OPEN ACCESS INTERNATIONAL JOURNAL OF SCIENCE & ENGINEERING

MoD-MFO: A Modified Moth Flame Optimization Algorithm for Function Optimization

Chitaranjan Dash[1], Kanakprava Biswal[2]

Department of Mathematics, Government College of Engineering, Kalahandi, Bhawanipatna, Odisha-766002, India

crdash1972@gmail.com[1], kanakpravabiswalrs@gmail.com[2]

Abstract: The Moth Flame Optimization (MFO) algorithm is a search algorithm based on a mechanism called transverse orientation. In this mechanism, the moths tend to maintain a fixed angle with respect to the moon. The main disadvantages of MFO algorithm are slow convergence rate and degeneration of the global search. To overcome these drawbacks, a modified moth flame optimization algorithm is introduced. The proposed method is called MoD-MFO . In the propose MoD-MFO , the mutualism phase from symbiosis Organisms Search (SOS) algorithm has been embedded into basic MFO algorithm. The proposed method is tested with twenty-four classical benchmark functions taken from literature for performance evaluation and the obtained results are compared with some of the state-of-the-art meta-heuristic algorithms with basic MFO algorithm. The results clearly showed that MoD-MFO outperformed MFO and the other meta-heuristics algorithms. Also, The efficiency of the proposed MoD-MFO has been measured by Friedmann rank test and found that the rank of MoD-MFO is least. Finally, the method endorsed in this paper has been applied to one real life problem and it was inferred that the output of the proposed algorithm is satisfactory.

Keywords: Optimization Algorithm. Moth flame optimization. Mutualism phase. Benchmark functions. Friedman Rank Test

---

I. INTRODUCTION

Optimization problems play an important role in both industrial application fields and the scientific research world. Many computational methods have been proposed to solve optimization problems in last few decades. In earlier, for solving optimization problems, a lot of mathematical computations are needed. However, by numerical methods, it is very complicated to solve the non-convex, nonlinear, highly constrained and high variables problems. To over come the drawbacks, such as additional mathematical calculations, initial guess, convergent issues in discrete optimization problems, a bunch of optimization algorithms called as meta-heuristics algorithms have been proposed in recent decades, such as Genetic Algorithm(GA) **[1]**, Differential Evolution (DE) **[2]**, Particle Swarm Optimization (PSO) **[3]**, Ant Colony Optimization (ACO) **[4]**, Butterfly Optimization Algorithm (BOA) **[5]**, Symbiosis organisms search (SOS) **[6]**, Artificial Bee Colony Optimization (ABC) **[7],** Moth flame optimization (MFO) **[8],** etc.

Nowadays, meta-heuristic algorithms are widely used in solving engineering problems for their simple implementation, simple mathematical operators, and less possibility of giving the local optimum solutions. Usually, these methods start with an initial set of solution and then run the process until the global optimal solutions of the objective function are obtained. Broadly we divide meta-heuristic algorithms into two categories i.e single solution-based methods and population-based methods. The single solution-based algorithms perform the search by single search agents, while a set of search agents are used in population-based methods. In the population-based methods, each solution updates its position depends on individual and social information. Moreover, various solutions could easily search the whole search space; hence, better final results can be obtained as compared to single solution-based methods.

There are various meta-heuristic algorithms introduced in the literature but according to the no free lunch (NFL) **[9]** theorem, no optimization method can solve all the optimization problems efficiently. In this paper, MFO is considered to be studied and analyzed deeply. MFO algorithm was first discovered in 2015 by Mirjalili **[8]**. The inspiration of MFO came from the navigation technique of moths in nature referred as transverse orientation. The author of the MFO algorithm, showed that MFO acquired very competitive results compared with other nature inspireed meta-heuristic optimization algorithms. Also, MFO provides competitive results in the field automatic mitosis detection in breast cancer histology images. The demerits of MFO are (a) suffered from entrapping at local optima and (b) low convergence rate. Thus, a lot of methods have been proposed to enhance the performance of MFO. Li et al. **[10]** have introduced Lévy-flight moth-flame optimization (LMFO) algorithm to improve the performance of MFO. Emery et al. **[11]** have introduced chaos parameter in the spiral equation of updating the position of moths. Motivated by the above works, in this paper, a novel improved MFO (namely, MoD-MFO) has been introduced in order to improve the performance of original MFO further by adding the mutualism phase of SOS with the basic MFO algorithm. The performance of our proposed algorithm has been exmined on a set of 24 (twenty-four) Benchmark test functions from literature and also, the obtained results have been compared with some state-of-the-art optimization algorithms and found that MoD-MFO performs better than the other Metaheuristic optimization algorithms.

The rest of this paper is organized as follows: in Sect. 2, we will review the moth flame optimization (MFO) algorithm. In sect. 3, mutualism phase is proposed. The proposed algorithm MoD-MFO is shown in sect. 4. The simulation results and Performance are present in sect. 5. Results and discussion are present in Sect. 6. The application of real world problem is shown in sect. 7. Finally conclusions are discussed in sect. 8.

Moth Flame Optimization:

Moths are basically belongs to the insects and they are very similar to the family of butterflies. The navigation technique of moths are very unique in nature which attaracts researcher to think on it. Moths fly in night using moon light and for navigation they utilized transverse orientation mechanism. Moths fly by keeping a fixed angle with respect to the moon for long journey in a straight path . The effectiveness of the transverse orientation mechanism depends on the distance of the flame (light source) i.e. when the flame is close to the moth, the moth starts mooving in a spiral path around the light. This spiral fly path eventuallyconverges the moth to the flame. Using this behaviour of moth and mathematical modelling, the MFO algorithm is developed by Mirjalili in 2015.

1.1 MFO Algorithm:

In this algorithm, the moths are expressed as the candidate solutions and their position is expressed as a vector of decision variables. Let us consider the following moths matrix

$$X=\begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n-1} & x_{1,n} \\ x_{2,1} & \ddots & \cdots & \cdots & x_{2,n} \\ \vdots & \cdots & \ddots & \cdots & \vdots \\ x_{N-1,1} & \cdots & \cdots & \ddots & x_{N-1,n} \\ x_{N,1} & x_{N,2} & \cdots & x_{N-1} & x_{N,n} \end{bmatrix} \quad (1.1)$$

Where N and n are the number of moths in initial population and the number of decision variables respectivily. The second key point of the MFO algorithm is the flame matrix. Here the size of the both moth's matrix (X) and flame matrix (FM) are same as each moth flies around its corresponding flame.

$$FM=\begin{bmatrix} Fm_{1,1} & Fm_{1,2} & \cdots & Fm_{1,n-1} & Fm_{1,n} \\ Fm_{2,1} & \ddots & \cdots & \cdots & Fm_{2,n} \\ \vdots & \cdots & \ddots & \cdots & \vdots \\ Fm_{N-1,1} & \cdots & \cdots & \ddots & Fm_{N-1,n} \\ Fm_{N,1} & Fm_{N,2} & \cdots & Fm_{N-1} & Fm_{N,n} \end{bmatrix} \quad (1.2)$$

In MFO algorithm, there are two important term moth and flame where, the moth flies around its corresponding flame to find better solutions and the flame is the best solution obtained by the moth. Since, the flying path of the moths is spiral around their corresponding flame, therefore, the author defined a logarithmic spiral function to set a spiral fly path for the moth

$$x_i^{K+1}=\left|x_i^K-Fm_i\right|\cdot e^{bt}\cdot\cos(2\pi t)+Fm_i \quad (1.3)$$

Where, t is a random uniform number between -1 and 1 which defines the closeness of the next position of the moth to its corresponding flame. An adaptive procedure has been proposed to decrease the values of the parameter t over the iterations, which enhance the effectiveness of both exploration and exploitation in first and last iterations respectivily.

$$a=-1+current_{iter}\left(\frac{-1}{max_{iter}}\right) \quad (1.4)$$

$$t=(a-1)\times r+1 \quad (1.5)$$

where max_{iter} is the maximum number of iterations, a is the convergence constant which decreases linearly from -1 to -2 over the course of iterations which proves that both exploration and exploitation happens in the MFO algorithm. To obtain the final solution, the number of flames is reduced over the iteration can be obtained by the following formula

$$\text{N.FM}=round\left(N.FM_{Lst\,it}-crnt.it\frac{(N.FM_{Lst\,it}-1)}{\max it}\right) \quad (1.6)$$

Where N.FM is the number of flame and $N.FM_{Lst\,it}$ is the number of flame in last iteration

SYMBIOSIS ORGANISMS SEARCH ALGORITHM

The symbiosis organisms search (SOS) algorithm is a natureinspired swarm-based metaheuristic approach which was first proposed by Cheng and Prayogo **[12]**. This algorithm replicates the symbiotic interaction schemes between different species to survive and grow in an environment, where each individual of different species is considered as a candidate solution in a search space. The effectiveness and robustness of SOS are validated for both benchmark and real life problems **[13]**. The SOS algorithm randomly initializes N organisms to generate the first ecosystem. Newsolutions are then updated in turn by using the mutualism, commensalism and parasitism phases. In the proposed method, we have used only the mutualism phase of SOS which is discussed below.

1.2 Mutualism phase:

A symbiotic relationship between two distinct species that produces individual benefits from the synergy is called mutualism. Let X_i and X_J represent the ith and jth organisms of the ecosystem, respectively, where X_J is a randomly chosen organism in the ecosystem. X_i interacts with X_J to create new candidate solutions by the following two Eqs. (1.7) and (1.8):

$$X_{inew}=X_i+r[0,1]\times\left(X_b-\left(\frac{X_i+X_j}{2}\right)\times BF_1\right)$$
(1.7)

$$X_{jnew}=X_j+r[0,1]\times\left(X_b-\left(\frac{X_i+X_j}{2}\right)\times BF_2\right)$$
(1.8)

where $r[0,1]$ is a uniformly distributed random number in the range $[0,1]$; X_b is the best organism in the ecosystem; BF_1 and BF_2 are the benefit factors randomly generated as either 1 or 2. These factors indicate the level of benefit to each organism, and Mutual Vector i.e. $\left(\frac{X_i+X_j}{2}\right)$ expresses the relationship characteristic between two organisms $X_i$and X_j . Subsequently, X_{inew} and X_{jnew} are compared with X_i and $X_j$to choose the fittest organism in each pair, respectively. In this phase, new organisms are generated based on the best organism X_b .

II.PROPOSED SYSTEM

In MFO, exploration and exploitation are obtained from the spiral movement of moths around the flame. The power of the exponent factor 't' gives a better clarification about exploration and exploitation. We know that the next position of moth is obtained from the spiral equation (3) The parameter t in the spiral equation defines how much the next position of the moth should be close to the flame (t = −1 is the closest position to the flame, while t = 1 shows the farthest). Exploration occurs when the next position is outside the space between the moth and flame and Exploitation happens when the next position lies inside the space between the moth and flame.

In order to increase the diversity of population against premature convergence and accelerate the convergence speed, this paper proposes an enhanced moth flame optimization (MoD-MFO). In other words, this approach is beneficial to obtain a better trade-off between the exploration and exploitation ability of MFO. We start the algorithm in the similar manner like MFO and then we apply mutualism phase [eqn (1.9) and eqn (1.10)] for position updating i.e. in mutualism phase, we take two organisms (here organism means moths) from the population for updating the position of each moth in each iteration and share information with another randomly chosen moth to update their respective positions in the search space. The formulation of mutual phase is present in below

$$x_{i\,new}^{K+1}= x_i^K+r[0,1]\times\left(x_b^K-\left(\frac{x_i^K+x_j^K}{2}\right)\times BF_1\right)$$
(1.9)

$$x_{j\,new}^{K+1}= x_j^K+r[0,1]\times\left(x_b^K-\left(\frac{x_i^K+x_j^K}{2}\right)\times BF_2\right)$$
(1.10)

Where where x_j^K is randomly selected another population and $x_{i\,new}^{K+1}$, $x_{j\,new}^{K+1}$ are the updated new populations and BF_1 and BF_2 are benefit factor of x_i^K & x_j^K , value of which is randomly taken as 1 or 2.

NO		MoD-MFO	MFO	SOS	ABC	PSO	GA	DE	BOA
F-1	MEAN	0.00E+00	0.00E+000	6.03E−173	2.5823E−01	1.76657E −51	2.2039E −02	2.6178E−165	0.0000E+00
	STD	0.00E+00	0.00E+000	0.00E+00	5.1309E−01	1.1246E −50	9.6789E −03	0.0000E +00	0.0000E+00
F-2	MEAN	2.78E-06	1.56E-001	3.55E−20	1.9286E −04	0.0000E +00	0.0000E +00	0.0000E +00	0.0000E+00
	STD	2.44E-06	3.05E-001	9.29E−20	3.1824E −04	0.0000E +00	0.0000E +00	0.0000E +00	0.0000E+00

Because of these feature, the proposed algorithm has potential to provide superior performance compared to MFO. In following section, a set of 24 (twenty four) benchmark functions are hired to verify the effectiveness of the proposed algorithm. The main steps of MoD-MFO can be simply presented in Algorithm 1.

Algorithm 1 MoD-MFO algorithm

Objective function f(X), X = ($X1$ $X2$………Xdim);
for i= 1:n (n=number of moths)
for j= 1:dim (dim=number of the decision variables)
Generate solutions of n organisms Xi(i =1,2,….n) using equation X (i,j)=LB(i)+(UB (i)-LB(i))*rand
end for end for
While Current iteration<Maximum number of iterations
if Iteration==1
Enter the number of flames equal to the number of moths in initial population
else
Linearly decrease the number of flames using eqn 1.6
end
FM=Fitness Function f(X);
if Iteration==1
Sort the moths based on FM Update the Flames Iteration=0;
else
Sort the moths based on FM from last iteration Update the Flames
end
Decrease the convergence constant
for j = 1 : n
for k= 1 : dim
Calculate parameters r and t using eqn 1.4 & 1.5
Update the position of moths with respect to their corresponding flame
end end
Randomly select one solution (i ≠ j);
Update the solution Xi & Xj according to Eqn. (2.4) and (2.5);
Calculate fitness value of the new solutions;
End while
Output: The best solution with the minimum fitness function value in the ecosystem;
Current iteration=Current iteration+1;

F-3	MEAN	0.00E+00	0.00E+000	0.00E+00	8.4339E −03	3.10418E −53	2.1954E −04	1.4054E−172	0.0000E+00
	STD	0.00E+00	0.00E+000	0.00E+00	2.3047E −02	2.10651E −52	1.0961E −04	0.0000E +00	0.0000E+00
F-4	MEAN	0.00E+00	0.00E+000	0.00E+00	2.2314E −01	1.31167E −31	2.3026E −02	1.3000E −01	0.0000E+00
	STD	0.00E+00	0.00E+000	0.00E+00	4.8282E −01	6.653E −31	1.0512E −02	4.8524E −01	0.0000E+00
F-5	MEAN	0.00E+00	0.00E+000	0.00E+00	0.0000E +00	3.0393E −02	0.0000E +00	0.0000E +00	0.0000E+00
	STD	0.00E+00	0.00E+000	0.00E+00	0.0000E +00	2.6192E −17	0.0000E +00	0.0000E +00	0.0000E+00
F-6	MEAN	8.88E-16	8.88E-016	1.00E−15	3.6522E +00	1.8186E +01	4.6683E −02	1.7183E +00	1.7183E+00
	STD	0.00E+00	0.00E+000	6.38E−16	6.9377E −01	5.4812E +00	1.6847E −02	0.0000E +00	0.0000E+00
F-7	MEAN	0.00E+00	0.00E+000	0.00E+00	5.4819E −01	4.0503E +01	5.5598E −02	9.22E −17	1.8472E−19
	STD	0.00E+00	0.00E+000	0.00E+00	2.7212E −01	4.1334E +01	2.9490E −02	2.0465E −17	2.6886E−20
F-8	MEAN	4.30E-07	4.60E-002	6.73E−23	3.0839E −02	4.9297E +00	1.3732E −04	5.9824E −02	4.4108E−01
	STD	4.12E-07	9.35E-002	1.95E−22	4.8229E −02	5.9951E +00	7.6553E −05	2.0758E −01	5.7467E−02
F-9	MEAN	-1.13E-10	-1.13E-010	0.00E+00	−9.5606E+00	−7.75E +00	− 9.660E +00	−9.6155E+00	−5.3382E+00
	STD	2.13E-17	5.15E-017	0.00E+00	6.8027E −02	0.7118249	5.8796E −09	4.5500E −02	− 5.6092+00
F-10	MEAN	0.00E+00	0.00E+000	0.00E+00	2.3624E +01	1.4453E +02	1.6458E	1.2475E +01	0.0000E +00
	STD	0.00E+00	0.00E+000	0.00E+00	5.4860E +00	3.7957E +01	+014.1090E +00	3.3093E +00	0.0000E +00
F-11	MEAN	2.37E-09	0.00E+000	4.85E−16	6.0589E +00	4.4402E −02	3.1474E −17	6.90E −16	0.0000E +00
	STD	7.29E-09	0.00E+000	2.74E−16	6.1601E −01	4.5318E −01	3.0784E −16	2.57579E−15	0.0000E +00
F12	MEAN	0.00E+00	0.00E+000	1.82E−01	8.6222E−16	0.0000E +00	1.68965E−17	0.0000E +00	0.0000E +00
	STD	0.00E+00	0.00E+000	1.20E−01	8.54387E−16	0.0000E +00	1.68964E−16	0.0000E +00	0.0000E +00
F-13	MEAN	2.69E+01	2.89E+001	−1.00E+00	1.2517E +02	2.5746E +02	3.7872E +01	2.3552E +01	2.8837E +01
	STD	1.28E+00	6.44E-002	0.00E+00	8.7737E +01	3.7311E +02	2.5254E +01	4.9866E +00	3.1281E −02
F-14	MEAN	0.00E+00	0.00E+000	2.27E−74	2.3684E−02	1.2030E+03	2.4244E−03	1.1423E−175	0.0000E +00
	STD	0.00E+00	0.00E+000	2.41E−74	4.1868E−02	5.4485E+02	1.0716E−03	0.0000E+00	0.0000E +00
F-15	MEAN	2.03E-197	0.00E+000	1.24E−86	1.0060E−01	1.0586E−01	4.6662E−02	1.0201E+00	6.996E−153

F-16	MEAN	0.00E+00	0.00E+000	−4.19E+02	1.4281E-01	7.2800E+01	5.4367E-02	8.15E-91	0.0000E+00
	STD	0.00E+00	0.00E+000	0.00E+00	6.8388E-02	2.2745E+01	1.6187E-02	1.2303E-90	0.0000E+00
F-17	MEAN	1.59E-05	1.04E+001.	3.00E+00	1.3240E-11	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	STD	2.23E-05	78E+00	1.94E-15	2.4241E-11	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
F-18	MEAN	3.00E+00	7.75E+000	2.62E-43	3.0000E+00	3.0000E+00	3.0000E+00	3.0000E+00	3.0000E+00
	STD	8.12E-08	6.57E+000	5.87E-43	6.4688E-12	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
F-19	MEAN	0.00E+00	0.00E+000	7.51E-13	1.1923E-04	0.0000E+00	8.3889E-104	0.0000E+00	0.0000E+00
	STD	0.00E+00	0.00E+000	3.99E-12	1.3039E-04	0.0000E+00	2.5167E-103	0.0000E+00	0.0000E+00
F-20	MEAN	1.31E-12	0.00E+000	4.62E-03	1.5150E+01	3.3671E+03	3.2713E-02	1.3099E+02	0.0000E+00
	STD	4.94E-12	0.00E+000	6.45E-03	1.1075E+01	9.5201E+02	1.1145E-02	1.8525E+02	0.0000E+00
F-21	MEAN	1.32E-02	1.53E-002	1.09E−117	7.1334E-02	1.9839E-04	1.2997E-04	6.2736E-04	2.8400E-02
	STD	8.93E-03	1.23E-002	3.94E−117	5.1357E-02	1.6590E-04	1.1326E-04	1.3123E-03	1.3179E-02
F-22	MEAN	0.00E+00	0.00E+000	−4.13E+03	−1.015E+01	−1.015E+01	− 8.900E+00	−1.015E+01	−1.015E+01
	STD	0.00E+00	0.00E+000	2.59E+02	0.0000E+00	0.0000E+00	2.7008E+00	0.0000E+00	0.0000E+00
F-23	MEAN	-3.79E-03	-3.18E-003	1.26E-07	3.6452E+04	7.6727E+05	−1.0350E+03	-4.8988E+03	−2.7507E+07
	STD	1.47E-12	8.43E-004	3.29E-07	1.9481E+04	4.0197E+05	1.1916E+03	2.7292E+01	5.6521E+07
F-24	MEAN	7.05E-06	8.63E-002	9.98E-01	−3.7912E-03	−3.7912E-03	−3.7912E-03	−3.7912E-03	−3.7912E-03
	STD	647E-06	1.02E-001	2.92E-16	5.7115E-18	0.0000E+00	4.5714E-19	4.5714E-19	0.0000E+00

Table 1 Mean and standard deviation results for comparing MoD-MFO with MFO, SOS, ABC, PSO, GA, DE, BOA]

Experimental Set-up:

The proposed algorithm has implemented in MATLAB R2015a, we use a maximum of 10,000 iterations for the stopping criterion of MoD-MFO . There are different way to stop the algorithm such as maximum CPU time used, maximum iteration number reached, the maximum number of iterations with no improvement, a particular value of error rate is reached or any other appropriate criteria. To reduce statistical errors and generate statistically significant results, each function is repeated for 30 runs. The mean, standard deviation of MoD-MFO and other algorithms used for comparison are recorded. In order to meet the requirement, we use one fixed combination of parameters for MoD-MFO in the simulation of all benchmark functions. The population size N is 50, the power exponent constant b is equal to 1 and t varies from -1to 1.

III.RESULT AND DISCUSSION:

The obtained simulation result of our proposed MoD-MFO has been compare with other 7 (seven) meta-heuristics MFO, GA, DE, PSO, BOA ,ABC and SOS which are the best performing and produce satisfactory performance when applied to global optimization problems.

These algorithms are widely employed to compare the performance of optimization algorithms. For this comparative analysis, the population size is fixed at 50, and the maximum number of iterations is fixed to be 10,000. Each algorithm is executed for 30 times, and results are calculated and tabulated.

Table 1 depicts the performance results of the proposed MoD-MFO and five basic state-of-the-art optimization techniques, namely Genetic Algorithm (GA) **[1]**, Differential Evolution (DE) **[2]**, Particle Swarm Optimization (PSO) **[3]**, Ant Colony Optimization (ACO) **[4]**, Butterfly Optimization Algorithm (BOA) **[5]**, Symbiosis organisms search (SOS) **[6]**, Artificial Bee Colony Optimization (ABC) **[7],** Moth flame optimization (MFO) **[8],** for 24 basic benchmark functions.

Table 2 shows the number of occasions where the mean performance of MoD-MFO is better than, similar to and worse than the above algorithms. From this table, it can be observed that MoD-MFO performs better than MFO, ABC, PSO, GA, DE, BOA and SOS in 10, 15, 14, 15, 15, 10 and 17 benchmark functions, respectively, similar results are seen in 10, 0, 2, 2, 1, 7 and 5 occasions, respectively, and worse results are obtained in, respectively, 4, 9, 8, 7, 8, 7, 3 benchmark functions.

Table 2 Performance results of MoD-MFO compared to MFO, ABC, PSO, GA, DE, BOA, SOS BOA, SOS on 24 benchmark functions.

Proposed Algorithm	MFO	ABC	PSO	GA	DE	BOA	SOS
Superior to	10	15	14	15	15	10	17
Similar to	10	0	2	2	1	7	5
Inferior to	4	9	8	7	8	7	3

The Friedman test is a non-parametric statistical test developed by Milton Friedman **[14]**. It is used to detect differences in treatments across multiple test attempts. The procedure involves ranking each row (or block) together, then considering the values of ranks by columns. In this paper we use the Friedman rank test (using IBM-SPSS software) from the mean performances of the algorithms for each benchmark functions. From Table 3, it can be seen that the rank of MoD-MFO is least which asserts that the performance of MoD-MFO is better than those of the compared algorithms

Table 3 Statistical Analysis (Friedman Rank Test)

Algorithm	Mean rank	rank
MUMFO	3.50	1
MFO	4.09	2
DE	4.66	3
GA	4.69	4
PSO	4.67	5
ABC	4.78	6
BOA	4.79	7
SOS	4.83	8

Real world application

The proposed MoD-MFO is employed to solve one real world problem(RWP), which is taken from**[16]**.

RWP: Optimal capacity of gas production facilities

$$Min\ f(x)=61.8+5.72\times x_1\times 0.2623\times\left[(40-x_1)\times \ln\frac{x_2}{200}\right]^{-0.85}+0.087\times$$

s.t $x_1\geq 17.5,\ x_2\geq 200,\ 17.5\leq x_1\leq 40,\ 300\leq x_1\leq 600;\backslash$

Table 4 Comparision performance of MoD-MFO with MFO, DE, Gravitation search algorithm (GSA)**[17]**, DE-GSA.

Item	DE	GSA	DE-GSA	MFO	MoD-MFO
x_1	17.5	17.5	17.5	17.5	17.5
x_2	600	600	600	600	600
$f(x)$	169.844	169.844	169.844	71.4495	**71.4468**

The experimental results of this problem is present in Table 4. In this table, the results of DE, GSA and DE-GSA are taken from **[16]**. It is observed that, The performance of our proposed method is better than other algorithms.

IV.CONCLUSION

This paper presents an enhanced moth flame optimization (MoD-MFO) which uses an additional mutualism step to improve the MFO algorithm. To evaluate the performance of MoD-MFO , various numerical experiments are conducted on a diverse subset of benchmark functions and compared with the basic MFO, ABC, PSO, GA, DE, BOA, SOS BOA. The simulation results indicate that proposed algorithm is able to make use of the global best solution in the optimization process which results in fast convergence. The additional mutualism step also enables the proposed algorithm to avoid the local optima trap and premature convergence.

V.REFERENCES :

1. Holand JH (1992) Genetic algorithms. Sci Am 267:66–72
2. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
3. Kennedy J, EberhartR(1995) Particle swarm optimization. In: Proceedings of ICNN'95—international conference on neural networks, vol 4, pp 1942–1948
4. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Mag 1(4):28–39
5. Arora S, Singh S (2015) Butterfly algorithm with levy flights for global optimization. In: International conference on signal processing, computing and control. IEEE, Solan, pp 220–224
6. Cheng MY, Prayogo D (2014) Symbiotic organisms search: a newmetaheuristic optimization algorithm. Comput Struct 139:98–112
7. Yi Y, He R (2014) Anovel artificial bee colony algorithm. In: 2014 sixth international conference on intelligent human–machine systems and cybernetics, vol 1, pp 271-274.
8. Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowled Based Syst 89:228–249
9. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1:67–82
10. Li Z, Zhou Y, Zhang S, Song J (2016) Lévy-flight moth-flame algorithm for function optimization and engineering design problems. Math Probl Eng 2016:1–22
11. Emary E, Zawbaa H (2016) Impact of chaos functions on modern swarm optimizers. PLoS One 11(7):1–26
12. Cheng MY, PrayogoD (2014) Symbiotic organisms search: a newmetaheuristic optimization algorithm. Comput Struct 139:98–112
13. Panda A, Pani S (2016) A symbiotic organism search algorithm with adaptive penalty function to solve multi-objective constrained optimization problems. Appl Soft Comput 46:344–360
14. Milton F (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association. 32: 675–701.
15. Li X, Yin M, Ma Z (2011) Hybrid differential evolution and gravitation search algorithm for unconstrained optimization. International journal of the physical sciences 6:5961-5981
16. Rashedi E, Nejamabadi-pour H, Saryajdi S (2009) GSA: A gravitation search algorithm. Information sciences 179:2232-2248