Worker Ant Optimization: An Algorithm for Complex Problems

Tian Naiyue

School of Big Data and Statistics, Guizhou University of Finance and Economics, Guiyang, Guizhou Province

Received: 04 September 2024/ Revised: 10 September 2024/ Accepted: 17 September 2024/ Published: 30-09-2024

Abstract— *The article introduces a novel metaheuristic algorithm called the Worker Ant Optimization (WAO) algorithm. This algorithm is mathematically modeled based on five natural behaviors of worker ants: avoiding danger, foraging, approaching food, decomposing food, and transporting food. The performance of WAO was evaluated using 23 classical test functions and compared with results from seven well-known metaheuristic algorithms. Simulation results demonstrate that the WAO algorithm exhibits significant advantages in terms of convergence speed, avoidance of local optima, and optimization accuracy. To assess the effectiveness of WAO in practical applications, the method was applied to three classical engineering design problems, validating the engineering applicability of the WAO optimization algorithm. WAO effectively explores the decision space and performs well across various evaluation metrics, demonstrating its capability to effectively address challenges in practical applications.*

Keywords— Classical test functions; Constrained optimization; Engineering constraint issues; Swarm optimization; WAO.

I. INTRODUCTION

The term "optimization problem" refers to a situation where the goal is to find feasible solutions under given constraints [1]. It involves the process of seeking optimal values for specific system parameters within existing solutions, aiming to meet a certain criterion at minimal cost [2]. Such criteria could include maximizing profit, minimizing costs, maximizing efficiency, or minimizing risks. Optimization problems find widespread applications in fields such as engineering, economics, management, and computer science.

Typically, an optimization problem comprises several elements: decision variables, constraints, and an objective function [3]. In practical applications, optimization problems can be highly complex, often involving conflicting objective functions and numerous constraints. To address these challenges, enhance system performance, and reduce computational costs, various optimization methods have been developed. These methods are generally categorized into two classes: mathematical methods and metaheuristic algorithms.

In academic and applied contexts, optimizing system parameters involves leveraging these methodologies to achieve desired outcomes efficiently and effectively.

In various real-world applications, particularly in fields like artificial intelligence and machine learning, optimization problems often exhibit discrete, unconstrained, or non-continuous characteristics [4]. Traditional mathematical programming methods rely on gradients, are sensitive to initial conditions, and struggle to solve such complex problems [5]. This limitation has spurred the development of metaheuristic algorithms [6].

Metaheuristic algorithms simulate the behavior of biological individuals or populations in nature to explore and optimize solution spaces. They are based on principles of simulation and natural inspiration for global optimization [7][8]. The optimization process begins by initializing a set of random feasible solutions in the problem space. Subsequently, these solutions are iteratively updated and improved according to algorithmic instructions. Upon completion, the algorithm identifies the optimal solution among the candidate solutions [9].

Due to their nature as stochastic searches, metaheuristic algorithms cannot guarantee finding the globally optimal solutions. However, they often converge near-optimal solutions that are accepted as quasi-optimal [10].

Metaheuristic algorithms draw inspiration from problem-solving methods observed in nature, such as cooperative behaviors among fish, birds, and ants. This intelligence emerges from interactions among simple individuals in a group, without the need for centralized control. Group members follow basic behavioral rules, exhibiting collective intelligent behavior through interaction. Algorithms based on simulating natural behaviors like population reproduction, hunting, and migration can solve complex optimization problems [11]. This research field is known as swarm intelligence [12]. In recent years, these algorithms have found wide applications in areas like image processing, path planning, and data mining, yielding significant research outcomes. Common swarm intelligence algorithms include Particle Swarm Optimization (PSO) [13], Sine Cosine Algorithm (SCA) [14], Raccoon Optimization Algorithm (ROA) [15], Genetic Algorithm (GA) [16], Harris Hawks Optimization (HHO) [17], Artificial Bee Colony (ABC) [18], Grey Wolf Optimizer (GWO) [19], and Whale Optimization Algorithm (WOA) [20].

This article proposes a novel metaheuristic algorithm—the Worker Ant Optimization (WAO), which simulates natural behaviors of worker ants. It evaluates WAO's performance in solving optimization problems using 23 classic test functions and compares its optimization results with seven well-known metaheuristic algorithms. The study tests WAO's performance in solving practical optimization problems in three engineering design scenarios.

The article introduces the WAO algorithm and models it in Section 2. Section 3 investigates the efficiency of WAO through simulation studies and analysis of practical applications. Section 4 examines WAO's efficiency in addressing real-world optimization problems. Finally, conclusions are drawn, and several future research directions are suggested.

II. WORKER ANT OPTIMIZATION ALGORITHM

2.1 Source of inspiration:

Worker ants in the black ant society undertake the crucial task of searching for and transporting food. The foraging behavior of worker ants demonstrates a high level of adaptability and intelligence. They are capable of adjusting their foraging strategies based on environmental changes, expanding their foraging range in times of food scarcity, and regulating the length and speed of foraging queues to meet different foraging needs.

During the food search process, worker ants release a chemical substance called pheromones to mark their foraging paths, which come in two types. When encountering natural enemies, worker ants release an alarm-type pheromone; when discovering food, they release a trail-type pheromone to guide other worker ants.

Other worker ants perceive these pheromones through their sense of smell and respond accordingly to their types. When receiving alarm-type pheromones, worker ants will avoid paths marked as dangerous; when receiving trail-type pheromones, worker ants will gather along the path indicated to locate the food. In addition to using pheromones to find paths, worker ants also observe the surrounding environment, memorize prominent landmarks, and use them to determine their position and direction, thereby adjusting their routes to find the optimal path.

After finding food, worker ants transport it back to the nest. If the food is too large, worker ants employ a decomposition strategy, cutting off and carrying back parts of the food until the task is completed.

Through this division of labor and pheromone communication system, worker ants in the black ant colony efficiently search for and transport food, ensuring the survival and reproduction of the entire ant colony. This behavioral strategy enables worker ants to cooperate to accomplish complex tasks and respond appropriately to different situations.

2.2 Algorithm initialization process:

The Ant Colony Optimization algorithm is based on the behavior of ants. In the WAO algorithm, ants are candidate solutions to the optimization problem, meaning that the position updates of each ant in the search space represent the values of decision variables. Therefore, each ant is represented as a vector, and the ant colony is mathematically characterized by a matrix. Similar to traditional optimization algorithms, the initialization stage of WAO involves generating random initial solutions. In this step, the following formula is used to generate vectors of decision variables:

$$x_{i,j} = lb_j + r * \left(ub_j - lb_j\right), \quad i = 1,2,\ldots,N, \quad j = 1,2,\ldots,m \tag{1}$$

In this context, $x_{i,j}$ represents the value of the j^{th} decision variable of the i^{th} candidate solution, r is a random number within the range of 0 to 1, and lb and ub denote the lower and upper bounds, respectively, of the j^{th} decision variable. The ant colony population can be mathematically represented by the matrix hh, referred to as the population matrix.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N*m} = \begin{bmatrix} X_{1,1} & \cdots & X_{1,j} & \cdots & X_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ X_{i,1} & \cdots & X_{i,j} & \cdots & X_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ X_{N,1} & \cdots & X_{N,j} & \cdots & X_{N,m} \end{bmatrix}_{N*m} \tag{2}$$

2.3 Mathematical model:

In WAO, the process of updating the position of worker ants (candidate solutions) is based on modeling four natural behaviors of the ants. These behaviors include:

a) Avoidance of danger behavior

b) Foraging behavior

c) Attraction to food behavior

d) Decomposition and transport of food behavior

2.3.1 Avoidance of danger behavior

This stage is modeled based on the danger avoidance behavior of worker ants. If a worker ant encounters a predator while searching for food, it releases an alarm pheromone. When other worker ants detect this alarm pheromone, they will avoid the route. The route where the predator is encountered is assumed to be the position of the best member. To simulate the behavior of the ants moving away in the opposite direction, an inverse learning update strategy based on the principle of convex lens imaging is used.

Inverse learning, proposed by Tizhoosh, is an optimization mechanism that expands the search range by calculating an inverse solution based on the current solution during the population optimization process. The current solution and the inverse solution's objective function values are compared, and the better solution is selected for the next iteration [21]. However, the inverse solution generated by the inverse learning strategy is at a fixed distance from the current solution, lacking randomness, and thus cannot effectively enhance the diversity of the population within the search space. Combining optimization algorithms with inverse learning can effectively improve the optimization performance of the algorithm. However, since the value generated by the inverse learning strategy is fixed, it cannot effectively help the algorithm escape local optima in the later stages of iteration and lacks randomness. Therefore, the lens imaging principle is introduced into the inverse learning strategy. As shown in the figure below, taking a two-dimensional space as an example, $[a, b]$ represents the search range of the solution, and the y axis represents the convex lens. Suppose there is an object p with height h and a projection on the $x$axis of x^*; this object is imaged by the convex lens on the other side as an inverted real image $p^*$with height $h^*$and a projection on the axis of x^*. According to the principle of convex lens imaging:

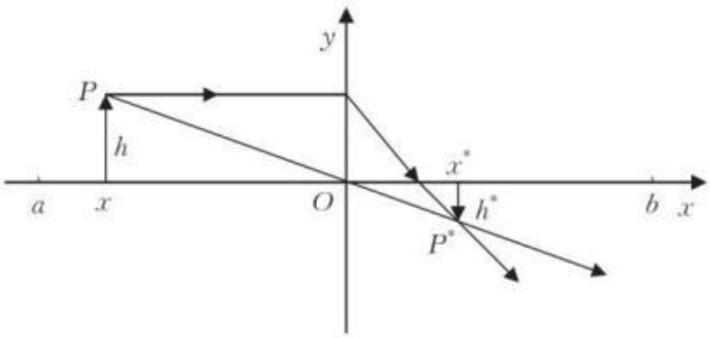

FIGURE 1: Schematic Diagram of the Reverse Learning Strategy for Lens Imaging

$$\frac{\frac{a+b}{2}-x}{x'-\frac{a+b}{2}} = \frac{h}{h'} \tag{3}$$

Let $n = \frac{h}{h'}$, then the equation can be rewritten as:

$$x' = \frac{a+b}{2} + \frac{a+b}{2n} - \frac{x}{n} \tag{4}$$

Further, the equation can be rewritten as the optimization algorithm update strategy:

$$n = \frac{5}{e^{-5.5*\frac{t}{t_{max}}}} \tag{5}$$

$$X_i^{t+1} = \frac{lb+ub}{2} + \frac{lb+ub}{2n} - \frac{X_{best}^t}{n} \quad (6)$$

2.3.2 Foraging behavior:

$$X_i^{t+1} = \begin{cases} r_1 * lb + r_2 * (ub - X_i^t), & p < 0.5 \\ X_i^t * e^{-\left(\frac{t}{r_3 * t_{max}}()^a\right)} \end{cases} \quad (7)$$

During foraging, worker ants adjust their strategies and intensity based on food availability. When food is scarce, they expand their foraging range and adjust their foraging intensity according to the length and speed of the foraging trail. The random walk strategy simulates the ants' behavior of randomly searching for food, while adaptive weights model the changes in foraging speed and intensity. The random walk strategy moves randomly within the search space to find the optimal solution. In each update, two random numbers within the interval [0,1] are generated to adjust the current solution's position, aiming to find better solutions. This randomness enhances the global search capability of the algorithm and helps avoid local optima. The adaptive weight strategy decreases exponentially with the number of iterations, allowing strong global search capabilities in the early stages and improving local search capabilities as the algorithm approaches the optimal solution. Here, t is the current iteration count, t_{max} is the maximum number of iterations, r_1, r_2, and $r_3$are random numbers within the range of 0 to 1, $lb$and $ub$are the bounds of the search space representing the ants' activity range, and $a$is a constant.

2.3.3 Approach behavior:

$$C_1 = \left(3 - e^{\left(\frac{t}{t_{max}}()^3\right)}()_4\right) \quad (8)$$

$$C_2 = \frac{X_{g\_best} + X_{best}^t}{2} \quad (9)$$

$$E = lb + r_5 * (ub - lb) \quad (10)$$

$$X_i^{t+1} = \begin{cases} X_i^t + C_1 * (C_2 - X_i^t), & p < 0.5 \\ X_i^t + r_6 * (X_i^t - E), & else \end{cases} \quad (11)$$

In this section, the behavior of worker ants approaching the food source is simulated. The ants use pheromones to follow and approach the food location denoted by C_2, and move with a step size determined by a nonlinear parameter $C_1$to get closer to the food source. The formula dynamically transitions the population from global search to local exploitation. In the early stages of the algorithm, a larger step size is maintained to enhance global search capability, while in the later stages, a smaller step size improves the precision of local development. In addition to using pheromones, ants also adjust their direction based on environmental observations and noticeable landmarks to find the best route to the food. Here, X_{best}^trepresents the position of the best individual in the population at the t^{th} iteration, $X_{g_best}$is the position of the globally best individual so far, $r_4$is a random number within the 2D [0,1] range, $r_5$is a random number between 0 and 1, and $r_6$is a random number following a standard normal distribution.

2.3.4 Food decomposition and transport:

$$D = 0.2 * \left(1 - e^{-5*\left(\frac{t}{t_{max}}\right)^3}\right) \quad (12)$$

$$X_i^{t+1} = \begin{cases} D * X_i^t - r_7, & p < 0.5 \\ X_{best}^t + r_8 * (X_{best}^t - X_i^t), & else \end{cases} \quad (13)$$

When the food is large, worker ants choose to cut it into smaller pieces for easier transport. The parameter Drepresents the proportion of food cut by the ants relative to the current food volume. Initially, when the food is large, the cutting proportion is small. As the cutting progresses and the food volume decreases, the cutting proportion increases until the remaining food can be transported directly. The mathematical formula representing the ants' behavior in transporting food to the nest approaches the optimal value. Random perturbations, generated by a normal distribution, are applied to the current optimal solution, with the perturbation magnitude determined by the difference between the current optimal solution and the current solution. This strategy leverages information from the current optimal solution and introduces randomness to increase diversity in the search process, preventing convergence to local optima and enhancing global search convergence and robustness. Here, X_{best}^trepresents the position of the best individual in the population at the t^{th} iteration, $r_7$is a random number in the [0,1] interval, $r_8$is a random number following a standard normal distribution, $t$is the current iteration count, and $t_{max}$is the maximum number of iterations.

2.3.5 Repetition process and flowchart of WAO:

The specific optimization scheme is shown in Figure 2. After updating the positions of all the ants in the search space based on the first and second stages, the WAO iteration is completed. The population update process is repeated using equations (4), (5), (9) and (11) until the final iteration of the algorithm. Once a WAO run is completed, the best solutions obtained throughout all iterations of the algorithm are returned as output.

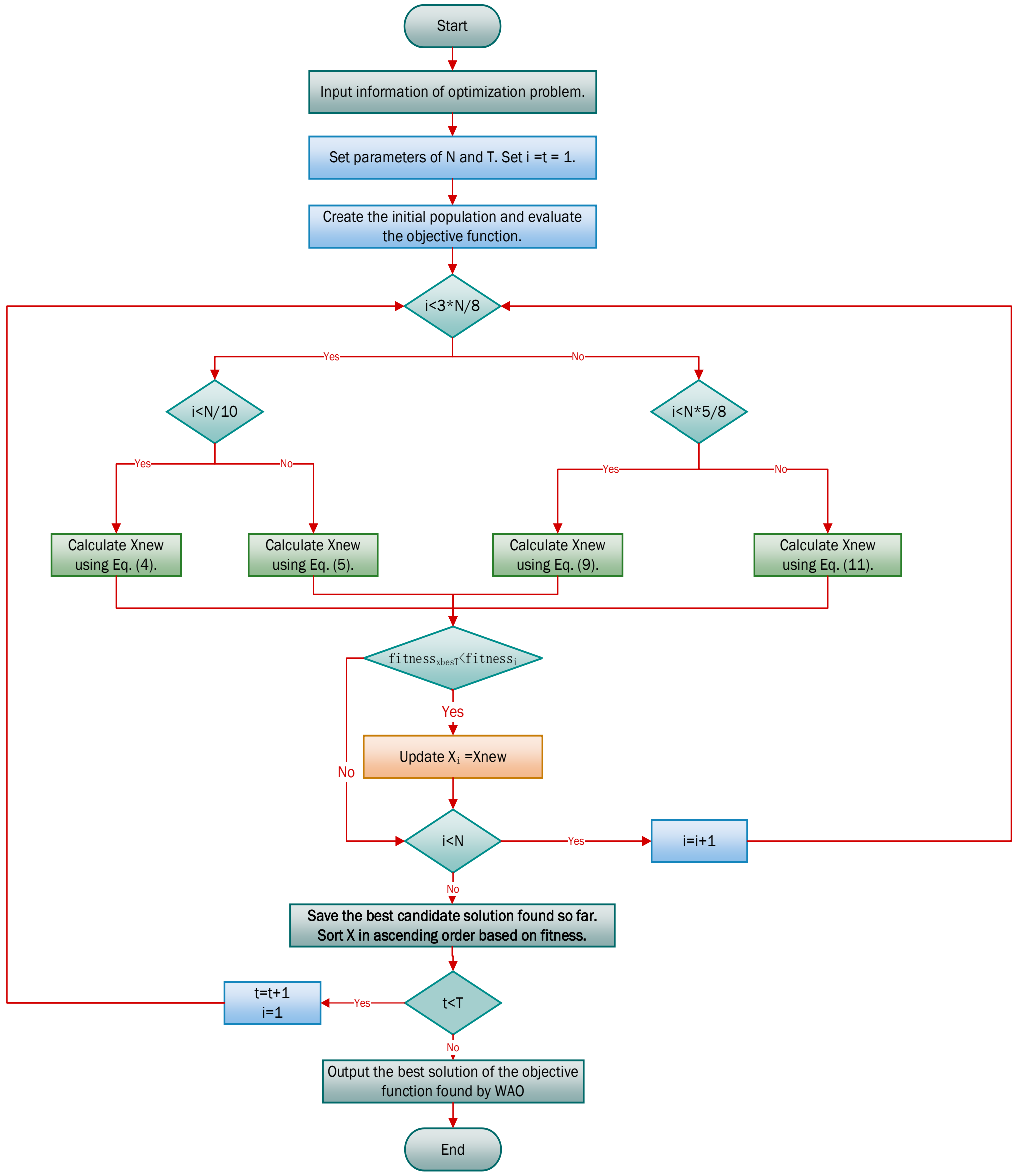

FIGURE 2: Flowchart of the worker ant optimization algorithm

III. EXPERIMENT SIMULATION AND ANALYSIS

To thoroughly validate the superiority of the WAO algorithm, this study tests it using 23 classical test functions, which effectively demonstrate the algorithm's optimization capability. Functions F1-F7 are unimodal test functions with a single theoretical optimal solution, used to assess the algorithm's convergence speed and accuracy. Functions F8-F23 are multimodal test functions with multiple local optima and one theoretical global optimum, used to evaluate the algorithm's global search capability and its ability to avoid local optima. Tables 1-3 provide the specific function forms, while Fig. 3 shows the function plots for F1-F23.

TABLE 1
UNIMODAL BENCHMARK FUNCTIONS

Function	Dim	Range	fmin
$F_1(x) = \sum_{i=1}^{n} x_i^2$	50	$[-100,100]$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	50	$[-10,10]$	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_i \right)^2$	50	$[-100,100]$	0
$F_4(x) = max\{\lvert x_i \rvert, 1 \le i \le n\}$	50	$[-100,100]$	0
$F_5(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2 \right]$	50	$[-30,30]$	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	50	$[-100,100]$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random0,1)$	50	$[-1.28,1.28]$	0

TABLE 2
MULTIMODAL BENCHMARK FUNCTIONS

Function	Dim	Range	fmin
$F_8(x) = \sum_{i=1}^{n} -x_i sin\left(\sqrt{\lvert x_i \rvert}\right)$	50	$[-500,500]$	-20949
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10cos(2\pi x_i) + 10 \right]$	50	$[-5.12,5.12]$	0
$F_{10}(x) = -20exp\left(-0.2 \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i) \right) + 20 + e$	50	$[-32,32]$	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{(i)}}\right) + 1$	50	$[-600,600]$	0
$F_{12}(x) = \frac{\pi}{n}\left\{ 10sin(\pi y_i) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + +10sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\} + \sum_{i=1}^{n} \mu\,(x_i, 10,100,4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $\mu(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a \le x_i \le a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	50	$[-50,50]$	0
$F_{13}(x) = 0.1\left\{ sin^2(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + sin^2(2\pi x_n)] \right\} + \sum_{i=1}^{n} \mu\,(x_i, 5,100,4)$	50	$[-50,50]$	0

To better assess the improved algorithm's performance, comparisons are made with currently mainstream and newly emerging optimization algorithms, including COA, GSA, WOA, GWO, GA, PSO, and HHO. For fairness, all algorithms are tested with the same dimensionality, a population size of 50, and each function is solved independently 30 times.

TABLE 3
FIXED-DIMENSION MULTIMODAL BENCHMARK FUNCTIONS

Function	Dim	Range	fmin
$F_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	$[-65,65]$	1
$F_{15}(x)=\sum_{j=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_i^2x_3+x_4}\right]^2$	4	$[-5,5]$	0.0003075
$F_{16}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_1^2+4x_1^4$	2	$[-5,5]$	-1.0316285
$F_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)+$ $10\left(1-\frac{1}{8\pi}\right)cos\,x_1+10$	2	$[-5,5]$	0.398
$F_{18}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-$ $14x_2+16x_1x_2+3_2^2)]*[30+(2x_1-3x_2)^2*$ $(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	$[-2,2]$	3
$F_{19}(x)=-\sum_{i=1}^{4}c_iexp\left(-\sum_{j3=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$	3	$[1,3]$	-3.86
$F_{20}(x)=-\sum_{i=1}^{4}c_iexp\left(-\sum_{j3=1}^{6}a_{ij}(x_j-p_{ij})^2\right)$	6	$[0,1]$	-3.32
$F_{21}(x)=-\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	4	$[0,10]$	-10.1532
$F_{22}(x)=-\sum_{i=1}^{7}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	4	$[0,10]$	-10.4028

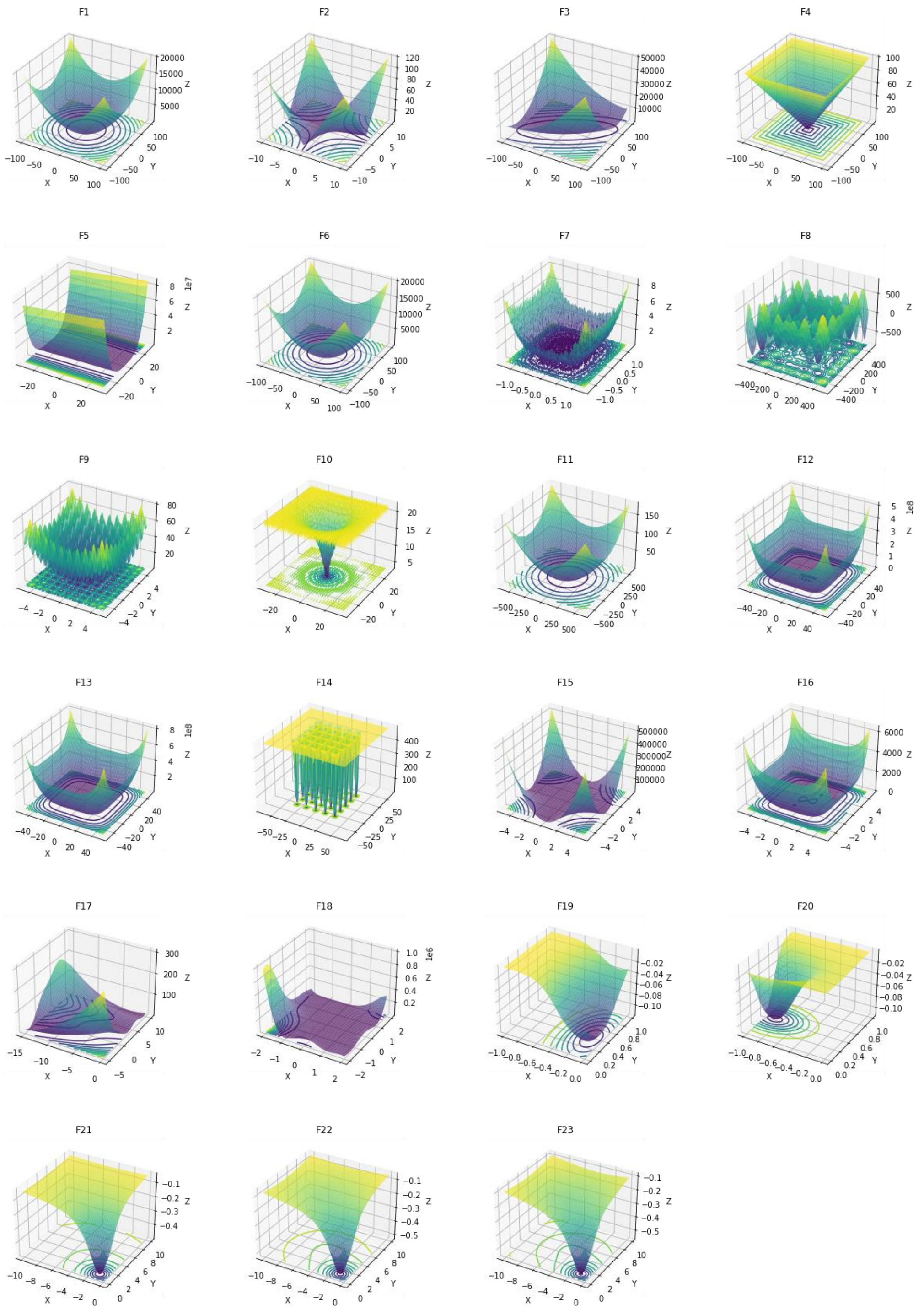

FIGURE 3: Classical function images

3.1 Comparison of different algorithms on classical test functions:

3.1.1 Analysis of solution accuracy:

The maximum iteration count is set to 300, and the test results are shown in Table 4, with the optimal results highlighted in bold. It is evident that WAO outperforms all comparison algorithms on unimodal test functions, indicating its superior development capability and fast convergence speed. For multimodal functions, WAO performs well on F12-F15 and F21-F23, demonstrating its ability to maintain good population diversity and avoid local optima. For functions F9-F12, F17, and F18, WAO's results are comparable to those of other algorithms. Overall, WAO shows better or comparable average optimization results across all 23 classic functions. However, it has slightly higher variance on F8, F16, F19, and F29 compared to other algorithms, and is slightly less effective than COA on F7, suggesting areas for further improvement.

TABLE 4
RESULT OF FIXED-DIMENSION MULTIMODAL BENCHMARK FUNCTIONS

Function	Item	COA	GSA	WOA	GWO	GA	PSO	HHO	WAO
F1	Mean	4.08E-227	5.99E-124	1.55E-54	9.71E-49	3.03E+02	4.93E+04	4.66E-142	**0.00E+00**
	STD	0.00E+00	3.28E-123	5.63E-54	2.15E-48	7.26E+01	2.02E+04	2.29E-141	**0.00E+00**
F2	Mean	3.15E-115	1.41E-79	5.87E-34	1.69E-31	6.77E+00	3.10E+19	5.00E-73	**0.00E+00**
	STD	1.01E-114	7.65E-79	2.76E-33	4.24E-31	9.38E-01	1.06E+20	2.57E-72	**0.00E+00**
F3	Mean	3.77E-228	1.06E-109	1.42E+05	3.68E+04	4.44E+04	1.35E+05	1.87E-85	**0.00E+00**
	STD	0.00E+00	5.83E-109	3.40E+04	1.24E+04	7.86E+03	6.03E+04	1.01E-84	**0.00E+00**
F4	Mean	1.38E-114	1.03E-82	8.78E+01	2.06E-10	2.44E+01	8.82E+01	3.98E-72	**0.00E+00**
	STD	6.20E-114	5.63E-82	4.02E+00	6.49E-10	2.64E+00	9.01E+00	1.42E-71	**0.00E+00**
F5	Mean	1.39E+00	1.02E-02	4.84E+01	4.86E+01	1.36E+04	3.40E+08	3.33E-04	**1.74E-07**
	STD	2.17E+00	2.28E-02	2.51E-01	4.88E-02	6.00E+03	1.70E+08	5.96E-04	**6.03E-07**
F6	Mean	7.49E-02	7.10E-04	1.61E+00	2.82E+00	3.02E+02	5.07E+04	2.85E-06	**1.22E-07**
	STD	4.72E-02	1.69E-03	5.57E-01	5.95E-01	7.51E+01	3.09E+04	3.68E-06	**2.51E-07**
F7	Mean	**1.31E-04**	1.46E-04	5.14E-03	8.41E-04	2.86E-01	1.74E+02	1.59E-04	1.93E-04
	STD	**1.05E-04**	1.13E-04	6.93E-03	5.39E-04	6.08E-02	1.29E+02	1.54E-04	1.58E-04
F8	Mean	-1.13E+04	**-2.09E+04**	-1.45E+04	-1.48E+04	-2.04E+04	-5.11E+03	-1.96E+04	**-2.09E+04**
	STD	1.63E+03	**4.19E-01**	5.24E+02	4.55E+02	1.00E+02	1.19E+03	2.72E+03	5.95E+01
F9	Mean	**0.00E+00**	**0.00E+00**	9.30E+01	**0.00E+00**	3.51E+01	6.53E+02	**0.00E+00**	**0.00E+00**
	STD	**0.00E+00**	**0.00E+00**	1.44E+02	**0.00E+00**	2.94E+00	8.14E+01	**0.00E+00**	**0.00E+00**
F10	Mean	**4.44E-16**	**4.44E-16**	4.00E-15	3.88E-15	4.51E+00	2.00E+01	**4.44E-16**	**4.44E-16**
	STD	**0.00E+00**	**0.00E+00**	2.59E-15	6.38E-16	3.67E-01	1.06E-03	**0.00E+00**	**0.00E+00**
F11	Mean	**0.00E+00**	**0.00E+00**	5.80E-03	**0.00E+00**	3.90E+00	5.14E+02	**0.00E+00**	**0.00E+00**
	STD	**0.00E+00**	**0.00E+00**	1.59E-02	**0.00E+00**	5.98E-01	2.57E+02	**0.00E+00**	**0.00E+00**
F12	Mean	1.83E-03	1.17E-05	9.54E-01	8.51E-02	2.22E+00	6.83E+08	2.91E-07	**1.11E-08**
	STD	1.48E-03	3.31E-05	3.48E+00	2.86E-02	6.86E-01	4.27E+08	3.02E-07	**1.97E-08**
F13	Mean	7.43E-02	7.34E-05	1.84E+00	1.32E+00	2.31E+01	1.40E+09	5.00E-06	**4.20E-07**
	STD	5.20E-02	1.07E-04	2.93E+00	2.87E-01	7.48E+00	7.74E+08	5.24E-06	**8.42E-07**
F14	Mean	1.03E+00	1.10E+00	2.70E+00	3.56E+00	1.04E+00	2.11E+00	3.16E+00	**9.98E-01**
	STD	1.82E-01	3.24E-01	2.92E+00	2.73E+00	1.91E-01	1.61E+00	2.69E+00	**1.25E-16**
F15	Mean	1.00E-03	3.74E-04	9.59E-04	6.57E-04	6.47E-03	1.13E-02	1.10E-03	**3.07E-04**
	STD	4.74E-04	6.53E-05	1.48E-03	4.20E-04	7.48E-03	9.41E-03	2.38E-03	**9.57E-19**
F16	Mean	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**
	STD	6.09E-07	7.95E-03	4.85E-16	2.25E-07	4.60E-03	5.26E-03	6.42E-16	**5.80E-16**
F17	Mean	**3.98E-01**	4.02E-01	**3.98E-01**	4.23E-01	4.17E-01	4.52E-01	**3.98E-01**	**3.98E-01**
	STD	9.77E-10	9.73E-03	9.03E-15	1.03E-01	3.90E-02	2.77E-01	2.34E-06	**0.00E+00**
F18	Mean	**3.00E+00**	**6.81E+00**	**3.00E+00**	6.60E+00	1.13E+01	**3.00E+00**	**3.00E+00**	**3.00E+00**
	STD	3.77E-06	9.69E+00	3.61E-15	9.18E+00	1.76E+01	5.49E-04	6.56E-15	**2.24E-15**
F19	Mean	**-3.86E+00**	-3.82E+00	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	-3.85E+00	**-3.86E+00**	**-3.86E+00**
	STD	3.67E-04	6.68E-02	1.91E-15	7.90E-04	6.04E-04	6.14E-02	1.44E-07	**2.44E-15**
F20	Mean	-3.25E+00	-3.06E+00	-3.27E+00	-3.25E+00	-3.29E+00	-2.85E+00	-3.25E+00	**-3.30E+00**
	STD	**4.73E-02**	1.00E-01	5.89E-02	7.38E-02	5.25E-02	2.87E-01	7.54E-02	4.76E-02
F21	Mean	-9.79E+00	-1.01E+01	-8.05E+00	-8.62E+00	-6.32E+00	-5.94E+00	-6.24E+00	**-1.02E+01**
	STD	3.48E-01	6.60E-03	3.06E+00	2.43E+00	3.50E+00	2.96E+00	2.16E+00	**5.39E-15**
F22	Mean	-1.01E+01	**-1.04E+01**	-8.53E+00	-8.01E+00	-5.01E+00	-5.17E+00	-6.51E+00	**-1.04E+01**
	STD	3.89E-01	8.25E-03	2.92E+00	2.96E+00	3.19E+00	3.58E+00	2.35E+00	**7.25E-16**
F23	Mean	-1.00E+01	**-1.05E+01**	-6.47E+00	-8.30E+00	-5.63E+00	-5.43E+00	-7.11E+00	**-1.05E+01**
	STD	7.92E-01	4.67E-03	3.68E+00	2.92E+00	3.62E+00	3.53E+00	2.61E+00	**2.00E-15**

3.1.2 Convergence analysis:

Comparing algorithm performance solely based on average values is not sufficient. To more intuitively demonstrate the performance of the WAO algorithm compared to other algorithms in terms of convergence accuracy and speed, Fig. 4 displays the convergence curves of each algorithm on the test functions. The horizontal axis represents the number of iterations, while the vertical axis reflects the convergence accuracy of the algorithm, i.e., the final fitness value. By comparing the evolutionary curves and convergence accuracy of seven algorithms on each test function, a detailed analysis of the experimental results was conducted. The results indicate that WAO outperforms other algorithms in most test functions. For single-peak functions F1-F6, WAO shows exceptional convergence speed and search capability compared to other optimization algorithms. For multi-peak functions, WAO exhibits better convergence accuracy and faster convergence speed on test functions F8-F13. On test functions F14-F23, WAO's convergence speed is comparable to that of other individual algorithms, quickly converging to global optima and demonstrating good capability in escaping local optima. However, WAO's convergence speed on test function F7 is slightly inferior to COA, indicating room for improvement. All algorithms achieved optimal or near-optimal solutions. Overall, compared to other established optimization algorithms, WAO performs well on both single-peak and multi-peak test functions, confirming its robustness across various function types and its applicability in solving complex optimization problems.

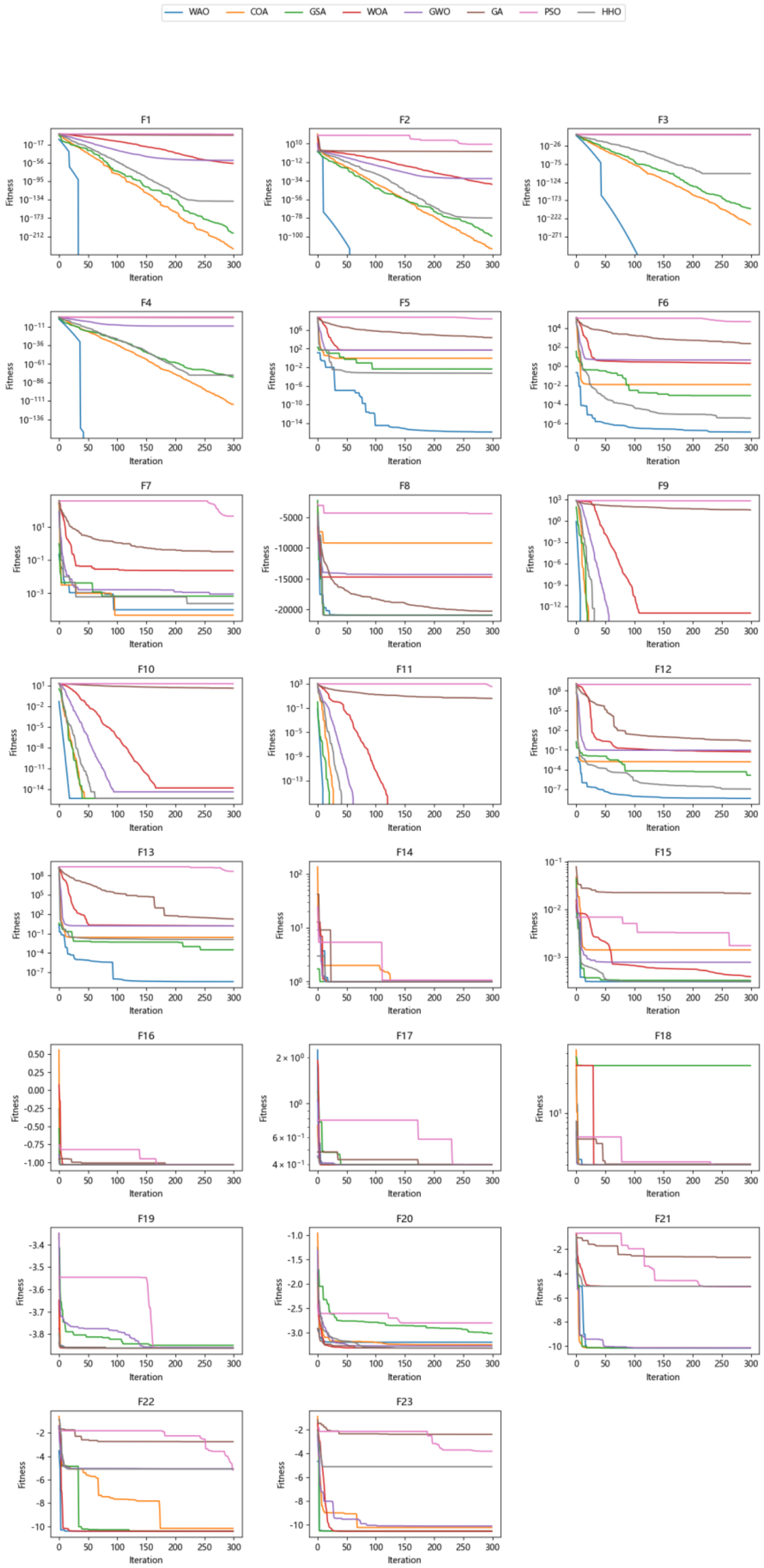

FIGURE 4: Comparison of convergence rates for different algorithms

3.2 Rank sum test:

This section employs the Wilcoxon signed-rank test to determine if there are statistically significant differences between the improved algorithm and the comparison algorithms, reducing the influence of chance in the tests. A significance level of 5% is set; if $p$is less than 5%, the null hypothesis is rejected, indicating a significant difference between the two algorithms; otherwise, the performance difference is not substantial. Results from 23 test functions are compared to assess WAO's statistical advantage. Table 5 displays the experimental results. Since the algorithm cannot be compared with itself, WAO's p are not included. Values in the table greater than 5% are bolded, and it is evident that most p are below 5%, indicating significant differences between WAO and the other seven comparison algorithms, suggesting that WAO has superior search capability.

TABLE 5
WILCOXON SIGNED-RANK TEST RESULTS

Function	COA	GSA	WOA	GWO	GA	PSO	HHO
F1	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F2	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F3	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F4	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F5	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F6	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	9.31323E-09
F7	**0.556113275**	**0.370740607**	3.72529E-09	8.32602E-07	1.86265E-09	1.86265E-09	**0.452164343**
F8	1.86265E-09	0.000152871	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	**0.164184082**
F9	**1**	**1**	0.00352902	**1**	1.86265E-09	1.86265E-09	**1**
F10	**1**	1.86265E-09	7.24259E-06	7.25244E-08	1.86265E-09	1.86265E-09	**1**
F11	**1**	**1**	0.022010526	**1**	1.86265E-09	1.86265E-09	**1**
F12	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	3.72529E-09
F13	1.86265E-09	5.58794E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	6.91041E-07
F14	1.82537E-06	1.86265E-09	5.99237E-05	1.86265E-09	1.86265E-09	1.86265E-09	6.71136E-06
F15	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F16	1.82537E-06	1.86265E-09	**0.202112635**	1.86265E-09	1.86265E-09	1.86265E-09	0.000499647
F17	1.86265E-09	1.86265E-09	0.000563755	1.86265E-09	1.86265E-09	1.86265E-09	0.000102642
F18	1.86265E-09	1.86265E-09	0.040256192	1.86265E-09	1.86265E-09	1.86265E-09	0.00041528
F19	1.86265E-09	1.86265E-09	0.000461595	1.86265E-09	1.86265E-09	1.86265E-09	1.82537E-06
F20	0.000283264	1.86265E-09	0.000460107	0.001232104	0.00256009	1.86265E-09	1.06096E-05
F21	1.86265E-09	1.86265E-09	2.73906E-06	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F22	1.86265E-09	1.86265E-09	3.65907E-05	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09
F23	1.86265E-09	1.86265E-09	3.43673E-06	1.86265E-09	1.86265E-09	1.86265E-09	1.86265E-09

IV. ENGINEERING DESIGN OPTIMIZATION PROBLEMS

The experimental parameter settings and test functions are the same. The engineering problems include the welding beam design optimization problem and the gearbox design problem, both using the same number of iterations (300) and population size (100) for optimization. The optimization results are compared with those obtained using COA, GSA, WOA, GWO, GA, PSO, and HHO algorithms.

4.1 Welded beam design problem:

The Welded beam design problem, as shown in Fig.5, is a minimization problem aimed at reducing manufacturing costs. The optimization algorithm focuses on minimizing the cost of manufacturing the welding beam by optimizing the beam's length l, height t, thickness b, and weld seam thickness h. Consequently, this problem is a classic nonlinear programming problem. Set $X = [x_1, x_2, x_3, x_4] = [h, l, t, b]$, the corresponding objective function f, constraints g, and the ranges of the design variables are as follows:

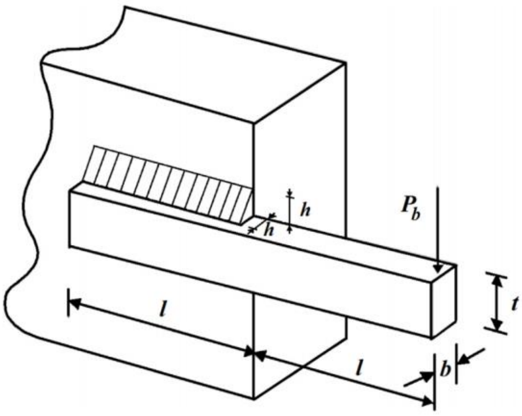

FIGURE 5: Schematic diagram of welded beam design

$$min\, f(x_1, x_2, x_3, x_4) = 1.1047x_1^2x_2 + 0.04811x_3x_4(14 + x_2) \tag{14}$$

Subject to

$$g_1(X) = \tau(X) - \tau_{max} \leq 0$$

$$g_2(X) = \sigma(X) - \sigma_{max} \leq 0$$

$$g_3(X) = \delta(X) - \delta_{max} \leq 0$$

$$g_4(X) = x_1 - x_4 \leq 0$$

$$g_5(X) = P - P_c(X) \leq 0$$

$$g_6(X) = 0.125 - x_1 \leq 0$$

$$g_7(X) = 1.10471x_1^2 + 0.04811x_4x_3(14.0 + x_2) - 5.0 \leq 0$$

$$0.1 < x_1 < 2$$

$$0.1 < x_2 < 10$$

$$0.1 < x_3 < 10$$

$$0.1 < x_4 < 2$$

In the constraints, the expressions for each function can be referenced using the following formulas:

$$\tau(X) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2}x_1x_2}$$

$$\tau'' = \frac{MR}{J}$$

$$M = P\left(L + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_2}{2}\right)^2}$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_2}{2}\right)^2\right]\right\}$$

$$\sigma(X) = \frac{6PL}{x_4x_3^2}$$

$$\delta(X) = \frac{6PL^3}{Ex_4x_3^2}$$

$$P_c(X) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$$\sigma_{max} = 30000psi$$

$$P = 6000lb$$

$$\delta_{max} = 0.25in$$

$$L = 14i$$

$$E = 3 * 10^6 psi$$

$$\tau_{max} = 136000psi$$

$$G = 1.2 * 10^7 psi$$

Set the population size to 100 and the number of iterations to 300. The results of each algorithm are shown in Tables 6 and 7.

TABLE 6
PERFORMANCE OF OPTIMIZATION ALGORITHMS ON THE WELDED BEAM DESIGN PROBLEM

Algorithm	Optimum variables				Optimum cost
	X1	X2	X3	X4	
WAO	0.2057	3.2349	9.0366	0.2057	1.6928
COA	0.2019	3.3597	9.0366	0.2057	1.7041
GSA	0.2061	3.3923	9.0805	0.2065	1.7285
WOA	0.143	5.0461	9.4019	0.204	1.8713
GWO	0.2004	3.3447	9.0441	0.2059	1.7024
GA	0.2077	3.4654	8.4791	0.2404	1.8779
PSO	0.1722	4.1204	9.0456	0.2058	1.7577
HHO	0.1577	4.6585	9.0493	0.2057	1.7986

TABLE 7
STATISTICAL RESULTS OF OPTIMIZATION ALGORITHMS ON WELDED BEAM DESIGN PROBLEM.

Algorithm	Best	Worst	Mean	Std	Median
WAO	1.6928	2.3649	1.7512	0.1595	1.6935
COA	1.7041	1.8995	1.7814	0.047	1.7787
GSA	1.7285	3.7164	1.9276	0.3626	1.8221
WOA	1.8713	4.961	2.9542	0.8289	2.7298
GWO	1.7024	3.1509	1.9544	0.3537	1.8016
GA	1.8779	3.777	2.4613	0.4042	2.3894
PSO	1.7577	2.6159	2.1221	0.247	2.0264
HHO	1.7986	3.9132	2.4908	0.4976	2.338

4.2 Speed reducer design problem:

In a mechanical system, the reducer is a crucial component of the gearbox. The optimization problem involves minimizing the weight of the reducer while adhering to constraints related to gear root bending stress, surface stress, shaft lateral deflection, and shaft stress. The variables to be optimized include the gear face width b, gear module m, number of teeth in the small gear p, length of the first shaft between bearings l_1, length of the second shaft between bearings l_2, diameter of the first shaft d_1, and diameter of the second shaft d_2. The integer variable $p$is specified with $X = [x_1, x_2, x_3, x_4, x_5, x_6, x_7] = [b, m, p, l_1, l_2, d_1, d_2]$, while the remaining variables are continuous.

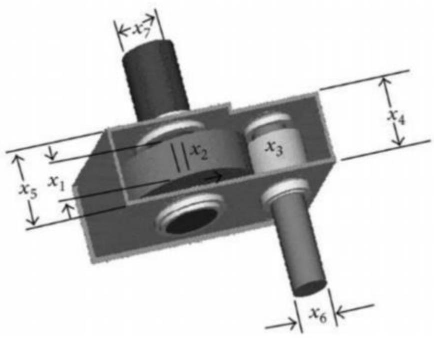

FIGURE 6: Schematic diagram of speed reducer design

The corresponding objective function f, constraints g, and variable ranges are as follows:

$min\, f(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$ （15）

Subject to

$$g_1(X) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0$$

$$g_2(X) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0$$

$$g_3(X) = \frac{1.93x_4^3}{x_2x_3x_6^2} - 1 \leq 0$$

$$g_4(X) = \frac{1.93x_5^3}{x_2x_3x_7^2} - 1 \leq 0$$

$$g_5(X) = \frac{\sqrt{\left(\frac{754x_4}{x_2x_3}\right)^2 + 16.9e^6}}{110x_6^3} - 1 \leq 0$$

$$g_6(X) = \frac{\sqrt{\left(\frac{754x_5}{x_2x_3}\right)^2 + 157.5e^6}}{85x_7^3} - 1 \leq 0$$

$$g_7(X) = \frac{x_2x_3}{40} - 1 \leq 0$$

$$g_8(X) = \frac{5x_2}{x_1} - 1 \leq 0$$

$$g_9(X) = \frac{x_1}{12x_2} - 1 \leq 0$$

$$g_{10}(X) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0$$

$$g_{11}(X) = \frac{1.1x_6 + 1.9}{x_5} - 1 \leq 0$$

$$2.6 < x_1 < 3.6$$

$0.7 < x_2 < 0.8$

$17 < x_3 < 28$

$7.3 < x_4 < 8.3$

$7.8 < x_5 < 8.3$

$2.9 < x_6 < 3.9$

$5.0 < x_7 < 5.5$

The optimization results for each algorithm are presented in the following table:

TABLE 8
PERFORMANCE OF OPTIMIZATION ALGORITHMS ON SPEED REDUCER DESIGN PROBLEM.

Algorithm	Optimum variables							Optimum cost
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
WAO	3.5	0.7	17	7.3	7.7153	3.3502	5.2867	2994.4711
COA	3.506	0.7	17	7.3	7.8226	3.3542	5.2867	3000.22
GSA	3.5054	0.7	17	8.0664	8.0664	3.4445	5.2869	3036.2795
WOA	3.5	0.7	17	7.7858	7.7858	3.3512	5.2868	3000.66
GWO	3.5001	0.7	17	7.3	7.9708	3.3502	5.2911	3003.06
GA	3.5008	0.7001	17.0012	7.3036	7.7288	3.3504	5.2867	2996.0666
PSO	3.6	0.7	17	7.3	8.3	3.3535	5.2874	3047.89
HHO	3.5067	0.7	17	7.3	7.7521	3.3502	5.2868	2997.9701

TABLE 9
STATISTICAL RESULTS OF OPTIMIZATION ALGORITHMS ON SPEED REDUCER DESIGN PROBLEM

Algorithm	Best	Worst	Mean	Std	Median
WAO	2994.4711	3007.8428	2997.0406	4.4774	2994.4809
COA	3000.2235	3050.3287	3020.9087	12.102	3017.5886
GSA	3036.2795	5296.003	3312.7957	581.8437	3143.7067
WOA	3000.6616	5285.8882	3518.3961	594.6483	3241.1305
GWO	3003.0627	5278.9856	3337.8872	650.8056	3128.7551
GA	2996.0666	3549.0969	3018.3054	98.6448	2999.037
PSO	3047.8934	3222.3128	3096.4834	65.7039	3058.8795
HHO	2997.9701	5105.7952	3691.6963	703.4508	3503.2547

4.3 Pressure vessel design problem:

In the pressure vessel design problem, the goal is to minimize the cost while meeting production requirements. The pressure vessel has covers at both ends, with one end featuring a hemispherical head. The design involves four optimization variables: $L$is the length of the cylindrical section of the vessel, $R$is the internal diameter, $T_s$is the thickness of the vessel wall, and $T_h$is the thickness of the head. Let $X = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L]$. The objective function f, constraints g, and the range of the design variables are specified as follows:

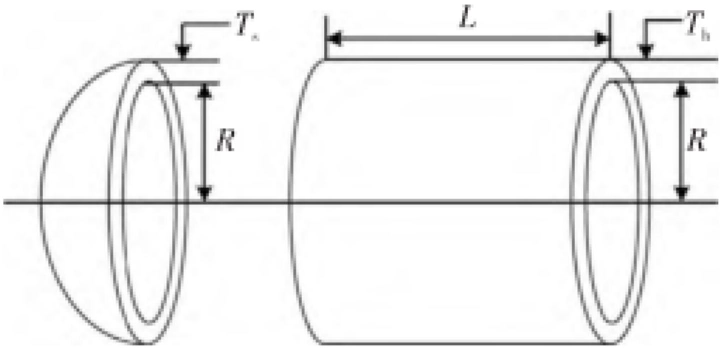

FIGURE 7: Schematic diagram of pressure vessel design

$$min\, f(x_1, x_2, x_3, x_4) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (16)$$

Subject to

$$g_1(X) = -x_1 + 0.0193x_3 \le 0$$

$$g_2(X) = -x_2 + 0.00954x_3 \le 0$$

$$g_3(X) = -\pi x_3^2x_4 - \frac{4\pi x_3^3}{3} + 1296000 \le 0$$

$$g_4(X) = x_4 - 240 \le 0$$

$$0.0625 < x_1 < 99.999$$

$$0.0625 < x_2 < 99.999$$

$$10 < x_3 < 200$$

$$10 < x_4 < 200$$

The optimization results for each algorithm are presented in the following table:

TABLE 10
PERFORMANCE OF OPTIMIZATION ALGORITHMS ON THE PRESSURE VESSEL DESIGN PROBLEM

Algorithm	Optimum variables				Optimum cost
	x_1	x_2	x_3	x_4	
WAO	0.7782	0.3846	40.3196	200	5885.3379
COA	0.8274	0.4589	42.8696	167.3251	6138.5222
GSA	1.0935	0.5401	55.7566	58.7088	6758.2153
WOA	0.9177	0.4504	47.2075	122.1683	6193.3793
GWO	0.8598	0.425	44.4798	149.2101	6048.4539
GA	1.0668	0.5797	53.7254	71.5534	6998.602
PSO	1.4299	0.6468	65.3164	10	8201.8786
HHO	0.8533	0.4366	44.2053	152.201	6078.8689

TABLE 11
STATISTICAL RESULTS OF OPTIMIZATION ALGORITHMS ON THE PRESSURE VESSEL DESIGN PROBLEM.

Algorithm	Best	Worst	Mean	Std	Median
WAO	5885.3379	7319.007	6433.909	514.0014	6261.0542
COA	6138.5222	12346.614	7755.2197	1376.762	7317.44
GSA	6758.2153	26704.47	11746.118	6174.4435	9119.3473
WOA	6193.3793	437875.79	105923.96	131415.26	45444.839
GWO	6048.4539	10080.049	6788.6016	751.3376	6725.7667
GA	6998.602	13656.272	8624.9883	1344.316	8481.7794
PSO	8201.8786	68129.769	23515.66	15195.015	14875.789
HHO	6078.8689	138840.5	11210.103	23706.136	6805.7329

V. CONCLUSION AND FUTURE WORK

This paper presents a novel metaheuristic optimization algorithm called the Worker Ant Optimization (WAO) algorithm, designed to simulate various activities of worker ants in nature, including behaviors such as avoiding danger, foraging, approaching food, and decomposing and transporting food. The paper develops a mathematical optimization model based on these natural activities of worker ants and rigorously evaluates the convergence speed and search accuracy of WAO across 23 classic test functions. The quality of WAO optimization results is compared with the performance of seven well-known algorithms. Simulation results show that WAO exhibits excellent convergence speed, achieves a suitable balance between exploration in global search and exploitation in local search, and demonstrates a strong ability to escape local optima, providing effective solutions for optimization problems. Additionally, the WAO method is applied to three engineering design optimization problems, and its applicability to engineering optimization is validated. The comparisons with seven well-known optimization algorithms further demonstrate the advantages of the WAO algorithm in optimizing complex global optimization problems.

However, it is worth noting that there is still room for improvement in the convergence speed of WAO. While the algorithm shows strong performance in other aspects, optimizing its convergence rate could further enhance its overall efficiency and effectiveness in solving complex optimization problems. Future research could focus on refining the algorithm's parameters and exploring hybrid approaches to accelerate convergence without compromising its robustness and solution quality.

REFERENCES

[1] Dhiman G. SSC: A hybrid nature-inspired meta-heuristic optimization algorithm for engineering applications[J]. Knowledge-Based Systems, 2021, 222: 106926.

[2] Hajipour V, Kheirkhah A S, Tavana M, et al. Novel Pareto-based meta-heuristics for solving multi-objective multi-item capacitated lot-sizing problems[J]. The International Journal of Advanced Manufacturing Technology, 2015, 80: 31-45.

[3] Dehghani M, Hubálovský Š, Trojovský P. Tasmanian devil optimization: a new bio-inspired optimization algorithm for solving optimization algorithm[J]. IEEE Access, 2022, 10: 19599-19620.

[4] Hajipour V, Mehdizadeh E, Tavakkoli-Moghaddam R. A novel Pareto-based multi-objective vibration dam** optimization algorithm to solve multi-objective optimization problems[J]. Scientia Iranica. Transaction E, Industrial Engineering, 2014, 21(6): 2368.

[5] Wu G. Across neighborhood search for numerical optimization[J]. Information Sciences, 2016, 329:

[6] Ezugwu A E, Shukla A K, Nath R, et al. Metaheuristics: a comprehensive overview and classification along with bibliometric analysis[J]. Artificial Intelligence Review, 2021, 54: 4237-4316.

[7] Nabil E. A modified flower pollination algorithm for global optimization[J]. Expert Systems with Applications, 2016, 57: 192-203.

[8] Wang G G. Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems[J]. Memetic Computing, 2018, 10(2): 151-164.

[9] Dehghani M, Montazeri Z, Dehghani A, et al. DM: Dehghani Method for modifying optimization algorithms[J]. Applied Sciences, 2020, 10(21): 7683.

[10] Dehghani M, Montazeri Z, Dehghani A, et al. Binary spring search algorithm for solving various optimization problems[J]. Applied Sciences, 2021, 11(3): 1286.

[11] Jerne N K .Towards a network theory of the immune system. Ann Inst Pasteur Immunol 125C:373-389[J].Annales d'immunologie, 1974, 125C(1-2):373-389.

[12] Beni G, Wang J. Swarm intelligence in cellular robotic systems[M]//Robots and biological systems: towards a new bionics?. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993: 703-712.

[13] Eberhart R, Kennedy J. A new optimizer using particle swarm theory[C]//MHS'95. Proceedings of the sixth international symposium on micro machine and human science. Ieee, 1995: 39-43.

[14] Tanyildizi E, Demir G. Golden sine algorithm: a novel math-inspired algorithm[J]. Advances in Electrical & Computer Engineering, 2017, 17(2).

[15] 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.

[16] Holland J H. Genetic algorithms[J]. Scientific american, 1992, 267(1): 66-73.

[17] Heidari A A, Mirjalili S, Faris H, et al. Harris hawks optimization: Algorithm and applications[J]. Future generation computer systems, 2019, 97: 849-872.

[18] 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: 459-471.

[19] Saremi S, Mirjalili S Z, Mirjalili S M. Evolutionary population dynamics and grey wolf optimizer[J]. Neural Computing and Applications, 2015, 26: 1257-1263.

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

[21] Tizhoosh H R. Opposition-based reinforcement learning[J]. Journal of Advanced Computational Intelligence and Intelligent Informatics, 2006, 10(3).