Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 20, No. 2, 2026

Muskrat Optimization Algorithm (MOA): A Nature-Inspired Metaheuristic for Complex Optimization Problems.

Lin Yuan

College of Computer Science and Engineering, Chongqing University of Technology, Chongqing 400054, China

Abstract: This paper proposes a novel metaheuristic optimization method, named the Muskrat Optimization Algorithm (MOA), for solving complex optimization problems. Inspired by the foraging and nesting behaviors of muskrats, MOA simulates their cooperative characteristics and spatial memory mechanism. The population is modeled as muskrats with three distinct search paths, and the optimization process is divided into two main phases: foraging and nesting. In the foraging phase, an opportunistic strategy is applied in the early stage to enhance global exploration, followed by a defensive strategy to improve convergence stability. In the nesting phase, a tentative nesting strategy is first adopted to maintain population diversity, and then a refined nesting strategy is employed to strengthen local exploitation. A dynamic memory updating mechanism further balances exploration and exploitation by adjusting historical search information. The performance of MOA is evaluated on the CEC2017 benchmark suite and three constrained engineering design problems. Experimental results demonstrate that MOA achieves competitive convergence accuracy, robustness, and overall optimization performance.

Keywords: Metaheuristic optimization; Muskrat optimization algorithm; Exploration and exploitation; Engineering design problems.

1. Introduction

In practical applications, many optimization problems exhibit non-convexity, nonlinearity, and multimodality[1]. With the increasing complexity of real-world problems, stochastic optimization methods have become an important means for solving a wide range of optimization tasks [2]. In optimization, problems can be classified according to different characteristics, such as continuous or discrete, static or dynamic, and single-objective or multi-objective. Such classifications help in designing solution methods that are well matched to problem characteristics, thereby achieving higher efficiency. Correspondingly, solution methods usually explore the search space through iterative procedures, continuously improving candidate solutions so that the search gradually converges toward the optimal solution [3].

With the continuous advancement of research, a variety of solution methods have been proposed for optimization problems, which can generally be classified into two categories: heuristic algorithms and metaheuristic algorithms [4]. When a problem is computationally difficult to solve exactly within a finite time, heuristic algorithms are widely adopted, aiming to obtain acceptable approximate solutions within a reasonable computational cost rather than exact optimal solutions [5]. On the other hand, metaheuristic optimization algorithms are a class of general-purpose methods for solving optimization problems. They do not rely on the specific structure of a given problem; instead, they seek solutions by simulating heuristic strategies derived from natural phenomena or human experience. Such algorithms usually exhibit good performance when dealing with complex, multimodal, and nonlinear optimization problems, and they do not depend on the convexity or concavity of the problem, nor are they sensitive to the choice of initial values [6].

Metaheuristic algorithms can generally be divided into two categories: single-solution-based methods and population-based methods [7]. In terms of more thoroughly exploring the solution space and seeking global optimal solutions, population-based methods usually outperform single-solution-based methods. According to the different sources of inspiration adopted in algorithm design, metaheuristic algorithms can be further classified into four major categories: swarm intelligence algorithms (Swarm Intelligence, SI), which perform optimization by simulating the movements and behaviors of animal groups and are mainly inspired by biological collectives in nature, such as flocks of birds, colonies of ants, or herds of animals; evolutionary algorithms (Evolutionary Algorithms, EA), which are inspired by biological evolutionary processes and employ operators such as chromosome crossover and mutation for search; physics-based algorithms (Physics-based Algorithms, PA), which guide the optimization process by simulating physical laws in the real world; and human behavior–based algorithms, which design optimization strategies by modeling human behaviors in social environments.

The first type of metaheuristic algorithms consists of swarm intelligence–based methods, which are typically inspired by animals exhibiting collective behaviors in nature. Several swarm intelligence algorithms have been proposed in recent years, such as Particle Swarm Optimization (PSO) [8], Grey Wolf Optimization (GWO) [9], Ant Colony Optimization (ACO) [10], Artificial Bee Colony (ABC) [11], Cuckoo Search (CS) [12], Whale Optimization Algorithm (WOA) [13], Artificial Hummingbird Algorithm (AHA) [14], Nutcracker Optimizer [15], Osprey Optimization Algorithm [16], and Horse Herd Optimization Algorithm (HOA) [17].

The second type of metaheuristic algorithms is evolutionary-based methods, which usually simulate the evolutionary processes of biological species in nature. Some well-known algorithms in this category include Genetic Algorithms (GA), which model individual evolution through chromosome mutation, recombination, and natural selection

[18]. Other notable algorithms belonging to this category include Differential Evolution (DE) [19], Differential Search Algorithm (DSA) [20], Genetic Programming (GP) [21], Forest Optimization Algorithm [22], Stochastic Fractal Search (SFS) [23], Biogeography-Based Optimizer (BBO) [24], Gradient Evolution Algorithm (GEA) [25], Evolution Strategy (ES) [26], and Evolutionary Programming (EP) [27].

The third type of metaheuristic algorithms is physics-based methods, which are typically inspired by natural phenomena governed by physical laws. Some classical algorithms in this category include Simulated Annealing (SA) [28], Gravitational Search Algorithm (GSA) [29], Artificial Chemical Reaction Optimization Algorithm (ACROA) [30], Black Hole (BH) [31], Charged System Search (CSS) [32], Ray Optimization (RO) [33], Gases Brownian Motion Optimization (GBMO) [34], Colliding Bodies Optimization (CBO) [35], Optics Inspired Optimization (OIO) [36], and Water Cycle Algorithm (WCA) [37].

The final type of metaheuristic algorithms is human behavior–based methods, which are usually inspired by human social behaviors. Representative algorithms in this category include Teaching–Learning-Based Optimization (TLBO) [38], Mine Blast Algorithm (MBA) [39], Interior Search Algorithm (ISA) [40], Exchange Market Algorithm (EMA) [41], and Brain Storm Optimization (BSO) [42].

In addition to the four main categories mentioned above, there are also mathematics-based and sports-inspired algorithms, such as QIO[43], TTAO[44], SCA[45], FGA[46], and SGO[47].Although metaheuristic algorithms have demonstrated strong problem-solving capabilities when dealing with highly complex and challenging real-world optimization problems, they inevitably suffer from several inherent limitations. In general, metaheuristic algorithms rely on evolutionary information provided by the best-performing individuals in the population to guide the search process [48]. Taking widely used population-based algorithms as an example, their optimization mechanisms are often constructed around elite individuals. Such elite-oriented search strategies tend to reduce population diversity during the iterative process, which may lead to premature convergence. Meanwhile, metaheuristic algorithms are inherently stochastic in nature, and purely random search also has certain limitations in terms of efficiency and stability. For instance, when solving most practical engineering problems, a large amount of computational resources is often required to obtain a satisfactory set of solutions [49]. These issues have motivated researchers to further explore and improve algorithmic designs in order to construct more stable and efficient optimization frameworks, particularly by addressing the tendency of traditional strategies to become trapped in local optima.

According to the No Free Lunch (NFL) theorem [50], there is no single optimization method that can consistently achieve optimal performance across all problem scenarios. Therefore, metaheuristic optimization algorithms usually exhibit strong performance only for specific types of problems, while their search efficiency and stability may be limited for others. In this paper, a novel metaheuristic optimization algorithm is proposed, inspired by the collective behavioral characteristics and spatial memory mechanisms exhibited by muskrats during foraging and nest-building processes in natural environments. The staged foraging strategies, dependence on historical experience, and risk-avoidance behaviors demonstrated by muskrats in complex wetland ecosystems provide a valuable biological inspiration for constructing an optimization algorithm that balances exploration capability and convergence stability. Based on these behavioral characteristics, a new optimization framework is designed, and its feasibility and reliability are systematically analyzed and evaluated on numerical optimization problems and engineering design problems.

The main contributions of this paper are summarized as follows:

(1) A novel metaheuristic optimization algorithm, termed Muskrat Optimization Algorithm (MOA), is proposed, which can effectively solve continuous optimization problems as well as complex engineering design problems;

(2) Inspired by the spatial memory and risk-avoidance behaviors of muskrats during foraging and nest-building, a memory-guided search mechanism is developed to dynamically balance exploration and exploitation during the search process, thereby reducing the probability of being trapped in local optima;

(3) Comparative experiments with several classical optimization algorithms are conducted to verify the competitive advantages of the proposed algorithm in terms of optimization accuracy, convergence speed, and stability.

The remainder of this paper is organized as follows: the section titled “Muskrat Optimization Algorithm” provides a detailed description of the MOA.; the “Experiments” section presents the simulation results and corresponding analyses; the “Engineering Design Problems” section introduces the engineering problems and discusses the experimental results; and the “Conclusion and Future Work” section summarizes this study and outlines several potential directions for future research.

2. Muskrat Optimization Algorithm

Inspired by the staged foraging behavior, nest-site selection strategy, and history-based spatial memory mechanism exhibited by muskrats in complex wetland environments, we propose the Muskrat Optimization Algorithm (MOA). The algorithm simulates how individual muskrats explore for food, develop nest areas, and utilize historical experience to continuously adjust their behaviors under dynamic risk and resource conditions, thereby approaching the optimal survival position.

2.1. Initialization

In MOA, let the total population size be denoted as PopSize, and the number of core muskrat individuals is defined as $N = \lfloor \mathrm{PopSize}/3 \rfloor$. Each core individual corresponds to three candidate behavioral states in one iteration, representing its current position and exploratory movements in different directions. The mathematical formulation is given in Eq. (1).

$$\overrightarrow{M}_{i,1} = r_0 \times (\overrightarrow{UP} - \overrightarrow{LOW}) + \overrightarrow{LOW} \tag{1}$$

Here, $\overrightarrow{M}_{i,1}$ denotes the first exploratory point of the i-th muskrat, which is a random number within the interval [0,1], and i is an integer ranging from 1 to $\lfloor \mathrm{PopSize}/3 \rfloor$. The parameters $\overrightarrow{LOW}$ and $\overrightarrow{UP}$ represent the lower and upper bounds of these variables, respectively.

2.2. Path attempt and Evaluation

In the proposed MOA, each muskrat generates several tentative positions along different directions according to its

current location, thereby simulating its path exploration behavior in natural environments. The mathematical formulation of this behavior is given in Eq. (2).

$$\begin{aligned}\overrightarrow{M}_{i,2} &= \overrightarrow{M}_{i,1} + \left(r_1 \times \left(\overrightarrow{f}_{up} - \overrightarrow{f}_{low}\right) + \overrightarrow{f}_{low}\right) \times k \\ \overrightarrow{M}_{i,3} &= \overrightarrow{M}_{i,2} + \left(r_2 \times \left(\overrightarrow{f}_{up} - \overrightarrow{f}_{low}\right) + \overrightarrow{f}_{low}\right) \times k\end{aligned} \tag{2}$$

Here, $\overrightarrow{M}_{i,2}$ denotes the second exploratory position of the i-th muskrat, and $\overrightarrow{M}_{i,3}$ denotes the third exploratory position of the i-th muskrat. The parameters r_1 and r_2 are random numbers uniformly distributed in the interval [0,1]. $\overrightarrow{f}_{up}$ and $\overrightarrow{f}_{low}$ epresent the upper and lower bounds of the exploration range, respectively, and k is a scaling factor. Their corresponding calculation formulas are given in Eq. (3).

$$\begin{aligned}\overrightarrow{f}_{up} &= \frac{\overrightarrow{UP}}{t} \\ \overrightarrow{f}_{low} &= \frac{\overrightarrow{LOW}}{t} \\ k(t) &= e^{\frac{-t}{T}}\end{aligned} \tag{3}$$

Here, $\overrightarrow{LOW}$ denotes the given lower bound of the variables, $\overrightarrow{UP}$ denotes the given upper bound of the variables, and T represents the maximum number of iterations.As time progresses, muskrats gradually become familiar with their environment, and their activity radius and exploration intensity exhibit an exponential decay trend. After probing along different directions, muskrats compare the survival gains of multiple paths and categorize them into three types—superior, moderate, and inferior strategies—which are then used to guide subsequent decision-making.

2.3. Foraging phase

In the opportunistic foraging stage, when the location of high-quality food sources is not yet clearly identified, muskrats tend to conduct large-scale exploratory foraging to broaden their understanding of the habitat environment. The characteristics of this behavior are described by the corresponding mathematical model, as shown in Eq. (4).

$$\overrightarrow{M}_{i,of}^{t+1} = \overrightarrow{M}_{i,rand1}^{t} + Fm_i \times \left(\overrightarrow{M}_{best}^{t} - \overrightarrow{M}_{worst}^{t}\right) + \vec{\lambda} \times \left(\overrightarrow{M}_{rand2,best}^{t} - \overrightarrow{M}_{rand3,worst}^{t}\right) \tag{4}$$

Here, $\overrightarrow{M}_{i,rand1}^{t}$ denotes a randomly selected probing path of a muskrat, $\overrightarrow{M}_{best}^{t}$ represents the global best path, and $\overrightarrow{M}_{worst}^{t}$ represents the global worst path. $\overrightarrow{M}_{rand2,best}^{t}$ denotes the superior strategy of a randomly selected muskrat individual, while $\overrightarrow{M}_{rand3,worst}^{t}$ denotes the inferior strategy of another randomly selected muskrat individual. Among the parameters, Fm_i is a dynamic memory factor sampled from a normal distribution centered around the historical successful value. Its mathematical formulation is given in Eq. (5).

$$\begin{aligned}Fm_i &\sim \mathcal{N}\left(HFm_i, 0.5\right) \\ Fm_i &= \max\left(0, Fm_i\right)\end{aligned} \tag{5}$$

The parameter λ is used to characterize the responsiveness of muskrats to non-local environmental information during foraging, thereby regulating the influence intensity of differential information from other individuals on the current search behavior. Its mathematical formulation is provided in Eq.(6).Here, the vector $\vec{r}$ is a random vector with elements uniformly distributed in the range [0,1].

$$\lambda(t) = \left(1 - \frac{e^{\frac{-t}{T}}}{3}\right) \times \vec{r} \tag{6}$$

In the defensive foraging stage, muskrats reduce ineffective long-distance movements and instead concentrate their foraging activities around known high-yield areas and regions near their nests to improve energy utilization efficiency. The characteristics of this behavior are described by the corresponding mathematical model, as shown in Eq. (7).

$$\overrightarrow{M}_{i,df}^{t+1} = \overrightarrow{M}_{i,rand4}^{t} + \overrightarrow{F}_{best} \times \left(\overrightarrow{M}_{i,Hbest}^{t} - \overrightarrow{M}_{i,best}^{t}\right) + \overrightarrow{F}_{worst} \times \left(\overrightarrow{M}_{i,best}^{t} - \overrightarrow{M}_{i,Hworst}^{t}\right) \tag{7}$$

Here, $\overrightarrow{M}_{i,rand4}^{t}$ denotes a randomly selected probing path of a muskrat. $\overrightarrow{F}_{best}$ and $\overrightarrow{F}_{worst}$ represent the response intensities of the muskrat to historically favorable and unfavorable experiences during foraging, respectively. Through a time-evolution mechanism, they achieve a dynamic balance between exploration and avoidance. Their mathematical formulations are given in Eq. (8).

$$\begin{aligned}\overrightarrow{F}_{best} &= \vec{r}_1 \times \frac{e^{\frac{-t}{T}}}{2} \\ \overrightarrow{F}_{worst} &= \vec{r}_2 \times \left(1 - \frac{e^{\frac{-t}{T}}}{2}\right)\end{aligned} \tag{8}$$

$\overrightarrow{F}_{best}$ denotes the attraction intensity toward successful foraging experiences, while $\overrightarrow{F}_{worst}$ represents the avoidance intensity toward unfavorable experiences.Both

$\overrightarrow{r}_1$ and $\overrightarrow{r}_2$ are random vectors within the range [0,1]. $\overrightarrow{M}^t_{i,Hbest}$ and $\overrightarrow{M}^t_{i,Hworst}$ denote the suitable habitat memory and avoidance memory of muskrat individuals, respectively.

$\overrightarrow{M}^t_{i,best}$ represents the optimal strategy of the i -th individual.

2.4. Nesting phase

During the trial nesting stage, muskrats maintain a relatively large activity radius around the temporary nest area and frequently adjust their positions to evaluate the potential benefits of the surrounding environment. The mathematical expression describing this behavior is given in Eq. (9).

$$\overrightarrow{M}^{t+1}_{i,tn}=\overrightarrow{M}^t_{i,ac}+\vec{\alpha}\times\left(\overrightarrow{M}^t_{i,best}-\overrightarrow{M}^t_{i,middle}\right)+\vec{\gamma}\times\left(\overrightarrow{M}^t_{c,best}-\overrightarrow{M}^t_{i,best}\right) \quad (9)$$

Here, $\overrightarrow{M}^t_{i,ac}$ represents the "activity center" of the temporary nest area. The vector $\vec{\alpha}$ is used to characterize the intensity of autonomous nesting adjustment. The vector $\vec{\gamma}$ controls the influence strength of the displacement difference from randomly selected individuals on the current individual's nesting decision. c is a random integer between 1 and N.The formulations of vector $\vec{\alpha}$ and vector $\vec{\gamma}$ are given in Eq. (10). Both $\overrightarrow{r}_3$ and $\overrightarrow{r}_4$ are random vectors within [0,1]. $\overrightarrow{M}^t_{i,ac}$ can be computed using Eq. (11).

$$\begin{aligned}\vec{\alpha}&=\left(\frac{e^{2\times\left(1-\frac{t}{T}\right)}}{2}\right)\times\overrightarrow{r}_3\\ \vec{\gamma}&=e^{\frac{-t}{T}}\times\overrightarrow{r}_4\end{aligned} \quad (10)$$

$$\overrightarrow{M}^t_{i,ac}=\frac{\left(\overrightarrow{M}^t_{i,best}+\overrightarrow{M}^t_{i,middle}\right)}{2} \quad (11)$$

In the elaborate nest-building stage, muskrat individuals utilize historical optimal and suboptimal information to perform fine-grained position adjustments within the local area. By introducing multi-directional differential perturbations, the algorithm can further improve solution accuracy while maintaining search stability. The mathematical modeling of this behavior is given in Eq. (12).

$$\overrightarrow{M}^{t+1}_{i,en}=\overrightarrow{M}^t_{i,best}+\vec{\alpha}\times\left(\overrightarrow{w}_1\times\overrightarrow{v}^t_1+\overrightarrow{w}_2\times\overrightarrow{v}^t_2\right) \quad (12)$$

The vectors $\overrightarrow{v}^t_1$ and $\overrightarrow{v}^t_2$ describe the spatial displacement directions of the current optimal nest relative to other high-quality nests. They represent the directional judgment formed by muskrats when comparing "why this nest site is better." Their mathematical formulations are given in Eq. (13).The parameters $\overrightarrow{w}_1$ and $\overrightarrow{w}_2$ characterize the variation in muskrats' reliance on reference information at different stages. Their mathematical expressions are provided in Eq.(14. Both $\overrightarrow{r}_5$ and $\overrightarrow{r}_6$ are random vectors with elements in the range [0,1].

$$\begin{aligned}\overrightarrow{v}^t_1&=\overrightarrow{M}^t_{i,best}-\overrightarrow{M}^t_{i,middle}\\ \overrightarrow{v}^t_2&=\overrightarrow{M}^t_{i,best}-\overrightarrow{M}^t_{i,worst}\end{aligned} \quad (13)$$

$$\begin{aligned}\overrightarrow{w}_1&=e^{\frac{-t}{T}}\times\overrightarrow{r}_5\\ \overrightarrow{w}_2&=\left(1-e^{\frac{-t}{T}}\right)\times\overrightarrow{r}_6\end{aligned} \quad (14)$$

2.5. Historical memory update mechanism

During long-term foraging, muskrats do not rely solely on immediate perception for decision-making, but exhibit a certain degree of learning and memory capability. In MOA, this behavior is characterized through a dual-channel historical memory structure. $\overrightarrow{M}^t_{i,Hbest}$ represents the historical best memory, storing locations where the individual achieved high rewards during past foraging. $\overrightarrow{M}^t_{i,Hworst}$ represents the historical worst memory, recording locations where the individual performed poorly during exploration and should be avoided.Unlike the strategy of "immediately replacing memory upon finding a better solution," MOA does not adopt a deterministic update approach. Instead, it introduces a probabilistic memory update mechanism, with the update rule given in Eq. (15).

$$\begin{aligned}\overrightarrow{M}^{t+1}_{i,Hbest}&=\begin{cases}\overrightarrow{M}^t_{i,best}, & if\ rand<p_{Hbest}\ and\ f\left(\overrightarrow{M}^t_{i,best}\right)<f\left(\overrightarrow{M}^t_{i,Hbest}\right)\\ \overrightarrow{M}^t_{i,Hbest}, & otherwise\end{cases}\\ \overrightarrow{M}^{t+1}_{i,Hworst}&=\begin{cases}\overrightarrow{M}^t_{i,worst}, & if\ rand<p_{Hworst}\ and\ f\left(\overrightarrow{M}^t_{i,worst}\right)>f\left(\overrightarrow{M}^t_{i,Hworst}\right)\\ \overrightarrow{M}^t_{i,Hworst}, & otherwise\end{cases}\end{aligned} \quad (15)$$

The parameters p_{Hbest} and p_{Hworst} are used to regulate the update intensities of positive experience reinforcement and negative experience inhibition, respectively. p_{Hbest} describes the tendency of a muskrat to consolidate a superior foraging outcome into long-term memory, while p_{Hworst} reflects the muskrat's sensitivity to unsuccessful experiences.

3. Experiments

The CEC2017 benchmark function set consists of 29 functions, and the optimization complexity increases with the problem dimension. Based on the different characteristics exhibited in the search space, these test functions can be classified into four categories: unimodal functions (F1, F3), simple multimodal functions (F4–F10), hybrid functions (F11–F20), and composition functions (F21–F30). To evaluate the performance of PMOA in local search, global search, and local optima avoidance, we conducted tests on the CEC2017 benchmark functions in 100-dimensional space. In this study, several well-known optimization algorithms were selected as comparison methods, including COA[51], DE[19], GJO[52], GWCA[53], GWO[9], PSO[8], RSO[2], and POA[54]. Except for PSO and DE, all comparison algorithms use the same parameter settings as in their original papers, while the parameter settings for PSO and DE are listed in Table 1.The time complexity of the MOA algorithm is $O(N\times D\times T)$, where initialization requires $O(N\times D)$time. The foraging, nest-building, and historical memory update

processes each require $O(N \times D)$ per iteration, population evaluation requires $O(N)$, and the overall iterative process costs $O(T)$. All experimental results were obtained from 30 independent runs. The mean and standard deviation over the 30 runs were used as the final statistical indicators. Detailed experimental results are presented in Tables 2-5.Under the 100-dimensional setting, the MOA algorithm achieved 13 first-place rankings, 6 second-place rankings, 8 third-place rankings, and 1 fourth-place ranking. Across all four types of functions in CEC2017, MOA performed significantly better than the comparison algorithms, ranking first overall.To visually illustrate the iterative process of the algorithm on the 100-dimensional test functions, several representative functions were selected from the four categories to plot the convergence curves for convergence analysis. The convergence curves of different algorithms under the 100-dimensional setting are shown in Figure. 1. As shown in Figure. 1, MOA achieves higher convergence accuracy on F1 compared with the other algorithms. On the multimodal function F5, although the convergence speed is relatively slower, its final convergence accuracy is significantly better than that of the other comparison algorithms. On the hybrid functions F17 and F18, MOA demonstrates the best performance among all comparison algorithms. On the composition functions F21 and F25, MOA attains higher convergence accuracy than the other algorithms. The above convergence analysis indicates that MOA exhibits good robustness in high-dimensional problems.To further demonstrate the statistical superiority of MOA, statistical tests were conducted on the results of the CEC2017 benchmark functions under the 100-dimensional setting. Table 6 presents the statistical results of the Wilcoxon rank-sum test at the 5% significance level. The symbols '+' and '–' indicate that the algorithm shows statistically significant difference and no significant difference, respectively. The results of the Wilcoxon rank-sum test confirm that MOA outperforms the other comparison algorithms in the 100-dimensional case.

Table 1. Parameter setting of the algorithms

Algorithm name	Parameters	Value
DE	F	0.5
	CR	0.9
PSO	w	0.3
	C1	2
	C2	2
	Vmax	6
	Vmin	-6

Table 2. Experimental results of MOA, COA, DE, GJO, GWCA, GWO, PSO, RSO and POA for F1-F9 with 100 dimensions (excluding F2).

Function	Type	MOA	COA	DE	GJO	GWCA	GWO	PSO	RSO	POA
F1	mean	9.94E+04	2.64E+11	3.50E+07	1.22E+11	9.74E+07	3.98E+10	1.61E+11	2.32E+11	1.29E+11
	std	1.33E+04	9.56E+09	2.49E+07	1.15E+10	1.58E+08	1.00E+10	1.35E+10	1.02E+10	1.52E+10
F3	mean	4.79E+05	3.47E+05	7.53E+05	2.91E+05	3.86E+05	3.20E+05	3.84E+05	3.31E+05	2.46E+05
	std	6.03E+04	1.23E+04	6.06E+04	2.04E+04	1.04E+05	3.23E+04	5.77E+04	1.48E+04	2.48E+04
F4	mean	7.52E+02	1.01E+05	7.76E+02	1.70E+04	1.04E+03	3.88E+03	3.85E+04	6.65E+04	2.03E+04
	std	4.93E+01	1.09E+04	4.97E+01	3.99E+03	1.34E+02	1.02E+03	5.05E+03	6.37E+03	6.32E+03
F5	mean	9.58E+02	2.10E+03	1.44E+03	1.54E+03	1.19E+03	1.15E+03	1.22E+03	1.91E+03	1.56E+03
	std	4.79E+01	4.72E+01	3.20E+01	7.92E+01	7.76E+01	5.19E+01	6.62E+01	6.46E+01	6.75E+01
F6	mean	6.46E+02	7.10E+02	6.02E+02	6.67E+02	6.54E+02	6.36E+02	6.58E+02	7.02E+02	6.79E+02
	std	4.12E+00	3.64E+00	5.31E-01	3.87E+00	4.52E+00	5.38E+00	3.11E+00	3.75E+00	3.33E+00
F7	mean	1.58E+03	3.96E+03	1.81E+03	2.85E+03	2.88E+03	1.99E+03	2.63E+03	3.70E+03	3.38E+03
	std	8.46E+01	5.91E+01	4.74E+01	1.61E+02	2.78E+02	2.21E+02	1.87E+02	1.06E+02	1.17E+02
F8	mean	1.30E+03	2.55E+03	1.74E+03	1.86E+03	1.55E+03	1.44E+03	1.64E+03	2.38E+03	2.00E+03
	std	6.32E+01	5.76E+01	2.98E+01	1.03E+02	8.03E+01	5.94E+01	5.42E+01	5.49E+01	6.65E+01
F9	mean	1.59E+04	7.65E+04	3.26E+03	5.31E+04	2.14E+04	3.61E+04	2.07E+04	6.90E+04	3.66E+04
	std	2.29E+03	3.39E+03	1.93E+03	1.13E+04	2.21E+03	1.14E+04	1.77E+03	8.00E+03	2.32E+03

Table 3. Experimental results of MOA, COA, DE, GJO, GWCA, GWO, PSO, RSO and POA for F10-F17 with 100 dimensions.

Function	Type	MOA	COA	DE	GJO	GWCA	GWO	PSO	RSO	POA
F10	mean	1.56E+04	3.23E+04	3.22E+04	2.36E+04	1.65E+04	1.74E+04	1.59E+04	3.05E+04	1.96E+04
	std	7.86E+02	7.69E+02	5.96E+02	4.95E+03	1.36E+03	4.87E+03	1.37E+03	2.02E+03	1.42E+03
F11	mean	3.17E+04	2.33E+05	1.24E+04	8.20E+04	4.17E+03	6.29E+04	3.95E+03	1.59E+05	5.94E+04
	std	1.15E+04	3.09E+04	3.89E+03	1.83E+04	1.65E+03	1.81E+04	4.53E+02	3.10E+04	1.42E+04
F12	mean	1.46E+08	1.93E+11	2.86E+07	3.61E+10	3.89E+07	6.52E+09	9.34E+10	1.51E+11	5.57E+10
	std	4.27E+07	1.65E+10	1.50E+07	1.10E+10	2.33E+07	3.26E+09	1.41E+10	1.70E+10	1.25E+10
F13	mean	6.20E+04	4.15E+10	6.40E+03	6.35E+09	1.34E+04	6.90E+08	1.65E+10	4.01E+10	1.05E+10
	std	2.45E+04	6.36E+09	4.67E+03	2.92E+09	9.33E+03	5.46E+08	4.08E+09	3.55E+09	3.69E+09
F14	mean	5.20E+05	8.13E+07	3.77E+05	1.28E+07	1.16E+06	7.36E+06	1.18E+06	2.78E+07	4.27E+06
	std	5.84E+05	3.50E+07	3.47E+05	5.89E+06	1.45E+06	4.59E+06	1.78E+06	1.69E+07	2.18E+06
F15	mean	5.54E+04	2.02E+10	7.50E+03	1.95E+09	5.46E+03	1.72E+08	4.30E+09	1.55E+10	3.50E+09
	std	2.43E+04	5.54E+09	6.69E+03	1.42E+09	3.53E+03	2.23E+08	1.76E+09	2.47E+09	2.41E+09
F16	mean	6.03E+03	2.25E+04	1.02E+04	8.79E+03	5.83E+03	6.05E+03	9.85E+03	1.83E+04	1.06E+04
	std	6.03E+02	2.51E+03	4.98E+02	1.63E+03	6.32E+02	6.78E+02	1.86E+03	1.56E+03	1.39E+03
F17	mean	4.63E+03	5.02E+06	7.34E+03	1.20E+04	5.70E+03	5.10E+03	1.14E+05	2.75E+05	2.93E+04
	std	3.51E+02	3.48E+06	3.23E+02	9.49E+03	5.44E+02	6.47E+02	1.55E+05	2.93E+05	3.17E+04

Table 4. Experimental results of MOA, COA, DE, GJO, GWCA, GWO, PSO, RSO and POA for F18-F25 with 100 dimensions

Function	Type	MOA	COA	DE	GJO	GWCA	GWO	PSO	RSO	POA
F18	mean	1.06E+06	2.11E+08	1.84E+06	1.16E+07	1.45E+06	4.55E+06	7.66E+06	3.95E+07	4.78E+06
	std	4.32E+05	1.11E+08	7.86E+05	6.38E+06	1.21E+06	2.25E+06	1.35E+07	3.88E+07	3.62E+06
F19	mean	4.99E+06	2.30E+10	6.27E+03	1.59E+09	6.10E+03	1.22E+08	4.61E+09	1.48E+10	2.26E+09
	std	2.30E+06	4.52E+09	5.57E+03	1.43E+09	3.86E+03	1.49E+08	1.49E+09	1.90E+09	2.11E+09
F20	mean	5.03E+03	7.77E+03	7.57E+03	6.15E+03	5.37E+03	5.03E+03	5.37E+03	6.85E+03	5.14E+03
	std	2.50E+02	3.40E+02	2.27E+02	1.05E+03	4.11E+02	7.72E+02	4.46E+02	5.69E+02	3.60E+02
F21	mean	2.87E+03	4.85E+03	3.27E+03	3.41E+03	3.18E+03	2.96E+03	4.03E+03	4.80E+03	3.79E+03
	std	5.40E+01	2.03E+02	2.87E+01	1.05E+02	1.11E+02	1.16E+02	1.75E+02	1.61E+02	1.17E+02
F22	mean	1.86E+04	3.48E+04	3.42E+04	2.63E+04	1.89E+04	2.06E+04	1.95E+04	3.35E+04	2.33E+04
	std	7.39E+02	6.69E+02	4.61E+02	4.38E+03	1.37E+03	3.66E+03	1.49E+03	1.50E+03	9.36E+02
F23	mean	3.58E+03	6.58E+03	3.75E+03	4.25E+03	4.11E+03	3.54E+03	5.69E+03	6.00E+03	4.69E+03
	std	1.44E+02	2.36E+02	3.56E+01	1.14E+02	2.07E+02	1.11E+02	3.72E+02	3.00E+02	2.34E+02
F24	mean	4.28E+03	1.01E+04	4.22E+03	5.56E+03	5.11E+03	4.22E+03	1.04E+04	8.65E+03	5.94E+03
	std	1.55E+02	7.79E+02	5.08E+01	2.98E+02	5.12E+02	1.94E+02	6.07E+02	6.87E+02	3.37E+02
F25	mean	3.42E+03	2.85E+04	3.47E+03	1.09E+04	3.73E+03	6.05E+03	1.50E+04	1.92E+04	1.12E+04
	std	5.97E+01	1.78E+03	6.68E+01	1.73E+03	1.50E+02	6.64E+02	1.75E+03	1.73E+03	1.57E+03

Table 5. Experimental results of MOA, COA, DE, GJO, GWCA, GWO, PSO, RSO and POA for F26-F30 with 100 dimensions

Function	Type	MOA	COA	DE	GJO	GWCA	GWO	PSO	RSO	POA
F26	mean	1.53E+04	5.11E+04	1.53E+04	2.56E+04	2.26E+04	1.43E+04	3.83E+04	4.03E+04	3.41E+04
	std	1.14E+03	2.18E+03	4.02E+02	2.16E+03	2.96E+03	9.07E+02	2.91E+03	2.90E+03	2.81E+03
F27	mean	3.79E+03	1.38E+04	3.42E+03	5.36E+03	4.04E+03	4.05E+03	9.44E+03	1.01E+04	5.46E+03
	std	1.13E+02	1.31E+03	3.79E+01	4.21E+02	2.55E+02	1.41E+02	1.49E+03	8.41E+02	4.66E+02
F28	mean	3.56E+03	2.95E+04	3.77E+03	1.44E+04	3.97E+03	7.90E+03	2.12E+04	2.14E+04	1.59E+04
	std	5.59E+01	1.48E+03	1.67E+02	1.78E+03	2.06E+02	1.22E+03	1.63E+03	1.48E+03	2.44E+03
F29	mean	8.76E+03	5.43E+05	9.24E+03	1.41E+04	7.78E+03	8.36E+03	2.48E+04	1.26E+05	1.57E+04
	std	8.27E+02	2.67E+05	4.92E+02	6.02E+03	6.30E+02	6.14E+02	1.17E+04	5.59E+04	4.97E+03
F30	mean	1.96E+07	3.99E+10	3.16E+04	4.93E+09	3.33E+04	6.88E+08	1.47E+10	3.32E+10	8.45E+09
	std	7.27E+06	4.75E+09	1.92E+04	2.36E+09	2.00E+04	5.45E+08	3.85E+09	2.40E+09	3.38E+09

Table 6. Wilcoxon rank-sum test results of the different algorithms on the 100-dimensional CEC'17 functions.

Function	COA	DE	GJO	GWCA	GWO	PSO	RSO	POA
F1	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
F3	3.02E-11	4.08E-11	3.02E-11	2.60E-05	3.34E-11	2.57E-07	3.02E-11	3.02E-11
F4	3.02E-11	4.84E-02	3.02E-11	4.08E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
F5	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.34E-11	3.02E-11	3.02E-11	3.02E-11
F6	3.02E-11	3.02E-11	3.02E-11	7.69E-08	3.65E-08	4.08E-11	3.02E-11	3.02E-11
F7	3.02E-11	7.39E-11	3.02E-11	3.02E-11	7.39E-11	3.02E-11	3.02E-11	3.02E-11
F8	3.02E-11	3.02E-11	3.02E-11	6.70E-11	1.20E-08	3.02E-11	3.02E-11	3.02E-11
F9	3.02E-11	3.02E-11	3.02E-11	7.38E-10	8.15E-11	1.10E-08	3.02E-11	3.02E-11
F10	3.02E-11	3.02E-11	8.99E-11	5.08E-03	1.81E-01	2.40E-01	3.02E-11	3.69E-11
F11	3.02E-11	2.87E-10	6.07E-11	3.02E-11	8.48E-09	3.02E-11	3.02E-11	8.48E-09
F12	3.02E-11	3.02E-11	3.02E-11	6.07E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
F13	3.02E-11	3.02E-11	3.02E-11	1.21E-10	3.02E-11	3.02E-11	3.02E-11	3.02E-11
F14	3.02E-11	5.19E-02	3.34E-11	1.27E-02	5.49E-11	1.87E-07	3.02E-11	8.99E-11
F15	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
F16	3.02E-11	3.02E-11	3.69E-11	1.76E-01	8.30E-01	4.62E-10	3.02E-11	3.02E-11
F17	3.02E-11	3.02E-11	3.34E-11	6.52E-09	6.91E-04	3.02E-11	3.02E-11	3.02E-11
F18	3.02E-11	2.60E-05	3.02E-11	6.20E-01	5.07E-10	5.60E-07	3.02E-11	2.37E-10
F19	3.02E-11	3.02E-11	3.02E-11	3.02E-11	6.72E-10	3.02E-11	3.02E-11	3.02E-11
F20	3.02E-11	3.02E-11	3.32E-06	8.12E-04	3.63E-01	9.03E-04	3.02E-11	7.24E-02
F21	3.02E-11	3.02E-11	3.02E-11	4.08E-11	2.28E-05	3.02E-11	3.02E-11	3.02E-11
F22	3.02E-11	3.02E-11	3.02E-11	6.95E-01	7.20E-05	1.50E-02	3.02E-11	3.02E-11
F23	3.02E-11	1.25E-05	3.02E-11	1.61E-10	4.12E-01	3.02E-11	3.02E-11	3.02E-11
F24	3.02E-11	1.45E-01	3.02E-11	6.12E-10	9.63E-02	3.02E-11	3.02E-11	3.02E-11
F25	3.02E-11	9.47E-03	3.02E-11	4.98E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
F26	3.02E-11	6.52E-01	3.02E-11	4.98E-11	8.56E-04	3.02E-11	3.02E-11	3.02E-11
F27	3.02E-11	3.02E-11	3.02E-11	3.16E-05	1.70E-08	3.02E-11	3.02E-11	3.02E-11
F28	3.02E-11	6.72E-10	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
F29	3.02E-11	2.51E-02	5.49E-11	2.13E-05	5.01E-02	3.02E-11	3.02E-11	3.02E-11
F30	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
+/-	29/0	26/3	29/0	26/3	23/6	28/1	29/0	28/1

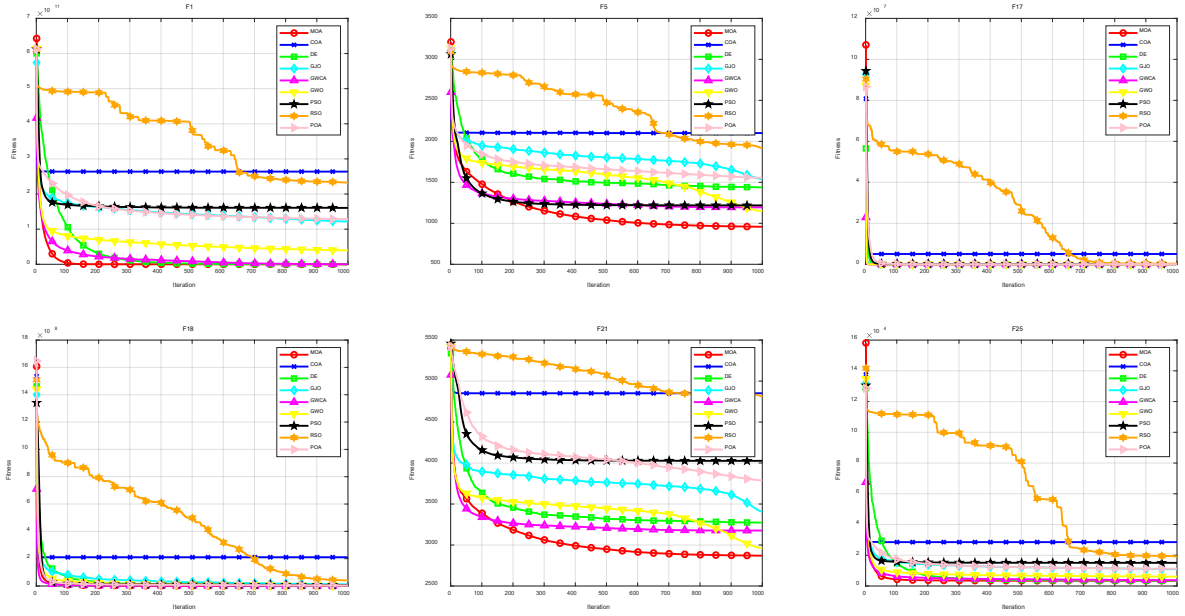

Figure 1. Convergence curves of the different algorithms on the 100-dimensional CEC'17 functions.

4. Engineering Design Problems

In this section, three classical real-world engineering optimization problems—the Three-bar truss design problem, the Gear train design problem, and the Rolling element bearing design problem—are selected to evaluate the optimization performance of the MOA algorithm through simulation. These problems require obtaining the optimal solution while satisfying multiple constraints. However, the presence of constraints and the limited feasible search space often significantly affect the performance of optimization algorithms. To effectively handle constrained optimization problems, a Constraint Handling Technique (CHT) is introduced in this study. This method penalizes constraint-violating solutions by incorporating penalty terms into the objective function, thereby achieving a balance between objective optimization and constraint satisfaction. To ensure experimental fairness, the population size of all algorithms is set to 60, the maximum number of iterations is fixed at 1000, and each algorithm is independently executed 30 times.

4.1. Three-bar truss design problem

The Three-bar truss design problem is an important engineering optimization problem. Its objective is to determine the ideal values of three variables to minimize the weight of the three-bar truss while satisfying specific constraints. The mathematical formulation of the problem is as follows:

$$
\begin{aligned}
&\textit{Minimize } f\left(\bar{x}\right) = l\left(x_2 + 2\sqrt{2}x_1\right) \\
&\textit{Subject to } g_1\left(\bar{x}\right) = \frac{x_2}{2x_2x_1 + \sqrt{2}x_1^2}P - \sigma \le 0, \\
&g_2\left(\bar{x}\right) = \frac{x_2 + \sqrt{2}x_1}{2x_2x_1 + \sqrt{2}x_1^2}P - \sigma \le 0, \\
&g_3\left(\bar{x}\right) = \frac{1}{x_1 + \sqrt{2}x_2}P - \sigma \le 0, \\
&l = 100, P = 2, \sigma = 2. \\
&\textit{With bounds } 0 \le x_1, x_2 \le 1.
\end{aligned}
\tag{16}
$$

Table 7. The best results of various optimizers for three-bar truss design problem.

	X1	X2	Optimum
MOA	0.7886648296940418	0.4082774722027158	263.8958468842678258
COA	0.789202	0.406761	263.8960000000 00015
DE	0.788675	0.408248	263.8960000000 00015
GJO	0.7882972896139562	0.4093279687389092	263.8969405046926227
GWCA	0.7885	0.408743	263.8960000000 00015
GWO	0.789382	0.406253	263.8960000000 00015
PSO	0.78855	0.408602	263.8960000000 00015
RSO	0.7966692260700101	0.3864331245811944	263.9753973048165676
POA	0.788675	0.408248	263.8960000000 00015

Table 7 presents the best experimental results of the MOA algorithm compared with the benchmark algorithms. It is worth noting that, for specific variables, the MOA algorithm achieved the lowest cost. Compared with the eight benchmark algorithms (COA, DE, GJO, GWCA, GWO, PSO, RSO, and POA), MOA demonstrates stronger competitiveness in

solving the Three-bar Truss Design problem.

4.2. Gear train design problem

Gear train design is an important engineering problem, with the core objective of minimizing the gear ratio in a compound gear train configuration. The specific details of this problem are described as follows.

Table 8 presents the best experimental results of the MOA algorithm and the comparison algorithms. The results indicate that for the gear train design problem, the MOA algorithm demonstrates comparable competitiveness with COA, GJO, GWO, PSO, and POA, and outperforms DE, GWCA, and RSO.

$$Minimize\ f\left(\vec{x}\right)=\left(i_{trg}-i_{tot}\right)^2=\left(\frac{1}{6.931}-\frac{x_1x_2}{x_3x_4}\right)^2 \quad (17)$$

$$With\ bounds\ 12\le x_i\le 60, i=1,2,...,4.$$

Table 8. The best results of various optimizers for gear train design problem.

	X1	X2	X3	X4	Optimum
MOA	49.247428320515958	15.5257717011500329	18.8966760344044786	42.6056459177648819	0.0000000000027009
COA	48.6084000000000032	16.1424999999999983	19.3171999999999997	43.3599000000000032	0.0000000000027009
DE	42.7304999999999993	16.4923999999999999	18.6085999999999991	48.5548000000000002	0.0000000000027009
GJO	49.3067232494204220	19.1776149715386381	15.8498153814671277	43.1338387612098444	0.0000000000027009
GWCA	53.2631000000000014	13.3132999999999999	20.1779000000000011	34.1293000000000006	0.0000000000230782
GWO	43.0870000000000033	18.6020000000000003	15.5954999999999995	49.4551999999999978	0.0000000000027009
PSO	43.2357999999999976	18.8232999999999997	16.4819999999999993	48.6803999999999988	0.0000000000027009
RSO	26.7662170813886782	12	12	36.7217987030912312	0.0000000182738024
POA	43.285499999999999	16.0253000000000014	19.2551999999999985	49.0236999999999981	0.0000000000027009

4.3. Rolling element bearing design problem

The rolling element bearing design problem is regarded as one of the most complex problems in the engineering field, involving ten optimization variables. Its main objective is to determine the values of these variables in order to maximize the dynamic load-carrying capacity while satisfying nine assembly and geometric constraints. The specific details of the problem are as follows: Table 9 presents the best experimental results of the MOA algorithm compared with the benchmark algorithms. The results show that, compared with the eight benchmark algorithms (COA, DE, GJO, GWCA, GWO, PSO, RSO, and POA), the proposed algorithm achieves higher optimization accuracy in solving the rolling element bearing design problem.

$$\begin{aligned}
&Maximize\ C_d=f_c U^{2/3} D_b^{1.8}, if\ D\le 25.4\ \ mm, C_d=3.647 f_c U^{2/3} D_b^{1.4}, if\ D\le 25.4\ \ mm,\\
&Subject\ to\ g_1(\vec{c})=\frac{\phi_0}{2\sin^{-1}(D_b/D_m)}\le 0, g_2(\vec{u})=2D_b-K_{D\min}(D-d)>0, g_3(\vec{u})=K_{D\max}(D-d)-2D_b>0,\\
&g_4(\vec{u})=\xi B_w-D_b\le 0, g_5(\vec{u})=D_m-0.5(D-d)\ge 0, g_6(\vec{u})=(0.5+e)(D+d)-D_m\ge 0,\\
&g_7(\vec{u})=0.5(D-D_m-D_b)-\varepsilon D_b\ge 0, g_8(\vec{u})=f_i\ge 0.515, g_9(\vec{u})=f_0\ge 0.515,\\
&Where\ f_c=37.91\left[1+\left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i(2f_0-1)}{f_0(2f_i-1)}\right)^{0.41}\right\}^{10/3-0.3}\right]\left(\frac{\gamma^{0.3}(1-\gamma)^{1.39}}{(1+\gamma)^{1/3}}\right)\left(\frac{2f_i}{2f_i-1}\right)^{0.41}\\
&x=\left[\left\{(D-d)/2-3(T/4)\right\}^2+\left\{D/2-T/4-D_b\right\}^2-\left\{d/2+T/4\right\}^2\right],\\
&y=2\left\{(D-d)/2-3(T/4)\right\}\left\{D/2-T/4-D_b\right\}, \phi_0=2\pi-\cos^{-1}\left(\frac{u}{y}\right), \gamma=\frac{D_b}{D_m}, f_i=\frac{r_i}{D_b}, f_0=\frac{r_0}{D_b},\\
&T=D-d-2D_b, D=160, d=90, B=30, r_i=r_0=11.033, 0.5(D+d)\le D_m\le 0.6(D+d), 0.6\le\xi\le 0.85\\
&0.15(D-d)\le D_b\le 0.45(D-d), 4\le Z\le 50, 0.515\le f_i\ and\ f_0\le 0.6, 0.4\le K_{D\min}\le 0.5, 0.6\le K_{D\max}\le 0.7, 0.3\le\varepsilon\le 0.4, 0.02\le e\le 0.1.
\end{aligned} \quad (18)$$

Table 9. The best results of various optimizers for rolling element bearing design problem.

	D_m	D_b	Z	f_i	f_0	$K_{D\min}$	$K_{D\max}$	ε	e	ξ	Optimum
MOA	125	21.8749974375242751	49.5695147122494575	0.515	0.515	0.4	0.6393927016082795	0.3	0.0201628820518617	0.6000117709449898	243431.4346000999794342
COA	125	21.8740999999999985	50	0.515	0.515	0.4	0.7	0.3	0.1	0.6	243413
DE	125	21.875	49.6345000000000027	0.515	0.515	0.471335	0.698163	0.3	0.0878019	0.664354	243431
GJO	125	21.8748824775102158	50	0.515	0.5153647595397965	0.4055496970355618	0.6957014445787131	0.3	0.0200420727716843	0.6280478470694437	243429.0034151372674387
GWCA	125	21.875	49.5581999999999994	0.515	0.515	0.5	0.699074	0.3	0.0660192	0.600718	243431
GWO	125	21.8749000000000002	49.7543000000000006	0.515	0.516706	0.451679	0.687107	0.3	0.0705661	0.615351	243428
PSO	125	21.875	49.8277000000000001	0.515	0.515	0.5	0.7	0.3	0.1	0.6	243431
RSO	125	18.2678543205019146	34.8014604047787657	0.515	0.515	0.4	0.6	0.3	0.02	0.6	139032.3499106741219293
POA	125	21.875	50	0.515	0.515	0.5	0.7	0.3	0.0677655	0.6	243431

5. Conclusion and Future Work

This paper proposes a novel meta-heuristic optimization algorithm called the Muskrat Optimization Algorithm (MOA). The algorithm is inspired by the collective behavioral characteristics and spatial memory mechanisms exhibited by muskrats during foraging and nest-building in natural environments. In MOA, the population is divided into three exploratory paths for each muskrat and separated into two stages: foraging and nesting. During the early foraging stage, an opportunistic foraging strategy is adopted, while a defensive foraging strategy is used in the later stage. During

the nesting stage, a trial nesting strategy is employed initially, followed by a refinement nesting strategy in the later stage. These strategies enhance global search capability and effectively avoid being trapped in local optima.To evaluate the performance of MOA, benchmark tests were conducted using 29 functions from the CEC2017 test suite in 100-dimensional space, and the experimental results were analyzed using the Wilcoxon rank-sum test. Finally, benchmark tests were performed on three classical engineering design problems. The experimental results and statistical analyses indicate that MOA achieves superior search accuracy compared with the comparison algorithms (COA, DE, GJO, GWCA, GWO, PSO, RSO, and POA).However, the algorithm still has areas for improvement: on hybrid functions, the advantages of MOA in terms of convergence speed and accuracy are not significant compared with other algorithms. Therefore, enhancing the optimization capability of MOA remains an important research direction. In the future, we aim to develop novel search strategies to further improve its exploration ability. Additionally, MOA can be extended to a multi-objective version to address complex multi-objective optimization scenarios.

References

[1] Gao H, Zhang Q. Alpha evolution: An efficient evolutionary algorithm with evolution path adaptation and matrix generation[J]. Engineering Applications of Artificial Intelligence, 2024, 137: 109202.

[2] Dhiman G, Garg M, Nagar A, et al. A novel algorithm for global optimization: Rat Swarm Optimizer[J]. Journal of Ambient Intelligence and Humanized Computing, 2021, 12(8): 8457-8482.

[3] Liu Q, Wu L, Xiao W, et al. A novel hybrid bat algorithm for solving continuous optimization problems[J]. Applied Soft Computing, 2018, 73: 67-82.

[4] Bouaouda A, Sayouti Y. Hybrid meta-heuristic algorithms for optimal sizing of hybrid renewable energy system: a review of the state-of-the-art[J]. Archives of Computational Methods in Engineering, 2022, 29(6): 4049.

[5] Bouaouda A, Hashim F A, Sayouti Y, et al. Pied kingfisher optimizer: a new bio-inspired algorithm for solving numerical optimization and industrial engineering problems[J]. Neural Computing and Applications, 2024, 36(25): 15455-15513.

[6] Jamil M, Yang X S. A literature survey of benchmark functions for global optimisation problems[J]. International Journal of Mathematical Modelling and Numerical Optimisation, 2013, 4(2): 150-194.

[7] Jackson W G, Özcan E, John R I. Move acceptance in local search metaheuristics for cross-domain search[J]. Expert Systems with Applications, 2018, 109: 131-151.

[8] Kennedy J, Eberhart R. Particle swarm optimization[C]//Proceedings of ICNN'95-international conference on neural networks. ieee, 1995, 4: 1942-1948.

[9] Mirjalili S, Mirjalili S M, Lewis A. Grey wolf optimizer[J]. Advances in engineering software, 2014, 69: 46-61.

[10] Dorigo M, Maniezzo V, Colorni A. Ant system: optimization by a colony of cooperating agents[J]. IEEE transactions on systems, man, and cybernetics, part b (cybernetics), 1996, 26(1): 29-41.

[11] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm[J]. Journal of global optimization, 2007, 39(3): 459-471.

[12] Yang X S, Deb S. Cuckoo search via Lévy flights[C]//2009 World congress on nature & biologically inspired computing (NaBIC). Ieee, 2009: 210-214.

[13] Mirjalili S, Lewis A. The whale optimization algorithm[J]. Advances in engineering software, 2016, 95: 51-67.

[14] Zhao W, Wang L, Mirjalili S. Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications[J]. Computer Methods in Applied Mechanics and Engineering, 2022, 388: 114194.

[15] Abdel-Basset M, Mohamed R, Jameel M, et al. Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems[J]. Knowledge-Based Systems, 2023, 262: 110248.

[16] Dehghani M, Trojovský P. Osprey optimization algorithm: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems[J]. Frontiers in Mechanical Engineering, 2023, 8: 1126450.

[17] MiarNaeimi F, Azizyan G, Rashki M. Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems[J]. Knowledge-Based Systems, 2021, 213: 106711.

[18] Reeves C R. Genetic algorithms[M]//Handbook of metaheuristics. Springer, Boston, MA, 2010: 109-139.

[19] Storn R, Price K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of global optimization, 1997, 11(4): 341-359.

[20] Civicioglu P. Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm[J]. Computers & Geosciences, 2012, 46: 229-247.

[21] Koza J R. Genetic programming as a means for programming computers by natural selection[J]. Statistics and computing, 1994, 4(2): 87-112.

[22] Ghaemi M, Feizi-Derakhshi M R. Forest optimization algorithm[J]. Expert systems with applications, 2014, 41(15): 6676-6687.

[23] Salimi H. Stochastic fractal search: a powerful metaheuristic algorithm[J]. Knowledge-based systems, 2015, 75: 1-18.

[24] Simon D. Biogeography-based optimization[J]. IEEE transactions on evolutionary computation, 2008, 12(6): 702-713.

[25] Kuo R J, Zulvia F E. The gradient evolution algorithm: A new metaheuristic[J]. Information Sciences, 2015, 316: 246-265.

[26] Beyer H G, Schwefel H P. Evolution strategies–a comprehensive introduction[J]. Natural computing, 2002, 1(1): 3-52.

[27] Yao X, Liu Y, Lin G. Evolutionary programming made faster[J]. IEEE Transactions on Evolutionary computation, 1999, 3(2): 82-102.

[28] Kirkpatrick S, Gelatt Jr C D, Vecchi M P. Optimization by simulated annealing[J]. science, 1983, 220(4598): 671-680.

[29] Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: a gravitational search algorithm[J]. Information sciences, 2009, 179(13): 2232-2248.

[30] Alatas B. ACROA: artificial chemical reaction optimization algorithm for global optimization[J]. Expert Systems with Applications, 2011, 38(10): 13170-13180.

[31] Hatamlou A. Black hole: A new heuristic optimization approach for data clustering[J]. Information sciences, 2013, 222: 175-184.

[32] Kaveh A, Talatahari S. A novel heuristic optimization method: charged system search[J]. Acta mechanica, 2010, 213(3): 267-289.

[33] Kaveh A, Khayatazad M. A new meta-heuristic method: ray optimization[J]. Computers & structures, 2012, 112: 283-294.

[34] Abdechiri M, Meybodi M R, Bahrami H. Gases Brownian motion optimization: an algorithm for optimization (GBMO)[J]. Applied Soft Computing, 2013, 13(5): 2932-2946.

[35] Kaveh A, Mahdavi V R. Colliding bodies optimization: a novel meta-heuristic method[J]. Computers & Structures, 2014, 139: 18-27.

[36] Kashan A H. A new metaheuristic for optimization: optics inspired optimization (OIO)[J]. Computers & operations research, 2015, 55: 99-125.

[37] Eskandar H, Sadollah A, Bahreininejad A, et al. Water cycle algorithm–A novel metaheuristic optimization method for solving constrained engineering optimization problems[J]. Computers & Structures, 2012, 110: 151-166.

[38] Rao R V, Savsani V J, Vakharia D P. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems[J]. Computer-aided design, 2011, 43(3): 303-315.

[39] Sadollah A, Bahreininejad A, Eskandar H, et al. Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems[J]. Applied Soft Computing, 2013, 13(5): 2592-2612.

[40] Gandomi A H. Interior search algorithm (ISA): a novel approach for global optimization[J]. ISA transactions, 2014, 53(4): 1168-1183.

[41] Ghorbani N, Babaei E. Exchange market algorithm[J]. Applied soft computing, 2014, 19: 177-187.

[42] Cheng S, Qin Q, Chen J, et al. Brain storm optimization algorithm: a review[J]. Artificial Intelligence Review, 2016, 46(4): 445-458.

[43] Zhao W, Wang L, Zhang Z, et al. Quadratic Interpolation Optimization (QIO): A new optimization algorithm based on generalized quadratic interpolation and its applications to real-world engineering problems[J]. Computer Methods in Applied Mechanics and Engineering, 2023, 417: 116446.

[44] Zhao S, Zhang T, Cai L, et al. Triangulation topology aggregation optimizer: A novel mathematics-based meta-heuristic algorithm for continuous optimization and engineering applications[J]. Expert Systems with Applications, 2024, 238: 121744.

[45] Mirjalili S. SCA: a sine cosine algorithm for solving optimization problems[J]. Knowledge-based systems, 2016, 96: 120-133.

[46] Fadakar E, Ebrahimi M. A new metaheuristic football game inspired algorithm[C]//2016 1st conference on swarm intelligence and evolutionary computation (CSIEC). IEEE, 2016: 6-11.

[47] Azizi M, Baghalzadeh Shishehgarkhaneh M, Basiri M, et al. Squid Game Optimizer (SGO): a novel metaheuristic algorithm[J]. Scientific reports, 2023, 13(1): 5373.

[48] Bertsimas D, Tsitsiklis J. Simulated annealing[J]. Statistical science, 1993, 8(1): 10-15.

[49] Boussaïd I, Lepagnot J, Siarry P. A survey on optimization metaheuristics[J]. Information sciences, 2013, 237: 82-117.

[50] Wolpert D H, Macready W G. No free lunch theorems for optimization[J]. IEEE transactions on evolutionary computation, 2002, 1(1): 67-82.

[51] Dehghani M, Montazeri Z, Trojovská E, et al. Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems[J]. Knowledge-based systems, 2023, 259: 110011.

[52] Chopra N, Ansari M M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications[J]. Expert Systems with Applications, 2022, 198: 116924.

[53] Guan Z, Ren C, Niu J, et al. Great Wall Construction Algorithm: A novel meta-heuristic algorithm for engineer problems[J]. Expert Systems with Applications, 2023, 233: 120905.

[54] Trojovský P, Dehghani M. Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications[J]. Sensors, 2022, 22(3): 855.