search [18] are introduced into the original MFO algorithm. The Gauss mutation is designed to enhance the population diversity. The chaotic search is employed to update the flame to improve the global search capacity of the algorithm. Gauss mutation, Cauchy mutation, and Levy-flight [19] are designed to disrupt the population. This method not only improves the diversity of the population but also sufficiently explores the potential solution. Double adaptive weights [20] are introduced into the MFO algorithm to improve the exploration ability of the algorithm. An improved moth-flame optimization based on the comprehensive learning mechanism and low discrepancy sequence (CLMFOLDS) is proposed in this paper. Specific contributions are shown as follows.

- LDS, which ensures that the algorithm searches the overall search space adequately, is introduced in the primary phase of the population to generate a random population with uniform distribution in the search space.
- CL strategy is employed to update location to assure the balance between global and local search.
- The elimination mechanism is employed to enhance the local searching capacity of the algorithm.
- External storage is used to store suboptimal solutions and ensure population diversity.

The rest of the article is shown as follows: The original MFO algorithm is given in Section II. The CLMFOLDS algorithm is designed in Section III. Section IV discusses the experimental results and analysis. The summary and outlook are described in Section V.

II. THE ORIGINAL MOTH-FLAME OPTIMIZATION ALGORITHM

A. *Moth Search Mechanism*

The moth is assumed to be the candidate solution in the MFO algorithm. Moths explore search space by spiraling around the flame. The matrix M and F are employed to represent the position vector of the moth and flame in the search space. OM and OF are designed to store the fitness value of the moth and flame. The matrix is expressed as follows:

$$M = \begin{bmatrix} m_{1,1} & \cdots & m_{1,d} \\ \vdots & \ddots & \vdots \\ m_{N,1} & \cdots & m_{N,d} \end{bmatrix} \tag{1}$$

$$OM = \begin{bmatrix} OM_1 \\ \vdots \\ OM_N \end{bmatrix} \tag{2}$$

$$F = \begin{bmatrix} F_{1,1} & \cdots & F_{1,d} \\ \vdots & \ddots & \vdots \\ F_{flame\ no,1} & \cdots & F_{flame\ no,d} \end{bmatrix} \tag{3}$$

$$OF = \begin{bmatrix} OF_1 \\ \vdots \\ OF_{flame\ no} \end{bmatrix} \tag{4}$$

where N, which is set to 100 in this paper, is the number of moths. d is the dimension of the problem, $flame\ no$ means the number of flames. MFO algorithm requires each moth to move around the corresponding flame and uses Eq. (5) to update the position of moths.

$$S(M_i, F_j) = D_i * e^{bt} * \cos(2\pi t) + F_j \tag{5}$$

where M_i denotes the i-th moth ($i = 1, \dots, N$), F_j represents the j-th flame ($j = 1, \dots, flame\ no$), S denotes the spiral function. b, which is set to 1, represents the constant of the logarithmic spiral function. t is the path coefficient, which is a random number between [-1,1]. $t = -1$ represents the position nearest to the flame. $t = 1$ expresses the utmost distance from the flame. Fig. 1 depicts the positions of moths corresponding to different t. The moth is able to sufficiently examine the search space by adjusting t. D_i , which is calculated by Eq. (6), describes the distance between the moth and corresponding flame.

$$D_i = |M_i - F_j| \tag{6}$$

The spiral equation shows that the moth circles the flame instead of searching between them. A model of updating the position of the moth around the flame is described in Fig. 2.

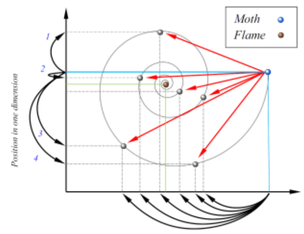

Fig. 1 The positions of moths corresponding to different t

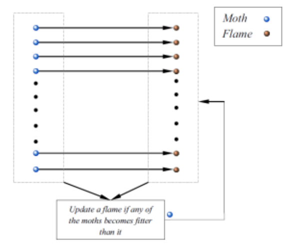

Fig. 2 The model of updating the position of moth

B. *Flame Renewal Mechanism*

The fitness of the new moth is calculated and the original flame is merged with the new moth. The merged population is sorted based on fitness. The first *flame no*-th individuals are employed as the new flame. An adaptive reduction method of flame number is proposed in Eq. (7).

$$flame\ no = round\left((N - L) * \frac{N - 1}{T}\right) \tag{7}$$

where $flame\ no$ means the number of flames, L represents the current number of iterations, T is the maximum number of iterations. The pseudocode of MFO is shown in Algorithm 1.

Algorithm 1: Pseudocode of MFO

Input: N, d, lb, ub, T
Initiate moth population M_i $(i = 1,2, \dots, N)$, calculate fitness value OM_i
While ($L <= T$)
    Use Eq. (7) to calculate the number of flames $flame\ no$
    IF $L == 1$
        $F = sort(M_L)$;
        $OF = sort(OM_L)$;
    else
        $F = sort(M_{L-1}, M_L)$;
        $OF = sort(OM_{L-1}, OM_L)$;
    End IF
    Select the first flame as the global optimum $good\_FF$
    Use Eq. (5) to replace the position of the moth
    $L = L + 1$;
End While
Return $good\_FF$

III. COMPREHENSIVE LEARNING MOTH-FLAME OPTIMIZATION WITH LOW DISCREPANCY SEQUENCE

A. *Low Discrepancy Sequence*

Pseudo-random number (PRN) method is usually designed to generate the initial population, as is shown in Eq. (8).

$$X = rand * (ub - lb) + lb \tag{8}$$

where ub and lb, which are set to 100 and -100, are the upper and lower bounds of the search space respectively, and $rand$ is a random number evenly distributed between [0,1].

100 individuals are generated within the interval [-100,100] by Eq. (8). The sampling points are not evenly distributed. LDS is proposed to initialize the population to solve the above problem. The discrepancy is calculated by Eq. (9).

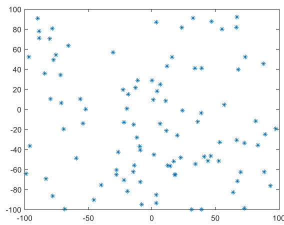

Fig. 3 The 100 individuals are generated using the PRN

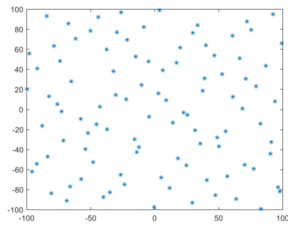

Fig. 4 The 100 individuals are generated using the LDS

$$D_N(P) = \sup\left|\frac{A(B)}{N} - \lambda_s(B)\right| \quad (9)$$

where B is the arbitrary area in space $[0,1]^n$, A is the number of points in the area B, N is the number of all points in the area B, $\lambda_s(B)$ is the volume of area B, and sup is the maximum. The van der corput sequence is calculated by (10) and (11).

$$i = \sum_{l=0}^{M-1} a_l(i)b^l \quad (10)$$

$$\Phi_b(i) = (b^{-1} \dots b^{-M})\left(a_0(i) \dots a_{M-1}(i)\right)^T = \sum_{l=0}^{M-1} a_l(i)b^{-l-1} \quad (11)$$

where b is a decimal number, which is set to an integer. i is represented as a binary number, the resulting number is flipped and placed after the decimal point. The resulting number is converted to a decimal number and used as the van der corput sequence for that number. The Halton sequence is used in LDS. The Halton sequence is defined in (12).

$$X_i = \left(\Phi_{b1}(i), \dots, \Phi_{bN}(i)\right) \quad (12)$$

where $b1...bN$ is prime to each other. Each dimension is constructed from the van der corput sequence based on a different number bN . LDS is employed to initialize the population, as is shown in Fig. 4. The randomness of the sampling process and the uniformity of the whole distribution are guaranteed. Fig. 4 is more evenly distributed than Fig. 3.

B. Comprehensive Learning Strategy

A highly adaptive moth is designed as the flame for the next iteration. The flame is generated from the moth in the original MFO, which leads to the low diversity of the flame, and further leads to the algorithm not exploring as much search space as possible. Each flame is required to learn from the optimal flame in the population by the CL strategy. The optimal flame means that every dimension of the individual is optimal. This flame does not belong to a population and is generated by using information from the entire population. The diversity of flames is enhanced as the result. The position of the moth is replaced by using (13) in the strategy.

$$M_i^d = D_i^d * e^{bt} * \cos(2\pi t) + F_j^d \quad (13)$$

where N is the number of moths, d is the dimension of the problem, $flame\ no$ means the number of flames. The learning rate is assumed to be Pc. The learning rate for each moth, which is derived from Eq. (14), adapts the algorithm to a different type of problem. Pc increases from 0.05 to 0.5.

$$Pc_i = 0.05 + 0.45 * \frac{\left(\exp\left(\frac{10(i-1)}{N-1}\right) - 1\right)}{(\exp(10) - 1)} \quad (14)$$

A random number is produced in the d-th dimension of the i-th moth. The d-th dimension of the optimal flame is used as the d-th dimension of the flame if the random number is less than Pc. Otherwise, the original flame corresponding to the moth is employed to update the position. The optimal flame is generated by the tournament selection method. Two flames are randomly selected from the flame set. The fitness values of the two flames are compared and the best one is selected as the optimal flame.

C. Elimination Mechanism

The elimination mechanism is employed to enhance the local search capacity of the algorithm. A large number of moths are demanded to explore the search space in the global search phase. A small number of elite individuals are required to search in the local area to enhance the convergence speed of the algorithm. The elimination mechanism is proposed to delete the inferior solutions in the population. Eq. (15) is employed to determine the number of individuals to be struck out in each generation.

$$num_L = N_{L+1} - \left(\frac{N_{min} - N_{max}}{T} * L + N_{max}\right) \quad (15)$$

where N_{L+1} is denoted as the size of the previous population, N_{min} is denoted as the minimum population, which is set to 3. N_{max}, which is set to 100, is expressed as the maximum population.

D. External Storage

The potential information of direction is provided by the suboptimal solution during the iteration. The original MFO algorithm only retains the information of the optimal solution and neglects the suboptimal solution. An external storage mechanism is proposed to store the suboptimal solution of each iteration. The worst individual in the external storage is deleted at the end of each iteration if the size of the storage exceeds the limit. The update formula of moth position changed as the external storage mechanism is proposed. The information about the movement of the moth towards the external storage is introduced based on (13), (16), and (17) and is used to update the position of the moth.

$$M_i^d = D_i^d * e^{bt} * \cos(2\pi t) + F_j^d + D_i'^d * e^{bt} * \cos(2\pi t) \quad (16)$$

$$D_i'^d = \left|M_i^d - F_k^d\right| \quad (17)$$

where F_k^d is the d-th dimension of the k-th ($k = 1, \dots, NP$) flame in the external storage, NP is the size of the external storage. Eq. (18) is used to calculate the size of the external storage.

$$archive.pop = N * archive.rate \quad (18)$$

where $archive.rate$ is set to 2.6 in this paper. The pseudocode of the CLMFOLDS algorithm is shown in Algorithm 2.

IV. EXPERIMENTAL STUDY AND DISCUSSION

The CEC 2017 benchmark is employed to assess the performance of CLMFOLDS. 29 functions are employed in the experiment, which include unimodal functions f_1, f_3 , simple multimodal functions $f_4 - f_{10}$, hybrid functions $f_{11} - f_{20}$, and composition functions $f_{21} - f_{30}$. The experiment is carried out in $10D$, $30D$, $50D$, and $100D$. D expresses the dimension of a problem. Each algorithm is independently run 51 times and the mean and standard are shown in TABLE I. Other variants of MFO are tested in the same experimental setting. The concrete variants include WEMFO [20], DMMFO [21], LGCMFO [19], OMFO [22], and the original MFO. The parameters of the comparison algorithm are set according to their respective papers. The experimental results are shown in TABLE I. The optimization are expressed by bold type. The results of the proposed algorithm are better than other algorithms on 25 functions. Therefore, 10-dimension data is selected when the page is limited.

Algorithm 2: Pseudocode of CLMFOLDS

Input: $N, d, ub, lb, archive.NP$
Initialize the population by using LDS $M_i(i = 1,2, \dots, N)$
Calculate fitness value $OM_i(i = 1,2, \dots N)$
While $(L <= T)$
    Use Eq. (7) to calculate the number of flames $flame\ no$
    IF $L == 1$
        $F = sort(M_L)$;
        $OF = sort(OM_L)$;
    else
        $F = sort(M_{L-1}, M_L)$;
        $OF = sort(OM_{L-1}, OM_L)$;
    End IF
    Select the first flame as the global optimum $good\_FF$
    For $i = 1: N$
        For $j = 1: d$
            Use Eq. (13) to replace the position of the moth M_L_new
        End For
    End For
    Calculate fitness value OM_L_new
    If $OM_L\_new > OM_L$
        $archive.pop = M_L\_new$;
    End If
    If $size(archive.pop) > archive.NP$
        $max\ (archive.pop) = Null$;
    End If
    Use Eq. (15) to calculate the number of individuals eliminated num_L
    For $i = 1: num_L$
        $max\ (M_L\_new) = Null$;
    End For
    Use Eq. (18) to calculate the size of external storage $archive.NP$
    L=L+1;
End While
return $Best\_FF$

The global optima of functions f_1, f_3, f_4, f_6, f_9 is found. Functions f_{15} and f_{19} are close to global optima. Due to the limitation of the article, one function of each type is chosen to draw the convergence diagram and the box plot. f_1, f_6, f_{14} and f_{30} are selected. Fig. 5 shows the convergence diagram of four functions.

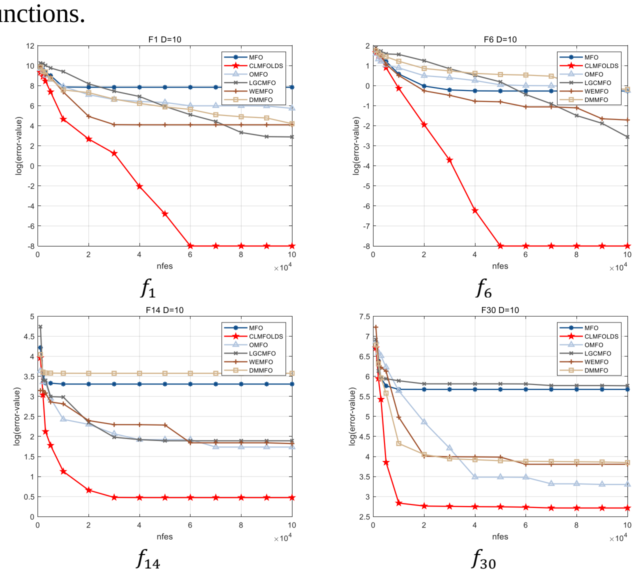

Fig. 5 The convergence curves of four functions

CLMFOLDS algorithm performs better than the other five algorithms by analyzing convergence graphs on four functions. CLMFOLDS algorithm has advantages of the fast speed of convergence and accurate solution. The population diversity and the global search capability of the algorithm are greatly improved through the use of CL strategy and external storage mechanism. The local search capability of the algorithm is enhanced by using the elimination mechanism. The box plots of f_1, f_6, f_{14} and f_{30} are shown in Fig. 6. The stability and accuracy of the CLMFOLDS algorithm are better than the other five algorithms.

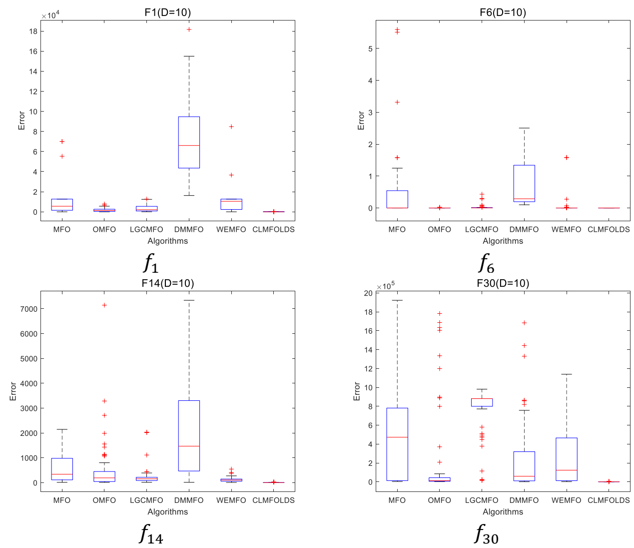

Fig. 6 The box plots of four functions

Wilcoxon test is used to demonstrate the performance of CLMFOLDS, which is a non-parametric statistical test. The results of the statistical analysis are shown in TABLE II. The confidence interval is set to 0.95 and 0.9 in the experiment. R^+ indicates that the sum of CLMFOLDS rankings is superior to other algorithms, and R^- is below. '+' indicates that CLMFOLDS is superior to the comparison algorithm, and '-' is poor. The 'yes' means the $p - value$ is less than 0.01 and 0.05. Friedman test, which is another non-parametric test, uses rank to check whether there is a significant difference among the six algorithms. The results are shown in Fig. 7. The mean rank of the CLMFOLDS is better than the other five contrast algorithms from the results of statistical analysis.

The purpose of policy validity analysis is to prove the effectiveness of each policy in the algorithm. The results are shown in Fig. 8. The CL strategy plays the most vital role in the CLMFOLDS algorithm. The CL strategy is far superior to the other three mechanisms in unimodal functions. LDS performs well in simple multimodal functions and hybrid functions.

TABLE I. THE RESULTS OF MEAN AND STANDARD

Fun	MFO		OMFO		LGCMFO		DMMFO		WEMFO		CLMFOLDS	
	Mean	***Std***	***Mean***	***Std***	***Mean***	***Std***	***Mean***	***Std***	***Mean***	***Std***	***Mean***	***Std***
1	2.78E+07	1.59E+08	4.54E+05	1.49E+05	3.39E+03	3.37E+03	7.22E+04	3.64E+04	8.07E+04	5.11E+05	**1.67E-15**	**4.62E-15**
3	4.24E+03	5.78E+03	3.42E+00	1.39E+00	1.01E+02	8.03E+01	3.39E+02	6.59E+02	1.65E+01	6.12E+01	**5.57E-15**	**1.71E-14**
4	7.28E+00	1.06E+01	6.10E+00	6.63E-01	6.34E+00	1.41E+00	5.69E+00	2.79E+00	6.33E+00	1.15E+00	**1.20E-10**	**4.91E-10**
5	2.52E+01	1.11E+01	1.24E+01	4.37E+00	1.83E+01	8.17E+00	1.78E+01	6.99E+00	2.12E+01	9.42E+00	**5.11E+00**	**3.63E+00**
6	5.06E-01	1.20E+00	5.36E-01	7.95E-02	2.94E-02	8.13E-02	1.63E+00	3.40E+00	7.04E-02	3.11E-01	**4.46E-14**	**5.61E-14**
7	3.20E+01	8.89E+00	2.11E+01	4.39E+00	3.24E+01	9.20E+00	3.54E+01	1.19E+01	3.45E+01	1.02E+01	**1.44E+01**	**2.53E+00**
8	2.65E+01	1.32E+01	1.30E+01	5.27E+00	2.00E+01	5.61E+00	1.60E+01	6.18E+00	2.07E+01	9.05E+00	**4.57E+00**	**2.37E+00**
9	1.82E+01	4.45E+01	2.35E-01	5.72E-02	2.62E-01	1.13E+00	2.16E+01	3.27E+01	5.20E+00	3.21E+01	**0.00E+00**	**0.00E+00**
10	8.73E+02	2.73E+02	3.98E+02	2.03E+02	7.47E+02	2.56E+02	4.18E+02	1.81E+02	7.31E+02	2.86E+02	**3.21E+02**	**1.83E+02**
11	3.37E+01	6.19E+01	7.86E+00	4.21E+00	1.19E+01	7.22E+00	1.92E+01	3.17E+01	1.68E+01	1.41E+01	**1.75E+00**	**1.34E+00**
12	1.22E+06	2.73E+06	1.08E+05	1.11E+05	2.97E+04	4.38E+04	1.75E+05	2.58E+05	5.85E+05	1.66E+06	**3.16E+02**	**1.19E+03**
13	1.08E+04	1.25E+04	4.78E+03	6.92E+03	8.75E+03	9.75E+03	4.53E+03	6.61E+03	9.68E+03	1.04E+04	**5.95E+00**	**2.43E+00**
14	6.69E+02	7.05E+02	6.56E+01	1.50E+01	2.51E+02	3.99E+02	2.56E+03	2.82E+03	1.24E+02	9.92E+01	**3.77E+00**	**6.27E+00**
15	3.16E+03	3.62E+03	2.81E+02	3.73E+02	8.20E+02	1.29E+03	1.24E+03	1.33E+03	4.84E+02	6.31E+02	**9.56E-01**	**1.26E+00**
16	1.06E+02	7.41E+01	2.42E+01	3.98E+01	4.76E+01	5.99E+01	1.12E+02	8.45E+01	5.89E+01	5.16E+01	**1.55E+01**	**2.63E+01**
17	4.88E+01	1.96E+01	3.25E+01	1.93E+01	3.12E+01	1.32E+01	2.33E+01	1.50E+01	3.81E+01	1.45E+01	**3.28E+00**	**5.97E+00**
18	2.22E+04	1.30E+04	1.04E+04	8.85E+03	2.00E+04	1.62E+04	2.19E+04	1.40E+04	2.26E+04	1.52E+04	**7.36E+00**	**9.41E+00**
19	8.02E+03	1.07E+04	5.58E+02	1.58E+03	3.71E+03	5.33E+03	8.11E+03	7.83E+03	2.66E+03	4.21E+03	**3.28E-01**	**5.70E-01**
20	3.90E+01	2.95E+01	2.25E+01	1.07E+01	3.52E+01	2.66E+01	2.09E+01	2.36E+01	4.02E+01	3.47E+01	**2.90E+00**	**6.90E+00**
21	1.75E+02	6.32E+01	1.58E+02	5.82E+01	1.04E+02	2.26E+01	1.40E+02	5.35E+01	1.48E+02	5.74E+01	1.60E+02	5.46E+01
22	9.45E+01	2.91E+01	9.42E+01	3.27E+01	9.85E+01	1.63E+01	8.83E+01	3.41E+01	9.61E+01	1.94E+01	9.58E+01	2.00E+01
23	3.26E+02	9.88E+00	3.13E+02	4.92E+00	3.18E+02	7.68E+00	3.26E+02	8.61E+00	3.22E+02	8.16E+00	**3.10E+02**	**4.12E+00**
24	3.53E+02	4.07E+01	3.05E+02	8.86E+01	3.22E+02	8.22E+01	2.57E+02	1.30E+02	3.25E+02	8.72E+01	3.25E+02	5.70E+01
25	4.31E+02	2.49E+01	4.24E+02	2.35E+01	4.21E+02	2.46E+01	4.08E+02	1.85E+01	4.27E+02	2.54E+01	4.16E+02	2.23E+01
26	3.76E+02	3.30E+01	3.11E+02	2.07E+01	3.27E+02	6.01E+01	3.19E+02	1.32E+02	3.68E+02	3.89E+01	**3.09E+02**	2.10E+01
27	3.93E+02	2.10E+00	3.90E+02	7.77E-01	3.91E+02	2.07E+00	3.95E+02	6.89E+00	3.92E+02	1.89E+00	**3.90E+02**	1.04E+00
28	4.87E+02	1.03E+02	3.78E+02	8.98E+01	4.17E+02	1.11E+02	4.27E+02	9.38E+01	4.39E+02	9.60E+01	**3.35E+02**	**7.37E+01**
29	2.97E+02	4.89E+01	2.55E+02	2.17E+01	2.70E+02	2.96E+01	2.73E+02	3.04E+01	2.93E+02	3.47E+01	**2.41E+02**	**8.91E+00**
30	4.53E+05	4.37E+05	1.00E+05	2.92E+05	7.19E+05	3.05E+05	2.60E+05	4.07E+05	2.49E+05	2.85E+05	**2.93E+04**	**1.45E+05**
Time(s)	7742		6743		3632		9083		6024		5251	

TABLE II. p-value of Wilcoxon rank-sum test

CLMFOLDS vs	R^+	R^-	Z	$p-value$	$\alpha=0.05$	$\alpha=0.1$
MFO	433.00	2.00	-4.660^b	.000	*yes*	*yes*
OMFO	406.00	29.00	-4.660^b	.000	*yes*	*yes*
LGCMFO	412.00	23.00	-4.660^b	.000	*yes*	*yes*
DMMFO	397.00	38.00	-4.660^b	.000	*yes*	*yes*
WEMFO	427.00	8.00	-4.660^b	.000	*yes*	*yes*

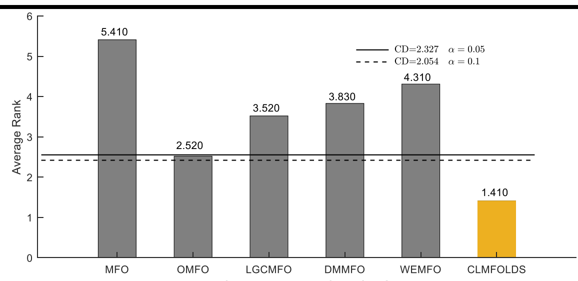

Fig. 7 The results of the Friedman test

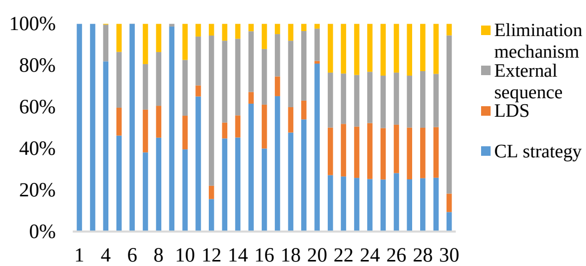

Fig. 8 The results of policy validity analysis

V. CONCLUSION

CLMFOLDS algorithm, which is employed to solve the complex continuous optimization problem, is proposed in this paper. LDS is employed in the initialization phase to generate the uniformly distributed random population in the search space to enhance the global search capability of the algorithm. The information of the whole population is used to update the position of the moth by the comprehensive learning strategy to improve the diversity of the flame and the global search capability of the algorithm. The external storage is introduced into the original MFO algorithm to store neglected suboptimal solutions to improve the diversity of the flame population. The elimination mechanism is employed to delete the unsuitable solution to improve the local search ability of the algorithm. The proposed CLMFOLDS is assessed on the CEC 2017 benchmark. The experimental results show that the CLMFOLDS is superior to the other 5 algorithms. The CLMFOLDS algorithm has not been extended to constrained and multi-objective problems. In the future, CLMFOLDS will get more application and research in practical engineering.

ACKNOWLEDGMENT

This work was financially supported by the National Natural Science Foundation of China under grant 62063021. It was also supported by the Key talent project of Gansu Province (ZZ2021G50700016), the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (21YF5WA086), Lanzhou Science Bureau project (2018-rc-98), and Project of Gansu Natural Science Foundation (21JR7RA204)， respectively.

References

[1] X. Cai et al., "An improved quantum-inspired cooperative co-evolution algorithm with muli-strategy and its application," Expert Syst. Appl., vol. 171, no. November 2019, p. 114629, 2021, doi: 10.1016/j.eswa.2021.114629.

[2] M. Gomes Pereira de Lacerda, L. F. de Araujo Pessoa, F. Buarque de Lima Neto, T. B. Ludermir, and H. Kuchen, "A systematic literature review on general parameter control for evolutionary and swarm-based algorithms," Swarm Evol. Comput., vol. 60, no. January 2020, p. 100777, 2021, doi: 10.1016/j.swevo.2020.100777.

[3] E. T. Bacalhau, L. Casacio, and A. T. de Azevedo, "New hybrid genetic algorithms to solve dynamic berth allocation problem," Expert Syst. Appl., vol. 167, p. 114198, 2021, doi: https://doi.org/10.1016/j.eswa.2020.114198.

[4] S. Molaei, H. Moazen, S. Najjar-Ghabel, and L. Farzinvash, "Particle swarm optimization with an enhanced learning strategy and crossover operator," Knowledge-Based Syst., vol. 215, p. 106768, 2021, doi: 10.1016/j.knosys.2021.106768.

[5] J. Lu, J. Zhang, and J. Sheng, "Enhanced multi-swarm cooperative particle swarm optimizer," Swarm Evol. Comput., no. July 2020, p. 100989, 2021, doi: 10.1016/j.swevo.2021.100989.

[6] L. Cai, Y. Hou, Y. Zhao, and J. Wang, "Application research and improvement of particle swarm optimization algorithm," Proc. 2020 IEEE Int. Conf. Power, Intell. Comput. Syst. ICPICS 2020, pp. 238–241, 2020, doi: 10.1109/ICPICS50287.2020.9202023.

[7] H. Zhao, C. Zhang, X. Zheng, C. Zhang, and B. Zhang, "A decomposition-based many-objective ant colony optimization algorithm with adaptive solution construction and selection approaches," Swarm Evol. Comput., no. March 2020, p. 100977, 2021, doi: 10.1016/j.swevo.2021.100977.

[8] T. Cura, "A rapidly converging artificial bee colony algorithm for portfolio optimization," Knowledge-Based Syst., vol. 233, p. 107505, 2021, doi: https://doi.org/10.1016/j.knosys.2021.107505.

[9] J. Ge, X. Zhang, G. Liu, and Y. Sun, "A Novel Feature Selection Algorithm Based on Artificial Bee Colony Algorithm and Genetic Algorithm," 2019 IEEE Int. Conf. Power, Intell. Comput. Syst. ICPICS 2019, pp. 131–135, 2019, doi: 10.1109/ICPICS47731.2019.8942410.

[10] X. Yu, W. Y. Xu, and C. L. Li, "Opposition-based learning grey wolf optimizer for global optimization," Knowledge-Based Syst., vol. 226, p. 107139, 2021, doi: 10.1016/j.knosys.2021.107139.

[11] S. Chakraborty, A. K. Saha, R. Chakraborty, and M. Saha, "An enhanced whale optimization algorithm for large scale optimization problems," Knowledge-Based Syst., vol. 233, p. 107543, 2021, doi: https://doi.org/10.1016/j.knosys.2021.107543.

[12] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," Knowledge-Based Syst., vol. 89, pp. 228–249, 2015, doi: 10.1016/j.knosys.2015.07.006.

[13] S. Mohseni, A. C. Brent, and D. Burmester, "A demand response-centred approach to the long-term equipment capacity planning of grid-independent micro-grids optimized by the moth-flame optimization algorithm," Energy Convers. Manag., vol. 200, p. 112105, 2019, doi: https://doi.org/10.1016/j.enconman.2019.112105.

[14] K. Zheng, D. Yang, B. Zhang, J. Xiong, J. Luo, and Y. Dong, "A group sparse representation method in frequency domain with adaptive parameters optimization of detecting incipient rolling bearing fault," J. Sound Vib., vol. 462, p. 114931, 2019, doi: https://doi.org/10.1016/j.jsv.2019.114931.

[15] S. Mahata, S. K. Saha, R. Kar, and D. Mandal, "A metaheuristic optimization approach to discretize the fractional order Laplacian operator without employing a discretization operator," Swarm Evol. Comput., vol. 44, pp. 534–545, 2019, doi: https://doi.org/10.1016/j.swevo.2018.06.007.

[16] X. Lei, M. Fang, and H. Fujita, "Moth–flame optimization-based algorithm with synthetic dynamic PPI networks for discovering protein complexes," Knowledge-Based Syst., vol. 172, pp. 76–85, 2019, doi: https://doi.org/10.1016/j.knosys.2019.02.011.

[17] H. Zhang et al., "Advanced orthogonal moth flame optimization with Broyden–Fletcher–Goldfarb–Shanno algorithm: Framework and real-world problems," Expert Syst. Appl., vol. 159, 2020, doi: 10.1016/j.eswa.2020.113617.

[18] Y. Xu et al., "An efficient chaotic mutative moth-flame-inspired optimizer for global optimization tasks," Expert Syst. Appl., vol. 129, pp. 135–155, 2019, doi: 10.1016/j.eswa.2019.03.043.

[19] Y. Xu, H. Chen, J. Luo, Q. Zhang, S. Jiao, and X. Zhang, "Enhanced Moth-flame optimizer with mutation strategy for global optimization," Inf. Sci. (Ny)., vol. 492, pp. 181–203, 2019, doi: 10.1016/j.ins.2019.04.022.

[20] W. Shan, Z. Qiao, A. A. Heidari, H. Chen, H. Turabieh, and Y. Teng, "Double adaptive weights for stabilization of moth flame optimizer: Balance analysis, engineering cases, and medical diagnosis," Knowledge-Based Syst., vol. 214, 2021, doi: 10.1016/j.knosys.2020.106728.

[21] L. Ma, C. Wang, N. gang Xie, M. Shi, Y. Ye, and L. Wang, "Moth-flame optimization algorithm based on diversity and mutation strategy," Appl. Intell., vol. 51, no. 8, pp. 5836–5872, 2021, doi: 10.1007/s10489-020-02081-9.

[22] S. Sapre and S. Mini, "Opposition-based moth flame optimization with Cauchy mutation and evolutionary boundary constraint handling for global optimization," Soft Comput., vol. 23, no. 15, pp. 6023–6041, 2018, doi: 10.1007/s00500-018-3586-y.