Indonesian Journal of Electrical Engineering and Computer Science
Vol. 35, No. 2, August 2024, pp. 1022~1031
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v35.i2.pp1022-1031

Improved moth search algorithm with mutation operator for numerical optimization problems

Sanaa A. A. Ghaleb[1,3,4,5], Mumtazimah Mohamad[1], Waheed Ali Hussein Mohammed Ghanem[2,3,4,5], Arifah Che Alhadi[2], Abdullah B. Nasser[6], Hanan Aldowah[7]
[1]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA), Terengganu, Malaysia
[2]Faculty of Computer Science and Mathematics, Universiti Malaysia Terengganu (UMT), Terengganu, Malaysia
[3]Faculty of Education, University of Lahej, Lahej, Yemen
[4]Faculty of Engineering, University of Aden, Aden, Yemen
[5]Faculty of Education, University of Aden, Aden, Yemen
[6]School of Technology and Innovation, University of Vaasa, Vaasa, Finland
[7]School of Management, Universiti Sains Malaysia (USM), Pulau Pinang, Malaysia

Article Info

Article history:

Received Feb 24, 2024
Revised Mar 11, 2024
Accepted Mar 30, 2024

Keywords:

Meta-heuristic
Optimization algorithm
Exploitation
Exploration
Moth search algorithm
Mutation operator

ABSTRACT

The moth search algorithm (MSA) is a meta-heuristic optimization technique inspired by moth behavior, has shown remarkable efficacy in solving optimization challenges. However, its poor exploration capability results in an imbalance between exploitation and exploration. To address this issue, this research introduces a new mutation operator to enhance exploration by increasing population diversity. The proposed enhanced moth search algorithm (EMSA) aims to expedite convergence and improve overall robustness by exploring new solutions more effectively. Evaluation on ten benchmark functions demonstrates EMSA's superior exploration capabilities, efficiently tackling optimization problems and yielding more optimal solutions within the search space. Compared to conventional MSA and other established algorithms, EMSA delivers well-balanced results, showcasing its effectiveness in optimizing the search space. In the future, the EMSA could potentially find applications in addressing real-world engineering optimization challenges.

Corresponding Author:

Mumtazimah Mohamad
Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA)
Kuala Nerus, Terengganu, 21300, Malaysia
Email: mumtaz@unisza.edu.my

1. INTRODUCTION

The field of optimization presents a significant challenge across various domains, where the task often involves finding the best solution among a multitude of possibilities while minimizing associated costs [1]. In recent years, metaheuristic algorithms have emerged as powerful tools for addressing such optimization problems [2]. Among these algorithms is the moth search algorithm (MSA) [3], which draws inspiration from the natural behaviors of moths. However, despite its simplicity and effectiveness, MSA suffers from limitations, particularly in achieving a balance between exploitation and exploration during the optimization process [4]. Existing solutions to optimization problems include a plethora of metaheuristic algorithms such as ant lion optimization (ALO) [5], [6]; dragonfly algorithm (DA) [7], [8]; monarch butterfly optimization (MBO) [9], [10] among others [11], [12]. While these algorithms have shown efficacy in various scenarios, each comes with its own set of strengths and weaknesses.

This study introduces a novel enhancement to the Moth Search Algorithm (MSA), a metaheuristic algorithm inspired by moth behavior, known for its efficient convergence towards optimal solutions but

limited in global exploration. The proposed enhancement incorporates a mutation operator to improve both exploration and exploitation in MSA. The MSA, while effective in local exploitation, can struggle with global exploration due to its reliance on strategies like decreasing values and random walks. To address this, we introduce a mutation operator into MSA to diversify the population and enhance exploration in the search space. This mutation operator combines local search techniques based on self-experience with global search techniques based on neighboring experience. The main goal is to expedite global convergence rates and avoid local optima traps. Simulation results demonstrate that integrating mutation operators into MSA significantly improves population diversity, enabling individuals (moths) to escape local optima more efficiently. The proposed Enhanced Moth Search Algorithm (EMSA) outperforms traditional MSA, especially with deterministic mutation operators, showcasing superior performance across various benchmark functions.

While this study introduced the innovative approach of utilizing a random mutation operator to improve the MSA, prior research has delved into refining traditional MSAs using different methodologies. For instance [13] introduced the CMSA algorithm, which utilizes chaos theory to improve the global convergence of MSA optimization. By integrating various chaotic maps into the MSA Logistic map, an effective balance between exploration and exploitation was sought. While the inclusion of chaotic maps showed promise in enhancing MSA performance, the results fell short of expectations due to the unsuitability of chaotic factors in handling benchmark functions. However, a notable drawback of this study is its dependence solely on four benchmark functions to assess the effectiveness of the CMSA. Strumberger *et al.* [14], the scholars have proposed a modified MSA for portfolio optimization. They utilized two distinct models for the portfolio selection problem: a straightforward one and a cardinality-constrained model. One of the primary disadvantages is that they produce a discrete solution space. Also, this study relies solely on four benchmark functions to assess the effectiveness of the modified MSA algorithm.

In their study referenced as [15], the authors introduced nine innovative algorithms under the umbrella of MSA, denoted as MSA 1 – MSA 9. To evaluate the influence of the "fly straight" and Lévy flight operators in the MSA, the Lévy flight operator was replaced with 9 alternative mutation operators, drawing inspiration from the global harmony search. The experimental findings indicated that these newly proposed MSA-based methods demonstrated enhanced overall performance. Nevertheless, the outcomes did not reach their optimum, primarily due to the discrepancy between the non-linear comfort zone parameter and the Levy flight factors in handling multiple optimization functions. In their study [16], researchers introduced a binary MSA algorithm. They integrated a self-learning flight strategy into the enhanced MSA to facilitate individual learning. Evaluation across 89 diverse benchmark instances revealed that while the proposed method improved global search capability and population diversity, its efficacy was constrained due to a deficiency in randomness. Finally, in their research [17], researchers introduced a novel approach called game model integrated enhanced MSA optimization (GMMSAO) to delve into design possibilities and streamline system optimization. GMMSAO incorporates a game model concept in its input phase, facilitating effective exploration of the system's design space across various configurations. This approach aims to identify the optimal solution within a reasonable timeframe by leveraging diverse search iterations. However, the absence of robust randomness could limit diversity during search iterations, restricting each search agent to predetermined positions. Moreover, numerous other application-specific modifications to the MSA can be found in the literature [18]-[20].

Previous research has primarily focused on refining traditional MSAs using different methodologies, introducing innovative algorithms under the umbrella of MSA, and proposing application-specific modifications. However, gaps remain in understanding how to enhance the MSA's efficacy in terms of convergence rate, solution quality, and exploration capabilities, especially when compared to other metaheuristic algorithms. Following the previous studies, we propose an enhanced version of the MSA (enhanced moth search algorithm) named EMSA. EMSA integrates a novel mutation operator into the conventional MSA algorithm to tackle issues such as local optima entrapment and slow convergence. Our approach aims to strike a balance between exploration and exploitation by employing a specialized mutation operator that enriches the diversity of the standard MSA, thereby facilitating the discovery of optimal solutions for global optimization problems. To assess the enhancement in solution accuracy, we conduct experiments using ten established test functions commonly utilized in swarm intelligence research for evaluating algorithm accuracy and performance, some of which were employed in the aforementioned studies. This study encompasses all of these functions. Our experimental results demonstrate that our suggested EMSA outperforms all other algorithms documented in the literature.

2. METHODS

2.1. Numerical optimization

Global optimization issues have been successfully solved in practice using metaheuristics. Global optimization seeks the most favorable solution from a set of viable options, aiming to maximize or minimize

a desired parameter within a specified range [21]-[23]. Identifying a vector inside a given domain that, out of a large number of conceivable solutions, delivers the optimal answer is, in other words, the process of optimization. In a variety of disciplines, including control theory, mathematics, management science, and computer science, optimization problems are sometimes known as mathematical programming or mathematical optimization. The fundamental idea behind optimization involves systematically evaluating a set of specified values, adhering to predefined forms and domains. For each input within this set, the corresponding output of a function is computed, aiming to identify the optimal values. The primary objective is to either maximize or minimize an objective function, subject to constraints within a designated domain. The formulation of an optimization problem typically follows the process outlined below.

$$Minimize/Maximize\ f(x): X \longrightarrow R;\ x\ represents\ a\ set\ of\ real\ numbers \quad (1)$$

Subject to: $f(x_0) \leq f(x)$ for all x in X ("minimize") / $f(x_0) \geq f(x)$ for all x in X ("maximize")
Where: $f(x)$: $Rn \longrightarrow R$ (is the goal to minimize or increase the objective function over x)

The optimization approach outlined above is specifically designed to address numerical problems, and many practical engineering and scientific issues can be formulated within this framework. In the previously mentioned (1), the function *f* serves as the objective function used to evaluate all feasible solutions within the search space, which is defined by the domain *x*. The choice of the objective function, denoted as *f*, depends on the specific application requiring optimization, whether it is geared towards minimizing or maximizing. The conventional method for articulating an optimization problem typically emphasizes minimization. It is essential to note that within a search space, multiple local minima may exist alongside the true extreme value, be it a minimum or maximum. The following statement holds for a point to be a local minimum at *x**.

$$f(x*) \leq f(x)\ For\ all\ x\ in\ some\ neighborhood \quad (2)$$

Generally, there are two main groups of optimization techniques: deterministic and stochastic. The primary distinction between them is that the deterministic method performs well and is effective when seeking a local optimum. Additionally, it yields the same final answer provided that one commences from an identical set of initial solutions. In contrast, a stochastic algorithm is more useful in optimization techniques where the solution space is frequently unreasonably large. The process commences with an initial set of solutions, employing random operators to navigate the solution space until identifying a global optimum. Notably, the outcome of a subsequent run of the same method, despite having the same initiation, may vary from the initial result.

2.2. Moth search algorithm

2.2.1. Levy flights

The moth will execute lévy flights to encircle the best butterfly because they are closer to it. In other words, as illustrated in (3), their positions are updated by completing lévy flights. The update for moth *i* can be expressed as follows: A power-law formula, as described in [3] can be employed to mathematically represent the lévy distribution, as demonstrated in (4).

$$x_i^{t+1} = x_i^t + \alpha L(s) \quad (3)$$

$$L(s) \sim |s|^{-B} \quad where\ 1 < \beta \leq 3\ is\ an\ index \quad (4)$$

The given text discusses a mathematical or computational model that involves a parameterized update rule for positions in a generation-based process. The notation used includes variables such as *t* for the current generation, x_i^{t+1} and x_i^t for the updated and typical positions at generation *t* respectively, and *L(s)* representing the step from Lévy flights. The scale factor for the problem is denoted as the parameter α, and its specification is provided in the context of the work.

$$\alpha = S_{max}/t^2 \quad (5)$$

Where S_{max} represents the maximum step size for walking, and its value is determined in accordance with the specific problem. The lévy distribution, denoted as *L(s)* in (3), can be expressed as follows:

$$L(s) = \frac{(\beta-1)\Gamma(\beta-1)sin(\frac{\pi(\beta-1)}{2})}{\pi\, s^{\beta}} \quad (6)$$

When the gamma function, $\Gamma(x)$, is greater than *s*.

The utilization of the lévy distribution with a parameter value of β = 1.5 provides a means to deduce the lévy flights undertaken by moths, as elucidated earlier.

2.2.2. Fly straightly

Some moths may fly in a line toward a light source when they are far away from it. Below is a description of this procedure. The flights of moths can be described as:

$$x_i^{t+1} = \lambda \times (x_i^t + \phi \times (x_{best}^t - x_i^t)) \quad (7)$$

Where, in this approach, ϕ is a scaling factor with the golden ratio as its base, and x_{best}^t is the top moth at generation *t*. A scale factor is λ. For simplicity, (7) or (8) will be used to update the position of moth *i* with a probability of 50%. The moth, conversely, could move towards its previous location, which is more distant from the light source. The final location of moth *i* in this scenario can be denoted as:

$$x_i^{t+1} = \lambda \times (x_i^t + \frac{1}{\phi} \times (x_{best}^t - x_i^t)) \quad (8)$$

The diagram in Figure 1 represents three key positions: x_{best}, x_i, and $x_{i,new}$, symbolizing the optimal, initial, and modified locations of a moth acting as a light source. x_{best} marks the most effective position, while x_i denotes the starting point, and $x_{i,new}$ signifies a modified location after optimization. As can be seen in Figure 1(a) presented the movement from x_i (initial position) to $x_{i,new}$ involves some form of optimization or improvement. This could be related to maximizing light intensity, minimizing energy expenditure, or achieving a better overall performance metric. The moth constantly adjusts its position in an iterative process aimed at reaching the optimal position (x_{best}). In Figure 1(b) the positions x_{best}, x_i, and $x_{i,new}$ create a feedback loop. The moth starts at x_i, evaluates its current state, makes adjustments to move towards x_{best}, and then repeats this cycle as needed. This feedback loop is crucial for continuous improvement and adaptation. Ideally, through these iterative adjustments, the moth converges towards x_{best}, indicating successful optimization or achievement of the desired goal.

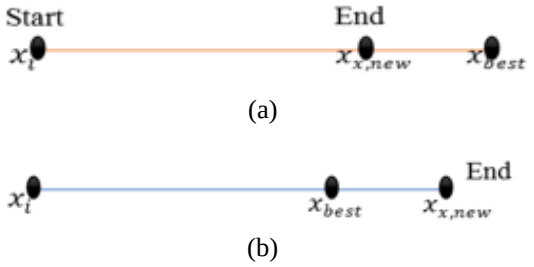


Figure 1. Removing of unwanted outer area from the image (a) original image and (b) cropped image

The parameter λ plays a crucial role as a scaling factor, influencing the convergence rate of the algorithm and augmenting the diversity within the population. Notably, the scaling factor is determined by a randomly generated value derived from the standard uniform distribution. Additionally, the Algorithm 1 encompasses pseudocode for MSA.

Algorithm 1: MSA algorithm [3]

```
Step 1: // Initialize parameters such as: Set t = 1; // Generation number
Generate MaxGen, max walk step S_max, the index β, and acceleration factor ϕ;
Step 2 Evaluate the fitness of each moth; // Evaluate fitness
Step 3 While (t < MaxGen) do
Sort all the moth individuals as per their fitness;  // Sort moths based on fitness
   For i=1 to NP/2 do // Update positions for moths in Subpopulation 1
        Generate  x_i^{t+1};   Eq. (3)
                Adjust the boundaries for the current moth in population;
     End for i
For i= NP/2+1 to NP do   // Update positions for moths in Subpopulation 2
        If rand > 0.5 then
            Generate  x_i^{t+1} by Eq. (7);
        Else
            Generate x_i^{t+1} by Eq. (8);
        End if
End for i
Update T; // Update global information if a better solution is found
```

```
t = t + 1; // Increment generation counter

Step 4 End while
Step 5 Output the best solution
End
```

2.3. The enhanced MSA (EMSA)

In this section, we present an enhanced version of EMSA, an algorithm built upon the typical MSA discussed earlier. The conventional moth algorithm demonstrates an ability to navigate the search space effectively. However, it occasionally becomes ensnared in local optima, impeding its ability to perform efficient global searches. To mitigate the risk of getting stuck in local optima within the MSA, it is essential to enhance the diversity of the search process. Numerous studies have proposed solutions to address this issue [24], [25]. some of which will be discussed in section 1.

The fundamental concept underlying the algorithm presented in our paper involves enhancing the MSA process with a highly effective operator. This operator comprises random-based modifications aimed at enhancing the diversity of the MSA, thereby facilitating more mutations within the examined solutions during the MSA search. Consequently, this allows the algorithm to escape potential local optima traps. Essentially, while the MSA primarily leverages solutions within its local neighborhood, the addition of the mutation operator enables exploration of new areas within the search space. The key distinction between EMSA and traditional MSA lies in the utilization of the added mutation operator to refine the typical MSA, generating a fresh solution for each iteration. Accordingly, the principles of exploitation and exploration emerge as pivotal characteristics in crafting a proficient optimization algorithm. The newly proposed algorithm, EMSA, and its detailed pseudocode are presented in Algorithm 2.

Algorithm 2: EMSA algorithm

```
Step 1: // Initialize parameters
Set t = 1; // Generation number
Initialize NP moths' P population at random using a uniform distribution;
Generate MaxGen, max walk step S_max, the index β, and acceleration factor ϕ;
// Generate a random population (X_i^d)
For i = 1 to N do
       For d = 1 to dim do
              (X_i^d) = Random Value ();
       End for d
End for i
Step 2:                    // Calculate the fitness of each moth
Calculate Fitness ();
T_d^ = the best moth;            // Initialize the best solution
Step 3 // Main loop
While (t < MaxGen) do
    Update Solution ();                                  // Update the solution using Eqs. (9), (10)
    // Iterate through the population
     For i=1 to N (moth in population) Do
        if ε_1 ≤ p Then                                                                  // Check probability condition
           MSA phase ();
         Else
            if ε_2 ≤ limit Then                                                        // Check another condition
           Randomly select a moth (r1);                  // Randomly select moths from the population
              x_{ir_1}^{t+1} = x_{r_1}^t;
              Randomly select a moth (r2);
              x_{ir_2}^{t+1} = x_{r_2}^t;
              if (r1≠ r2) Then                                                          // Update the positions based on conditions
                            x_i^{t+1} = w × (x_{ir_1}^t − x_{best}^t) × 2× (rand-1);
              Else
             x_i^{t+1} = w × (x_{ir_2}^t − x_{best}^t) × 2× (rand-1);
              End if
       Else
              x_i^{t+1} = x_{min}^t + rand × ( x_{max}^t − x_{min}^t);      // Update the position randomly
        End if
    End if
End for i
Update T (); // Update the best solution
t = t + 1; // Increment generation counter
```

```
Step 4 End while //
Return the best solution of T; // Return the best solution of T
```

To enhance solution diversity, randomization is introduced into the MSA component. For increased diversity of solutions, randomization is necessary. While x_i^(t+1) plays a comparative role in MSA, it is only applicable to certain local jump adjustments, rendering it a local search equivalent. By leveraging randomness, the algorithm can explore diverse regions with significant variability to find the overall best solution. Additionally, considering the moth's prior likelihood of existence and evaluating the best and worst moths, the mutation operator introduces random adjustments to the moth's movement within the search space, fostering increased diversity within the population. In this study, the MSA's exploitation step is utilized in tandem with the mutation operator to balance 50% of the MSA-calculated search space. These combined techniques expedite convergence towards the optimal solution, enhancing solutions that may initially fall outside the acceptable range. To summarize, the proposed method consists of two primary components: the starting phase and the updating phase. Throughout the iterative process, the model assesses the degree of aging probability values for individuals, aiding in the selection of the appropriate search space (either the primary or updating phase) to refine the overall strategy.

The proposed algorithm, EMSA, extends the typical MSA introduced in the previous section, aiming to enhance its performance by integrating a highly efficient operator. The core idea behind this mutation operator involves a set of random-based adjustments designed to improve the diversity of the MSA algorithm, facilitating an increase in mutations within the solutions explored during the search process, helping to avoid traps of local optima. Importantly, the enhanced MSA algorithm demonstrates proficiency in both exploiting solutions in the local neighborhood and exploring new areas in the search space simultaneously, crucial features in developing an effective optimization algorithm [3]. The EMSA algorithm distinguishes itself from the MSA by integrating a mutation operator designed to enhance the typical MSA. This augmentation results in the creation of unique solutions for each iteration. The algorithm's primary improvement lies in the introduction of the mutation operator, aiming to augment population diversity, consequently refining search efficiency, and hastening convergence to the optimal value.

Notably, EMSA retains all parameters of the MSA while incorporating 2 additional control parameters, namely p and limit. These variables play a pivotal role in achieving balance, with p representing a value within the [0, 1] range, and limit involving two random numbers drawn from a uniform distribution. When $\varepsilon_1 \leq p$,, the generation of a new solution is initiated through the utilization of the MSA phase. If $\varepsilon_1 > p$, the new solution will be generated using the mutation operator. Limit is the control parameter for the new mutation operator, and if it is set to $\varepsilon_2 \leq Limit$, it will randomly select two moths from the population, $x_{r_1}^t \; and \; x_{r_2}^t$. The population size is denoted by *N*, and the integers in the range [1, *N*] represent the values of (r_1) and (r_2). In (9) alters the value of x_i^{t+1} if (r_1) and (r_2) are distinct. Otherwise, x_i^{t+1} is not modified by (10). However, x_i^{t+1} is arbitrarily changed from the set of workable solutions if $\varepsilon_2 > Limit$. In the population under study, the best moth is represented by the string x_{best}^t, where *t* is the generation in question. A variable called *rand* produces random numbers using the same distribution between 0 and 1.

$$x_i^{t+1} = w \times \left(x_{ir_1}^t - \; x_{best}^t\right) \times 2 \times (rand - 1); \tag{9}$$

$$x_i^{t+1} = w \times \left(x_{ir_2}^t - \; x_{best}^t\right) \times 2 \times (rand - 1); \tag{10}$$

The EMSA plays a pivotal role in advancing MSA capabilities by seamlessly incorporating a novel mutation operator. This integration aims to augment population diversity, fostering a broader range of genetic variations. The ultimate objective is to optimize MSA performance, expediting the convergence process towards the optimal point.

3. EXPERIMENTAL EVALUATION

In this study, ten benchmark functions are utilized for assessment, comparing their efficacy with both traditional MSA and various metaheuristic algorithms. Evaluation criteria focus on achieving optimal solutions and identifying the best ones. A consistent population size of 50 is maintained, with the maximum number of generations unchanged. To ensure reliability, each experiment is run 30 times, minimizing the influence of chance.

The research is conducted on a laptop with an Intel Core i7 processor (2.4 GHz) and 8 GB of RAM. The EMSA introduced in this study, is derived from the typical MSA implementation.

All algorithms, including ALO, DA, and MBO, are executed on MATLAB R2020b, using Windows 8. Ten test optimization functions refer to Table 1 for details.

Table 1. Optimization functions

No	Name	Equation	Range		Opt
1	Sphere	$f(x)=\sum_{i=1}^{n} x_i^2$	-5.12	5.12	0
2	Schwefel 2.22	$f(x)=\sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	-10	10	0
3	Schwefel 2.21	$f(x)=max_i\{\lvert x_i\rvert, 1\le i\le n\}$	-100	100	0
4	Rosenbrock	$f(x)=\sum_{i=1}^{n-1}\left[100\sqrt{\lvert x_i-x_i^2\rvert}+(1-x_i)^2\right]$	-30	30	0
5	Step	$f(x)=\sum_{i=1}^{n}[\lvert x_i\rvert]$	-100	100	0
6	Quartic	$f(x)=\sum_{i=1}^{n} ix_i^4+rand[0,1)$	-1.28	1.28	0
7	Rastrigin	$f(x)=\sum_{i=1}^{n}[x_i^2-10\cos 2\pi x_i+10]$	-5.12	5.12	0
8	Ackley	$f(x)=-20e^{\left(-0.2\times\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)}-e^{\left[\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right]}+20+e^1$	-32.8	32.8	0
9	Griewank	$f(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	-600	600	0
10	Penalized No.2	$f(x)=0.1\times\{sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2[1+sin^2(3\pi x_{i+1})]$ $+(x_n-1)^2[1+sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	-50	50	0

4. RESULTS AND DISCUSSION

In this part, we thoroughly evaluate the EMSA algorithm, comparing its effectiveness as a versatile numerical optimizer with various metaheuristic algorithms. Our evaluation entails a meticulous comparison between EMSA and selected benchmarks, with a focus on global optimization. We systematically test EMSA across a range of optimization functions detailed in Table 1 (section 3), comparing its performance against ALO, DA, MSA, and MBO. Each experiment includes testing across different parameters of benchmark functions, with dimensions of 30 and 90, a population size of 50, and a consistent number of generations. To ensure reliability, we conduct 30 runs for each experiment.

Initially, we validate EMSA using ten benchmark functional datasets, comparing its performance with MSA. Contrasting EMSA with MSA, we find that essential parameters such as the maximum step S_{max}, acceleration factor ϕ, and index β are required for EMSA, set to specific values. Parameters for ALO, DA, and MBO are set based on typical works by Mirjalili. For MBO, specific parameter values including *BAR*, migration Peri, maximum step S_{max}, keep, and migration ratio ρ are assigned. Our experiments cover functional dimensions of 30 and 90, mirroring previous experiments, and evaluate EMSA against ALO, DA, MSA, and MBO using ten functions. Table 2 presents a comparative analysis of the results, showing that EMSA consistently outperforms other algorithms, achieving optimal solutions within the search space. The table displays the minimum values achieved by EMSA after 50 generations, repeated 30 times, for each benchmark function, emphasizing the best outcomes to underscore its outstanding performance.

The findings from Table 2 clearly indicate that the EMSA algorithm, as proposed, exhibits better performance compared to the alternative algorithm. Across all tested functions ($f1$ through $f10$), the EMSA algorithm consistently shows superior performance. Furthermore, the average convergence values achieved by EMSA surpass those of MSA, highlighting EMSA's superior convergence stability. This stability is consistently observed across various trials, including those with 30 and 90 dimensions.

Furthermore, as depicted in Figures 2 and 3, we present the results of EMSA in comparison to established algorithms like ALO, DA, and MBO. Figure 2 shows the effectiveness of the Penalized (10) function, conducted at dimensions of 90. This graphical representation highlights EMSA's superiority over MSA in achieving optimal performance, characterized by faster convergence speed and superior results. These conclusions align with prior studies [13], [6], emphasizing the importance of comprehensive comparisons with existing methodologies to keep pace with the constant evolution of optimization algorithms.

To address this need, we evaluated a newly proposed algorithm against a subset of established algorithms. Figure 3 presents outcomes for the Rosenbrock function (4) across 90 dimensions, revealing

EMSA's outperformance of conventional MSA and several other metaheuristic algorithms including DA, MBO, and ALO, in terms of both convergence speed and overall optimization effectiveness. The results in Figure 3 show EMSA's superior performance compared to the other methods, with MSA closely following as the second-best performer. Performing EMSA with a random mutation operator during optimization yielded better results compared to other general conventional algorithms.

Table 2. Comparative performance analysis of EMSA against established algorithms

F	D	ALO	DA	MBO	MSA	EMSA	F	ALO	DA	MBO	MSA	EMSA
1	30	7.130E+01	1.300E+02	2.210E+01	2.230E+01	1.020E+01	6	6.730E+01	1.230E+02	3.970E+01	5.480E+01	1.230E+01
	90	1.720E+02	3.090E+02	4.810E+01	4.840E+01	1.700E+01		1.740E+02	3.070E+02	5.100E+01	4.970E+01	1.760E+01
2	30	6.730E+01	1.250E+02	2.150E+01	2.240E+01	9.440E+00	7	5.000E+01	8.830E+01	2.470E+01	2.520E+01	9.700E+00
	90	1.750E+02	3.030E+02	4.830E+01	4.850E+01	1.740E+01		8.850E+01	1.870E+02	3.910E+01	3.540E+01	1.240E+01
3	30	6.450E+01	1.330E+02	2.170E+01	2.140E+01	9.430E+00	8	6.580E+01	1.270E+02	2.210E+01	2.190E+01	1.010E+01
	90	1.750E+02	3.150E+02	4.990E+01	4.790E+01	1.690E+01		1.770E+02	3.050E+02	5.010E+01	4.880E+01	1.780E+01
4	30	6.690E+01	1.290E+02	2.230E+01	2.180E+01	9.480E+00	9	6.580E+01	1.200E+02	2.180E+01	2.140E+01	9.360E+00
	90	1.750E+02	3.040E+02	4.880E+01	4.830E+01	1.700E+01		1.710E+02	2.950E+02	4.830E+01	4.820E+01	1.690E+01
5	30	6.500E+01	1.310E+02	2.120E+01	2.120E+01	9.190E+00	10	6.810E+01	1.270E+02	2.180E+01	2.210E+01	9.410E+00
	90	1.720E+02	3.120E+02	4.630E+01	4.930E+01	1.650E+01		1.730E+02	3.040E+02	4.900E+01	4.830E+01	1.760E+01

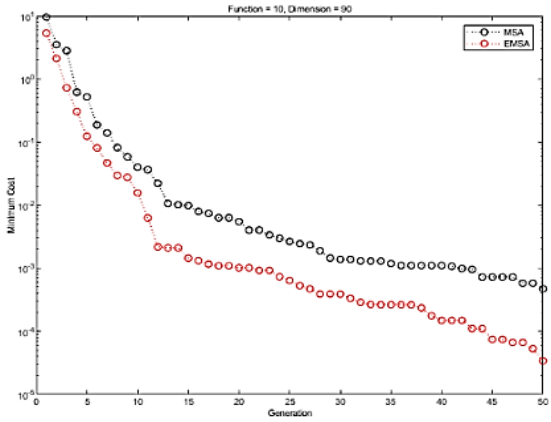


Figure 2. Convergence curves of the best objective function of EMSA and MSA algorithms for Function 10 Dimension 90

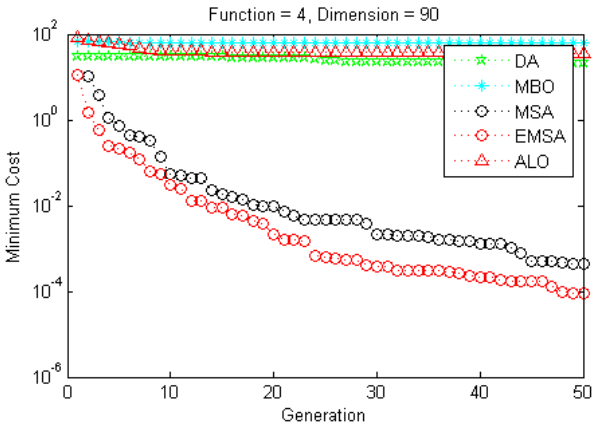


Figure 3. Convergence curves of the best objective function of EMSA and established algorithms for Function 4, Dimension 90

5. CONCLUSION

This article has introduced a random mutation operator mechanism aimed at improving MSA. This mechanism has been integrated into the original MSA to boost population diversity and increase the randomness of the search agent's (moth's) movement. This integration is intended to facilitate a more effective exploration of the solution space. The effectiveness of this mutation operator is assessed across ten benchmark test functions, revealing significant performance enhancements in EMSA. In general, the outcomes of all conducted experiments demonstrated that the mutation operator has the capability to greatly enhance the performance of MSA. Comparative analysis against established algorithms like ALO, DA, and MBO consistently demonstrates EMSA's competitive advantages across most benchmark functions. The primary factor behind the enhanced performance is EMSA's ability to strike a more optimal balance between exploration and exploitation, thereby preventing the algorithm from becoming trapped in local optima during the optimization process. The study also highlights potential applications of EMSA in training artificial neural networks for tasks like medical diagnosis, face recognition, and botnet detection, presenting exciting opportunities for further research.

ACKNOWLEDGEMENTS

This research was supported by the Center of Research Excellence and Incubation Management (CRIEM) of Universiti Sultan Zainal Abidin, Also partially supported by the Universiti Malaysia Terengganu (UMT/TAPE RG 2020/55225), Terengganu, Malaysia.

REFERENCES

[1] M. Baghel, Shikha Agrawal, and S. Silakari, "Survey of metaheuristic algorithms for combinatorial optimization," *International Journal of Computer Applications*, vol. 58, no. 19, pp. 21–31, 2012, doi: 10.5120/9391-3813.

[2] Z. Zhang, C. Huang, K. Dong, and H. Huang, "Birds foraging search: a novel population-based algorithm for global optimization," *Memetic Computing*, vol. 11, no. 3, pp. 221–250, 2019, doi: 10.1007/s12293-019-00286-1.

[3] G. G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Computing*, vol. 10, no. 2, pp. 151–164, 2018, doi: 10.1007/s12293-016-0212-3.

[4] S. A. A. Ghaleb, M. Mohamad, E. F. H. Syed Abdullah, and W. A. H. M. Ghanem, "Integrating mutation operator into grasshopper optimization algorithm for global optimization," *Soft Computing*, vol. 25, no. 13, pp. 8281–8324, 2021, doi: 10.1007/s00500-021-05752-y.

[5] A. A. A. Asmael and B. Al-Nedawe, "Energy efficient WSN using hybrid modification PEGASIS with ant lion optimization," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 23, no. 1, pp. 273–284, 2021, doi: 10.11591/ijeecs.v23.i1.pp273-284.

[6] H. Kilic, U. Yuzgec, and C. Karakuzu, "A novel improved antlion optimizer algorithm and its comparative performance," *Neural Computing and Applications*, vol. 32, no. 8, pp. 3803–3824, 2020, doi: 10.1007/s00521-018-3871-9.

[7] S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Computing and Applications*, vol. 27, no. 4, pp. 1053–1073, 2016, doi: 10.1007/s00521-015-1920-1.

[8] S. M. C. Sapul, R. Setthawong, and P. Setthawong, "New hybrid flower pollination algorithm with dragonfly algorithm and jaccard index to enhance solving university course timetable problem," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 20, no. 3, pp. 1556–1568, 2020, doi: 10.11591/ijeecs.v20.i3.pp1556-1568.

[9] P. Soltani and E. Hadavandi, "A monarch butterfly optimization-based neural network simulator for prediction of siro-spun yarn tenacity," *Soft Computing*, vol. 23, no. 20, pp. 10521–10535, 2019, doi: 10.1007/s00500-018-3624-9.

[10] H. Faris, I. Aljarah, and S. Mirjalili, "Improved monarch butterfly optimization for unconstrained global search and neural network training," *Applied Intelligence*, vol. 48, no. 2, pp. 445–464, 2018, doi: 10.1007/s10489-017-0967-3.

[11] Y. Wang, Q. Luo, and Y. Zhou, "Improved grey wolf optimizer for multiple unmanned aerial vehicles task allocation," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 30, no. 1, pp. 577–585, 2023, doi: 10.11591/ijeecs.v30.i1.pp577-585.

[12] Y. Y. Koay, J. D. Tan, C. W. Lim, S. P. Koh, S. K. Tiong, and K. Ali, "An adaptive gravitational search algorithm for global optimization," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 16, no. 2, pp. 724–729, 2019, doi: 10.11591/ijeecs.v16.i2.pp724-729.

[13] G. Huang, B. He, F. Meng, and D. Rodriguez, "Evaluation of a multi-objective model in energy generation under the influence of different hydrological conditions based on Moth Search Algorithm," *International Journal of Ambient Energy*, vol. 43, no. 1, pp. 3888–3899, 2022, doi: 10.1080/01430750.2020.1861091.

[14] I. Strumberger, E. Tuba, N. Bacanin, and M. Tuba, "Modified moth search algorithm for portfolio optimization," *Smart Innovation, Systems and Technologies*, vol. 165, no. July, pp. 445–453, 2020, doi: 10.1007/978-981-15-0077-0_45.

[15] Y. Feng and G. G. Wang, "A binary moth search algorithm based on self-learning for multidimensional knapsack problems," *Future Generation Computer Systems*, vol. 126, pp. 48–64, 2022, doi: 10.1016/j.future.2021.07.033.

[16] Y. H. Feng and G. G. Wang, "Binary moth search algorithm for discounted 0-1 knapsack problem," *IEEE Access*, vol. 6, pp. 10708–10719, 2018, doi: 10.1109/ACCESS.2018.2809445.

[17] M. Mariyappan and I. Veluchamy, "Game model-combined improved moth search approach for reconfigurable asymmetric multi-processor system-on-chip architecture," *Engineering Optimization*, vol. 55, no. 4, pp. 685–702, 2023.

[18] J. Li, Y. H. Yang, Q. An, H. Lei, Q. Deng, and G. G. Wang, "Moth search: variants, hybrids, and applications," *Mathematics*, vol. 10, no. 21, pp. 1–19, 2022, doi: 10.3390/math10214162.

[19] Y. Feng, H. An, and X. Gao, "The importance of transfer function in solving set-union knapsack problem based on discrete moth search algorithm," *Mathematics*, vol. 7, no. 1, p. 17, 2018, doi: 10.3390/math7010017.

[20] H. Chaudhary and R. Banati, "Improving convergence in swarm algorithms by controlling range of random movement," *Natural Computing*, vol. 20, no. 3, pp. 513–560, 2021, doi: 10.11591/ijeecs.v99.i1.pp1-1x.

[21] B. Hartke, "Global optimization," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 1, no. 6, pp. 879–887, 2011, doi: 10.1002/wcms.70.

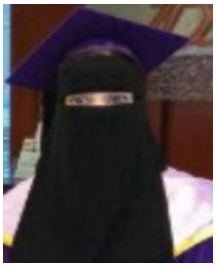
[22] W. A. H. M. Ghanem and A. Jantan, "An enhanced Bat algorithm with mutation operator for numerical optimization problems," *Neural Computing and Applications*, vol. 31, pp. 617–651, 2019, doi: 10.1007/s00521-017-3021-9.

[23] W. A. H. M. Ghanem and A. Jantan, "Hybridizing bat algorithm with modified pitch adjustment operator for numerical optimization problems," *EAI/Springer Innovations in Communication and Computing*, vol. 10, no. 46, pp. 1–39, 2018, doi: 10.1007/978-3-319-70542-2_5.

[24] W. A. H. M. Ghanem and A. Jantan, "A novel hybrid artificial bee colony with monarch butterfly optimization for global optimization problems," *EAI/Springer Innovations in Communication and Computing*, pp. 27–38, 2018, doi: 10.1007/978-3-319-70542-2_3.

[25] M. A. Elaziz, S. Xiong, K. P. N. Jayasena, and L. Li, "Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution," *Knowledge-Based Systems*, vol. 169, pp. 39–52, 2019, doi: 10.1016/j.knosys.2019.01.023.

BIOGRAPHIES OF AUTHORS



Sanaa A. A. Ghaleb received the bachelor's degree from the University of Aden, Yemen, in 2011, and the M. Sc degree from Universiti Sains Malaysia, Malaysia, in 2017. She received the Ph.D. degree from the Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Malaysia. Her research interests include technology-enhanced learning, instructional design and technology, computer networks and information security, cybersecurity, machine learning, artificial intelligence, swarm intelligence, and metaheuristic. She can be contacted at email: sanaaghaleb.sg@gmail.com.

Mumtazimah Mohamad was born in Terengganu, Malaysia. She received the bachelor's degree in information technology from Universiti Kebangsaan Malaysia, in 2000, the M.Sc. degree in computer science from Universiti Putra Malaysia, and the Ph.D. degree in computer science from Universiti Malaysia Terengganu, in 2014. She was a Junior Lecturer, in 2000. Currently, she is an Associate Professor with the Department of Computer Science, Faculty of Informatics and Computing (FIK), Universiti Sultan Zainal Abidin, Terengganu, Malaysia. She has published over 50 research articles in peer-reviewed journals, book chapters, and proceeding. She has appointed a reviewer and technical committee for many conferences and journals and worked as a researcher in several national funded research and development projects. Her research interests include pattern recognition, machine learning, artificial intelligence, and parallel processing. She can be contacted at email: mumtaz@unisza.edu.my.

Waheed Ali Hussein Mohammed Ghanem received the B.Sc. degree in computer sciences and engineering from Aden University, Yemen, in 2003, and the M.Sc. degree in computer science and the Ph.D. degree in network and communication protocols from Universiti Sains Malaysia, in 2013 and 2019, respectively. His research interests include computer and network security, cybersecurity, machine learning, artificial intelligence, swarm intelligence, optimization algorithms, and information technology. He can be contacted at email: waheedghanem@umt.edu.my.

Arifah Che Alhadi received her BSc with honors in Information Science from Universiti Kebangsaan Malaysia in 2001. She obtained her MSc in Information Technology from the same university in 2005. Additionally, she earned her Ph.D. in Computer Science from Universiti Malaysia Terengganu in 2019. She currently holds a position as a senior lecturer in the Faculty of Computer Science and Mathematics at Universiti Malaysia Terengganu, Malaysia. Her research interests encompass information retrieval and information systems. She can be contacted at email: arifah_hadi@umt.edu.my.

Abdullah B. Nasser received the B.Sc. degree from Hodeidah University, Yemen, in 2006, the M.Sc. degree from Universiti Sains Malaysia, Malaysia, in 2014, and the Ph.D. degree in computer science (software engineering) from Universiti Malaysia Pahang, in 2018. He is currently a University Lecturer in programming and software engineering with the, Finland. He is the author of many scientific papers published in renowned journals and conferences. His research interests include optimization algorithms, software testing, and AI, including software defect prediction and feature selection. He can be contacted at email: nasser.abdullah@uwasa.fi.

Hanan Aldowah is currently a senior lecturer at Universiti Sains Malaysia (USM), where she has earned both a Ph.D. and a Master's degree. Her extensive expertise spans diverse domains, including data analytics, data mining, machine learning, artificial intelligence, internet of things, engineering pedagogical research, cognitive maps, information science, E-learning, VR and augmented reality, and instructional technology. Aldowah's impactful research operates at the intersection of computer science and educational technology. She can be contacted at email: hanan@usm.my.