Konya Mühendislik Bilimleri Dergisi, c. 11, s. 2, 449-467, 2023
Konya Journal of Engineering Sciences, v. 11, n. 2, 449-467, 2023
ISSN: 2667-8055 (Electronic)
DOI: 10.36306/konjes.1165964

Bin_MRFOA: A NOVEL MANTA RAY FORAGING OPTIMIZATION ALGORITHM FOR BINARY OPTIMIZATION

* **Gülnur YILDIZDAN**

Selcuk University, Kulu Vocational School, Konya, TÜRKİYE
gavsar@selcuk.edu.tr

Highlights

- A new binary version of the manta ray foraging optimization algorithm (Bin_MRFOA) was proposed for solving binary optimization problems.
- Bin_MRFOA was tested for eight different transfer functions on the classic test function, and the effect of transfer functions on performance was examined.
- The most successful version of Bin_MRFOA was run on eighteen CEC2005 benchmark functions, and the results were compared with the algorithms in the literature.
- The results revealed that Bin_MRFOA is a successful, competitive, and preferable algorithm.

***Corresponding Author:** Gülnur YILDIZDAN, gavsar@selcuk.edu.tr

Konya Mühendislik Bilimleri Dergisi, c. 11, s. 2, 449-467, 2023
Konya Journal of Engineering Sciences, v. 11, n. 2, 449-467, 2023
ISSN: 2667-8055 (Electronic)
DOI: 10.36306/konjes.1165964

Bin_MRFOA: A NOVEL MANTA RAY FORAGING OPTIMIZATION ALGORITHM FOR BINARY OPTIMIZATION

* Gülnur YILDIZDAN

Selcuk University, Kulu Vocational School, Konya, TÜRKİYE
gavsar@selcuk.edu.tr

(Geliş/Received: 23.08.2022; Kabul/Accepted in Revised Form: 02.03.2023)

ABSTRACT: Optimization problems occur in three different structures: continuous, discrete, and hybrid. Metaheuristic algorithms, which are frequently preferred in the solution of optimization problems today, are mostly proposed for continuous problems and are discretized with subsequent modifications. In this study, a novel binary version (Bin_MRFOA) of the manta ray foraging optimization algorithm, which was frequently used in the solution of continuous optimization problems before, was proposed to be used in the solution of binary optimization problems. The Bin_MRFOA was first tested on ten classical benchmark functions, and the effect of the transfer function on performance was examined by comparing the variants obtained using eight different transfer functions. Then the most successful Bin_MRFOA variant was run on the eighteen CEC2005 benchmark functions. The results were compared with the algorithms in the literature and interpreted with Wilcoxon signed-rank and Friedman tests, which are nonparametric tests. The results revealed that Bin_MRFOA is a successful, competitive, and preferable algorithm compared to the literature.

Keywords: *Binary Optimization, Manta Ray Foraging Optimization, S-shaped and V-shaped Transfer Functions*

1. INTRODUCTION

Today, evolutionary computing has become an effective method of choice for solving complex optimization problems. Evolutionary computation is handled in two groups, evolutionary algorithms (EA) and swarm intelligence-based algorithms. EAs are nature-inspired algorithms (genetic algorithm, differential evolution algorithm, etc.), while swarm intelligence-based algorithms (particle swarm algorithm, bat algorithm, etc.) are inspired by the social behavior of animals to solve problems. Optimization problems appear in three different structures: continuous, discrete, or hybrid. Continuous problems can take an infinite number of input values in a given range and produce an infinite number of output values in response to these inputs. A binary optimization problem is represented as a binary-based problem space, and it is a type of combinatorial optimization problem. [1]. In continuous optimization, the search space is continuous and the search agents receive continuous values. In binary optimization, search agents scattered throughout the search space take the value "0" to represent absence and "1" to represent presence.

Many studies have been published in the literature to propose binary versions of metaheuristic algorithms. Korkmaz et al. developed the basic artificial algae algorithm using a new solution update rule and used it for solving binary optimization problems [2]. Wang et al. proposed the binary version of the chimpanzee optimization algorithm and used the proposed algorithm for the continuous optimization task [3]. Al-Tashi et al. proposed a binary algorithm using a hybrid of gray wolf optimization and particle swarm optimization for use in solving feature selection problems. [4]. Baş and Ülker proposed and used a binary version of the social spider algorithm for continuous optimization [5]. Aslan et al. proposed a novel optimizer predicated on the Jaya algorithm and the xor logic operator and tested this algorithm on the CEC 2015 benchmark functions and uncapacitated facility location problems [6]. Hussein et al. proposed an adaptation of the original whale optimization algorithm to deal with binary optimization problems [7]. Çınar and Kıran presented a new tree seed algorithm for binary optimization problems developed using logic gates and similarity measurement techniques. [8]. Rizk-Allah et al. proposed a novel binary version of the salp swarm algorithm based on a developed arctan transform so that the salp swarm algorithm can be adapted to binary problems [9]. Arora and Anand developed a binary variant of the butterfly optimization algorithm and used it to select feature subsets for classification [10]. Mafarja et al. proposed a binary dragonfly-based wrapper-feature selection algorithm. The algorithm was tested with eighteen benchmark datasets and eight different transfer functions mapping the continuous search space to the discrete search space [11]. Akan et al. presented a binary variant of the battle royale optimization algorithm [12]. Abdel-Basset et al. introduced a binary version of the marine predator optimization algorithm using a wide variety of transfer functions to map continuous values to binary [13]. Kaya proposed the binary galactic swarm optimization algorithm, which employs the binary artificial algae algorithm as the primary search algorithm [14]. Chauhan and Yadav designed a new binary variant of the artificial electric field algorithm to enhance its performance in discrete problems [15]. Şahman and Çınar mapped the tree seed algorithm to binary search space with the help of transfer functions to solve binary optimization problems. The proposed algorithm was used to solve the uncapacitated facility location problems for different sizes [16]. Dehghani et al. proposed the Binary Spring Search Algorithm based on the simulation of Hooke's Law (physics) for the conventional system of weights and springs to solve binary problems. The performance of the proposed algorithm was extensively validated for functions with unimodal and multimodal features [17]. Beheshti proposed the x-shaped binary PSO algorithm using a new x-shaped transfer function to enhance the exploration and exploitation capabilities of binary PSO in binary search space. The proposed algorithm was run on the 0–1 multidimensional knapsack problems, maximization functions, and minimization functions [18]. Kalra et al. presented the binary Emperor Penguin Optimizer algorithm for efficient solution of binary nature problems, leveraging the power of the standard Emperor Penguin Optimizer. The performance of the algorithm is evaluated over twenty-nine benchmark functions and binary feature selection problem [19]. Chantar et al. proposed an advanced binary grey wolf optimizer within a wrapper feature selection approach for solving Arabic text classification problems. The binary grey wolf optimizer was used in this algorithm as a wrapper-based method of feature selection [20]. Nadimi-Shahraki et al. presented a wrapper feature selection approach based on the Aquila optimizer. The proposed S-shaped binary Aquila optimizer and V-shaped binary Aquila optimizer were used for feature selection in medical datasets and real COVID-19 datasets [21]. He et al. proposed a new binary differential evolution algorithm based on taper-shaped transfer functions. The algorithm was used to solve the knapsack problem and the uncapacity facility location problem [22]. Hakli proposed a new binary algorithm based on the elephant herding optimization algorithm in order to develop a powerful algorithm that can deal with binary problems. The proposed method was applied to the problems of 0-1 knapsack, uncapacitated facility location, and wind turbine placement [23]. Pourrajabian et al. investigated the robustness and accuracy of the continuous and binary genetic algorithm approaches for the wind turbine blade design problem [24]. Mohammadzadeh and Gharehchopogh presented three efficient binary methods based on the Symbiotic Organism Search algorithm to solve the feature selection problem. In these methods, the S-shaped transfer function, the V-shaped transfer function, and two new operators named binary mutualism and binary commensalism were used to make the algorithm binary. The proposed methods were tested on the standard UCI dataset and on the spam e-mail dataset [25]. Ghosh et al. proposed a new feature selection approach based on the Manta ray foraging optimization algorithm, which models the foraging behavior of manta rays [26]. Feng and Wang proposed a self-learning-based binary moth search algorithm for solving multidimensional knapsack problems [27]. Xi et al. proposed the binary African Vulture Optimization Algorithm to solve discrete optimization problems. This algorithm uses the X-shaped transfer function. The algorithm was tested on benchmark problems, engineering problems, and the uncapacitated facility location problem [28].

The performances of binary optimization algorithms proposed in the literature are also frequently investigated when optimizing continuous functions [29-31]. Evolutionary computing or swarm intelligence algorithms can be used to optimize functions with continuous decision variables. However, because the problem can be generalized as desired and the optimum solutions are known, comparing binary optimization methods with each other on these problems is simple and informative. Therefore, most researchers also test the performance of binary optimization algorithms on such continuous functions [32]. With this motivation, in this study, the manta ray search optimization algorithm was discretized to be used in the solution of binary problems, and the binary version of the algorithm was proposed. The proposed Bin_MRFOA algorithm was tested on continuous problems such as CEC2005 and classical benchmark functions. The remaining of this paper is organized as follows: The material and method are detailed in Section 2. Experimental results obtained from Bin_MRFOA on classical and CEC2005 benchmark functions are shown in Section 3. The discussion and conclusions about Bin_MRFOA are presented in Section 4.

2. MATERIAL AND METHOD

2.1. Manta Ray Foraging Optimization Algorithm (MRFOA)

Manta rays are organisms that feed on plankton, which is a type of aquatic microfauna. They use the angular heads of their mouths to absorb water and prey during feeding and have upgraded rabbles to filter their prey out of the water. Manta rays are creatures that work in an organized way to find the best food. The manta ray foraging optimization algorithm was created based on the manta rays' mentioned features. This algorithm, proposed by Zhao et al. in 2020, mimics three different search strategies: chain, cyclone, and tumble foraging [33].

MRFOA, like many meta-heuristic algorithms, the initialization step is randomly created as given in Equation 1.

$$X_i^d = Lb_i^d + rand * \left(Ub_i^d - Lb_i^d\right) \qquad i = 1, \dots, N \qquad d = 1, \dots, D \tag{1}$$

where D is the number of dimensions, N is the population size, and Lb and Ub are the lower and upper limits for the dimensions.

2.1.1. Chain foraging

In MRFOA, manta rays can detect a plankton's position and swim towards it. The higher the plankton density in a location, the better the location, and manta rays are thought to tend to be directed towards high-density areas. Manta rays form a foraging chain by aligning themselves from head to tail for this reason. With the exception of the first individual, they have a tendency to look both at the food and at the individual in front of them. Each individual is updated after each iteration using both the solution that came before it and the best solution thus far. The mathematical model of chain foraging is expressed as given in Equation 2.

$$X_{i,d}^{t+1} = \begin{cases} X_{best,t}^t + r * \left(X_{best,d}^t - X_{i,d}^t\right) + \alpha * \left(X_{best,d}^t - X_{i,d}^t\right) & i = 1 \\ X_{i,d}^t + r * \left(X_{i-1,\mathrm{d}}^t - X_{i,d}^t\right) + \alpha * \left(X_{best,d}^t - X_{i,d}^t\right) & i = 2, \dots, N \end{cases} \tag{2}$$

$$\alpha = 2 * r * \sqrt{|\log(r)|} \tag{3}$$

In Equation 2, r is a vector consisting of random numbers in the interval $[0,1]$, α is the weight coefficient given in Equation 3, $X_{i,d}^t$ is the position of the ith individual at time t of the dth dimension, and $X_{best,d}^t$ is the high-density best location.

2.1.2. Cyclone foraging

When a swarm of manta rays spots a piece of plankton in deep water, in addition to spiraling toward the piece of plankton, each manta ray swims towards the one in front of it. In other words, swarms of manta rays move in a spiral and line up. The mathematical model of the aforementioned cyclone foraging is as given in Equation 4.

$$X_{i,d}^{t+1} = \begin{cases} X_{best,d}^{t} + r * \left(X_{best,d}^{t} - X_{i,d}^{t}\right) + \beta * \left(X_{best,d}^{t} - X_{i,d}^{t}\right) & i = 1 \\ X_{best}^{d}(t) + r * \left(X_{i-1,\mathrm{d}}^{t} - X_{i,d}^{t}\right) + \beta * \left(X_{best,d}^{t} - X_{i,d}^{t}\right) & i = 2, \ldots., N \end{cases} \tag{4}$$

$$\beta = 2e^{r1\frac{T-t+1}{T}} * \sin(2\pi r1) \tag{5}$$

where β is the weight coefficient given in Equation 5, T denotes the maximum number of iterations, and r1 is a random number between 0 and 1.

Furthermore, to improve population search and exploration capabilities, MRFOA generates a new location at random during the optimization process and then makes a spiral search at that location. As a result, MRFOA conducts a comprehensive global search, and its mathematical model is shown in Equation 7.

$$X_{r,d}^{t} = Lb^{d} + r * \left(Ub^{d} - Lb^{d}\right) \tag{6}$$

$$X_{i,d}^{t+1} = \begin{cases} X_{r,d}^{t} + r * \left(X_{r,d}^{t} - X_{i,d}^{t}\right) + \beta * \left(X_{r,d}^{t} - X_{i,d}^{t}\right) & i = 1 \\ X_{r,d}^{t} + r * \left(X_{i-1,\mathrm{d}}^{t} - X_{i,d}^{t}\right) + \beta * \left(X_{r,d}^{t} - X_{i,d}^{t}\right) & i = 2, \ldots., N \end{cases} \tag{7}$$

where $X_{r,d}^{t}$ represents a random location in the search space(Equation 6), Lb^{d} and Ub^{d} are the lower and upper limit values for the dth dimension, respectively.

2.1.3. Somersault foraging

Each individual swims back and forth around a food location viewed as a pivot and somersaults to a new position in this strategy. As a result, individuals update their location based on the best location found thus far. Equation 8 is used to create a mathematical model of this behavior.

$$X_{i,d}^{t+1} = X_{i,d}^{t} + S * \left(r2 * X_{best,d}^{t} - r3 * X_{i,d}^{t}\right), \qquad i = 1, \ldots, N \tag{8}$$

In the Equation, $r2$ and $r3$ are two random numbers in the interval [0, 1]. S is a somersault factor, which determines the somersault distance.
The pseudocode of MRFOA is given in Figure 1.

```
1.  Determine $T_{max}$, $N, Ub, Lb, t = 1$
2.  Initialize population according to Equation 1.
3.  Compute the fitness of each individual $f(X_i)$
4.  Obtain best individual $(X_{best})$
5.    While stop criterion is not satisfied  do
6.        For  i=1 to N
7.          If  rand < 0.5 then              // Cyclone foraging
8.              If   t / Tmax < rand    then
9.                  Generate a random individual($X_r(t)$) according to Equation 6.
10.                 Create a new candidate individual ($X_i(t+1)$) according to Equation 7.
11.             Else
12.                 Create a new candidate individual ($X_i(t+1)$) according to Equation 4.
13.             End If
14.         Else            //Chain foraging
15.             Create a new candidate individual ($X_i(t+1)$) according to Equation 2.
16.         End If
17.         Compute the fitness of each individual $f(X_i(t+1))$
18.           If  $f(X_i(t+1)) < f(X_{best})$ then
19.             $f(X_{best}) = f(X_i(t+1))$
20.           End If
21.       End For
22.       For i=1 to N       //Somersault foraging
23.         Create a new candidate individual ($X_i(t+1)$)according to Equation 8.
24.         Compute the fitness of the individual $f(X_i(t+1))$
25.           If  $f(X_i(t+1)) < f(X_{best})$ then
26.             $f(X_{best}) = f(X_i(t+1))$
27.           End If
28.       End For
29.    End While
```

Figure 1. Pseudocode of MRFOA

2.2. Binary Manta Ray Foraging Optimization Algorithm (Bin_MRFOA)

The manta ray search optimization algorithm is a swarm intelligence-based algorithm proposed for continuous optimization, with advantages such as strong global search capability, few parameters to adjust, and robustness [34]. The objective of this study is to present a binary MRFOA for solving continuous optimization tasks. MRFOA was chosen for this study because it is a new heuristic algorithm and binary MRFOA studies are rare in the literature. Initially, a manta ray population of N individuals is created. Each manta ray has n dimensions created with binary values generated using Equation 9. It randomly generates 0 or 1 for each ith position of each manta ray (X_i). The dth dimension is given a value of 1 if a random number in the range of (0,1) is greater than 0.5, otherwise a value of 0.

$$X_i^d = \begin{cases} 1 & if\ rand(0,1) > 0.5 \\ 0 & otherwise \end{cases} \tag{9}$$

The decision variables of the comparison functions used in this study take continuous values. For this reason, in the proposed Bin_MRFOA, binary values are converted to continuous values to solve these problems and calculate the fitness values of each individual. This conversion is done according to the equation given in Equation 10.

$$C_j = Low_j + \frac{(High_j - Low_j)DecValue_j}{MaxValue} \tag{10}$$

In Equation 10, while Cj denotes the continuous value of the jth dimension of the individual, and $High_j$ and Low_j denote the upper and lower bound values of the jth dimension. $DecValue_j$ indicates the integer value in the decimal number system of the binary number in the jth dimension. $MaxValue$

represents the maximum decimal integer value that the binary number can take, according to the bit length determined for each dimension [32]. For example, assuming that the binary values are represented by 4 bits, the binary value in the jth dimension is "1110" and the continuous value of the dimension is in the range of [-2, +2], the continuous value conversion is calculated as follows:

$$C_j = -2 + \frac{(2-(-2)) \times 14}{2^4 - 1} = -2 + \frac{56}{15} = 1.73$$

The transfer function is the most important aspect of binary optimization. Eight different transfer functions are used in this study to map MRFOA's continuous search space to the binary search space. Transfer functions use a real value as input and normalize it to a value between 0 and 1 using one of the equations in Table 1 [35-37]. Equation 11 is used to convert this number between 0 and 1 to a binary value.

$$F(a) = \begin{cases} 1 & if\ F(a) > rand(0,1) \\ 0 & otherwise \end{cases} \tag{11}$$

Table 1. Transfer functions

V-Shaped		S-Shaped	
V1	$F(a) = \left\lvert \frac{2}{\pi} arcTan\left(\frac{\pi}{2}a\right) \right\rvert$	S1	$F(a) = \frac{1}{1+e^{-a}}$
V2	$F(a) = \lvert \tanh(a) \rvert$	S2	$F(a) = \frac{1}{1+e^{-2*a}}$
V3	$F(a) = \left\lvert \frac{a}{\sqrt{1+a^2}} \right\rvert$	S3	$F(a) = \frac{1}{1+e^{-\frac{a}{2}}}$
V4	$F(a) = \left\lvert \mathrm{erf}\left(\frac{\sqrt{\pi}}{2} a\right) \right\rvert$	S4	$F(a) = \frac{1}{1+e^{-\frac{a}{3}}}$

Binary solution space is well suited for logic gates with binary input and output values. Exclusive-or (Xor) gate is often used in logic circuits. According to the truth table given in Table 2, the probability of the output being 0 or 1 in the Xor gate (⊕) is equal. Because of this feature, it supports diversity and is frequently used in the literature[6, 8]. In this study, the Xor gate was employed according to the formula given in Equation 12 for candidate solution(CX_i) generation.

$$CX_i^d = X_i^d \oplus X_{best}^d \tag{12}$$

Table 2. Xor truth table

X_i^d	X_k^d	$X_i^d \oplus X_k^d$
0	0	0
0	1	1
1	0	1
1	1	0
	50%=0	50%=1

The pseudo-code of Bin_MRFOA is shown in Figure 2.

```
1.  Determine $T_{max}, N, Ub, Lb, t = 1$
2.  Initialize population according to Equation 9.
3.  Convert binary values to continuous values according to Equation 10.
4.  Compute the fitness value $f(X_i)$ of each individual in the population.
5.  Obtain best individual $(X_{best})$
6.   While stop criterion is not satisfied  do
7.        For  i=1 to N
8.          If  rand < 0.5 then              // Cyclone foraging
9.              If   t / Tmax < rand     then
10.                 Generate a random individual$(X_r(t))$ according to Equation 6.
11.                 Create a new candidate individual $(X_i(t+1))$ according to Equation 7.
12.             Else
13.                 Create a new candidate individual $(X_i(t+1))$ according to Equation 4.
14.             End If
15.           Else           //Chain foraging
16.             Create a new candidate individual $(X_i(t+1))$ according to Equation 2.
17.           End If
18.           Convert $X_i(t+1)$ to binary form with the help of selected transfer functions.
19.           Generate candidate solution using XOR gate according to Equation 12.
20.           Convert binary values to continuous values according to Equation 10.
21.           Compute the fitness of candidate individual $f(X_i(t+1))$
22.            If  $f(X_i(t+1)) < f(X_{best})$ then
23.              $f(X_{best}) = f(X_i(t+1))$
24.            End If
25.         End For
26.        For i=1 to N       //Somersault foraging
27.           Create a new candidate individual $(X_i(t+1))$according to Equation 8.
28.           Convert $X_i(t+1)$ to binary form with the help of selected transfer functions.
29.           Generate candidate solution using XOR gate according to Equation 12.
30.           Convert binary values to continuous values according to Equation 10.
31.           Compute the fitness of the candidate individual $f(X_i(t+1))$
32.            If  $f(X_i(t+1)) < f(X_{best})$ then
33.                $f(X_{best}) = f(X_i(t+1))$
34.            End If
35.         End For
36.    End While
```

Figure 2. Pseudocode of Bin_MRFOA

3. EXPERIMENTAL RESULTS

In this section, the performance of the proposed Bin_MRFOA on the classical and CEC2005 benchmark functions is examined. Classical functions have been used to determine the ideal population size and the best performing transfer function for the algorithm, and these functions and their properties are given in Table 3. The classical functions used in the testing process consist of a total of ten functions, three of which are multimodal and seven of which are unimodal. For classical functions, the maximum iteration is 500, and the dimension is 10. Each function was run independently 30 times and the mean and standard deviation values of the obtained results were found.

Firstly, the proposed algorithm was run for five different population size (N) values to determine the effect of population size selection on performance. During this test, sigmoid (S1) was used as the transfer function, and the results are given in Table 4. The best mean value obtained for each function was shown in bold in the table. When the results were examined, it was seen that the best mean values for the functions could not be obtained from a single population size value and varied for each function. Therefore, the results obtained for different population values were evaluated statistically, and the Friedman test was applied, which enables sorting between the groups by comparing the mean in cases where the assumption of normality between the dependent groups is not provided [38-40]. The mean rank values obtained as a result of this test, which was applied separately for mean and standard deviation values, are presented comparatively in the graphic in Figure 3. According to the graphic, the smallest mean rank values in both comparisons were obtained when the population size was 50. Based on this result, the population size was taken as 50 in the tests that will be mentioned in the following sections.

Table 3. Classic benchmark functions

Function	Range	f_{min}	Characteristic
$F1(x)=\sum_{i=1}^{d} x_i^2$	$[-100,100]$	0	T
$F2(x)=\sum_{i=1}^{d} x_i^2+\prod_{i=1}^{d}\lvert x_i\rvert$	$[-10,10]$	0	T
$F3(x)=\sum_{i=1}^{d}(\sum_{j=1}^{i} x_j)^2$	$[-100,100]$	0	T
$F4(x)=max_i\{\lvert x_i\rvert, 1\le i\le d\}$	$[-100,100]$	0	T
$F5(x)=\sum_{i=1}^{d-1}[100\,(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	$[-30,30]$	0	T
$F6(x)=\sum_{i=1}^{d}([x_i+0.5])^2$	$[-100,100]$	0	T
$F7(x)=\sum_{i=1}^{d} i*x_i^4+rand[0,1)$	$[-1.28,1.28]$	0	T
$F8=\sum_{i=1}^{d}-x_i\sin(\sqrt{\lvert x_i\rvert})$	$[-500,500]$	−418.9829 ×d	Ç
$F9=\sum_{i=1}^{d}[x_i^2-10\cos(2\pi x_i)+10]$	$[-5.12,5.12]$	0	Ç
$F10(x)=\sum_{i=1}^{d}-20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right)-\exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right)+20+e$	$[-32,32]$	0	Ç

Table 4. Comparative results for various population sizes (N)

Function	N=10		N=20		N=30		N=40		N=50	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*	*Mean*	*Std*	*Mean*	*Std*
F1	4,401E+03	1,348E+03	**4,021E+03**	1,280E+03	4,024E+03	1,140E+03	4,452E+03	9,856E+02	4,444E+03	8,892E+02
F2	**1,549E+01**	2,941E+00	1,665E+01	2,314E+00	1,702E+01	1,912E+00	1,739E+01	2,540E+00	1,697E+01	2,742E+00
F3	4,383E+03	1,285E+03	**4,283E+03**	1,229E+03	4,422E+03	8,495E+02	4,681E+03	7,793E+02	4,349E+03	9,205E+02
F4	3,580E+01	4,858E+00	3,629E+01	3,860E+00	3,540E+01	4,774E+00	3,495E+01	3,680E+00	**3,461E+01**	4,408E+00
F5	**1,460E+06**	8,290E+05	1,838E+06	9,119E+05	1,980E+06	1,034E+06	1,984E+06	9,700E+05	1,746E+06	6,008E+05
F6	**3,486E+03**	1,131E+03	4,211E+03	9,086E+02	3,823E+03	1,094E+03	4,113E+03	9,859E+02	4,181E+03	9,194E+02
F7	8,553E-01	2,245E-01	**7,979E-01**	2,417E-01	9,486E-01	2,829E-01	9,494E-01	3,419E-01	8,522E-01	3,202E-01
F8	-2,396E+03	1,552E+02	-2,400E+03	1,459E+02	-2,447E+03	1,580E+02	**-2,452E+03**	1,832E+02	-2,441E+03	1,532E+02
F9	6,819E+01	5,556E+00	6,860E+01	6,843E+00	6,580E+01	6,651E+00	6,622E+01	8,105E+00	**6,290E+01**	9,100E+00
F10	1,628E+01	7,251E-01	1,625E+01	6,321E-01	1,652E+01	6,671E-01	**1,576E+01**	1,199E+00	1,649E+01	6,230E-01

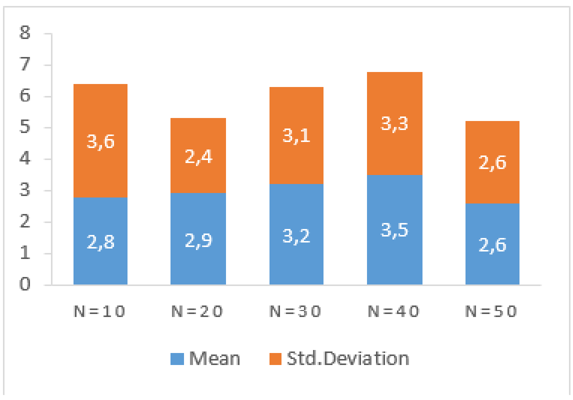


Figure 3. Comparison of Friedman test mean rank values

After determining the ideal population size, variants of the proposed algorithm were created using eight different transfer functions to determine the effect of the selected transfer function on performance and to determine the most successful transfer function. Comparison results of Bin_MRFOA variants are given in Table 5. When the results in the table were examined, it was found that the variant obtained with the V2 transfer function in four of the functions and the variant obtained with the V4 transfer function in the remaining six functions achieved a more successful mean value. It was determined that the best values were found by the variants created with the V2 and V4 transfer functions, except for the F9 function. The sums of the mean and standard deviation values acquired by Bin_MRFOA variants for all functions are compared in the graphics given in Figure 4 and Figure 5, respectively. According to the graphics, it was seen that the smallest sum values were generally obtained by variants created using V-shaped transfer functions, and the lowest sum values were found from the V4 variant in both graphics.

Table 5. Bin_MRFOA variants' statistical results on classical benchmark functions

	Transfer Func	Best	Median	Worst	Mean	Std
$F1$	$S1$	2.969E+03	4.533E+03	5.794E+03	4.562E+03	7.734E+02
	$S2$	2.354E+03	3.597E+03	6.206E+03	3.928E+03	1.035E+03
	$S3$	2.617E+03	4.399E+03	6.477E+03	4.447E+03	8.707E+02
	$S4$	2.093E+03	4.677E+03	5.843E+03	4.431E+03	1.130E+03
	$V1$	9.775E+00	1.399E+02	1.806E+03	2.427E+02	4.058E+02
	$V2$	**7.270E-26**	6.828E+01	6.513E+02	**1.334E+02**	1.681E+02
	$V3$	1.852E+03	3.540E+03	5.073E+03	3.376E+03	9.564E+02
	$V4$	9.241E-01	2.215E+01	1.657E+03	2.179E+02	4.473E+02
$F2$	$S1$	6.752E+00	1.611E+01	2.096E+01	1.608E+01	3.000E+00
	$S2$	1.344E+01	1.632E+01	1.916E+01	1.629E+01	1.735E+00
	$S3$	1.382E+01	1.711E+01	2.146E+01	1.751E+01	1.903E+00
	$S4$	1.067E+01	1.635E+01	2.023E+01	1.664E+01	2.310E+00
	$V1$	**8.882E-14**	1.206E+00	5.054E+00	1.426E+00	1.336E+00
	$V2$	8.457E-11	3.340E-01	6.987E+00	7.399E-01	1.512E+00
	$V3$	5.727E+00	1.144E+01	1.572E+01	1.168E+01	2.577E+00
	$V4$	**8.882E-14**	1.675E-01	4.445E+00	**5.390E-01**	1.110E+00
$F3$	$S1$	1.008E+03	4.385E+03	5.538E+03	4.083E+03	1.253E+03
	$S2$	3.046E+03	4.907E+03	5.974E+03	4.698E+03	7.693E+02
	$S3$	1.987E+03	4.903E+03	6.813E+03	4.676E+03	1.031E+03
	$S4$	2.013E+03	4.287E+03	5.970E+03	4.250E+03	1.067E+03
	$V1$	1.639E+02	1.464E+03	5.440E+03	1.739E+03	1.313E+03

	$V2$	**2.911E-07**	1.115E+03	3.749E+03	1.325E+03	1.151E+03
	$V3$	2.377E+03	4.697E+03	5.376E+03	4.386E+03	8.376E+02
	$V4$	1.424E+02	9.446E+02	3.131E+03	**1.166E+03**	8.025E+02
$F4$	$S1$	3.014E+01	3.604E+01	4.276E+01	3.605E+01	2.932E+00
	$S2$	2.754E+01	3.705E+01	4.528E+01	3.647E+01	4.456E+00
	$S3$	2.145E+01	3.519E+01	4.335E+01	3.537E+01	4.344E+00
	$S4$	2.761E+01	3.258E+01	4.320E+01	3.358E+01	4.561E+00
	$V1$	6.941E+00	1.866E+01	3.542E+01	2.005E+01	7.507E+00
	$V2$	3.308E+00	1.689E+01	3.056E+01	1.617E+01	8.693E+00
	$V3$	2.283E+01	3.393E+01	4.037E+01	3.368E+01	4.874E+00
	$V4$	**2.700E-13**	7.975E+00	2.334E+01	**9.261E+00**	6.448E+00
$F5$	$S1$	5.922E+05	1.712E+06	3.646E+06	1.888E+06	1.008E+06
	$S2$	3.061E+05	1.535E+06	4.363E+06	1.781E+06	9.780E+05
	$S3$	3.065E+05	1.775E+06	3.322E+06	1.692E+06	8.229E+05
	$S4$	6.667E+05	2.071E+06	4.906E+06	2.050E+06	1.079E+06
	$V1$	2.234E+03	5.772E+04	2.308E+06	2.275E+05	5.277E+05
	$V2$	4.984E+02	5.693E+03	2.375E+06	1.325E+05	5.283E+05
	$V3$	3.888E+05	1.158E+06	2.423E+06	1.281E+06	6.055E+05
	$V4$	**2.501E+01**	1.028E+03	7.733E+04	**5.041E+03**	1.709E+04
$F6$	$S1$	1.678E+03	4.209E+03	6.106E+03	4.292E+03	1.165E+03
	$S2$	2.797E+03	4.369E+03	6.474E+03	4.372E+03	1.006E+03
	$S3$	2.229E+03	4.513E+03	5.339E+03	4.234E+03	8.727E+02
	$S4$	2.903E+03	4.097E+03	5.547E+03	4.194E+03	7.752E+02
	$V1$	4.218E+00	1.413E+02	1.718E+03	2.693E+02	4.013E+02
	$V2$	1.827E+00	1.960E+01	9.749E+02	**1.095E+02**	2.338E+02
	$V3$	8.050E+02	3.311E+03	5.322E+03	3.428E+03	1.132E+03
	$V4$	**1.304E+00**	5.812E+01	1.722E+03	1.780E+02	3.852E+02
$F7$	$S1$	3.066E-01	9.120E-01	1.445E+00	9.333E-01	3.088E-01
	$S2$	3.964E-01	8.927E-01	1.351E+00	8.764E-01	2.748E-01
	$S3$	4.031E-01	7.404E-01	1.283E+00	8.074E-01	2.573E-01
	$S4$	3.377E-01	9.161E-01	1.299E+00	8.687E-01	2.453E-01
	$V1$	2.571E-02	8.980E-02	5.158E-01	1.391E-01	1.431E-01
	$V2$	**3.431E-04**	6.138E-02	2.537E-01	8.173E-02	6.613E-02
	$V3$	3.054E-01	6.883E-01	1.016E+00	6.939E-01	1.998E-01
	$V4$	1.113E-02	3.975E-02	4.447E-01	**6.540E-02**	9.410E-02
$F8$	$S1$	-2.682E+03	-2.412E+03	-2.186E+03	-2.408E+03	1.330E+02
	$S2$	-2.896E+03	-2.551E+03	-2.276E+03	-2.559E+03	1.737E+02
	$S3$	-2.741E+03	-2.396E+03	-2.143E+03	-2.424E+03	1.732E+02
	$S4$	-2.928E+03	-2.424E+03	-2.189E+03	-2.430E+03	1.957E+02
	$V1$	-4.027E+03	-3.856E+03	-3.392E+03	-3.825E+03	1.616E+02
	$V2$	-4.123E+03	-4.058E+03	-3.566E+03	**-3.982E+03**	1.566E+02
	$V3$	-3.722E+03	-3.347E+03	-2.721E+03	-3.317E+03	2.612E+02
	$V4$	**-4.160E+03**	-3.973E+03	-3.393E+03	-3.949E+03	1.968E+02
$F9$	$S1$	3.318E+01	6.580E+01	7.638E+01	6.370E+01	9.759E+00
	$S2$	4.597E+01	6.758E+01	7.579E+01	6.347E+01	8.716E+00
	$S3$	4.286E+01	6.738E+01	8.201E+01	6.502E+01	1.029E+01
	$S4$	5.693E+01	6.790E+01	7.685E+01	6.767E+01	5.344E+00

	$V1$	**4.690E-12**	2.245E+01	5.457E+01	2.484E+01	1.403E+01
	$V2$	2.472E+00	7.367E+00	2.246E+01	**1.010E+01**	6.485E+00
	$V3$	3.841E+01	5.726E+01	7.189E+01	5.738E+01	8.095E+00
	$V4$	4.079E-01	8.802E+00	4.062E+01	1.112E+01	9.420E+00
$F10$	$S1$	1.352E+01	1.657E+01	1.729E+01	1.624E+01	9.814E-01
	$S2$	1.180E+01	1.606E+01	1.711E+01	1.588E+01	1.198E+00
	$S3$	1.306E+01	1.630E+01	1.722E+01	1.591E+01	1.309E+00
	$S4$	1.388E+01	1.651E+01	1.753E+01	1.636E+01	8.847E-01
	$V1$	3.265E+00	5.826E+00	1.170E+01	6.343E+00	2.544E+00
	$V2$	1.429E+00	3.956E+00	9.940E+00	4.210E+00	2.015E+00
	$V3$	1.102E+01	1.474E+01	1.650E+01	1.428E+01	1.551E+00
	$V4$	**2.080E-01**	3.177E+00	9.385E+00	**3.864E+00**	2.468E+00

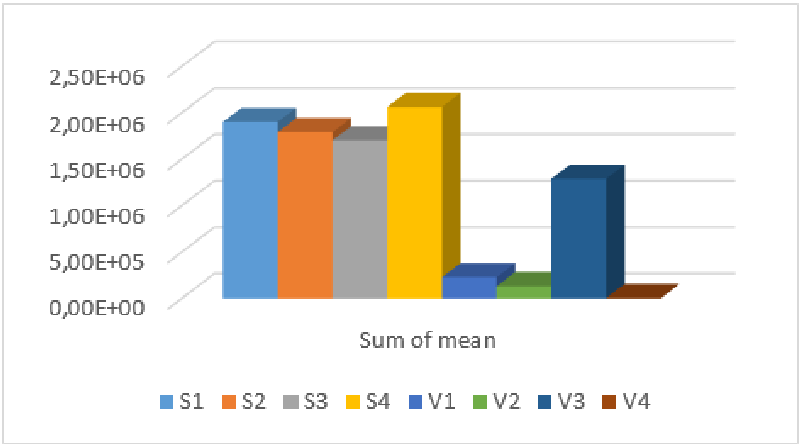


Figure 4. Comparision of the sum of the mean results

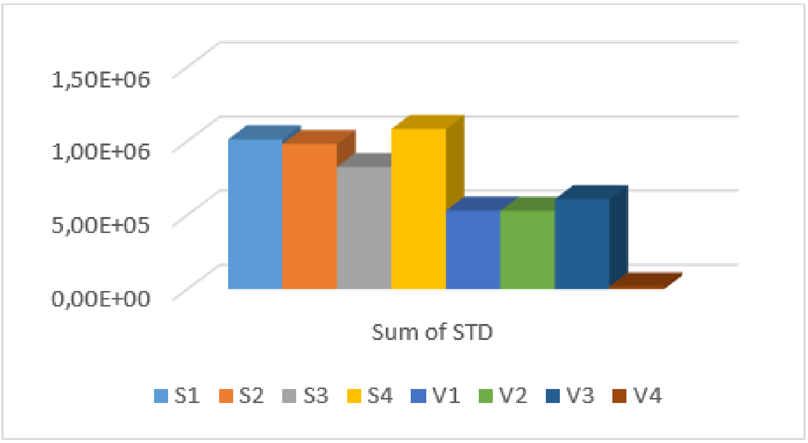
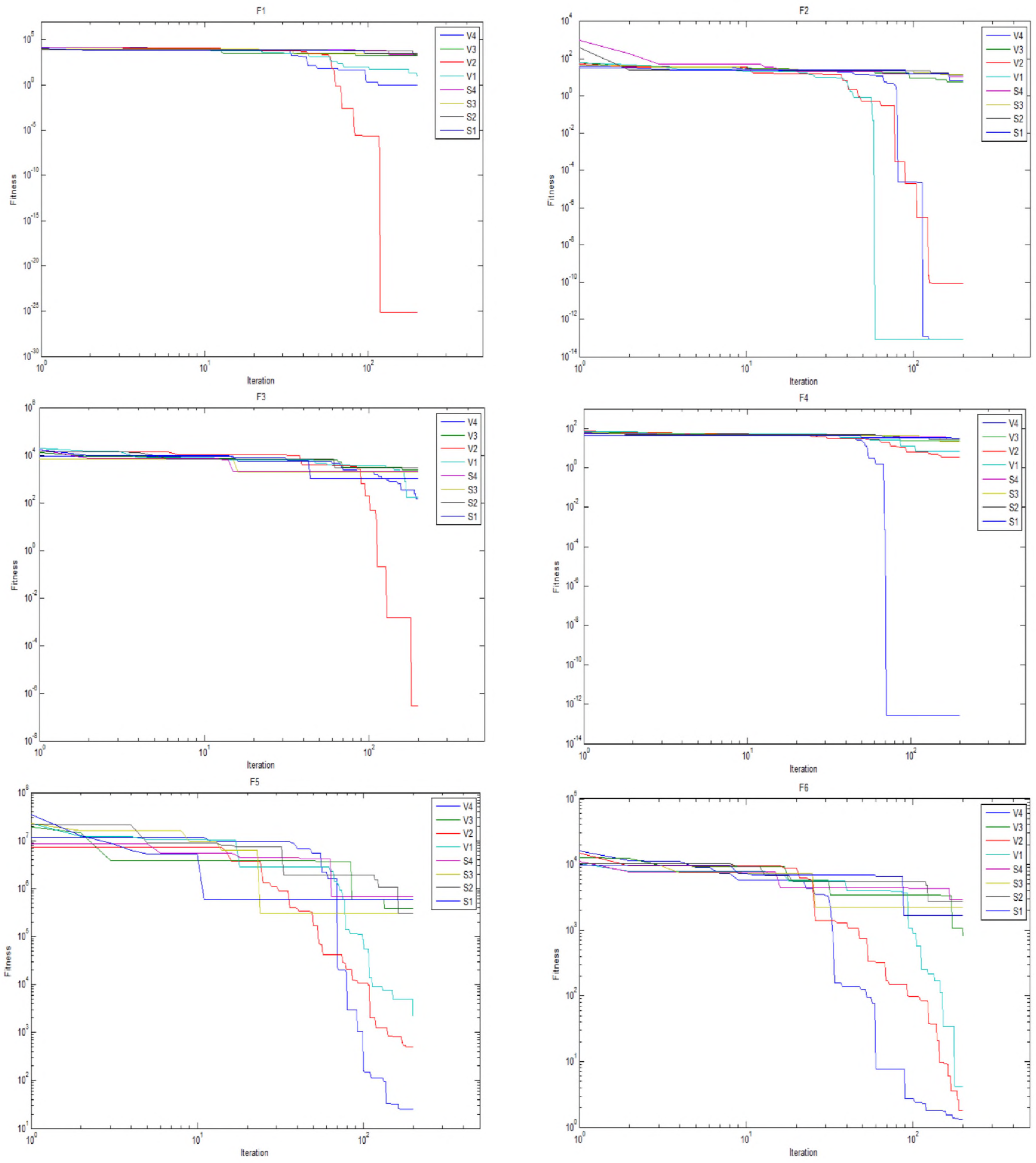


Figure 5. Comparision of the sum of the standard deviation results

In addition, the convergence graphics obtained from the variants for each function and plotted according to the best value are presented in Figure 6. When the graphics are examined, the V2 variant in the F1, F3, and F7 functions, the V4 variant in the F4, F5, F6, F8, and F10 functions, and the V1 variant in the remaining F2 and F9 functions converged faster and found the best value. In light of these results, it can be said that the variants using the transfer function with a V-shaped converge faster than the other variants.


F1
F2
F3
F4
F5
F6
Fitness
Iteration
V4
V3
V2
V1
S4
S3
S2
S1

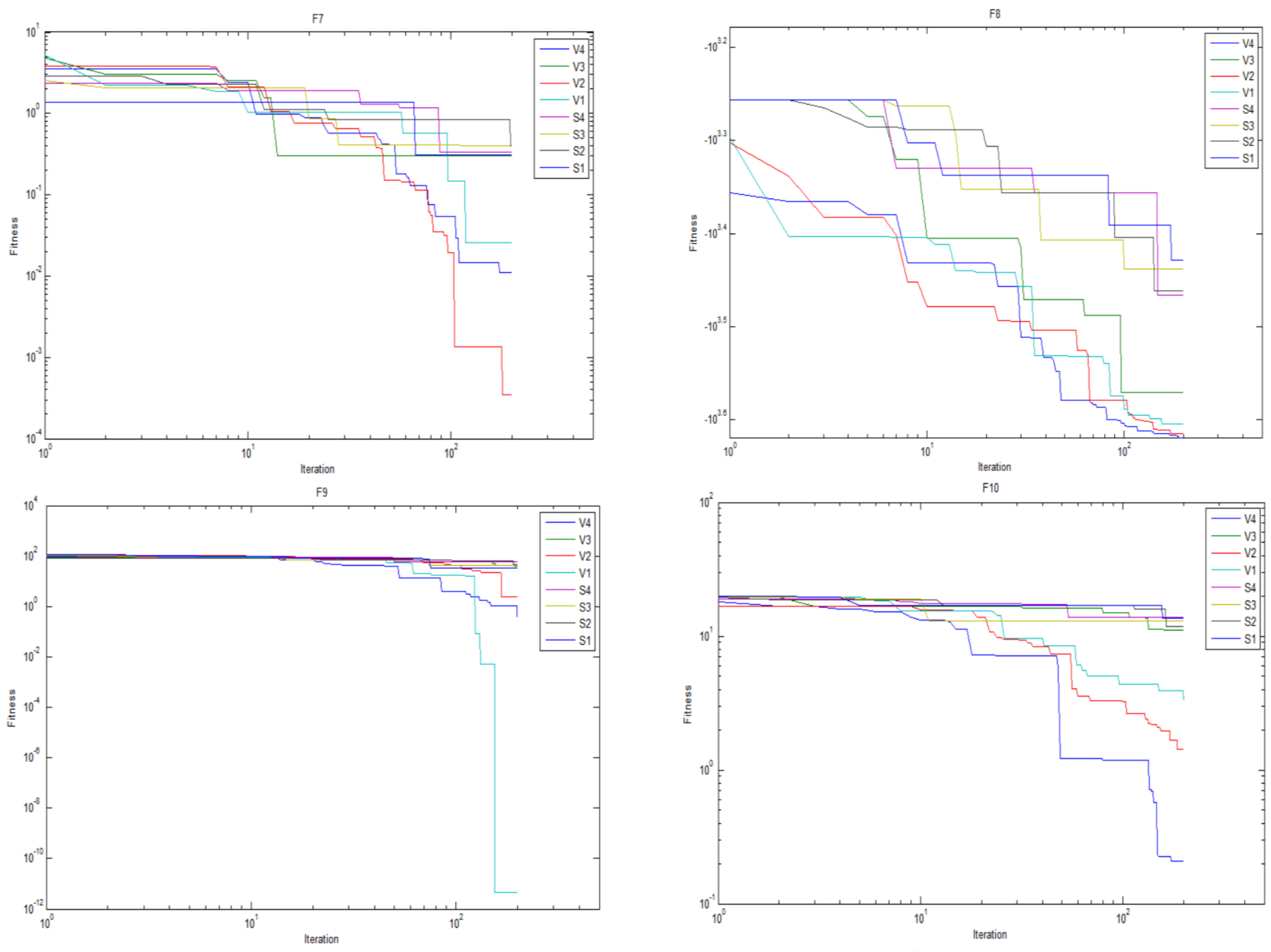
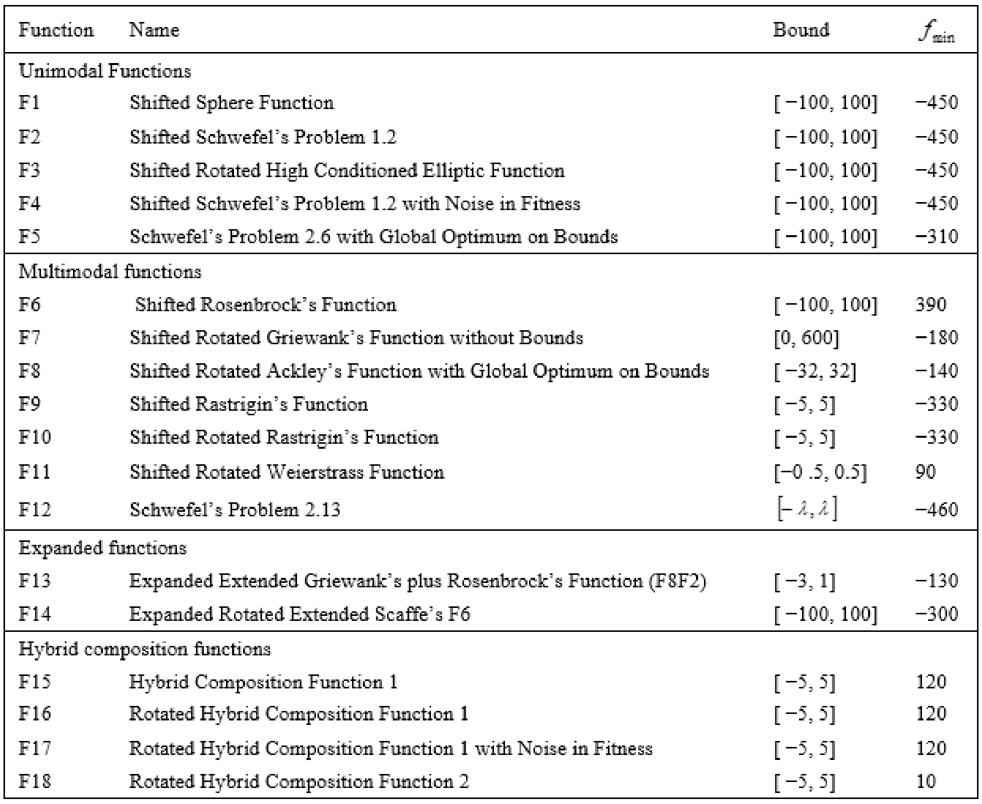


Figure 6. Convergence graphics for the variants on classic benchmark functions

After the performance evaluation of Bin_MRFOA on the classical benchmark functions, the algorithm was run on the CEC2005 benchmark functions. The findings were compared with the algorithms in the literature. The properties of the CEC2005 benchmark functions used are given in Table 6. To make a fair comparison, the algorithms were run under the same conditions as the compared algorithms. According to this, parameter settings are as follows: the population size (N) is set to 40, and the maximum iteration number as the stopping criterion is set to 500. All of binary algorithms are independently run 50 times. The dimensions of functions F1-F14 are 5, while hybrid composition functions F15-F18 are 10. In addition, the Bin_MRFOA variant obtained by using the V4 transfer function, which was found to be more successful in the previous section, was used for literature comparison.

Table 6. CEC2005 benchmark functions



Function	Name	Bound	f_{min}
Unimodal Functions			
F1	Shifted Sphere Function	[−100, 100]	−450
F2	Shifted Schwefel's Problem 1.2	[−100, 100]	−450
F3	Shifted Rotated High Conditioned Elliptic Function	[−100, 100]	−450
F4	Shifted Schwefel's Problem 1.2 with Noise in Fitness	[−100, 100]	−450
F5	Schwefel's Problem 2.6 with Global Optimum on Bounds	[−100, 100]	−310
Multimodal functions			
F6	Shifted Rosenbrock's Function	[−100, 100]	390
F7	Shifted Rotated Griewank's Function without Bounds	[0, 600]	−180
F8	Shifted Rotated Ackley's Function with Global Optimum on Bounds	[−32, 32]	−140
F9	Shifted Rastrigin's Function	[−5, 5]	−330
F10	Shifted Rotated Rastrigin's Function	[−5, 5]	−330
F11	Shifted Rotated Weierstrass Function	[−0 .5, 0.5]	90
F12	Schwefel's Problem 2.13	$[-\lambda, \lambda]$	−460
Expanded functions			
F13	Expanded Extended Griewank's plus Rosenbrock's Function (F8F2)	[−3, 1]	−130
F14	Expanded Rotated Extended Scaffe's F6	[−100, 100]	−300
Hybrid composition functions			
F15	Hybrid Composition Function 1	[−5, 5]	120
F16	Rotated Hybrid Composition Function 1	[−5, 5]	120
F17	Rotated Hybrid Composition Function 1 with Noise in Fitness	[−5, 5]	120
F18	Rotated Hybrid Composition Function 2	[−5, 5]	10

In the literature comparison, the Bin_MRFOA was compared to the findings of the algorithms mentioned in the studies of Beheshti [41] and Baş [42]. Comparison results are showed in Table 7. According to Table 7, Bin_MRFOA found a more successful mean value in 10 of the 18 functions. Similarly, 12 of the 18 functions obtained a better standard deviation value. Pairwise comparisons of the algorithms were made using the Wilcoxon signed-rank test. The Wilcoxon signed-rank test gets to decide the difference between two samples and offers an alternate position test influenced by the amplitude and sign of these differences. This test is also used to decide whether one algorithm outperforms another [40, 43]. The symbols +, -, and ≈ show the number of test functions in which the Bin_MRFOA is better, worse, and equal than the other algorithm, respectively. The significance level of the Wilcoxon signed-rank test was set at 0.05. Accordingly, since the obtained p value was less than 0.05, it was determined that there was a significant difference between the Bin_MRFOA and BPSO algorithms. In other words, the Bin_MRFOA algorithm outperformed BPSO. Since the p value of the remaining algorithms was greater than 0.05, there was no significant difference between them and Bin_MRFOA. The algorithms had similar performances. In addition, in the ranking among algorithms based on the Friedman test, the Bin_MRFOA algorithm took second place with a mean rank of 2,61. According to this result, it can be said that the proposed algorithm is a successful and competitive algorithm compared to the literature.

Table 7. The comparison results of Bin_MRFOA and algorithms in the literature on CEC2005 benchmark functions

Function		**TVMS-BPSO [41]**	**MS-BPSO [41]**	**TV-BPSO [41]**	**BPSO [41]**	**BinSSA4 [42]**	**Bin_MRFOA**
F1	Mean	-446.97	-420.73	-446.1	-432.56	-431.45	**-450.00**
	Std	5.4981	60.906	5.9235	16.237	8.412	**0.00**
F2	Mean	-438.98	-395.92	-436.31	-421.95	-435.47	**-450.00**
	Std	17.377	181.1	22.081	24.169	21.452	**0.00**
F3	Mean	98.287	3.47E+05	1.05E+05	2.98E+05	**98.265**	1.95E+05
	Std	1.13E+05	3.68E+05	1.11E+05	2.59E+05	**1.02E+05**	3.35E+05
F4	Mean	-441.74	-406.59	-433.73	-409.16	-445.45	**-450.00**
	Std	12.37	56.808	25.236	31.834	13.42	**0.00**
F5	Mean	-241.97	-225.78	-127.65	-4.5091	-125.41	**-310.00**
	Std	106.14	694.32	270.05	106.95	220.41	**0.00**
F6	Mean	503.45	10.293	1625	4926.8	2412.4	**390.00**
	Std	146.36	22.647	7752.8	9628.2	4124.5	**0.00**
F7	Mean	266.25	266.66	266.43	269.87	266.55	**220.20**
	Std	265.9	265.95	266.04	267.37	274.14	**182.71**
F8	Mean	-120.05	-120.01	**-120.24**	-120.05	-120.05	-120.06
	Std	0.74112	**0.31108**	1.7162	0.62672	0.7452	0.50
F9	Mean	-328.55	-328.03	-328.54	-323.75	-328.45	**-329.99**
	Std	1.3196	1.3623	1.1191	2.2942	1.4285	**0.03**
F10	Mean	**-325.6**	-322.63	-325.14	-320	-321.41	-322.10
	Std	**1.9871**	4.0479	2.6135	3.6903	3.4152	3.37
F11	Mean	91.167	91.698	91.293	92.447	**91.165**	91.91
	Std	0.65567	0.70823	0.67296	**0.44508**	0.64855	0.63
F12	Mean	-238.13	-64.804	-224.79	-98.765	-121.41	**-460.00**
	Std	222.15	457.07	277.22	237.39	250.411	**0.00**
F13	Mean	-129.77	-129.66	-129.78	-129.3	-129.65	**-130.00**
	Std	0.095197	0.21671	0.10462	0.22568	0.4122	**0.00**
F14	Mean	**-298.87**	-298.75	-298.86	-298.57	-298.77	-297.78
	Std	0.28931	0.35812	0.31671	**0.17735**	0.3214	0.25
F15	Mean	379.16	388.99	406.85	639.77	390.41	**120.00**
	Std	154.18	146.18	157.16	101.61	185.411	**0.00**
F16	Mean	261.86	280.98	267	373.48	**260.41**	312.76
	Std	17.854	33.481	18.718	23.068	18.12	**13.90**
F17	Mean	276.62	278.69	**270.22**	399.26	352.41	311.48
	Std	24.911	27.625	22.168	23.941	22.412	**25.83**
F18	Mean	795.26	911.25	**776.61**	976.16	845.74	988.43
	Std	208.3	136.76	207.36	**92.887**	205.11	53.06
Wilcoxon signed-rank test							
(+ /-/≈)		(11/7/0)	(11/7/0)	(10/8/0)	(16/2/0)	(13/5/0)	
p value		(≈) 0,472	(≈) 0,215	(≈) 0,420	(+) 0,001	(≈) 0,145	
Friedman test							
Mean Rank Value		2,17	4.39	2.78	5.44	3.61	2.61
Rank		1	5	3	6	4	2

4. CONCLUSIONS AND DISCUSSION

The manta ray foraging optimization algorithm has advantages such as strong global search capability, few parameters to adjust, and robustness. In this study, a new binary version of the manta ray foraging optimization algorithm was proposed to benefit from the mentioned advantages. At the same time, thanks to the proposed Bin_MRFOA, since the problem can be generalized as desired and the optimum solutions are known, a simple and informative algorithm was obtained to compare with other algorithms on binary optimization problems. The proposed algorithm was first tested for ten classical benchmark functions. Firstly, with this test process, the ideal population size was found. Then, the results of the variants obtained using eight different transfer functions were compared, and it was decided that the most effective transfer function was V4. Furthermore, as a consequence of these comparisons, it was concluded that the transfer functions of V-shaped are more effective than the transfer functions of S-shaped in general. Then, the most successful variant was run on eighteen CEC2005 benchmark test functions. The obtained results were compared with the algorithms in the literature. The algorithms' performance was interpreted by the Wilcoxon signed-rank and Friedman tests. According to Wilcoxon signed-rank test results, the Bin_MRFOA algorithm outperformed one of the compared algorithms. It performed similarly with the remaining four algorithms and no significant difference was found between them. In addition, among the compared algorithms, the proposed Bin_MRFOA algorithm took second place, revealing that it is a successful and competitive algorithm. Despite successful results, the optimal value could not be reached for functions except for F1, F2, F4, F5, F6, F12, F13, and F15 in all compared algorithms. It would be useful to consider the reason for this situation as a separate research topic. To comment, though, these functions are expensive functions created through rotation, shifting, and hybridization. Most metaheuristic algorithms experience performance decreases in such expensive functions when compared to classical functions. In light of these results, it can be said that the binary algorithms in the study should be strengthened with approaches such as mutation, crossover, similarity measurement technology, etc. that will contribute to the ability to create candidate individuals. In a future study, the performance of Bin_MRFOA on different binary problems can be examined.

Declaration of Ethical Standards

The article complies with ethical rules.

Credit Authorship Contribution Statement

Gülnur YILDIZDAN Conceptualization, Methodology, Designed the Experiments, Writing original draft, Supervision, and Writing review and Editing.

Declaration of Competing Interest

The author declares no conflict of interest.

Funding / Acknowledgements

The author declares that she has no financial support.

Data Availability

Data will be made available on request.

REFERENCES

[1] R. M. Rizk-Allah, "Hybridizing sine cosine algorithm with multi-orthogonal search strategy for engineering design problems," Journal of Computational Design and Engineering, vol. 5, no. 2, pp. 249-273, 2018.
[2] S. Korkmaz, A. Babalik, and M. S. Kiran, "An artificial algae algorithm for solving binary optimization problems," International Journal of Machine Learning and Cybernetics, vol. 9, no. 7, pp. 1233-1247, 2018.
[3] J. Wang, M. Khishe, M. Kaveh, and H. Mohammadi, "Binary Chimp Optimization Algorithm (BChOA): a New Binary Meta-heuristic for Solving Optimization Problems," Cognitive Computation, vol. 13, no. 5, pp. 1297-1316, 2021.
[4] Q. Al-Tashi, S. J. A. Kadir, H. M. Rais, S. Mirjalili, and H. Alhussian, "Binary optimization using hybrid grey wolf optimization for feature selection," Ieee Access, vol. 7, pp. 39496-39508, 2019.

[5] E. Baş and E. Ülker, "A binary social spider algorithm for continuous optimization task," Soft Computing, vol. 24, no. 17, pp. 12953-12979, 2020.

[6] M. Aslan, M. Gunduz, and M. S. Kiran, "JayaX: Jaya algorithm with xor operator for binary optimization," Applied Soft Computing, vol. 82, p. 105576, 2019.

[7] A. G. Hussien, A. E. Hassanien, E. H. Houssein, M. Amin, and A. T. Azar, "New binary whale optimization algorithm for discrete optimization problems," Engineering Optimization, vol. 52, no. 6, pp. 945-959, 2020.

[8] A. C. Cinar and M. S. Kiran, "Similarity and logic gate-based tree-seed algorithms for binary optimization," Computers & Industrial Engineering, vol. 115, pp. 631-646, 2018.

[9] R. M. Rizk-Allah, A. E. Hassanien, M. Elhoseny, and M. Gunasekaran, "A new binary salp swarm algorithm: development and application for optimization tasks," Neural Computing and Applications, vol. 31, no. 5, pp. 1641-1663, 2019.

[10] S. Arora and P. Anand, "Binary butterfly optimization approaches for feature selection," Expert Systems with Applications, vol. 116, pp. 147-160, 2019.

[11] M. Mafarja et al., "Binary dragonfly optimization for feature selection using time-varying transfer functions," Knowledge-Based Systems, vol. 161, pp. 185-204, 2018.

[12] T. Akan, S. Agahian, and R. Dehkharghani, "Binbro: Binary battle royale optimizer algorithm," Expert Systems with Applications, vol. 195, p. 116599, 2022.

[13] M. Abdel-Basset, R. Mohamed, R. K. Chakrabortty, M. Ryan, and S. Mirjalili, "New binary marine predators optimization algorithms for 0–1 knapsack problems," Computers & Industrial Engineering, vol. 151, p. 106949, 2021.

[14] E. Kaya, "BinGSO: galactic swarm optimization powered by binary artificial algae algorithm for solving uncapacitated facility location problems," Neural Computing and Applications, pp. 1-20, 2022.

[15] D. Chauhan and A. Yadav, "Binary Artificial Electric Field Algorithm," Evolutionary Intelligence, pp. 1-29, 2022.

[16] M. A. Sahman and A. C. Cinar, "Binary tree-seed algorithms with S-shaped and V-shaped transfer functions," International Journal of Intelligent Systems and Applications in Engineering, vol. 7, no. 2, pp. 111-117, 2019.

[17] M. Dehghani et al., "Binary Spring Search Algorithm for Solving Various Optimization Problems," Applied Sciences, vol. 11, no. 3, p. 1286, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/3/1286.

[18] Z. Beheshti, "A novel x-shaped binary particle swarm optimization," Soft Computing, vol. 25, no. 4, pp. 3013-3042, 2021/02/01 2021, doi: 10.1007/s00500-020-05360-2.

[19] M. Kalra, V. Kumar, M. Kaur, S. A. Idris, Ş. Öztürk, and H. Alshazly, "A Novel Binary Emperor Penguin Optimizer for Feature Selection Tasks," Comput. Mater. Contin, vol. 70, pp. 6239-6255, 2022.

[20] H. Chantar, M. Mafarja, H. Alsawalqah, A. A. Heidari, I. Aljarah, and H. Faris, "Feature selection using binary grey wolf optimizer with elite-based crossover for Arabic text classification," Neural Computing and Applications, vol. 32, no. 16, pp. 12201-12220, 2020/08/01 2020, doi: 10.1007/s00521-019-04368-6.

[21] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, and L. Abualigah, "Binary Aquila Optimizer for Selecting Effective Features from Medical Data: A COVID-19 Case Study," Mathematics, vol. 10, no. 11, p. 1929, 2022. [Online]. Available: https://www.mdpi.com/2227-7390/10/11/1929.

[22] Y. He, F. Zhang, S. Mirjalili, and T. Zhang, "Novel binary differential evolution algorithm based on Taper-shaped transfer functions for binary optimization problems," Swarm and Evolutionary Computation, vol. 69, p. 101022, 2022/03/01/ 2022, doi: https://doi.org/10.1016/j.swevo.2021.101022.

[23] H. Hakli, "BinEHO: a new binary variant based on elephant herding optimization algorithm," Neural Computing and Applications, vol. 32, no. 22, pp. 16971-16991, 2020/11/01 2020, doi: 10.1007/s00521-020-04917-4.

[24] A. Pourrajabian, M. Dehghan, and S. Rahgozar, "Genetic algorithms for the design and optimization of horizontal axis wind turbine (HAWT) blades: A continuous approach or a binary one?," Sustainable Energy Technologies and Assessments, vol. 44, p. 101022, 2021/04/01/ 2021, doi: https://doi.org/10.1016/j.seta.2021.101022.

[25] H. Mohammadzadeh and F. S. Gharehchopogh, "Feature selection with binary symbiotic organisms search algorithm for email spam detection," International Journal of Information Technology & Decision Making, vol. 20, no. 01, pp. 469-515, 2021.

[26] K. K. Ghosh, R. Guha, S. K. Bera, N. Kumar, and R. Sarkar, "S-shaped versus V-shaped transfer functions for binary Manta ray foraging optimization in feature selection problem," Neural Computing and Applications, vol. 33, no. 17, pp. 11027-11041, 2021/09/01 2021, doi: 10.1007/s00521-020-05560-9.

[27] 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, 2022/01/01/ 2022, doi: https://doi.org/10.1016/j.future.2021.07.033.

[28] M. Xi, Q. Song, M. Xu, and Z. Zhou, "Binary African vultures optimization algorithm for various optimization problems," International Journal of Machine Learning and Cybernetics, 2022/11/16 2022, doi: 10.1007/s13042-022-01703-7.

[29] P. Hu, J.-S. Pan, S.-C. Chu, and C. Sun, "Multi-surrogate assisted binary particle swarm optimization algorithm and its application for feature selection," Applied Soft Computing, vol. 121, p. 108736, 2022/05/01/ 2022, doi: https://doi.org/10.1016/j.asoc.2022.108736.

[30] A. Banitalebi, M. I. Abd Aziz, and Z. A. Aziz, "A self-adaptive binary differential evolution algorithm for large scale binary optimization problems," Information Sciences, vol. 367, pp. 487-511, 2016.

[31] A. Mohammadzadeh, M. Masdari, F. S. Gharehchopogh, and A. Jafarian, "Improved chaotic binary grey wolf optimization algorithm for workflow scheduling in green cloud computing," Evolutionary Intelligence, vol. 14, no. 4, pp. 1997-2025, 2021/12/01 2021, doi: 10.1007/s12065-020-00479-5.

[32] S. Korkmaz, "İkili optimizasyon problemlerinin çözümü için yapay alg algoritması tabanlı yeni yaklaşımlar," Doktora Tezi Doktora Tezi, Konya Teknik Üniversitesi, Konya, 2019.

[33] W. Zhao, Z. Zhang, and L. Wang, "Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications," Engineering Applications of Artificial Intelligence, vol. 87, p. 103300, 2020.

[34] G. Hu, M. Li, X. Wang, G. Wei, and C.-T. Chang, "An enhanced manta ray foraging optimization algorithm for shape optimization of complex CCG-Ball curves," Knowledge-Based Systems, vol. 240, p. 108071, 2022/03/15/ 2022, doi: https://doi.org/10.1016/j.knosys.2021.108071.

[35] M. Beşkirli, İ. Koç, H. Haklı, and H. Kodaz, "A new optimization algorithm for solving wind turbine placement problem: Binary artificial algae algorithm," Renewable Energy, vol. 121, pp. 301-308, 2018/06/01/ 2018, doi: https://doi.org/10.1016/j.renene.2017.12.087.

[36] M. Beşkirli, I. Koc, and H. Kodaz, "Optimal placement of wind turbines using novel binary invasive weed optimization," Tehnički vjesnik, vol. 26, no. 1, pp. 56-63, 2019.

[37] A. Beşkirli and İ. Dağ, "A new binary variant with transfer functions of Harris Hawks Optimization for binary wind turbine micrositing," Energy Reports, vol. 6, pp. 668-673, 2020/12/01/ 2020, doi: https://doi.org/10.1016/j.egyr.2020.11.154.

[38] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization," Journal of Heuristics, vol. 15, no. 6, pp. 617-644, 2009.

[39] T. Eftimov, P. Korošec, and B. Koroušić Seljak, "A Novel Approach to statistical comparison of meta-heuristic stochastic optimization algorithms using deep statistics," Information Sciences, vol. 417, pp. 186-215, 2017/11/01/ 2017, doi: https://doi.org/10.1016/j.ins.2017.07.015.

[40] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," Swarm and Evolutionary Computation, vol. 1, no. 1, pp. 3-18, 2011.

[41] Z. Beheshti, "A time-varying mirrored S-shaped transfer function for binary particle swarm optimization," Information Sciences, vol. 512, pp. 1503-1542, 2020.

[42] E. Baş and E. Ülker, "A binary social spider algorithm for uncapacitated facility location problem," Expert Systems with Applications, vol. 161, p. 113618, 2020.

[43] I. Tariq et al., "MOGSABAT: a metaheuristic hybrid algorithm for solving multi-objective optimisation problems," Neural Computing and Applications, vol. 32, no. 8, pp. 3101-3115, 2020/04/01 2020, doi: 10.1007/s00521-018-3808-3.