Tikrit Journal of Pure Science
ISSN: 1813 – 1662 (Print) --- E-ISSN: 2415 – 1726 (Online)
Journal Homepage: http://tjps.tu.edu.iq/index.php/j

Improving Moth-Flame Optimization Algorithm by using Slime-Mould Algorithm

Sami N. Hussein[1] , Nazar K. Hussein[2]
[1] *College of Computers Sciences and Mathematics, University of Mosul*
[2] *College of Computers Sciences and Mathematics, University of Tikrit*

DOI: http://dx.doi.org/10.25130/tjps.26.2021.011

ARTICLE INFO.
Article history:
-Received: 3 / 9 / 2021
-Accepted: 24 / 10 / 2021
-Available online: / / 2022
Keywords: Moth-Flame Optimization, Slime molds algorithm, optimization, swarm, heuristics.

Corresponding Author:
Name: Sami N. Hussein
E-mail:
Sami.csp107@student.uomosul.edu.iq
Nazar.dikhil@tu.edu.iq

ABSTRACT

The MFO algorithm is one of the modern optimization algorithms based on swarm intelligence, and the SMA algorithm is also one of the latest algorithms in the same field and has the advantages of fast convergence, high convergence accuracy, robust and robust. In this research paper, we introduce an optimized algorithm for MFO based on the SMA algorithm to get better performance using the features in the two algorithms, and two different algorithms are proposed in this field. The two predicted new algorithms were tested with standard test functions and the results were encouraging compared to the standard algorithms.

1- Introduction

In previous studies, there are many algorithms that are inspired by nature and that the best algorithms are based on swarm. Most algorithms are formulated in formulas that are unique and differ with each other. For example, there is the Moth-FLAM algorithm which is one of the modern algorithms that uses swarm intelligence to obtain optimal solutions to real issues, which are closely related to butterflies in nature, they live in different climates, they have pairs of wings. Its life cycle passes through the larval and adult stages. It is transmitted from larvae to moths. The most interesting fact of this moth's navigation techniques at night, it has the ability to travel at night using moonlight [1]. 160,000 species of these insects are found in nature. It has a mechanism called browser orientation. Navigate this way by maintaining a constant angle with the moon [2]. Also, mold is a single cell and visible through the eye without the aid of hardware. These cells display a wide range of intelligent behavior in obtaining the optimal pathway for the chilled food [3]. Due to its unique shape and form, more than one diet can be used at the same time by forming a vein network and connected to each other. If there is enough food in the environment [4]. Each algorithm has its own unique performance and best obtained implementations. Some scientists have proposed algorithms with similar nomenclature, but the method in terms of design and use of this algorithm is very different from the algorithms proposed in this paper.

2- Moth flame optimization (MFO)

2-1 Inspiration

Moths are insects that are closely related to butterflies. 160,000 species of these insects exist in nature, in different climates. It has two pairs of wings, and its life cycle goes through larval and adult stages. It moves from the larval stage to the moth by means of tracks. The special night-navigation techniques of this moth is the most interesting fact as it has the ability to travel at night using moonlight. It possesses a mechanism called transverse orientation. Navigating in this way by maintaining a constant angle with the moon is a successful mechanism for traveling very long distances in a straight path. Figure 1 shows an example to illustrate the mechanism of the transverse alignment. This mechanism ensures flight in a straight line [2]. Although this mechanism is effective, we usually notice some moths flying around lamp lights and litter. In fact, moths are tricked by lights because they display these behaviors. This is due to the inefficiency of the transverse directional mechanism, as it is

advantageous to mobility in a straight course only when the light is too far away. So while you see the mites light that is made by human beings, they are trying to maintain a similar angle of light to fly in a straight line. It will maintain a similar angle to the light source in the form of a useless and sometimes fatal vortex to mites [1]. Figure 2: A conceptual model for this behavior. We notice that moths eventually approach the lights.

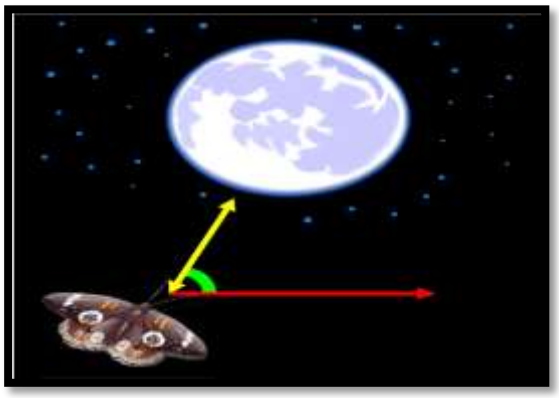

Fig. 1: shows an example to illustrate the mechanism of the transverse direction

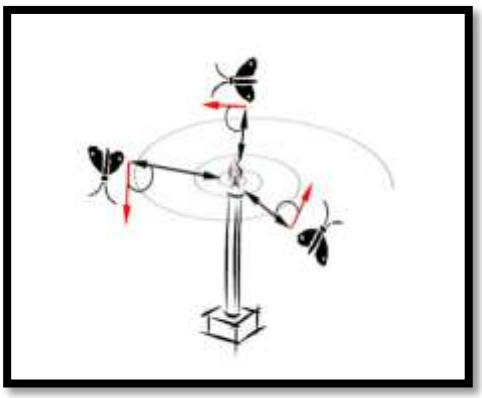

Fig. 2: A conceptual model for this behavior.

2-2 Mathematical representation of MFO

To allow the candidate solutions to be mites, the problem variables will be in the position of the mites in the space. P is a helical function as the moths move towards the search space and we represent a group of moths in a matrix as follows [2]:

$$M=\begin{bmatrix} m_{1,1} & \cdots & m_{1,d} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,d} \end{bmatrix} \quad \ldots.(1)$$

Let n represent moth and d be the variables.

Also suppose there is an array to store fitness function values as follows:

$$OM=\begin{bmatrix} OM_1 \\ OM_2 \\ \vdots \\ OM_n \end{bmatrix} \quad \ldots. (2)$$

Assume that Flame has a suggested algorithm. Consider it with a matrix similar to the moth matrix as follows :

$$F=\begin{bmatrix} f_{1,1} & \cdots & f_{1,d} \\ \vdots & \ddots & \vdots \\ f_{n,1} & \cdots & f_{n,d} \end{bmatrix} \quad \ldots..(3)$$

Let n represent of flame and d be the variables.

Let's get another matrix to store the corresponding relevance values as follows :

$$OF=\begin{bmatrix} OF_1 \\ OF_2 \\ \vdots \\ OF_n \end{bmatrix} \quad \ldots..(4)$$

Each moth updated its location with respect to fire according to the following equation:

$Mi = P(Mi , Fj) \ldots.(5)$

where Mi stands for moth and Fj stands for flame.

There are also other types of helical functions that can be used in relation to the following rules:

1.The moth represents the release of the spiral.

2. The end of the coil represents the position of the flame.

3. The oscillation range of the coil is restricted to the search space only.

$P(Mi , Fj) = Di \cdot e^{bt} \cdot \cos(2\pi t) + Fj \ldots..(6)$

Di mean distance between i and j, and t is the number located between [-1.1], and b fixed to determine the shape of D as well as we use the following equation to find P:

$Di = |Fj - Mi | \ldots. (7)$

Mi represents the first moth, Fj is the flame, and Di is the distance from the moth to the flame.

Equation (2) Helical flight simulator mechanism area. The next position of the moth in relation to the flame is updated. Since the helical equation T determines how close the next position from the moth to the fire is (T = -1 is the moth closest to the flame, and T = 1 represents the location of the moth furthest from the fire). The helical motion is the main component of the determinant method because it showed how to update the position of the moth around the bonfire. The chiral equation allows the flex to fly "around" a flame and not necessarily, how far it is. This method allows to explore the area, and exploit it more extensively. Figure 3 shows different function of T on the curve.

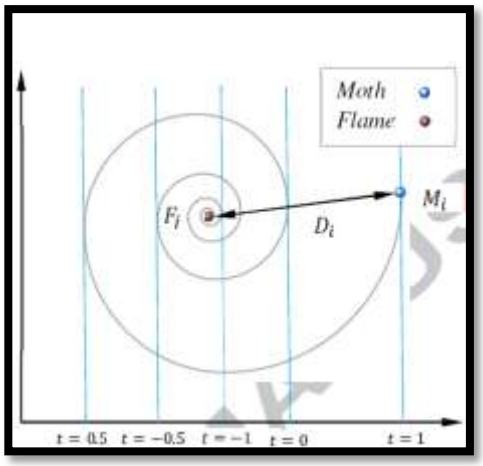

Fig. 3: space around flame, and the position with respect to t

The disadvantages of this method are that the moth constantly updates its location in the search space, which can reduce the exploitation of the best solutions. Therefore, the number of flames is adaptively reduced by a number of iterations using the following formula:

$$\text{flame} = \text{round}\left(\text{N} - \text{L} * \frac{\text{N}-1}{\text{T}}\right) \quad \ldots. \quad (8)$$

L is the current iteration, N is the maximum number of flames, and T is the maximum number of iterations.

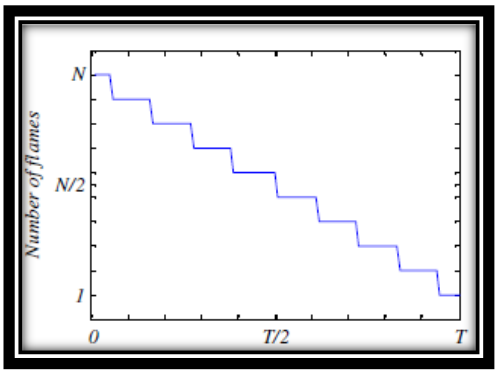

Fig. 4: Adaptation of the flame setting according to the frequency cycle

Figure 4 shows a high number of flames in the first steps, which makes it difficult to exploit so the mites update their positions according to the best flame in the final steps of the replication. To balance exploration and exploitation upon gradual decline within the search space.

Algorithm : Moth Flame Optimization[2]

- Initialize the values of MFO
- Initialize the position of the Moth as Initialize population
- For i from 1 to size of population
- Evaluate the fitness of each search agent
- And for
- While t≤ $Max\_iteraition$
- update the flame by use Equation(8)
- Evaluate the fitness of Moth position as MO
- If t=1
- Best-flame=sort(Moth-pos(t))
- Best-flame-fitness= sort(fitness (t))
- Elso
- Best-flame=sort(Moth-pos(t-1), Moth-pos(t))
- Best-flame-fitness= sort(fitness (t-1), fitness (t))
- End if
- Compute a=-i+(t*((-1)/ $Max\_iteraition$))
- For i from 1 to size of population
- For j from 1 to dim
- If i ≤ flame Moth
- Compute D using Equation (7)
- b=1 , t=(a-1)*rand +1
- Moth-pos(i,j)= D · e^{b*t} · cos(2*pi*t) + best-flame(i,j)
- Else if i> flame
- Compute D using Equation (7) , b=1 , t=(a-1)*rand+1
- Moth-pos(i,j)= D · e^{b*t} · cos(2*pi*t) + best-flame(flame moth , i)
- end if
- end for
- end for
- end While

3- Slime molds algorithm (SMA)

A new inspired optimization algorithm appeared by slime mold behaviors in obtaining the optimal path for food finding. Some scholars in this article have proposed algorithms with similar nomenclature, but the method differs from the design of this algorithm and the use is quite different from the other algorithms proposed. [4] To solve the problem of improving one goal by simulation. As shown in Figure 5. Due to its unique shape and pattern, more than one food source can be used at the same time by forming an intravenous network and deliver it together. When there is enough food in the environment, it has the ability to grow to more than 900 square centimeters. [5]

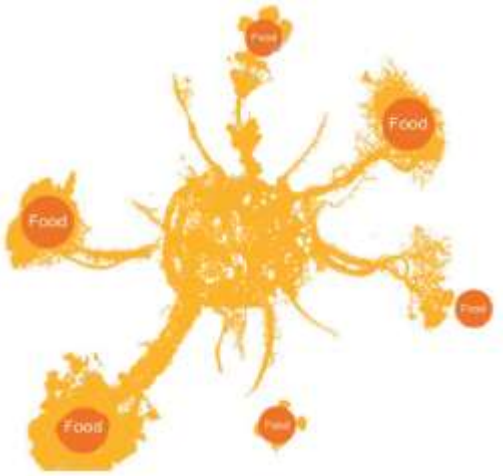

Fig. 5: Spread to get the best food mould

Slime mold's venous bifurcation evolves with different phase contraction mode [6], so that even if the mold finds a better food source, He can still divide himself within the search space to exploit all the resources in order to find good food at once [7]. Experiments have shown that when slime mold finds good quality food, the probability of leaving this area is very low [8]. When you leave the area and go to look in another area, they lack full information about the new exploration , and the best way to estimate when to leave in a new situation is to evaluate the guiding or pilot rules based on insufficient information.

3-1 Mathematical model:

When foraging for food in slime mold, the smell in the air is the medium used to reach the food in the first stage. On this basis, the behavior of slime mold was formulated as follows to simulate the shrinkage mode [9]:

$$\vec{S}(t+1) = \begin{cases} \vec{S_b} + \vec{vb} * \left(\vec{W} * \vec{S_A}(t) - \vec{S_B}(t)\right), r < p \\ \vec{vc} * \vec{S}(t) \qquad r \geq p \end{cases} \quad \ldots\ldots (9)$$

$\vec{vb}$ It represents a value located between [a, -a] as:

$\vec{vb}$ = [a,-a] …… (10)

$a = \operatorname{arctanh}\left(-\left(\frac{t}{t_{max}}\right)+1\right)$ …. (11)

$\vec{vc}$ Linear gradient from 1 to 0.

t represents the current iteration.

t_{max} Represent the highest end to repeat.

$\vec{S_b}$ The carrier, which indicates the highest concentration site odor has been reached so far.

$\vec{S}(t+1)$ Represents the current location of slime mold (SM).

$\vec{S}(t)$ The current location for SM.

$\vec{S_A}$ and $\vec{S_B}$ Two vectors representing the location of any two residents.

r random variable between 0 and 1.

To calculate $\vec{W}$ we use the following equation:

$$\overrightarrow{W(SmellIndex(\imath))} = \begin{cases} 1 + r.\log\left(\frac{bF - S(i)}{bF - wF} + 1\right), condition \\ 1 - r.\log\left(\frac{bF - S(i)}{bF - wF} + 1\right), others \end{cases} \quad \ldots\ldots (12)$$

$p = \tanh|(i) - DF| \ldots\ldots (13)$

Where $S\ (i)$ represents fitness for x and DF Represent the best fitness function obtained.

$SmellIndex$ = sort(S)…….(14)

Where indicates that (i) classify the first part of the population. r It represents a random value in the interval [0,1], bF It is the perfect solution that has been reached in the current iterative,

wF Represents the worst fitness value that has been repetitively performed, $SmellIndex$ Rated sequence indicates fitness values.

3-2 The Iterations

When procedures are made and exploited, and upon implementation of the iteration the positions of each individual will be updated and within the same iteration they will be directed to the global optimization.

Mold slime mimics their search patterns to fine-tune them according to food quality. When the focus of food is the basic content, the weight increases near the region of higher concentration and when the concentration of food decreases, the weight decreases in the area, and this leads to exploration of other areas. Figure 6 illustrates the process, evaluate the fitness values of the slime mold.

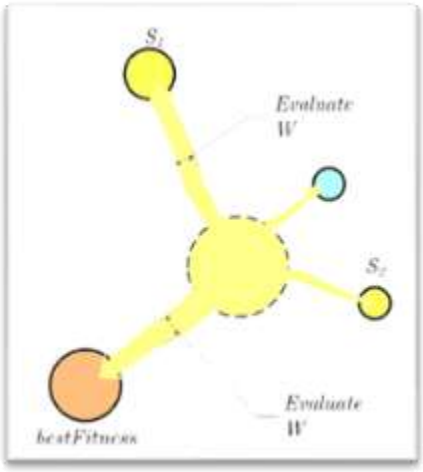

Fig. 6: Assessment of fitness

Mathematical formula for updating the slime mold site based on the principle above it will be as follows:

$$\vec{X^*} = \begin{cases} rand.(UB - LB) + LB, & rand < z \\ \vec{X_b(t)} + \vec{vb}.\left(\vec{W} * \vec{X_A}(t) - \vec{X_B}(t)\right), r < p \\ \vec{vc}.\ \vec{X(t)} \qquad r \geq p \end{cases} \quad \ldots. (14)$$

rand and r two digits were choosers randomly between [0.1].

UB and LB They represent the upper bound and the lower bound of the search space for the problem.

Z Possibility to determine whether SMA will move to another source of food or search for the best around In relative to $\vec{W}$,

$\vec{vb}$ and $\vec{vc}$ It is used to mimic the various venous network.

Finally, SMA steps are summarized and presented by the algorithm.

Algorithm : Slime mold algorithm

- Initialize the parameters;
- Initialize the positions of slime mold $Xi(i = 1,2, \ldots, n)$;
- While $(t \leq Max\_iteraition)$;
- Calculate the fitness of all slime mold;
- $update\ bestFitness, Xb$
- Calculate the W by Eq. (12);
- For $each\ search\ portion\ update\ p, vb, vc$;
- $update\ positions\ by$ Eq. (14);
- End **For**
- $t = t + 1$;
- Return $bestFitness, Xb$;
- End While

4- The proposed Algorithms

In this section, we will introduce two types of hybrid or improved algorithm of MFO by use SMA Algorithms has a good exploitation in the practice, then we will use those properties to improve the performance of MFO.

The exploration of MFO with improving exploitation by SMA will by strong the performance of the new algorithm. We are suggested two types of improving MFO by use SMA depended on the use of exploitation of SMA.

1- The first proposed Algorithms MFOSMA

In this method, the MFO algorithm has been improved by using the v_c exploit factor of the SMA algorithm, for the purpose of increasing the exploitation of local solutions within the global solution area, as the v_c factor is a strong catalyst in the exploitation process, and also the improvement condition was set after using this factor for the purpose of removing non-conforming solutions. And use only good solutions even if the sub-solutions are the same. The algorithm is as follows:

Algorithms MFOSMA

- Initialize the values of MFO
- Initialize the position of the Moth as Initialize population
- For i from 1 to size of population
- Evaluate the fitness of each search agent
- And for
- While t≤ $Max\_iteraition$
- update the flame by use Equation(8)
- Evaluate the fitness of Moth position as MO
- If t=1
- Best-flame=sort(Moth-pos(t))
- Best-flame-fitness= sort(fitness (t))
- Elso
- Best-flame=sort(Moth-pos(t-1), Moth-pos(t))
- Best-flame-fitness= sort(fitness (t-1), fitness (t))
- End if
- Compute a=-i+(t*((-1)/ $Max\_iteraition$))
- Compute b_2= 1-t/($Max\_iteraition$)
- For i from 1 to size of population
- For j from 1 to dim
- If i ≤ flame Moth
- Compute D using Equation (7)
- b=1 , t=(a-1)*rand +1
- Moth-pos(i,j)= D · e^{b*t} · cos(2*pi*t) + best-flame(i,j)
- Elso if i> flame
- Compute D using Equation (7) , b=1 , t=(a-1)*rand+1
- Moth-pos(i,j)= D · e^{b*t} · cos(2*pi*t) + best-flame(flame moth , i)
- end if
- end for
- $x = v_c * Moth - pos(i,j);$
- If fitness(x) ≤ fitness(Moth-pos(i,j);
- $Moth - pos(i,j) = x$;
- else ;
- $Moth - pos(i,j) = Moth - pos(i,j)$;
- end if ;
- end for;
- end While ;

2- The second proposed Algorithms MFOSMA2

In this method, it was proposed to improve the MFO algorithm through the feature of exploitation and exploration together in the SMA algorithm, where the v_c factor was used with the exploitation part of the MFO algorithm, as well as the v_b factor was used with the exploration part of the MFO algorithm, and the improvement condition was also set after using this factor for the purpose of the dimensions of the not-so-good solutions and the use of the good solutions only, even if the sub-solutions were the same, and the algorithm proved successful compared to other algorithms. The algorithm is as follows:

Algorithms MFOSMA2

- Initialize the values of MFO
- Initialize the position of the Moth as Initialize population
- For i from 1 to size of population
- Evaluate the fitness of each search agent
- And for
- While t≤ $Max\_iteraition$
- update the flame by use Equation(8)
- Evaluate the fitness of Moth position as MO
- If t=1
- Best-flame=sort(Moth-pos(t))
- Best-flame-fitness= sort(fitness (t))
- Elso
- Best-flame=sort(Moth-pos(t-1), Moth-pos(t))
- Best-flame-fitness= sort(fitness (t-1), fitness (t))
- End if
- Compute a=-i+(t*((-1)/ $Max\_iteraition$))
- Compute b_2= 1-t/($Max\_iteraition$)
- For i from 1 to size of population
- compute v_c = rand (-b_2, b_2)
- compute v_b = rand (-a_2, a_2)
- for j in the range (1 , dim)
- if i≤ flame-moth
- Moth-pos(i,j) =v_c(j)* Moth-pos(i,j)+Best flame-pos(j)
- end if
- if i> flame-moth
- compute D using Equation (7)
- b=1 , t=(a-1)*rand +1
- Moth-pos(i,j)= D · e^{b*t} · cos(2*pi*t) + sort population
- A=rand in (1,N) , B = rand in (1,N)
- Moth-pos(i,j)= v_b(j)* ((Moth-pos(A,j))- ((Moth-pos(B,j))
- end if
- end for
- end for
- end While

4- Results and discussion

It is common in this area to measure the performance of the algorithms on a set of mathematical functions with a global improvement known. Also, we follow the same process and employ 23 standard function. tests and divided it into three groups, one typical multi-media vehicle. The standardization of functions (F1-F7) to measure the appropriate exploitation of algorithms because they contain a global improvement of one and there is no local.

On the contrary, Multimedia Functions (F8-F13) has a large number of translation improvements and is

useful for checking translation improvements. Finally, Mobile Composite Functions (F14-F23) works with a range of different alignment and multimedia testing functions. Because this search space is very difficult, it is very similar to real search spaces, and it is useful to measure the performance of algorithms in terms of balanced exploration and exploitation of the mathematical formula of the test functions used in Table (1, 2, 3). And since cascading algorithms are optimization techniques Randomization, so we will use the same method to generate and report results across 30 independent orbits. Standard deviation just means the overall performance of the algorithms. In addition to the mean and standard deviation, in order to verify the performance of the proposed MFOSMA algorithm as another version of the algorithms.

Well-known algorithms and modern mixtures with the following algorithms were selected:

PSO [10], MVO [11], GWO [12], MFO [2], CS [13], WOA [14], HHO [15] and SMA [3].

Note that we used 30 numbers of search agents and 1000 iterations of each of the algorithms. This number can be reduced to 20 or 10 additionally. The following table shows the functions that were used:

Table1: The following table shows the functions

S	Dim	Range	F min
$F_1(x)=\sum_{i=1}^{n} x_i^2$	30	$[-100,100]$	0
$F_2(x)=\sum_{i=1}^{n} \lvert x_i\rvert + \prod_{j=1}^{n} \lvert x_i\rvert$	30	$[-10,10]$	0
$F_3(x)=\sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	30	$[-100,100]$	0
$F_4(x)=\max_i\{\lvert x_i\rvert, 1\le i\le n\}$	30	$[-100,100]$	0
$F_5(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2(x_i-1)^2\right]$	30	$[-30,30]$	0
$F_6(x)=\sum_{i=1}^{n}([x_i+0.5])^2$	30	$[-10,10]$	0
$F_7(x)=\sum_{i=1}^{n} i x_i^4+\text{random}[0,1)$	30	$[-1.28,1.28]$	0
$F_8(x)=\sum_{i=1}^{n} -x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	30	$[-500,500]$	$418.9829\times \text{Dim}$
$F_9(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	30	$[-5.12,5.12]$	0
$F_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	30	$[-32,32]$	0
$F_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30	$[-600,600]$	0
$F_{12}(x)=\frac{\pi}{n}\left\{10\sin(\pi y_1)+\sum_{i=1}^{n}(y_i-1)^2[1+10\sin^2(\pi y_{i+})]\right\}$ $+\sum_{i=1}^{n}u(x_i,10,100,4)\ y_i=1+\frac{x_i+1}{4}, \quad u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a<x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	30	$[-50,50]$	0
$F_{13}(x)=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]$ $+(x_n-1)^2[1+\sin^2(3\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	n	$[-50,50]$	0
$F_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	$[-65,65]$	0
$F_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_i(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4	$[-5,5]$	0.0003
$F_{16}(x)=4x_1^2-2.1x_1^2+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2	$[-5,5]$	-1.0316
$F_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	$[-5,5]$	0.398
$F_{18}(x)=\left[1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]$ $\times\left[30+(2x_1-3x_2)^2\times\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]$	2	$[-2,2]$	3
$F_{19}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_y\left(x_j-p_{ij}\right)^2\right)$	3	$[1,3]$	-3.86
$F_{20}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{6}a_y\left(x_j-p_{ij}\right)^2\right)$	6	$[0,1]$	-3.32
$F_{21}(x)=-\sum_{i=1}^{5}\left[(X-a)_i(X-a)_i^2+c_i\right]^{-1}$	4	$[0,10]$	-10.1532
$F_{22}(x)=-\sum_{i=1}^{7}\left[(X-a)_i(X-a)_i^2+c_i\right]^{-1}$	4	$[0,10]$	-10.4028
$F_{23}(x)=-\sum_{i=1}^{10}\left[(X-a)_i(X-a)_i^2+c_i\right]^{-1}$	4	$[0,10]$	-10.5363

The superiority of the MFOSMA algorithm is shown in Table 1, and it gives very competitive results compared to the rest of the algorithms in functions F1, F2, F3, F4 and F7. However, this mechanism is mostly used for exploration, so search clients spend a large number of iterations to explore search spaces and avoid local solutions. This mechanism slows down the exploitation of the MFOSMA algorithm and prevents the algorithm from finding a very accurate approximation to global optimization. The MFOSMA algorithm displays the best results in 5 out of 7 individual conditional test sites. It is clear that

updating the locations of the mites with respect to the best torch increases the convergence and improves the accuracy of the results. As discussed above, single media functions are suitable for scaling exploitation algorithms. Therefore, these results indicate a significant exploit of the MFOSMA algorithm.

Table 1: Results of multimodal benchmark functions (F1-F7)

		F1	F2	F3	F4	F5	F6	F7
PSO	**AVERAGE**	2.03E-06	6.01E+00	2.52E+01	7.96E-01	6.05E+01	**5.53E-07**	5.36E+00
	STD	6.06E+00	7.70E+00	1.31E+01	1.91E-01	3.15E+01	**1.13E-06**	7.95E+00
MVO	**AVERAGE**	5.76E-01	5.66E-01	9.11E+01	1.51E+00	4.74E+02	5.07E-01	2.44E-02
	STD	1.76E-01	2.52E-01	4.07E+01	6.10E-01	7.05E+02	1.57E-01	8.67E-03
GWO	**AVERAGE**	5.65E-50	1.14E-29	1.58E-09	3.37E-12	2.71E+01	9.60E-01	1.21E-03
	STD	1.53E-49	1.14E-29	1.53E-09	4.95E-12	6.58E-01	4.33E-01	4.70E-04
MFO	**AVERAGE**	2.33E+03	3.80E+01	1.92E+04	5.96E+01	9.37E+03	3.34E+03	3.90E+00
	STD	5.68E+03	2.34E+01	1.62E+04	1.12E+01	2.74E+04	6.63E+03	1.00E+01
CS	**AVERAGE**	1.10E-02	6.28E-02	3.43E+02	1.13E+01	9.24E+01	1.42E-02	1.00E-01
	STD	8.28E-03	1.15E+02	1.44E+02	4.14E+00	5.15E+01	1.03E-02	5.81E-02
WOA	**AVERAGE**	2.94E-127	2.08E-96	2.94E+04	3.91E+01	2.75E+01	4.45E-01	1.56E-03
	STD	1.61E-126	1.10E-95	1.32E+04	2.87E+01	6.43E-01	2.56E-01	1.49E-03
HHO	**AVERAGE**	3.25E-123	3.01E-65	1.03E-89	3.44E-59	**3.96E-03**	1.03E-04	1.05E-04
	STD	1.78E-122	1.19E-64	5.62E-89	1.88E-58	**5.61E-03**	1.38E-04	1.03E-04
SMA	**AVERAGE**	**0.00E+00**	2.07E-152	0.00E+00	1.79E-172	3.07E+00	1.26E-03	1.31E-04
	STD	**0.00E+00**	1.10E-151	**0.00E+00**	**0.00E+00**	8.19E+00	5.93E-04	1.14E-04
MFOSMA	**AVERAGE**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.63E+01	2.08E-06	**6.16E-05**
	STD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.54E-01	7.09E-06	**6.95E-05**
MFOSMA2	**AVERAGE**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.78E+01	2.73E+00	8.62E-05
	STD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	7.11E-01	4.80E-01	8.96E-05

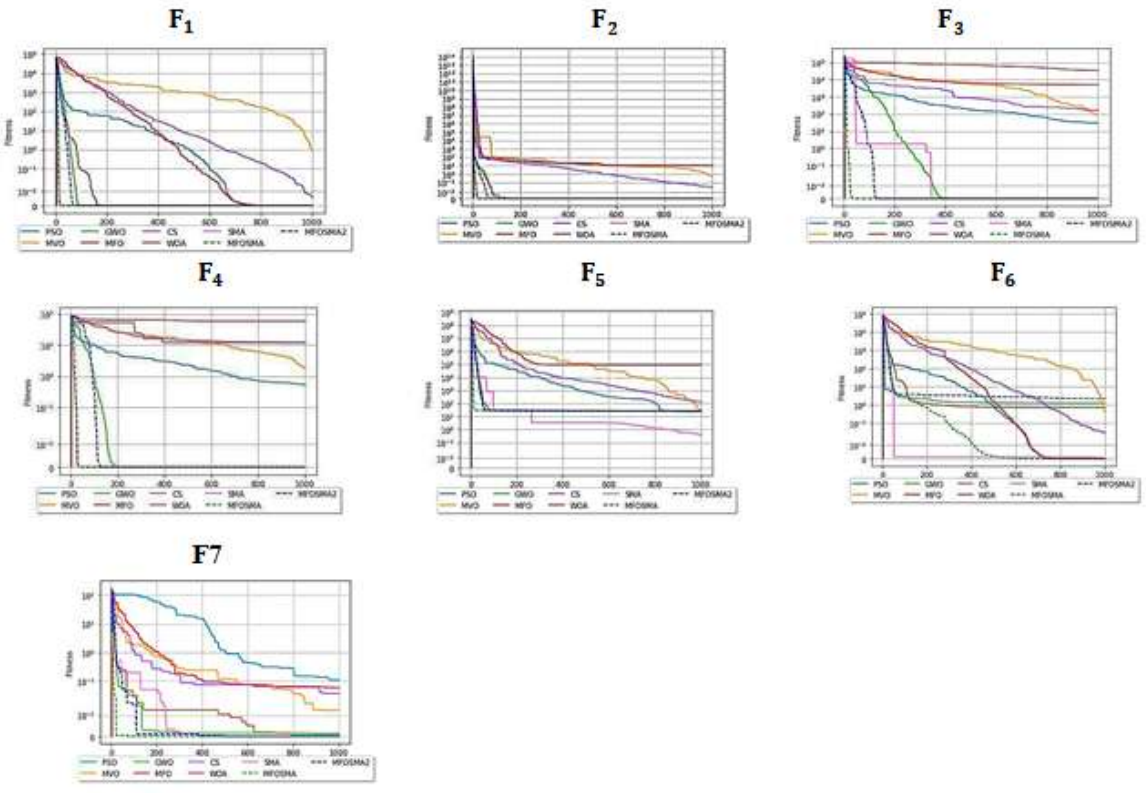

Fig. 7: convergence rate multimodal benchmark functions (F8-F13).

It can be seen that the MFOSMA algorithm is significantly superior to other algorithms in F8, F9, F11 and F12. The MFOSMA algorithm fails to return the best results in F10 and F13. According to the values in Table 2, in other words, the performance of the MFOSMA algorithm is very good and it can be considered as the best algorithm when solving F11. However, the F9 algorithm, is superior to other algorithms in the test function of the test function. Due to the low inconsistency in the results of MFOSMA and the rest of the algorithms, but it seems that MNF failed to exploit and improve the optimization obtained. However, other multimedia test function results strongly prove that the high detection of MFOSMA algorithm is a suitable mechanism for avoiding local solutions. Since the

multimedia functions have an exponential number of local solutions, there are results that show that the MFOSMA algorithm is able to explore the search space on a large scale and find promising results within the search areas of the search. In addition, avoiding the local optimization of this algorithm is another issue that can be inferred from these results. Statistical results of the algorithms on a multimedia test function are presented in Table 3.

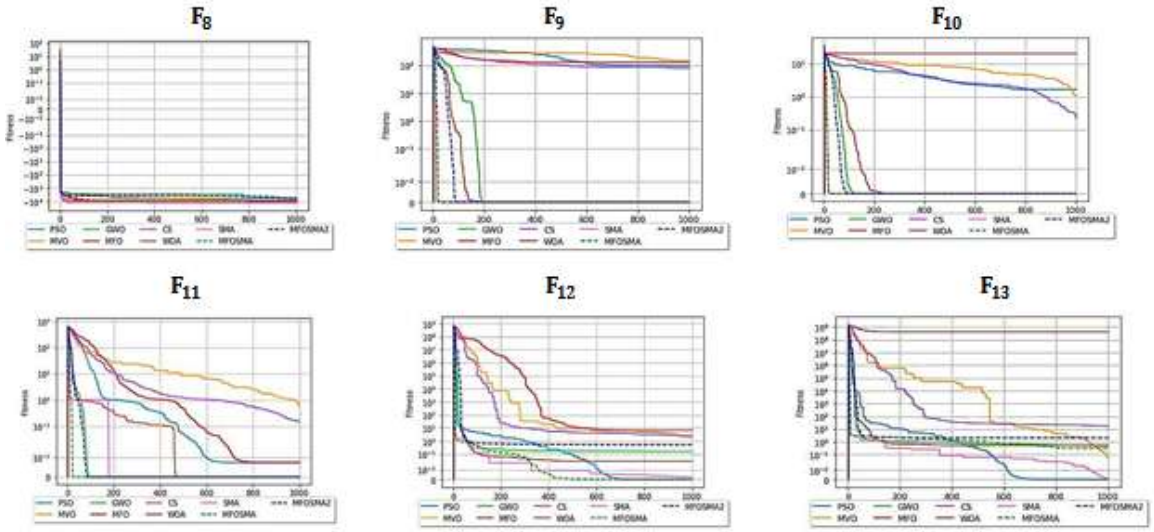

Fig. 8: convergence rate multimodal benchmark functions (F8-F13).

Table 2: Results of multimodal benchmark functions (F1-F7).

		F8	F9	F10	F11	F12	F13
PSO	AVERAGE	-5.59E+03	1.09E+02	2.09E-01	8.45E-03	1.00E-01	4.03E-03
	STD	3.07E+03	3.52E+01	4.82E-01	9.22E-03	7.45E-02	5.38E-03
MVO	AVERAGE	-7.82E+03	1.18E+02	1.53E+00	6.24E-01	1.94E+00	1.09E-01
	STD	7.06E+02	2.78E+01	7.42E-01	9.69E-02	8.66E-01	6.68E-02
GWO	AVERAGE	-5.97E+03	4.28E-01	2.19E-14	2.04E-03	1.53E-01	7.88E-01
	STD	8.45E+02	1.30E+00	9.36E-04	5.84E-03	7.83E-02	1.98E-01
MFO	AVERAGE	-8.55E+03	1.58E+02	1.38E+01	2.11E+01	8.82E+00	1.37E+07
	STD	1.15E+03	3.96E+01	8.93E+00	4.55E+01	5.58E+00	7.49E+07
CS	AVERAGE	-1.02E+04	7.28E+01	1.76E+00	1.25E-01	2.74E+00	1.02E+01
	STD	1.21E+03	5.39E+01	8.51E-01	8.09E-02	1.05E+00	6.60E+00
WOA	AVERAGE	-1.05E+04	**0.00E+00**	3.64E-15	**0.00E+00**	4.81E-02	5.45E-01
	STD	2.00E+03	**0.00E+00**	2.35E-15	**0.00E+00**	5.12E-02	2.23E-01
HHO	AVERAGE	-1.25E+04	**0.00E+00**	**4.44E-16**	**0.00E+00**	5.62E-06	**4.81E-05**
	STD	5.12E+02	**0.00E+00**	**5.01E-32**	**0.00E+00**	9.03E-06	**7.24E-05**
SMA	AVERAGE	**-1.26E+04**	**0.00E+00**	**4.44E-16**	**0.00E+00**	1.66E-03	2.24E-03
	STD	**1.26E-02**	**0.00E+00**	**5.01E-32**	**0.00E+00**	1.36E-03	3.80E-03
MFOSMA	AVERAGE	-8.40E+03	**0.00E+00**	**4.44E-16**	**0.00E+00**	**3.98E-07**	9.25E-02
	STD	8.32E+02	**0.00E+00**	**5.01E-32**	**0.00E+00**	**1.64E-06**	1.04E-01
MFOSMA2	AVERAGE	-4.92E+03	**0.00E+00**	**4.44E-16**	**0.00E+00**	5.13E-01	1.80E+00
	STD	7.98E+02	**0.00E+00**	**5.01E-32**	**0.00E+00**	1.54E-01	2.47E-01

The results, belonging to F14 to F23, are recorded in the timeline. For example, that the MFOSMA algorithm provides better results in some functions in the field of work (F19, F20, F21, F22 and F23), the results of this algorithm are very difficult complex functions in the exploration of a common space. We discussed the results showing that the MFOSMA algorithm is very good in terms of arrogance and readiness. The results indirectly that the algorithm approaches a point in the search space and improves the initial solutions. Figure 9. Search history, path in the first dimension, average fitness and affinity rate.

Table 3: Results of fixed-dimension multimodal benchmark functions (F14-F23)

		F14	F15	F16	F17	F18	F19	F20	F21	F22	F23
PSO	AVE	1.84E+02	4.84E-03	-1.03E+00	**3.98E-01**	5.70E+00	**-3.86E+00**	8.93E+00	-7.42E+00	-6.67E+00	-9.23E+00
	STD	8.17E+01	5.22E-01	0.00E+00	1.13E-16	1.48E+01	3.39E-03	1.14E-01	2.78E+00	3.33E+00	2.59E+00
MVO	AVE	3.30E+02	8.56E-03	-1.03E+00	4.75E-01	8.40E+00	**-3.86E+00**	**-3.27E+00**	-7.34E+00	-5.94E+00	-8.15E+00
	STD	6.99E+01	1.36E-02	1.26E-07	4.21E-01	2.06E+01	1.45E-06	6.11E-02	2.67E+00	3.23E+00	3.41E+00
GWO	AVE	1.27E+01	1.65E-03	-1.03E+00	**3.98E-01**	5.70E+00	**-3.86E+00**	**-3.26E+00**	-8.93E+00	-9.61E+00	**-1.05E+01**
	STD	9.03E-15	5.09E-03	9.66E-09	4.23E+00	1.48E+01	1.51E-03	7.10E-02	2.43E+00	1.83E+00	3.02E-04
MFO	AVE	5.00E+02	1.51E-03	-1.03E+00	**3.98E-01**	**3.00E+00**	**-3.86E+00**	**-3.21E+00**	-6.26E+00	-5.08E+00	-8.09E+00
	STD	1.57E-03	3.58E-03	0.00E+00	1.13E-16	**0.00E+00**	**2.26E-15**	4.81E-02	2.92E+00	3.49E+00	3.50E+00
CS	AVE	5.00E+02	**3.52E-04**	**-1.03E+00**	**3.98E-01**	**3.00E+00**	**-3.86E+00**	**-3.32E+00**	**-1.01E+01**	**-1.02E+01**	**-1.05E+01**
	STD	2.50E-02	**1.08E-04**	1.01E+00	1.13E-16	**0.00E+00**	**2.26E-15**	**9.03E-16**	**5.42E-15**	**5.42E-15**	**5.42E-15**
WOA	AVE	1.27E+01	6.29E-04	-1.03E+00	**3.98E-01**	**3.00E+00**	**-3.86E+00**	**-3.21E+00**	-8.25E+00	-7.79E+00	-7.80E+00
	STD	9.03E-15	3.77E-04	5.71E-10	4.16E-06	1.29E-04	1.12E-02	1.35E-01	2.48E+00	2.82E+00	2.78E+00
HHO	AVE	**9.98E-01**	4.12E-04	-1.03E+00	**3.98E-01**	7.50E+00	**-3.76E+00**	**-3.29E+00**	-5.21E+00	-5.23E+00	-5.47E+00
	STD	3.04E-06	3.15E-04	2.27E-09	1.50E-06	1.02E+01	2.67E-01	5.61E-02	9.24E-01	9.32E-01	1.36E+00
SMA	AVE	1.00E+00	5.21E-04	-1.03E+00	**3.98E-01**	**3.00E+00**	**-3.86E+00**	**-3.25E+00**	**-1.01E+01**	**-1.02E+01**	**-1.05E+01**
	STD	9.40E-03	2.68E-04	**0.00E+00**	2.07E-09	1.83E-10	1.24E-08	5.92E-02	3.45E-06	2.74E-06	2.05E-06
MFOSMA	AVE	1.27E+01	5.25E-04	-1.03E+00	**3.98E-01**	**3.00E+00**	**-3.86E+00**	**-3.24E+00**	-5.72E+00	-6.09E+00	-8.52E+00
	STD	**9.03E-15**	1.67E-04	**0.00E+00**	**1.13E-16**	**0.00E+00**	1.44E-03	1.07E-01	1.75E+00	2.52E+00	2.81E+00
MFOSMA 2	AVE	1.27E+01	4.90E-04	-1.03E+00	3.98E-01	**3.00E+00**	**-3.86E+00**	**-3.12E+00**	-7.43E+00	-7.62E+00	-8.92E+00
	STD	**9.03E-15**	3.27E-04	1.70E-06	7.28E-05	1.15E-05	3.66E-03	1.63E-01	2.81E+00	2.97E+00	2.65E+00

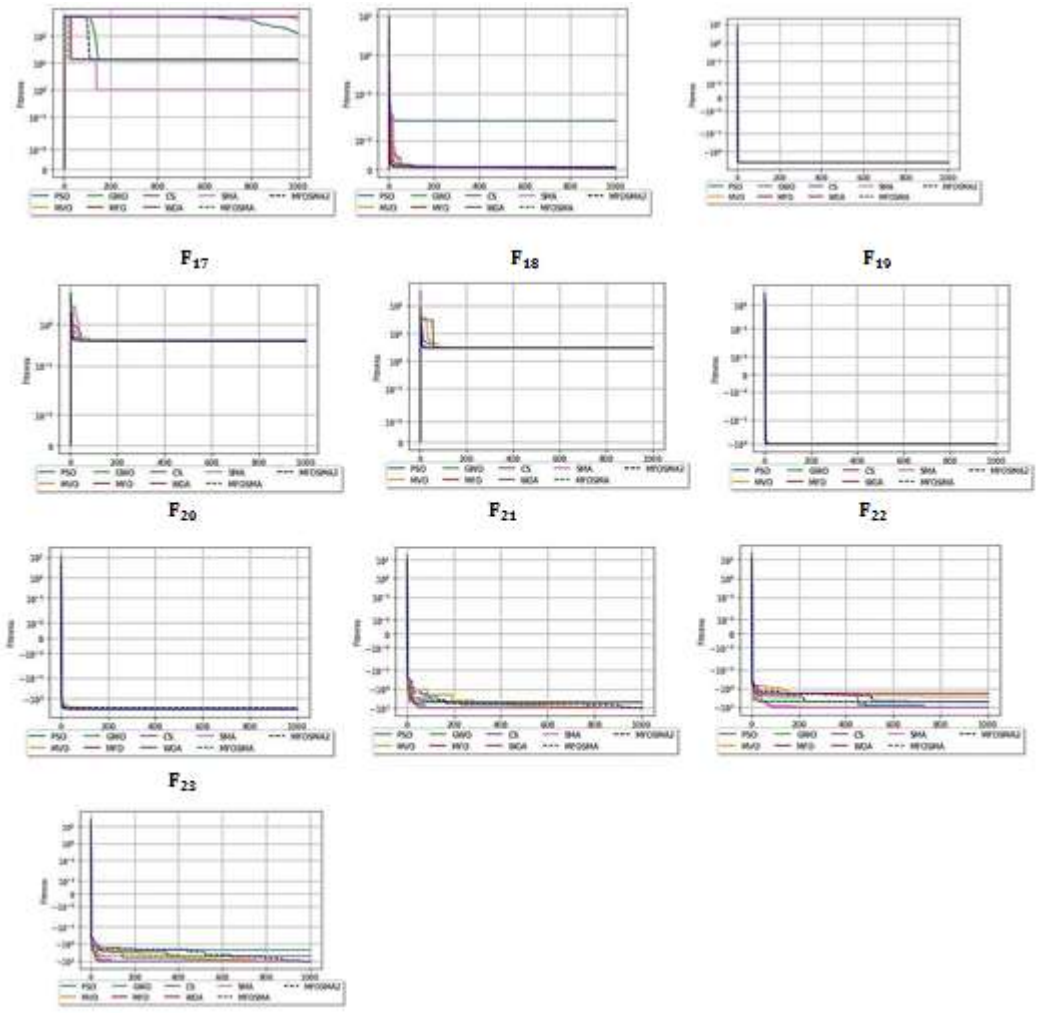

Fig. 9: convergence rate multimodal benchmark functions (F14-F23).

5- Conclusion

Episodic renders are designed to propose a new random population-based algorithm. Indeed. MFOSMA Performance Evaluation Results were compared to PSO, MVO, GWO, MFO, CS, WOA, HHO and SMA for their potential for improvement. 23 test functions were employed to measure MFOSMA from different expectations-views. The results look good and competitive in terms of cross-functional tasks, finding and resolving errors. In the first test phase. After that, the results prove to be very effective in solving problems.

Notes can be the following conclusion:

• It has the ability to update locations by nearby solutions around flames.

• Avoid high local solutions.

• Updating each Flame site increases the exploration of the search space and reduces the possibility of local solutions.
• Adapting the exploitation areas and accommodating the search space.
• Judgment in external solutions
• MFOSMA Algorithm Available to Solve Real Problems According to No Free Food (NFL) theory, there is no optimization algorithm to solve all optimization problems. Therefore it has the ability to outperform the rest of the algorithms in the case of the study, and can be considered as an alternative optimizer to solve the optimization problems in this study.

References

[1] Frank, K. D., Rich, C., & Longcore, T. (2006). Effects of artificial night lighting on moths. Ecological consequences of artificial night lighting, 305-344.

[2] Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowledge-based systems, 89, 228-249.

[3] Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. Future Generation Computer Systems, 111, 300-323.

[4] Monismith, D. R., & Mayfield, B. E. (2008, September). Slime mold as a model for numerical optimization. In 2008 IEEE swarm intelligence symposium (pp. 1-8). IEEE.

[5] Kessler, D. (1982). Plasmodial structure and motility. Cell biology of Physarum and Didymium /edited by Henry C. Aldrich, John W. Daniel

[6] Nakagaki, T., Yamada, H., & Ueda, T. (2000). Interaction between cell shape and contraction pattern in the Physarum plasmodium. Biophysical chemistry, 84(3), 195-204.

[7] Beekman, M., & Latty, T. (2015). Brainless but multi-headed: decision making by the acellular slime mould Physarum polycephalum. Journal of molecular biology, 427(23), 3734-3743.

[8] Latty, T., & Beekman, M. (2015). Slime moulds use heuristics based on within-patch experience to decide when to leave. The Journal of experimental biology, 218(8), 1175-1179.

[9] Nakagaki, T., Yamada, H., & Ueda, T. (2000). Interaction between cell shape and contraction pattern in the Physarum plasmodium. Biophysical chemistry, 84(3), 195-204.

[10] Eberhart, R., & Kennedy, J. (1995, November). Particle swarm optimization. In Proceedings of the IEEE international conference on neural networks (Vol. 4, pp. 1942-1948).

[11] Sharma, M., Bansal, R. K., Prakash, S., & Asefi, S. (2021). MVO algorithm based LFC design of a six-area hybrid diverse power system integrating IPFC and RFB. IETE Journal of Research, 67(3), 394-407.

[12] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.

[13] Yang, X. S., & Deb, S. (2009, December). Cuckoo search via Lévy flights. In 2009 World congress on nature & biologically inspired computing (NaBIC) (pp. 210-214). Ieee.

[14] Husein I., Riski A., Pradjaningsih A, (2020), "Application of Whale Optimization Algorithm (WOA) on Quadratic Knapsack 0-1 Problem "E-ISSN: 2580-5754,P-ISSN:2580-569X, Vol. 4, No. 1,pp. 13-20

[15] Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. Future generation computer systems, 97, 849-872.

تطوير خوارزمية عثة - لهب المحسنة باستخدام خوارزمية عفن الوحل

سامي ناظم حسين [1] ، نزار خلف حسين [2]

[1]قسم الرياضيات ، كلية علوم الحاسوب والرياضيات ، جامعة الموصل ، الموصل ، العراق

[2]قسم الرياضيات ، كلية علوم الحاسوب والرياضيات ، جامعة تكريت ، تكريت ، العراق

الملخص

تعد خوارزمية MFO واحدة من خوارزميات التحسين الحديثة التي تعتمد على ذكاء السرب ، ولان خوارزمية SMA أيضًا واحدة من أحدث الخوارزميات في نفس المجال وتمتلك مزايا التقارب السريع ودقة التقارب العالية والقوية والمتانة. في ورقة البحث هذه ، قدمنا خوارزمية محسّنة لـ MFO استنادًا إلى خوارزمية SMA للحصول على أداء أفضل باستخدام الميزات الموجودة في الخوارزميتين ، وقد تم اقتراح خوارزميتين مختلفتين في هذا المجال. تم اختبار الخوارزميتين الجديدتين المتوقعتين بوظائف اختبار قياسية وكانت النتائج مشجعة مقارنة بالخوارزميات القياسية.