Artificial Intelligence Review
https://doi.org/10.1007/s10462-023-10567-4

ORIGINAL PAPER

Crayfish optimization algorithm

Heming Jia[1] · **Honghua Rao**[1] · **Changsheng Wen**[1] · **Seyedali Mirjalili**[2,3]

Accepted: 26 July 2023

Abstract
This paper proposes a meta heuristic optimization algorithm, called Crayfish Optimization Algorithm (COA), which simulates crayfish's summer resort behavior, competition behavior and foraging behavior. The three behaviors are divided into three different stages to balance the exploration and exploitation of algorithm. The three stages are summer resort stage, competition stage and foraging stage. The summer resort stage represents the exploration stage of the COA. The competition stage and foraging stage represent the exploitation stage of the COA. Exploration and exploitation of COA are regulated by temperature. When the temperature is too high, crayfish will enter the cave for summer vacation or compete for the same cave. When the temperature is appropriate, crayfish have different foraging behaviors according to the size of food. Among them, the amount of food eaten by crayfish is related to food intake. Through temperature regulate exploration and exploitation process in COA, the COA has higher randomness and global optimization effect. To verify the optimization effect of COA, in the experimental part, 23 standard benchmark functions and CEC2014 benchmark functions are used to test, and 9 algorithms are selected for comparative experiments. The experimental results show that COA can balance the exploration and exploitation, and achieve good optimization effect. Finally, the COA is tested in five engineering problems, and finally achieves better results. The source code website for COA is https://github.com/rao12138/COA-s-code.

Keywords Crayfish optimization algorithm · Summer resort stage · Competition stage · Foraging stage

✉ Heming Jia
jiaheming@fjsmu.edu.cn

Honghua Rao
20200862235@fjsmu.edu.cn

Changsheng Wen
20200862204@fjsmu.edu.cn

[1] School of Information Engineering, Sanming University, Sanming 365004, China

[2] Centre for Artificial Intelligence Research and Optimisation, Torrens University, Adelaide, SA 5000, Australia

[3] University Research and Innovation Center, Obuda University, 1034 Budapest, Hungary

Published online: 02 September 2023

1 Introduction

In recent years, with the ever-increasing complexity of solving problems and the fuzziness of the final results, the demand for better optimization algorithms is increased. The optimization problem can be expressed as a continuous or combined design search space, which is a process of searching for the maximum or minimum value of the function $f(x)$. The parameter x is the solution within the search range. The optimization problem will go through many iterations to find the best solution $x_{best}\{x_1, x_2,..., x_D\}$. D is the dimension of solution. In each iteration, a new solution $x_{new}\{x_1, x_2,..., x_D\}$ will be obtained. If the solution x_{new} is better than x_{best}, then $x_{best} = x_{new}$. Then, these iterations continue until the solution found meets some predefined criteria. This final solution (close to the optimal solution) is called the optimal solution. This is the process of meta-heuristic optimization algorithm (Ezugwu et al. 2021). In order to get the best solution, the meta-heuristic optimization algorithm usually initializes multiple solutions within the solution range for iterative solution. During each iteration, solutions are updated with heuristic-based decision making onely relying on the objective function (aka. cost function). This has certain requirements for the algorithm. If all the solutions are too close to a certain region, it is easy to cause the algorithm to fall into local optimization and cannot obtain a better solution. If the update formula value is too large (or too small), resulting in the solution leaving the search scope, the optimization performance of the algorithm will be reduced. Local optimization refers to the minimum (or maximum) of the solution within a certain range, but not the minimum (or maximum) of the whole region. The global optimum is the minimum (or maximum) that can be solved.

Compared with traditional optimization algorithms, meta-heuristic algorithms are easier to understand and easier to implement. Traditional optimization algorithms generally aim at structured problems and have relatively clear descriptions of problems and conditions, such as linear programming (Dantzig 2002), integer programming (Chen et al. 2022), mixed programming (Daryalal et al. 2022), constrained and unconstrained conditions (Gautier and Granot 1994), etc. They have clear structural information, fixeds structures and parameters, and unique and clear global optimum. Traditional optimization algorithms can be theoretically analyzed in terms of computational complexity and convergence. In the single-extremum problem, the traditional optimization algorithm is good enough, but there are still many deficiencies in solving the multi-extremum problem. Meta-heuristic algorithm can have a good balance between jumping out of the local optimum and converging to a point through effective design, which increases the likelihood of finding the global optimum for a given optimization problem. Because the meta-heuristic algorithm can find the global optimal solution of the optimization problem, the optimization result is independent of the initial condition, the algorithm is independent of the solution domain and has strong robustness, and other excellent characteristics, it is widely used in application practice (Mzili et al. 2022; Jia et al. 2022a, b). Its practical value has also been reflected in recent years, and has been applied in many fields, such as multiple travelling salesman problem (Mzili et al. 2023), multilevel threshold image segmentation (Liu et al. 2022), ship routing and scheduling problem (Das et al. 2022), feature selection (Jia et al. 2022a, b) and multi-level image segmentation (Qi et al. 2021), etc. Nowadays, many scholars have proposed many kinds of algorithms according to the characteristics of nature, which can be roughly divided into four categories: Swarm intelligence algorithms, Physics-based Algorithms, Evolutionary algorithms, Human-inspired algorithms (Wen et al. 2022). As shown in Fig. 1.

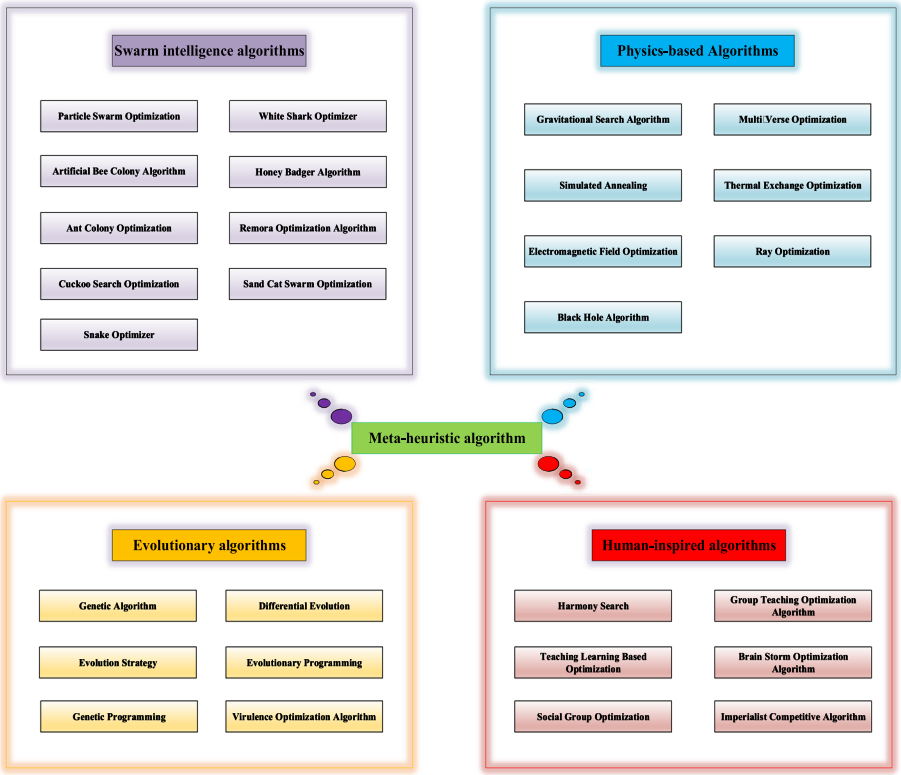

Fig. 1 Classification of meta-heuristic optimization algorithms

The algorithm inspired by the behavior of animals found in nature is called swarm intelligence algorithm (SI). SI was first proposed by Kennedy and Eberhart as Particle Swarm Optimization (PSO) (Kennedy and Eberhart 1995). PSO algorithm is inspired by the collective foraging behavior of birds. In PSO algorithm, candidate solutions are considered as particles in search space. In order to find a better solution, each particle is guided by the local optimum and the global optimum for space search. The change of solution is obtained by using the moving speed and inertia weight of particles in the search space. Artificial Bee Colony Algorithm (ABC) (Karaboga and Basturk 2007), Ant Colony Optimization (ACO) (Dorigo et al. 2006), and Cuckoo Search Optimization (CS) (Gandomi et al. 2013) are classic algorithms proposed after PSO algorithm. In ABC algorithm, the division of labor behavior of bees is used to find better solutions in the search space. The change of the solution is obtained by the behavior of bees searching for honey source and transporting honey source in the search space. In ant colony optimization, ants represent candidate solutions. Each ant explores and releases pheromones in the search space. The strength of pheromone is determined by the solution concentration. Ants will preferentially walk the route with high pheromone concentration and further explore nearby. CS algorithm utilizes the invasion behavior of cuckoo laying eggs in the nest. The cuckoo egg is a new solution. If this solution is better than the existing solution, it may replace the existing solution. These are classic algorithms proposed earlier. In the last decade, with more in-depth research,

many new algorithms have been proposed. For example, Snake Optimizer (SO) (Hashim and Hussien 2022)is inspired by the feeding and breeding behavior of snakes, White Shark Optimizer (WSO) (Braik et al. 2022)is inspired by the unique sense of hearing and smell of white sharks during navigation and foraging, Honey Badger Algorithm (Hashim et al. 2022)is inspired by the foraging stage of the honey badger, the Remora Optimization Algorithm (ROA) (Jia et al. 2021)inspired by the parasitic behavior of the remora, and the latest Sand Cat Swarm Optimization (SCSO) (Seyyedabbasi and Kiani 2022) inspired by the hunting behavior of the sand cat.

The algorithm inspired by physical phenomena is called Physics-based Algorithms. There are many representative works in Physics-based Algorithms. It is very famous that Rashedi et al. proposed the Gravitational Search Algorithm (GSA) based on physics (Rashedi et al. 2009)and Kirkpatrick et al. proposed the Simulated Annealing (SA) (Kirkpatrick et al. 1983). Inspired by the law of universal gravitation and the interaction between particles, the GSA represents a candidate solution. All objects move towards heavier objects, and the weight of the object represents the excellent solution. SA simulates the principle of solid annealing, benefits from the research results of material statistical mechanics, and this algorithm is also based on probability. Inspired by the behavior of electromagnets, Abedinpurshotorban et al. proposed the Electromagnetic Field Optimization (EFO) (Abedinpourshotorban et al. 2016). The corresponding algorithm is also Black Hole Algorithm (BH) inspired by black holes (Hatamlou 2013). Multi-Verse Optimization (MVO) is an algorithm based on the concepts of black and white holes and wormholes in the multiverse theory (Mirjalili et al. 2016a, b). Thermal Exchange Optimization (TEO) is based on Newton's cooling law (Kaveh and Dadras 2017). And Ray Optimization (RO) (Kaveh and Khayatazad 2012) proposed by Kaveh et al.

Evolutionary algorithms are a kind of meta-heuristic algorithm mimic evolution mechanism of nature. The most famous evolutionary algorithm is Genetic Algorithm (GA) (Holland 1992). GA simulates the biological evolution process of natural selection and genetic mechanism of Darwin's biological evolution theory. GA binary-coded genes to realize hybridization and mutation in population organisms. Each individual in the population represents a candidate solution. Algorithms similar to GA principles include Evolution Strategy (ES) (Beyer and Schwefel 2002), Genetic Programming (GP) (Banzhaf et al. 2000), and Differential Evolution (DE) (Storn and Price 1997). The principle of ES algorithm is similar to that of GA, but ES algorithm uses DNA sequence instead of binary coding. GP algorithm is also an evolutionary algorithm based on GA. In DE algorithm, the mutation is generated by the difference of the parent, and the new individual is generated by crossing with the parent individual, and the parent individual is directly selected. The previous algorithms are inspired by biological genes, and there are other evolutionary algorithms such as Evolutionary Programming (EP) (Sinha et al. 2003)and Virulence Optimization Algorithm (VOA) (Jaderyan and Khotanlou 2016). EP algorithm simulates the evolutionary behavior of organisms, and realizes the algorithm principle from the objective biological representation. VOA simulates the process of virus invasion and host cell immunity.

Human - inspired algorithms are meta-heuristic algorithms inspired by human behavior habits. Human-inspired algorithms is famous for Harmony Search (HS), which simulates music performance to find an excellent harmony process (Geem et al. 2001). Similar to this algorithm are Teaching Learning Based Optimization(TLBO) (Rao et al. 2012), Social Group Optimization (SGO) (Satapathy and Naik 2016), Group Teaching Optimization Algorithm (GTOA) (Zhang and Jin 2020), Brain Storm Optimization Algorithm (BSO) (Cheng et al. 2016)and Imperialist Competitive Algorithm (ICA) (Xing et al. 2014). TLBO algorithm simulates the traditional classroom teaching process, realizes the classroom

teaching process through the teacher's teaching and student's learning, and the knowledge reserve of each student represents the candidate solution. SGO algorithm is a kind of social group-based learning proposed by Satapathy and Naik in 2016. In the SGO algorithm, everyone will gain knowledge and have the ability to solve problems to a certain extent. The best individual is the best solution. The best individuals try to spread knowledge among all individuals, which in turn will improve the knowledge level of the whole team members, and the overall knowledge level will be improved. GTOA simulates the group teaching mechanism in the classroom. It divides students into two groups, namely elite students and ordinary students. The teacher will adopt different teaching methods for the two groups so that each student can get better teaching. The inspiration of BSO is that human beings can think of creative ways to solve problems. ICA takes advantage of the concept of colonies and empires. The less powerful empire is collapsing, while the powerful empire is surviving for a long time. The new empire was formed by invading the old empire with weaker strength.

Although many meta-heuristic algorithms have been proposed in recent years, according to NFL theorem (Wolpert and Macready 1997), no one algorithm can solve all problems. When an algorithm has a very good effect in one problem, it may not have a good effect in other problems. This paper proposes a new meta-heuristic optimization algorithm, Crayfish Optimization Algorithm (COA), based on crayfish's foraging behavior and summer resort behavior. COA proposed a new method to simulate crayfish's foraging, summer resort and competition behavior. By defining the temperature, the exploration and exploitation ability of the algorithm is balanced. And the COA has this excellent optimization effect in CEC2014 benchmark function and 23 standard benchmark functions. In engineering problems, the results obtained by COA are also very excellent. For some traditional optimization algorithms, such as SA, GWO, ABC, GA, etc. SA simulated the solid-state annealing process, GWO simulated the cooperative hunting of wolves, ABC was inspired by the intelligent foraging behavior of bees, and GA simulated the evolution process of natural organisms. These algorithms have their own innovation, and have good optimization effect. Compared with these algorithms, COA puts forward many variable to control algorithm's exploration and exploitation, which have better randomness and optimization effect. To verify that COA is a meaningful work. In the experiment, COA is compared with GA and GWO algorithm, and the results also show that COA has better optimization effect.

The main features and contributions of COA are as follows:

- Define temperature to control the behavior of crayfish, and define the feeding amount of crayfish through temperature.
- When the temperature is too high, the COA enters the competition stage or the summer resort stage. At the stage of competition, other crayfish also took a fancy to the cave, and two crayfish will compete for the cave. During the summer resort stage, there are no other crayfish competing for the cave. At this time, the crayfish will directly enter the cave.
- When the temperature is suitable for foraging, COA enters the foraging stage. At this time, the crayfish will look for food to eat. When the food is too large, the crayfish will use its claws to tear up the food and eat it. When the food is suitable, the crayfish will use the second walking foot and the third walking foot to eat alternately. The feeding of crayfish is also affected by the feeding amount, which also controls the feeding of crayfish.
- In order to find a suitable food factor, we have done parameter analysis on the food factor in the experimental part.

The organizational structure of this paper is as follows: In the second section, we introduce the proposed Crayfish optimization algorithm (COA). The third part is the experimental results of COA and 9 comparison algorithms in 23 standard test functions and CEC2014 test functions, as well as the parameter analysis of food factors in COA. Five constrained engineering problems are used to verify the effectiveness of COA in engineering problems. In the fourth part, this paper analyze the optimization characteristics and limitations of COA. Finally, the fifth section makes a summary and prospects for the future.

2 Crayfish optimization algorithm(COA)

The crayfish looks like shrimp, and its shell is very hard. In animal taxonomy, it belongs to Arthropoda, Crustacea and Decapoda. It is generally considered as a key species of freshwater habitat (Kouba et al. 2014). Crayfish live in a wide variety of freshwater habitats such as lakes, rivers and streams to cave pools, temporary ponds and only seasonally wet fields (Berrill and Chenoweth 1982). Crayfish have the habit of digging holes and appearing at night (Graham et al. 2022). More than 600 kinds of freshwater crayfish recorded in the world have the ability to dig, regardless of geographical location or habitat (Crandall and De Grave 2017). All crayfish have caves. The form and function of caves vary with species. This change was obtained from the classification of crayfish based on cave ecology (Florey and Moore 2019). Crayfish dig holes to avoid natural enemies and summer heat (Payette and McGaw 2003). Not only that, caves can also be used to avoid predators, prevent drying, foraging, and hatching care. Crayfish is a kind of wide temperature aquatic animal. It can adapt to various weather conditions, can provide commercial pellet feed, and can also be raised cheaply. After six months of cultivation, crayfish can grow to 40–60 g (Jones and Ruscoe 2001). After many years of research, more profound studies have been conducted on crayfish growth. Crayfish like all deciduae, temperature changes in the living environment prolong or shorten crayfish growth rates. So, temperature had an important effect on crayfish survival (Allan et al. 2006). It can survive between 11 and 44 °C. However, it does not mean it is suitable for its growth. The temperature suitable for the growth of crayfish is 15 to 30 °C. The most suitable temperature for survival of crayfish is 25 °C (García-Guerrero et al. 2013). If the water temperature is too low, it is easy to cause crayfish to eat less or not, thus affecting the growth of crayfish, or even death. When the water temperature is too high, it is easy to cause the crayfish to go ashore without oxygen, and even cause death. The schematic diagram of crayfish is shown in Fig. 2.

Fig. 2 Schematic diagram of crayfish

Crayfish is an omnivorous animal that can feed on all kinds of grains, cakes, vegetables, terrestrial grasses, aquatic plants in water bodies, epiphytic algae, zooplankton, aquatic insects, small benthos and animal carcasses (Larson and Olden 2011). When crayfish are preying, they usually catch large food with their claws and tear it up before sending it to the 2nd and 3rd walking feet for holding and gnawing. For small food, use the second and third walking feet to hold and nibble directly. As shown in Fig. 3. After hunting, they often hide quickly or use pincers to protect them from being robbed by other crayfish (Bellman and Krasne 1983).

2.1 Source of inspiration

COA is inspired by crayfish's foraging, summer vacation and competitive behavior. Foraging stage and competition stage are the exploitation stage of COA, and summer resort stage is the exploration stage of COA. In order to reflect the characteristics of swarm intelligence optimization, the crawfish colony X is defined at the initial stage of the algorithm. X_i is the position of the ith crayfish, indicating a solution. ($X_i = \{X_{i,1}, X_{i,1}, X_{i,1} \ \ldots \ X_{i,dim}\}$, dim is the characteristic quantity of optimization problem, also known as dimension). X_i brings in the function $f(\cdot)$ to obtain a solution, also known as fitness value. The exploration and exploitation of COA is regulated by temperature, which is a random constant that represents the temperature of the environment in which the individual is located. When the temperature is too high, COA will enter the summer resort stage or competition stage. During the summer resort stage, update the new solution according to the individual position X_i and the cave position X_{shade}. When the temperature is appropriate, COA will enter the foraging stage. In the foraging stage, the location of food is the best location, also known as the optimal solution. The size of food is obtained by the current solution $fitness_i$ (the solution obtained by X_i) and the optimal solution $fitness_{food}$ (the solution obtained by the optimal solution). When the food is suitable, crayfish get new solutions according to their position X_i, food intake constant p and food position X_{food} update. When the food is too large, crayfish will use its claw foot to tear up the food, and then use the second walking foot and the third walking foot to

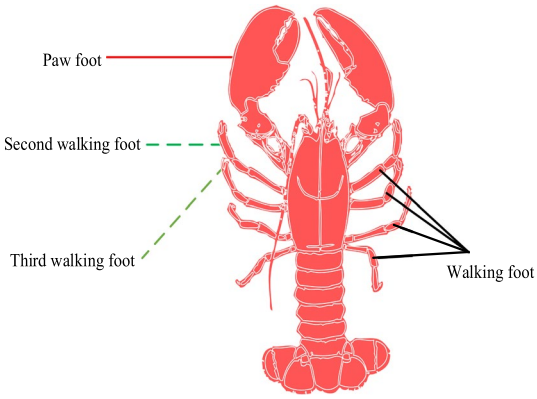

Fig. 3 Structure diagram of crayfish

eat alternately. We used the sine and cosine formula to simulate the alternating feeding behavior of crayfish. Food intake is controlled in crayfish. Food intake is determined by temperature and presents a positive distribution.

2.2 Initialize population

In the multi-dimensional optimization problem, each crayfish is a matrix of $1 \times dim$. Each column matrix represents a solution to a problem. In a set of variables ($X_{i,1}$, $X_{i,2}$, ..., $X_{i,dim}$), each variable X_i must lie between the upper and lower boundaries. The initialization of COA is to randomly generate a group of candidate solutions X in the space. Candidate solution X is based on population size N and dimension dim. The initialization of COA algorithm is shown in Eq. (1).

$$X = [X_1, X_2, \cdots, X_N] = \begin{bmatrix} X_{1,1} & \cdots & X_{1,j} & \cdots & X_{1,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ X_{i,1} & \cdots & X_{i,j} & \cdots & X_{i,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ X_{N,1} & \cdots & X_{N,j} & \cdots & X_{N,dim} \end{bmatrix}, \tag{1}$$

where X is the initial population position, N is the number of populations, dim is the population dimension, $X_{i,j}$ is the position of individual i in the j dimension, and $X_{i,j}$ value is obtained from Eq. (2).

$$X_{i,j} = lb_j + (ub_j - lb_j) \times rand, \tag{2}$$

where lb_j represents the lower bound of the jth dimension, ub_j represents the upper bound of the jth dimension, and $rand$is a random number of Ezugwu et al. (2021).

2.3 Define temperature and intake of crayfish

The change of temperature will affect the behavior of crayfish and make it enter different stages. Temperature is defined as Eq. (3). When the temperature is higher than 30 °C, the crayfish will choose a cool place for summer vacation. At the appropriate temperature, crayfish will conduct foraging behavior. The feeding amount of crayfish is affected by temperature. The feeding range of crayfish is between 15, 30, and 25 °C is the best. Therefore, the feeding amount of crayfish can be approximated to the normal distribution, so that the feeding amount is affected by temperature. Because crayfish have strong foraging behavior between 20 and 30 ℃. So, COA defines a range of temperatures from 20 to 35 ℃. The mathematical model of crayfish intake is shown in Eq. (4). The schematic diagram of food intake is shown in Fig. 4.

$$temp = rand \times 15 + 20, \tag{3}$$

where, $temp$ represents the temperature of the environment where the crayfish is located.

$$p = C_1 \times \left(\frac{1}{\sqrt{2 \times \pi} \times \sigma)} \times \exp\left(-\frac{(temp - \mu)^2}{2\sigma^2} \right) \right). \tag{4}$$

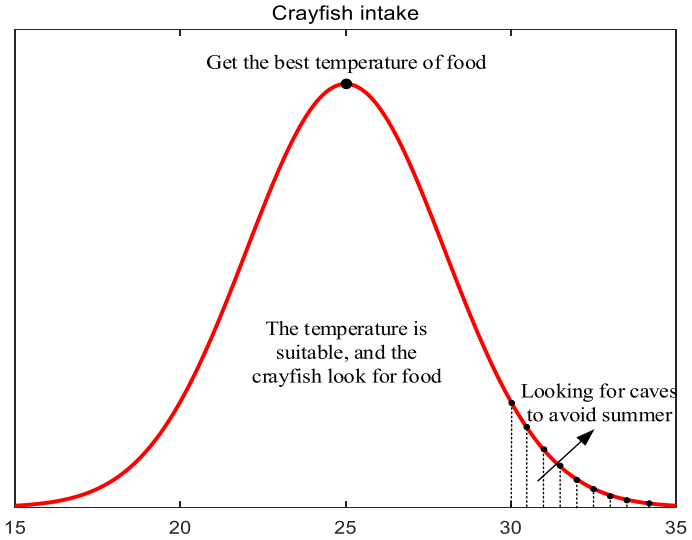

Fig. 4 Effect of temperature on crayfish intake

Among them, μ refers to the temperature most suitable for crayfish, σ and C_1 are used to control the intake of crayfish at different temperatures.

2.4 Summer resort stage (exploration)

When $temp > 30$, the temperature is too high. At this time, the crayfish will choose to join the cave for summer vacation. The cave X_{shade} is defined as follows:

$$X_{shade} = \left(X_G + X_L\right)/2, \tag{5}$$

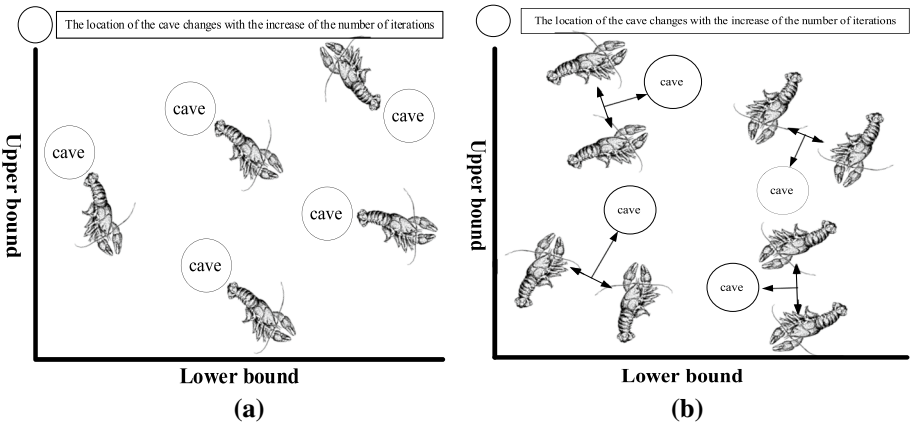

Fig. 5 **A** The crayfish enters the cave. **B** The crayfish compete with each other to get the cave

where X_G represents the optimal position obtained so far by the number of iterations, and X_L represents the optimal position of the current population.

Crayfish fighting for caves is a random event. When $rand < 0.5$, it means that there is no other crawfish competing for caves, and the crawfish will directly enter the cave for summer vacation. As shown in Fig. 5A. At this time, the crayfish will enter the cave for summer resort using Eq. (6):

$$X_{i,j}^{t+1} = X_{i,j}^{t} + C_2 \times rand \times \left(X_{shade} - X_{i,j}^{t} \right), \tag{6}$$

where t represents the current iteration number, and $t+1$ represents the next generation iteration number, C_2 is a decreasing curve, as shown in Eq. (7).

$$C_2 = 2 - (t/T), \tag{7}$$

where T represents the maximum number of iterations.

At the Summer resort stage, the goal of crayfish is to approach the cave, which represents the optimal solution. At this point, the crayfish will approach the cave. This brings individuals closer to the optimal solution and enhances the exploitation ability of COA. Enable the algorithm to converge faster.

2.5 Competition stage (exploitation)

When $temp > 30$ and $rand \geq 0.5$, it means that other crayfish are also interested in the cave. At this time, they will fight to get the cave. As shown in Fig. 5B. The crayfish competes for the cave through Eq. (8).

$$X_{i,j}^{t+1} = X_{i,j}^{t} - X_{z,j}^{t} + X_{shade}, \tag{8}$$

where z represents the random individual of crayfish, as shown in Eq. (9):

$$z = round(rand \times (N-1)) + 1. \tag{9}$$

In the Competition stage, crayfish compete with each other, and crayfish X_i adjusts their position based on the position X_z of another crayfish. By adjusting the position, the search range of COA is expanded, and the exploration ability of the algorithm is enhanced.

2.6 Foraging stage (exploitation)

When $temp \leq 30$, the temperature is suitable for the crayfish to feed. At this time, the crayfish will move towards the food. After finding the food, the crayfish will judge the size of the food. If the food is too large, the crayfish will tear the food with its claws and eat it with the second and third walking feet alternately. The food location X_{food} is defined as:

$$X_{food} = X_G. \tag{10}$$

Food size Q is defined as:

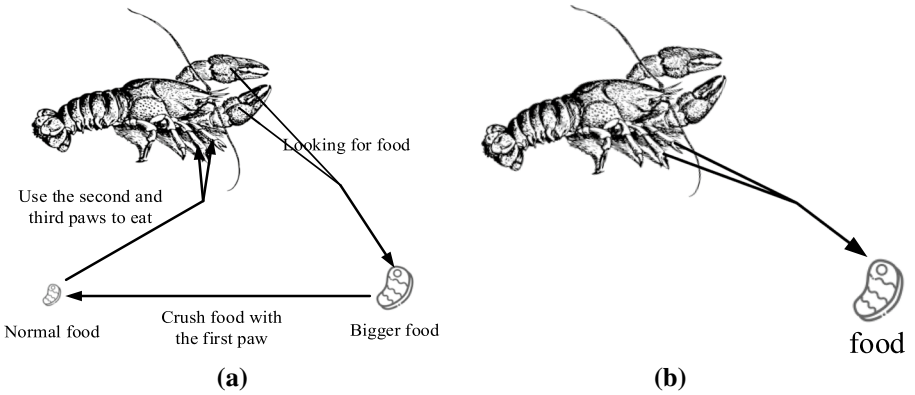

Fig. 6 **A** Crayfish shred food before eating. **B** Crayfish eat directly

$$Q = C_3 \times rand \times \left(fitness_i / fitness_{food}\right), \tag{11}$$

where C_3 is the food factor, representing the largest food, and the value is constant 3, $fitness_i$ represents the fitness value of the ith crayfish, and $fitness_{food}$ represents the fitness value of the food location.

The crayfish's judgment of food size comes from the size of the largest food. When $Q > (C_3+1)/2$, it means that the food is too big. At this time, the crayfish will tear the food with the first claw foot. As shown in Fig. 6A. The mathematical equation is as follows:

$$X_{food} = \exp\left(-\frac{1}{Q}\right) \times X_{food}. \tag{12}$$

when the food is shredded and becomes smaller, the second and third paws will alternately pick up the food and put it into the mouth. In order to simulate the alternating process, the combination of sine function and cosine function is used to simulate the alternating process. As shown in Fig. 6B. Not only that, the food obtained by crayfish is also related to the food intake, so the equation for foraging is as follows:

$$X_{i,j}^{t+1} = X_{i,j}^{t} + X_{food} \times p \times (\cos(2 \times \pi \times rand) - \sin(2 \times \pi \times rand)). \tag{13}$$

when $Q \le (C_3+1)/2$, the crayfish just needs to move towards the food and eat directly, the equation is as follows:

$$X_{i,j}^{t+1} = \left(X_{i,j}^{t} - X_{food}\right) \times p + p \times rand \times X_{i,j}^{t}. \tag{14}$$

In the Foraging stage, crayfish use different feeding methods based on the size of their food Q, and food X_{food} represents the optimal solution. When the size of the food Q is suitable for crayfish to eat, the crayfish will approach the food. When Q is too large, it indicates

that there is a significant difference between the crayfish and the optimal solution. Therefore, X_{food} should be reduced and brought closer to the food. And control the randomness of the food intake enhancement algorithm for crayfish. Through the foraging stage, COA will approach the optimal solution, enhancing the exploitation ability of the algorithm and making it have good convergence ability.

2.7 Algorithm implementation

Step 1: Parameter definition and initialization of population

Define the number of iterations T, population N, dimension dim, upper bound ub, and lower bound lb. Initialize population X based on upper and lower bounds. The population initialization is obtained from Eq. (2).

Step 2: Temperature definition

The ambient temperature of crayfish is defined according to Eq. (3) to make COA enter different stages.

Step 3: Summer resort stage and competition stage

3.1. When $temp > 30$ and $rand < 0.5$, COA entered the summer resort stage. At this time, COA obtains a new position $X_{i,j}^{t+1}$ according to the cave position (X_{shade}) and crayfish position ($X_{i,j}^{t}$). The updated formula is Eq. (6). Then go to step 5.

3.2. When $temp > 30$ and $rand \geq 0.5$, COA enters the competitive stage. At this time, the two crayfish will compete for the cave according to Eq. (8), and obtain a new position $X_{i,j}^{t+1}$ according to the cave position (X_{shade}) and the position ($X_{i,j}^{t}$, $X_{z,j}^{t}$) of the two crayfish. Then go to step 5.

Step 4: Foraging stage

4.1. When $\text{temp} \leq 30$, COA enters the foraging stage, the food intake p and the food size Q are defined according to Eqs. (4) and (11).

4.2. If $Q > (C_3 + 1)/2$, shred the food according to Eq. (12). After that, obtain the new position through Eq. (13) and proceed to step 5.

4.3. If $Q \leq (C_3 + 1)/2$, obtain a new position through Eq. (14) and proceed to step 5.

Step 5: Evaluation function

Evaluate the population and determine whether to exit the cycle. If not, return to step 2.

Step 6: output the best fitness value $fitness_{best}$

This section introduces a new swarm intelligence optimization algorithm. It establishes a mathematical model based on the behaviors of crayfish such as summer resort, competition and foraging. So far, there is no optimization literature about crayfish. Compared with other optimization algorithms, COA is special in that it simulates the summer resort mechanism, competition mechanism and foraging mechanism of crayfish. And COA also simulated the process of crayfish decomposing large food. From the results on solving 53 mathematical functions and 5 engineering problems, it can be seen that COA has great advantages compared with existing meta-heuristic algorithms.

2.8 Complexity analysis

2.8.1 Time complexity

The time complexity is related to the number of crayfish (N), the dimension of the problem (D), the number of iterations of the COA algorithm (T) and the evaluation cost of the function (C). Therefore, the time complexity of COA is expressed as follows:

$$\begin{aligned} O(\text{COA}) = {} & O(\text{define algorithm parameters}) + O(\text{initialize population}) \\ & + O(\text{function evaluation cost}) + O(\text{population update}). \end{aligned} \tag{15}$$

The time complexity of COA is as follows:

1. Define the time O (1) of the required parameter.
2. Position time of initialization population O ($N \times D$).
3. Calculate the time cost O ($T \times N \times C$) of the evaluation function.
4. Time required for updating population position O ($T \times N \times D$).

Therefore, the time complexity of COA is:

$$O(COA) = O(1 + N \times D + T \times N \times C + T \times N \times D). \tag{16}$$

When setting optimization parameters, T is usually a large value. So, $1 << T \times N \times C$, $1 << T \times N \times D$, $N \times D << T \times N \times C$ and $N \times D << T \times N \times D$. Therefore, Eq. (16) can be simplified into Eq. (17).

$$O(COA) \cong O(T \times N \times (C + D)). \tag{17}$$

The above formula shows that the time complexity of COA is polynomial. Therefore, the time complexity of COA is very low, and the proposed COA can be regarded as an optimization algorithm with very high computational efficiency.

2.8.2 Spatial complexity

The relationship between the spatial complexity of COA and the amount of memory space depends on the number of crayfish (N) and the problem dimension (D). Because the space is determined during initialization. Therefore, the spatial complexity of COA is as follows:

$$O(N \times D). \tag{18}$$

2.9 Pseudo code and flow chart of COA

The pseudo code of COA is as follows:

Algorithm 1.Crayfish optimization algorithm pseudo-code

Initialization iterations T, population N, dimension dim

Randomly generate an initial population

Calculate the fitness value of the population to get X_G, X_L

While $t<T$

  Defining temperature temp by Eq.(3)

  If $temp>30$

    Define cave X_{shade} according to Eq.(5)

    If $rand<0.5$

      Crayfish conducts the summer resort stage according to Eq.(6)

    Else

      Crayfish compete for caves through Eq.(8)

    End

  Else

    The food intake p and food size Q are obtained by Eq.(4) and Eq.(11)

    If $Q>2$

      Crayfish shreds food by Eq.(12)

      Crayfish foraging according to Eq.(13)

    Else

      Crayfish foraging according to Eq.(14)

    End

  End

  Update fitness values, X_G, X_L

  $t=t+1$

End

The flow chart of COA is shown in Fig. 7.

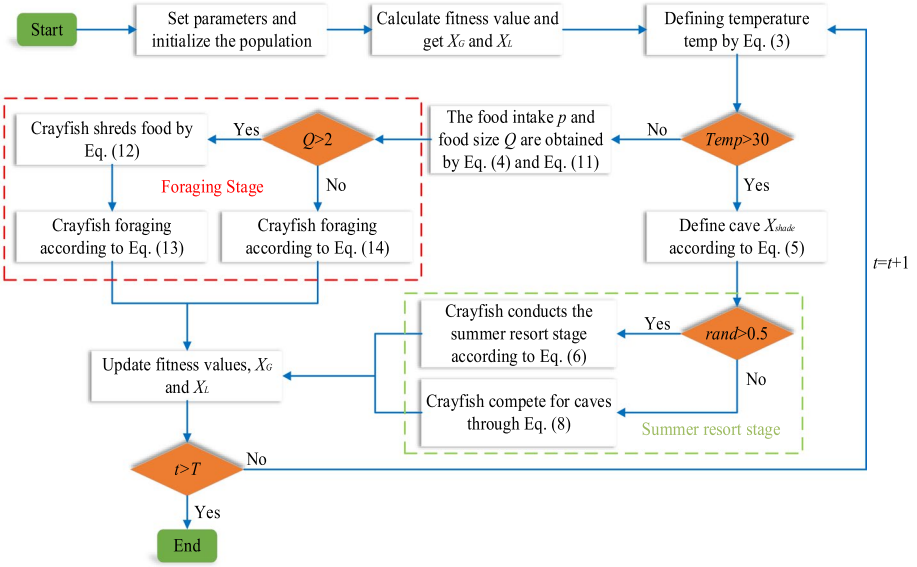

Fig. 7 Flow chart of COA

3 Experimental results and discussion

To prove that COA has good optimization effect and convergence performance. This paper selects 23 standard test functions and CEC2014 test functions to verify the optimization capability of COA. The summary is divided into three parts. The first part is the experimental results and analysis of 23 standard reference functions. The second part is the experimental results and analysis of CEC2014 test function. The third part is the parameter sensitivity analysis of food factor C3 in COA. All the experiments in this paper are completed on the computer with the 11th Gen Intel(R) Core(TM) i7-11700 processor with the primary frequency of 2.50GHz, 16GB memory, and the operating system of 64-bit windows 11 using matlab2021a.

In order to better show the optimization effect of COA. We used 9 classic comparison algorithms for comparison and verification. The 9 comparison algorithms are: Sine Cosine Algorithm (SCA) (Mirjalili 2016a, b), Sand Cat Swarm Optimization (SCSO) (Seyyedabbasi and Kiani 2022), Remora Optimization Algorithm (ROA) (Jia et al. 2021), Genetic Algorithms (GA) (Holland 1992), Sooty Tern Optimization Algorithm (STOA) (Dhiman and Kaur 2019), Bald Eagle Search (BES) (Alsattar et al. 2020), Prairie Dog Optimization Algorithm (PDO) (Ezugwu et al. 2022), Whale Optimization Algorithm (WOA) (Mirjalili and Lewis 2016), Arithmetic Optimization Algorithm (AOA) (Abualigah et al. 2021a, b). These algorithms have received high attention in swarm intelligence optimization

algorithms and have good optimization effect and representativeness. Compared with these algorithms, it can show the superiority of COA. The parameter settings of these ten algorithms are shown in Table1.

3.1 Experiments on 23 standard test functions

In this paper, 23 standard test functions are used to verify the optimization effect of COA, and nine comparison algorithms are used for comparative experiments. The expressions of 23 standard test functions are shown in Tables 2, 3 and 4 (Wu et al. 2022). Three-dimensional view is shown in Figs. 8, 9, 10. The distribution of COA in some 23 standard test functions is shown in Fig. 11.

To increase the fairness of the comparison experiment, we set the number of population $N=30$, the maximum number of iterations $T=500$, and the dimension $dim=30/500$ in the experiment.

Table 1 Parameter setting of each algorithm

Algorithm	Parameters	Value
AOA	*MOP_Max*	1
	MOP_Min	0.2
	A	5
	Mu	0.499
WOA	$\vec{A}$	1
	$\vec{C}$	[− 1, 1]
	b	0.75
	l	[− 1, 1]
PDO	*MOP_Max*	1
	MOP_Min	0.2
	A	5
	Mu	0.499
BES	α	[1.5, 2.0]
	r	[0, 1]
STOA	*Sa*	[0, 2]
	b	1
GA	*Type*	Real coded
	Selection	Roulette wheel
	Crossover	0.7
	Mutation	0.01
ROA	*C*	0.1
SCSO	*SM*	2
	Roulette wheel selection	[0, 360]
	C	0.35
SCA	α	2
COA	C_1	0.2
	C_3	3
	μ	25
	σ	3

Table 2 Equation of unimodal test function

F	*dim*	Range	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30/500	[− 100, 100]	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30/500	[− 10, 10]	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	30/500	[− 100, 100]	0
$F_4(x) = \max \left\{ \lvert x_i \rvert, 1 \le i \le n \right\}$	30/500	[− 100, 100]	0
$F_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + \left(x_i - 1 \right)^2 \right]$	30/500	[− 30, 30]	0
$F_6(x) = \sum_{i=1}^{n} \left(x_i + 5 \right)^2$	30/500	[− 100, 100]	0
$F_7(x) = \sum_{i=1}^{n} i \times x_i^4 + random[0, 1)$	30/500	[− 1.28, 1.28]	0

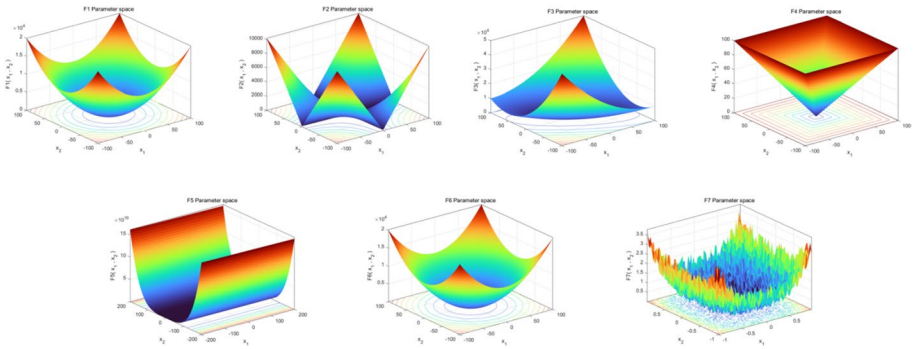

Fig. 8 Schematic diagram of unimodal test functions

Table 3 Equation of multimodal test functions

F	*dim*	Range	f_{min}
$F_8(x) = \sum_{i=1}^{n} - x_i \sin(\sqrt{\lvert x_i \rvert})$	30/500	[− 500, 500]	− 418.9829 × *dim*
$F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30/500	[− 5.12, 5.12]	0
$F_{10}(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e)$	30/500	[− 32, 32]	0
$F_{11}(x) = \frac{1}{400}\sum_{i=1}^{n} x_i^2 - \Pi_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30/500	[− 600, 600]	0
$F_{12}(x) = \frac{\pi}{n}\left\{ 10\sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\}$ $+ \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, where $y_i = 1 + \frac{x_i + 1}{4}$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a^m) & x_i \end{cases}$	30/500	[− 50, 50]	0
$F_{13}(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)]$ $+ (x_n - 1)^2[1 + \sin^2(2\pi x_n)]) + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30/500	[− 50, 50]	0

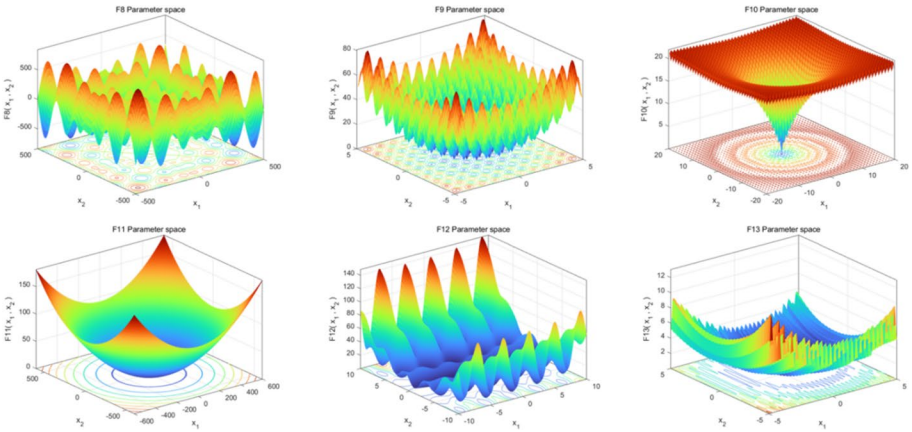

Fig. 9 Schematic diagram of multimodal test functions

Table 4 Equation of Fixed-dimension multimodal test functions

F	*dim*	Range	f_{min}
$F_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	[− 65, 65]	1
$F_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4	[− 5, 5]	0.00030
$F_{16}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+x_2^4$	2	[− 5, 5]	− 1.0316
$F_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	[− 5, 5]	0.398
$F_{18}(x)=\left[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right]$ $\times\left[30+(2x_1-3x_2)^2\times(18-32x_2+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right]$	5	[− 2, 2]	3
$F_{19}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$	3	[− 1, 2]	− 3.86
$F_{20}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{6}a_{ij}(x_j-p_{ij})^2\right)$	6	[0, 1]	− 3.32
$F_{21}(x)=-\sum_{i=1}^{5}\left[(X-a_i)(X-a_i)^T+c_i\right]^{-1}$	4	[0, 10]	− 10.1532
$F_{22}(x)=-\sum_{i=1}^{7}\left[(X-a_i)(X-a_i)^T+c_i^{-1}\right]$	4	[0, 10]	− 10.4028
$F_{23}(x)=-\sum_{i=1}^{10}\left[(X-a_i)(X-a_i)^T+c_i^{-1}\right]$	4	[0, 10]	− 10.5363

3.1.1 Analysis of statistical results of 23 standard test functions

Tables 5, 6 and 7 shows the statistical results of data obtained by COA and 9 comparison algorithms running independently for 30 times. Where, mean is the average fitness value, std is the standard deviation of fitness value, and the bold part is the best result obtained by the 10 algorithms in the same function.

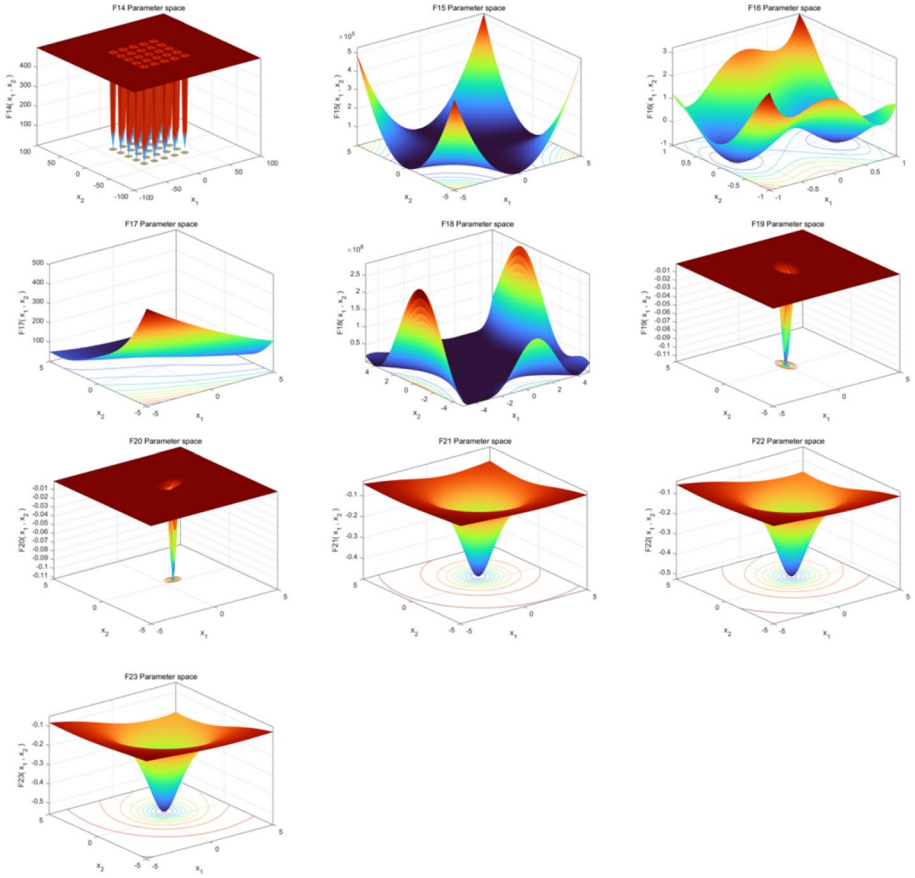

Fig. 10 Schematic diagram of fixed-dimension multimodal test functions

It can be seen from Tables 5, 6 and 7 that COA has achieved good results. Table 5 is the results of 23 standard benchmark functions with dimension 30 obtained by ten algorithms. According to the data. In F1–F4, both COA and PDO algorithms achieve theoretical optimal values. ROA achieves the theoretical optimum value in F1. The fitness values obtained by SCA, GA, STOA and AOA algorithms in F1–F4 are general. Although COA does not achieve optimal results in F5, F6, and F8, its fitness value is superior to most algorithms. PDO reaches the optimal average in F5, but std isn't as good as AOA. WOA achieves the best average in F6, but std isn't as good as GA. ROA achieves the best mean in F8, but std isn't as good as SCA. In F7, COA is obviously superior to other algorithms and can find better fitness values. The fitness values obtained by other comparative algorithms are not significantly different, but they are not as good as COA. In F9–F11, COA, SCSO, ROA and PDO algorithms have achieved the best results, indicating that these algorithms have good optimization effects. In F12 and F13, although COA does not get the best results, the fitness value obtained is better than SCA, GA, STOA and AOA on the whole. Table 6 shows the results obtained by ten algorithms on 23 standard benchmark functions with a dimension of 500. The high-dimensional optimization test of the algorithm is even more challenging. In F1–F4, COA and PDO algorithms still achieved theoretical optimal values.

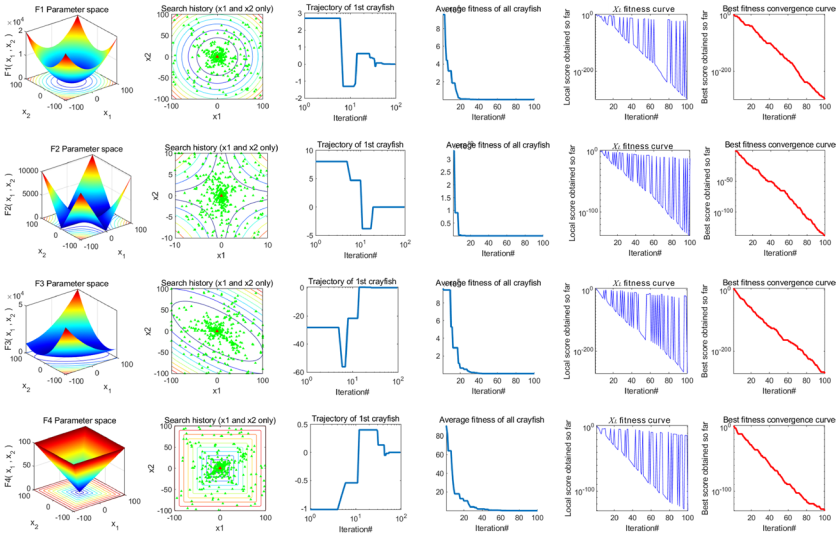

Fig. 11 Historical search distribution of COA in some 23 standard test functions

The values obtained by SCA, SCSO, GA, and AOA algorithms are significantly lower than the data in Table 5. In F5, F6, and F8, COA does not achieve the best performance in the comparison algorithm, but it also outperformed SCA, GA, and AOA. Other comparative algorithms don't achieve the best results in all of these test functions. In F5, the std of BES algorithm is not as good as PDO algorithm. In F6, the std of ROA isn't as good as AOA. In F8, the std of ROA is not as good as SCA. In F7, COA achieves the best mean and std. Compared with Table 5, The SCA shows a significant increase in mean and std. In F9 - F11, COA, SCSO, ROA, BES and PDO algorithms have achieved the best results. In F12 and F13, although COA does not get the best results. The fitness value obtained by ROA is better. In F7, COA achieves the best fitness values in F15, F16, F17, F18, F19 and F20. In F14, F21, F22 and F23, although COA does not achieve the best results overall. To sum up, COA has a good optimization effect in 23 standard test functions.

3.1.2 Convergence curve analysis of 23 standard test functions

Figures 12, 13 and 14 are the convergence curves of COA and 9 comparison algorithms on 23 standard reference functions. It can be seen from the comparison between Figs. 12 and 13 that COA has good convergence ability in both high and low dimensions. In F1–F4, COA can quickly converge and obtain a good fitness value. ROA and BES also have good convergence ability. AOA can quickly converge in the later stage of F2. The convergence ability of other algorithms is average. In F5, COA and many algorithms have found a good value in the early stage of iteration. In the F6 and F8 functions of 30 dimensions, COA can converge well, and the results obtained are better than SCA, PDO, GA, AOA, SCSO algorithm, but weaker than WOA. In the F6 function of 500 dimensions, COA is difficult to converge. ROA, WOA, and BES can converge lower and obtain better fitness values. In the F8 function of 500 dimensions, COA can continuously converge, but the final fitness value obtained is not as good as ROA, BES, and WOA. In F7, COA can jump out of the local optimum and obtain the best fitness value. In F9–F11, COA, ROA and STOA can

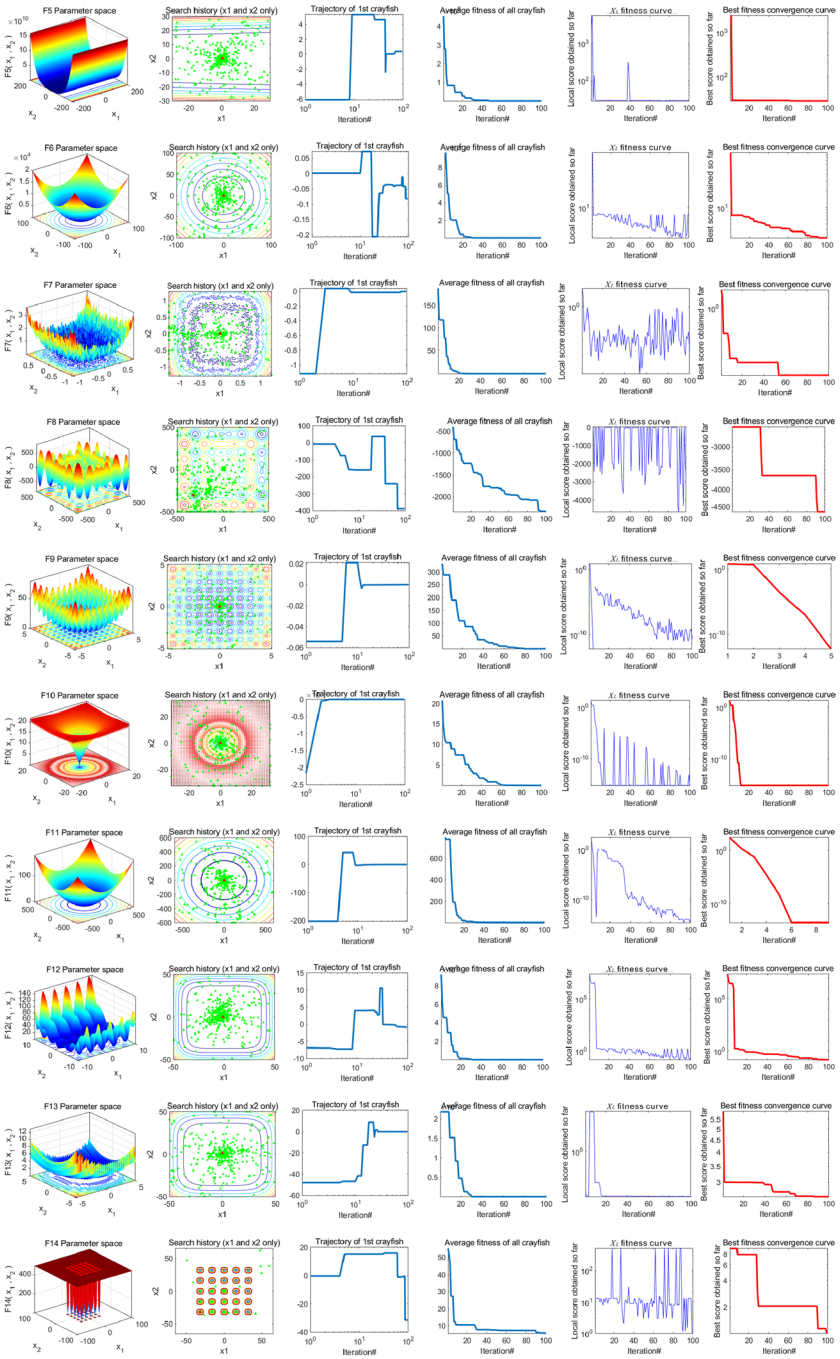

Fig. 11 (continued)

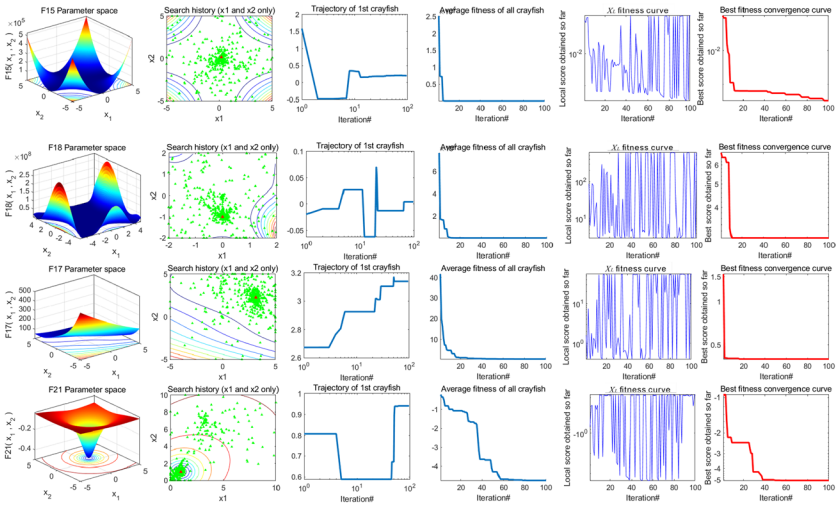

Fig. 11 (continued)

quickly converge and obtain the optimal fitness value. GA, SCA, and STOA are difficult to converge and cannot obtain better fitness values. COA fell into local optimum in F12–F13, but still can get a better result. As can be seen from Fig. 14, COA can find a good fitness value in the Fixed - dimension multimodal test functions. GA is difficult to converge in many functions. SCA is difficult to converge in F15 and F21. STOA is difficult to converge in F22. The other algorithms can converge, but the convergence results are not satisfactory. In F14, F15 and F23, COA can obtain better fitness values in convergence.

3.1.3 Wilcoxon rank sum test results analysis of 23 standard test functions

The above experiments are all the results obtained by independent operation, and there is no direct comparison of algorithm differences. Wilcoxon's rank sum test is a nonparametric statistical test. The differences between different algorithms are found through the comparison between algorithms. The significance level is 5%. Less than 5% indicates a significant difference between the two algorithms. Table 8 shows the statistical results of COA and 9 different algorithms in 30 runs. From the data in the table, it can be seen that COA is significantly different from most algorithms. However, there are many results equal to 1 in F1–F4, which is because COA and these algorithms have reached the theoretical optimal value and obtained the same results. In F9–F11, the results obtained by COA are the same as those obtained by SCSO algorithm, ROA, BES and PDO algorithm. Among other functions, COA is different from these functions, and there are significant differences in most functions.

3.2 Experiments on CEC2014 test function

The 23 standard test functions are easy to find good fitness values due to their simple formulas, and are suitable for testing the ability of the algorithm to solve simple problems. However, this is not enough to fully prove the optimization effect of COA. In order to

Table 5 Statistical results of 10 algorithms in 23 standard reference functions (F1–F13) at $dim = 30$

F	Metric	COA	SCA	SCSO	ROA	GA	STOA	BES	PDO	WOA	AOA
F1	Mean	**0**	12.46449973	3.8119×10^{-113}	**0**	0.022503892	4.61083×10^{-07}	1.47×10^{-313}	**0**	1.13447×10^{-73}	2.37053×10^{-19}
	Std	**0**	27.67489163	1.3814×10^{-112}	**0**	0.007222287	9.11719×10^{-07}	**0**	**0**	6.21015×10^{-73}	1.2965×10^{-18}
F2	Mean	**0**	0.018464765	7.29043×10^{-1}	3.3611×10^{-162}	0.479849018	1.55971×10^{-05}	7.9658×10^{-149}	**0**	5.09636×10^{-51}	**0**
	Std	**0**	0.028799132	2.42417×10^{-0}	1.8329×10^{-161}	0.083778024	1.985×10^{-05}	4.363×10^{-148}	**0**	1.30301×10^{-50}	**0**
F3	Mean	**0**	9008.355414	2.61049×10^{-4}	2.39×10^{-310}	20713.91316	0.168974565	105.0227622	**0**	47560.79019	0.005001788
	Std	**0**	5224.6003	1.42981×10^{-93}	**0**	9570.036865	0.278483497	575.2333588	**0**	12268.73585	0.013242648
F4	Mean	**0**	39.70886538	3.44357×10^{-50}	2.4637×10^{-171}	0.285352582	0.040069253	1.0679×10^{-165}	**0**	54.03888827	0.02791644
	Std	**0**	14.02416817	1.56579×10^{-49}	**0**	0.050452749	0.055150167	**0**	**0**	31.6723265	0.018174524
F5	Mean	27.04692264	205501.322	28.18300294	27.80896101	161.0511146	28.33546284	25.46547976	**15.0713288**	27.94174031	28.45948082
	Std	0.692370942	520914.5013	0.81476117	5.05990838	471.4022029	0.485680484	8.756375257	12.74886492	0.464489068	**0.183961739**
F6	Mean	0.656887855	32.7079201	1.94342455	0.620921533	8.03634482	2.623888828	1.339297968	3.223407748	**0.492862701**	3.223114001
	Std	0.35286518	53.54549005	0.582094136	0.326067601	**0.111967573**	0.480703812	2.805186813	1.58692712	0.31013118	0.253798125
F7	Mean	$\mathbf{5.44863 \times 10^{-05}}$	0.267326423	0.000191213	0.000128756	0.189916509	0.007002442	0.006925512	0.000118025	0.003892378	7.95394×10^{-05}
	Std	$\mathbf{4.67701 \times 10^{-05}}$	0.476019827	0.000400525	0.000131071	0.067701627	0.004814106	0.00407002	0.000113049	0.004496881	7.60094×10^{-05}
F8	Mean	− 6584.343718	− 3715.632157	− 6952.755325	**− 12470.17226**	− 4667.893903	− 5140.392629	− 9416.846541	− 3735.066753	− 9909.281431	− 5448.307177
	Std	1273.657393	**271.6001687**	794.0502982	325.1840758	710.7387755	458.1062999	2168.020448	315.422366	1767.466011	416.3662867
F9	Mean	**0**	43.77229707	**0**	**0**	2.80082631	16.7383095	5.787913569	**0**	1.89478×10^{-15}	**0**
	Std	**0**	43.75306076	**0**	**0**	0.721483284	32.70724162	31.70170823	**0**	1.03781×10^{-14}	**0**
F10	Mean	$\mathbf{8.88178 \times 10^{-16}}$	14.93964441	$\mathbf{8.88178 \times 10^{-16}}$	$\mathbf{8.88178 \times 10^{-16}}$	0.138070402	19.95952256	$\mathbf{8.88178 \times 10^{-16}}$	$\mathbf{8.88178 \times 10^{-16}}$	4.67774×10^{-15}	$\mathbf{8.88178 \times 10^{-16}}$
	Std	**0**	8.135436092	**0**	**0**	0.036162017	0.002136186	**0**	**0**	2.27261×10^{-15}	**0**
F11	Mean	**0**	1.059670224	**0**	**0**	0.046316477	0.034186644	**0**	**0**	0.00462941	0.193948952
	Std	**0**	0.48857014	**0**	**0**	0.170763591	0.040153506	**0**	**0**	0.025356322	0.169482023
F12	Mean	0.025348284	20962.86841	0.097671919	0.033242504	1.669823181	0.31333614	0.110616614	0.380788659	**0.021075793**	0.514650426
	Std	**0.014422954**	98874.98673	0.043992774	0.024627405	0.162703241	0.179657448	0.319716246	0.405975454	0.017064804	0.049691798
F13	Mean	2.374527441	459225.0641	2.551752655	0.273904618	**0.004510582**	1.936451722	1.245972856	2.975681786	0.545746244	2.80577021
	Std	0.255270082	2257569.144	0.28642411	0.170791653	**0.003060361**	0.275378631	1.437468285	0.051991296	0.316269416	0.105027131

The bold part represents the best solution obtained from 10 comparative algorithms

Table 6 Statistical results of 10 algorithms in 23 standard reference functions (F1–F13) at $dim = 500$

F	Metric	COA	SCA	SCSO	ROA	GA	STOA	BES	PDO	WOA	AOA
F1	Mean	**0**	208719.0464	3.39179×10^{-97}	**0**	71.33481105	6.609821353	**0**	**0**	1.13518×10^{-68}	0.64147806
	Std	**0**	69313.99552	1.5566×10^{-96}	**0**	2.396205008	5.276233151	**0**	**0**	6.16721×10^{-68}	0.037147403
F2	Mean	**0**	117.1552721	4.22125×10^{-52}	2.3316×10^{-164}	138.2148376	0.134654023	3.5395×10^{-175}	**0**	2.91115×10^{-49}	0.001615272
	Std	**0**	55.78154533	8.55249×10^{-52}	**0**	2.668723277	0.098282568	**0**	**0**	1.02817×10^{-48}	0.001721245
F3	Mean	**0**	7215844.766	1.18369×10^{-83}	8.6637×10^{-262}	684046.1523	543352.4997	42561.21841	**0**	31440651.5	118028.2174
	Std	**0**	1444445.152	4.52616×10^{-52}	**0**	114747.5046	196649.8487	233071.6542	**0**	10955203.81	646280.0445
F4	Mean	**0**	98.96703548	2.04007×10^{-44}	1.0043×10^{-167}	0.968174779	98.95602501	1.5634×10^{-173}	**0**	73.77019619	0.179769518
	Std	**0**	0.468656426	1.04135×10^{-43}	**0**	0.01075101	0.495887124	**0**	**0**	25.64707783	0.019969361
F5	Mean	497.9366009	2034721536	498.4648312	494.4706234	5120.50857	16781.42798	**432.203889**	498.9936145	496.2162896	499.0643095
	Std	0.136467611	532404040.3	0.098183562	0.261164756	115.4902541	18604.7166	166.4715812	**0.008165841**	0.435087353	0.071506572
F6	Mean	96.87159589	191050.2693	104.6773376	**10.79568869**	343.0934191	127.2350145	14.26320755	94.36529832	33.65602761	115.761869
	Std	3.411636092	82269.45429	4.618431618	6.133875287	4.726841821	12.81767032	37.65489595	32.27900651	10.64954754	**1.422800492**
F7	Mean	$\mathbf{4.8611\times10^{-05}}$	16030.05062	0.000203497	0.000132661	4508.57485	0.430000372	0.006400406	8.90149×10^{-05}	0.004531407	0.000117497
	Std	$\mathbf{4.77191\times10^{-05}}$	3467.713624	0.000271903	0.000115447	300.8306339	0.160724817	0.003679235	8.30906×10^{-05}	0.006327394	9.52148×10^{-05}
F8	Mean	− 34071.82575	− 15411.72537	− 60184.8166	**− 207478.6338**	− 32941.4469	− 24109.22472	− 142980.7981	− 20079.39926	− 175642.3913	− 22868.14244
	Std	9866.925652	**1065.519182**	5143.412085	4584.466098	1696.214419	3785.890731	33878.25697	4179.19732	29518.02548	2083.666427
F9	Mean	**0**	1186.625395	**0**	**0**	2394.809596	27.97133478	**0**	**0**	**0**	6.40253×10^{-06}
	Std	**0**	648.478523	**0**	**0**	68.1865915	27.17647785	**0**	**0**	**0**	7.13504×10^{-06}
F10	Mean	$\mathbf{8.88178\times10^{-16}}$	19.70321343	$\mathbf{8.88178\times10^{-16}}$	$\mathbf{8.88178\times10^{-16}}$	2.910072729	19.96658645	$\mathbf{8.88178\times10^{-16}}$	$\mathbf{8.88178\times10^{-16}}$	4.20404×10^{-15}	0.007972383
	Std	**0**	2.818042918	**0**	**0**	0.028963117	5.98475×10^{-5}	**0**	**0**	3.22298×10^{-15}	0.000370792
F11	Mean	**0**	1864.149525	**0**	**0**	0.259956812	0.711903797	**0**	**0**	**0**	9843.799101
	Std	**0**	866.6441082	**0**	**0**	0.106728832	0.359814264	**0**	**0**	**0**	2959.619342
F12	Mean	0.753407718	6313653523	0.776198191	**0.02454821**	2.804968451	4.935038622	0.123590644	0.689486897	0.101053955	1.080502296
	Std	0.047786234	1005888468	0.067069386	0.017695387	0.047293276	1.787375706	0.366571559	0.462958009	0.044859387	**0.012761585**
F13	Mean	49.7873507	9731656312	49.83415324	**5.876894347**	10.85681785	168.0156556	13.88921152	49.99493465	17.81196225	50.20708253
	Std	0.140161519	1613171502	0.078576467	3.770046105	0.354397982	89.24050371	21.88438865	**0.02468735**	5.801741945	0.041330919

The bold part represents the best solution obtained from 10 comparative algorithms

Table 7 Statistical results of 10 algorithms in 23 standard reference functions (F14–F23)

F	Metric	COA	SCA	SCSO	ROA	GA	STOA	BES	PDO	WOA	AOA
F14	Mean	2.829476547	**2.57016816**	4.03718315	6.071634461	9.318267186	3.093406185	3.480227177	6.358916504	3.673497251	10.35453717
	Std	3.281606393	2.920527603	3.871664309	5.164598891	4.204359036	3.220770711	**1.58700685**	4.267170968	4.044674107	3.602044039
F15	Mean	**0.000499408**	0.001127175	0.000545188	0.001010503	0.014566567	0.003047933	0.00932339	0.006163561	0.001325962	0.025235382
	Std	**0.000148677**	0.000386225	0.000370128	0.001004594	0.025962911	0.005875084	0.010437186	0.018859093	0.003373334	0.034533571
F16	Mean	**− 1.031628453**	− 1.031554839	− 1.031628453	− 1.031628403	− 0.99561839	− 1.031625412	− 0.951406842	− 1.03023582	− 1.031628452	− 1.031628237
	Std	$\mathbf{5.19188\times10^{-12}}$	6.52219×10^{-05}	9.62211×10^{-10}	6.50165×10^{-08}	0.024723677	3.25873×10^{-06}	0.225405483	0.00548254	4.95973×10^{-09}	1.6069×10^{-07}
F17	Mean	**0.397887364**	0.399492751	0.397887376	0.412780395	1.460668701	0.398031077	0.757287449	0.40028073	0.39789633	0.397887457
	Std	$\mathbf{7.74122\times10^{-09}}$	0.001785387	2.78611×10^{-08}	0.081544731	1.078628111	0.000136105	0.732705463	0.012236169	2.53568×10^{-05}	7.48666×10^{-08}
F18	Mean	**3.000000006**	3.000145684	3.000007398	5.791256864	42.88429031	3.000139519	8.090817908	6.600497184	3.900715376	12.90118036
	Std	$\mathbf{1.15996\times10^{-08}}$	0.000432085	1.12561×10^{-05}	8.524680029	52.10266424	0.000203366	14.24040818	15.42893513	4.933113195	13.23267004
F19	Mean	**− 3.862782146**	− 3.848667789	− 3.859096559	− 3.771599101	− 3.557801253	− 3.854938336	− 3.510252638	− 3.767361251	− 3.831055073	− 3.799312287
	Std	$\mathbf{2.5294\times10^{-09}}$	0.01152759	0.004571861	0.236816345	0.595095875	0.008226989	0.348705568	0.522516818	0.140607667	0.200062499
F20	Mean	**− 3.290290069**	− 2.806495907	− 3.152070062	− 3.023967959	− 3.262908475	− 2.876254534	− 2.775236123	− 2.896539341	− 3.195863792	− 3.035689294
	Std	**0.053475408**	0.419455317	0.331211473	0.289960988	0.061944364	0.474712261	0.441704747	0.508525503	0.177458867	0.119319761
F21	Mean	− 8.555154038	− 2.20940134	− 4.961516657	**− 9.154127078**	− 1.19255615	− 2.850126732	− 5.330587931	− 4.83527145	− 7.863039543	− 3.538902608
	Std	2.359307838	1.867496159	2.604189146	2.199255752	1.329118858	3.636756823	2.703734084	2.532990771	3.117102379	**1.202001362**
F22	Mean	− 8.775387596	− 3.464927974	− 6.049351626	**− 9.72171656**	− 1.353185836	− 5.036201839	− 5.700156163	− 5.444428866	− 6.851462897	− 3.669530121
	Std	2.458651625	2.106447091	2.658240888	2.016294864	**1.250003445**	4.563548323	2.99837909	2.678034798	3.004990346	1.695917019
F23	Mean	− 9.274550765	− 3.902539156	− 6.267328247	**− 9.665476398**	− 2.029169243	− 6.235687737	− 5.897951606	− 5.542313518	− 6.162742247	− 3.507946668
	Std	2.326394508	1.908662981	3.023990395	2.213987964	1.75051273	4.455115567	2.861348286	3.062704835	3.270045392	**1.429785563**

The bold part represents the best solution obtained from 10 comparative algorithms

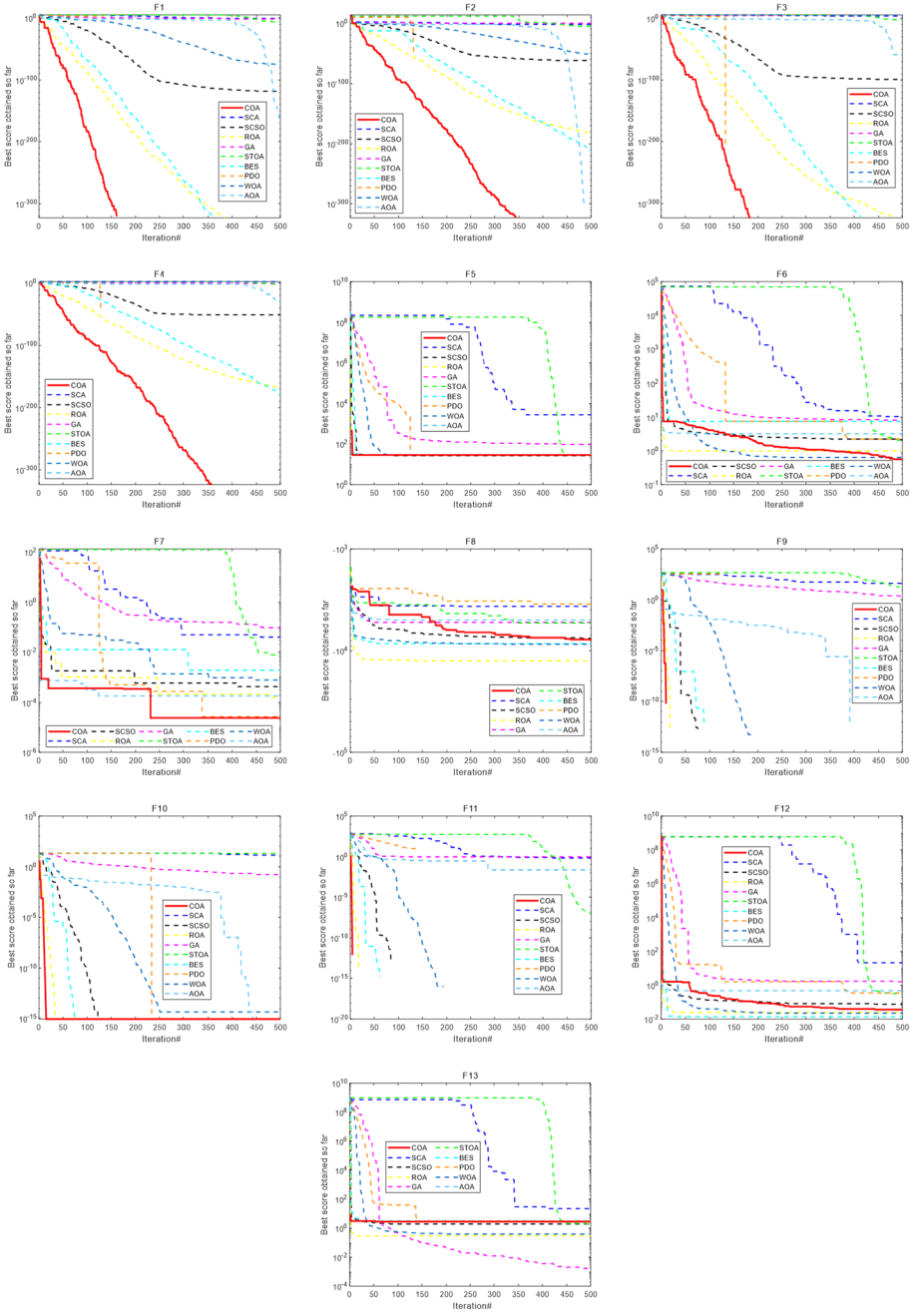

Fig. 12 10 algorithms' convergence in standard test functions(F1–F13) with $dim = 30$

prove that COA still has a good effect in solving complex problems, this section selects CEC2014 test function to test the ability of COA to solve complex problems. The content of CEC2014 test function is shown in the literature (Rao et al. 2022). Compared with

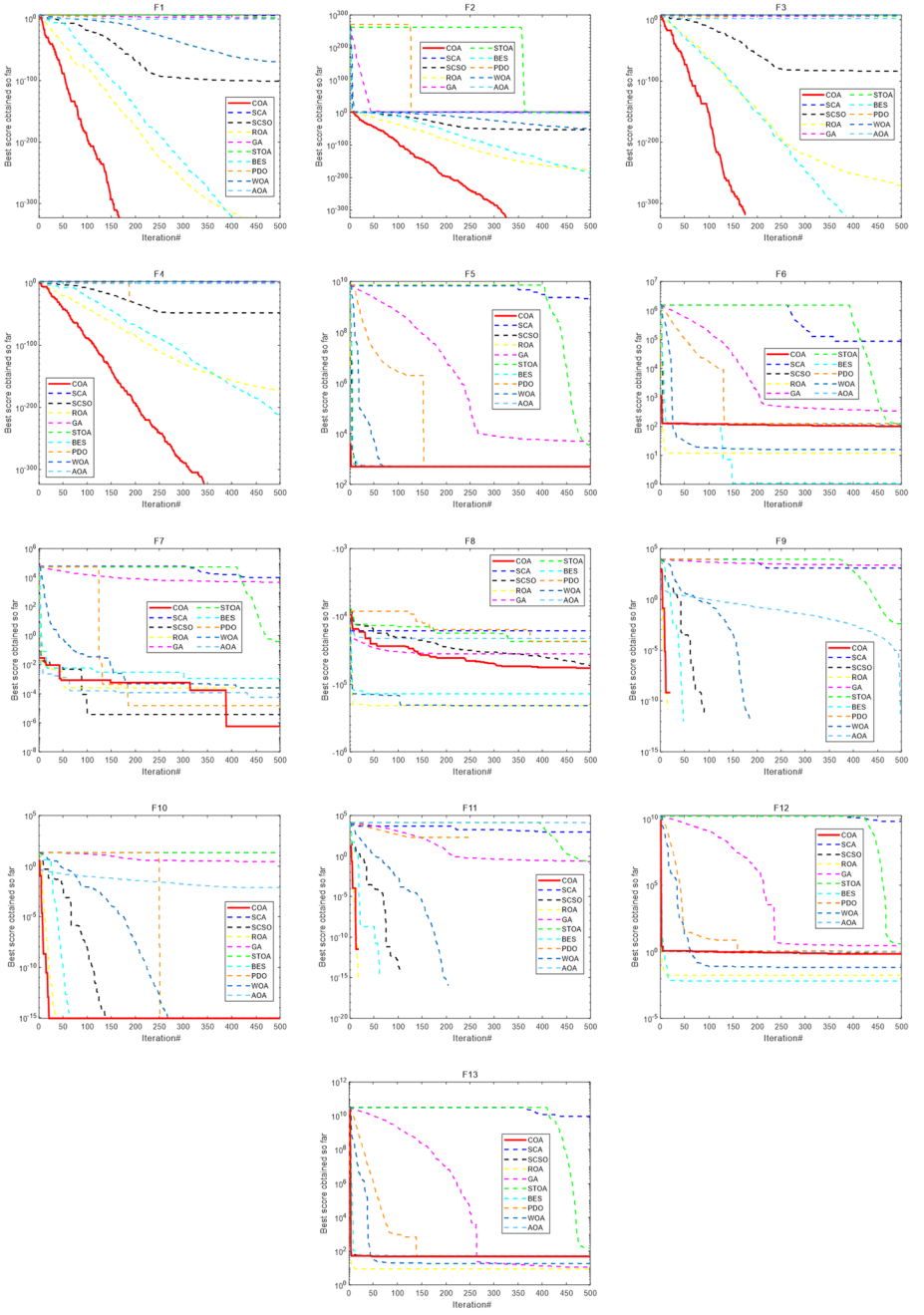

Fig. 13 10 algorithms' convergence in standard test functions(F1–F13) with $dim = 500$

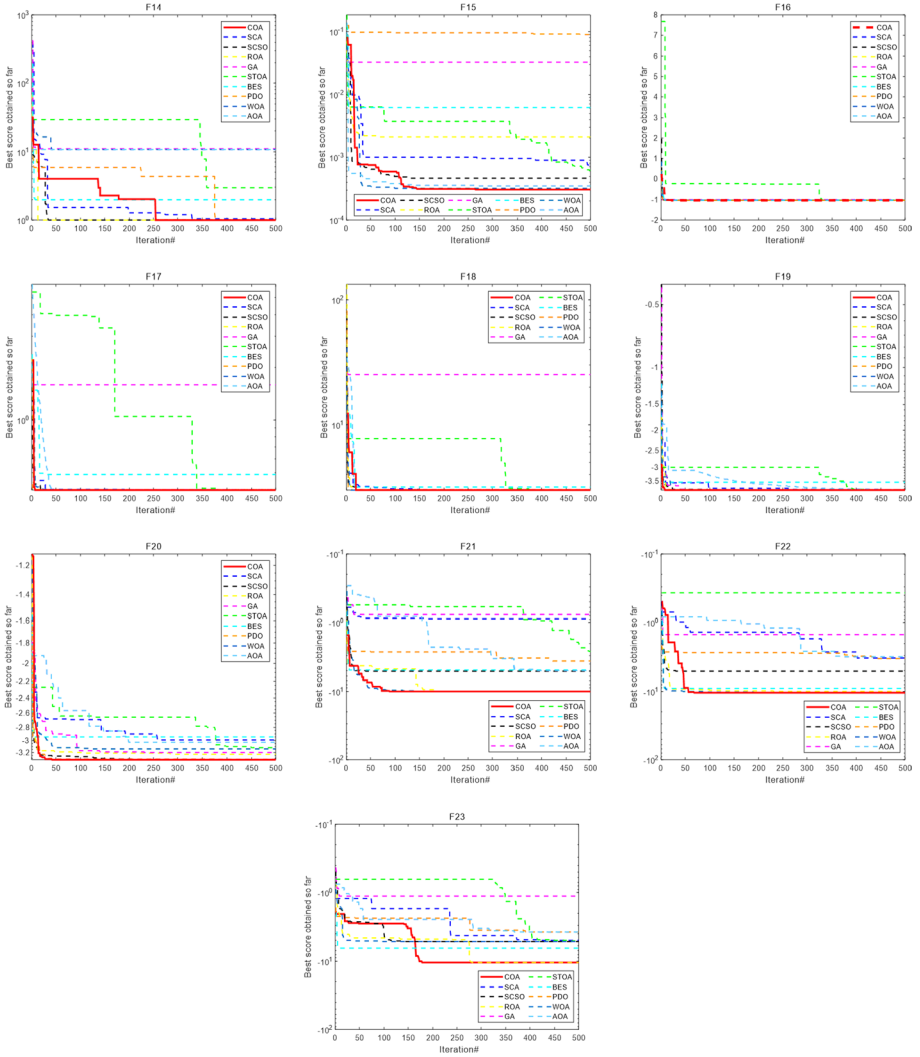

Fig. 14 10 algorithms' convergence in standard test functions(F14–F23)

the recent CEC test functions, CEC2014 test functions are more comprehensive. In addition, many of the recent CEC functions are taken from CEC2014 test functions. Therefore, CEC2014 test functions is still representative. Considering these factors, this paper selects CEC2014 as the test function.

3.2.1 Analysis of statistical results of CEC2014 test functions

Table 9 shows the statistical results of COA and 9 comparison algorithms independently run for 30 times. Where mean is the average fitness value, and std is the standard deviation

Table 8 Experimental results of Wilcoxon rank sum test on 23 standard test functions

F	*dim*	COA vs SCA	COA vs SCSO	COA vs ROA	COA vs GA	COA vs STOA	COA vs BES	COA vs PDO	COA vs WOA	COA vs AOA
F1	30	1.7344×10^{-06}	1.7344×10^{-06}	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}
	500	1.7344×10^{-06}	1.7344×10^{-06}	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}
F2	30	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1	1.7344×10^{-06}	1
	500	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1	1.7344×10^{-06}	1.7344×10^{-06}
F3	30	1.7344×10^{-06}	1.7344×10^{-06}	0.0078125	1.7344×10^{-06}	1.7344×10^{-06}	0.0625	1	1.7344×10^{-06}	1.7344×10^{-06}
	500	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	0.0625	1	1.7344×10^{-06}	1.7344×10^{-06}
F4	30	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1	1.7344×10^{-06}	1.7344×10^{-06}
	500	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1	1.7344×10^{-06}	1.7344×10^{-06}
F5	30	1.7344×10^{-06}	0.00089443	3.11232×10^{-05}	4.28569×10^{-06}	2.16302×10^{-05}	0.057096495	0.008729668	6.98378×10^{-06}	2.60333×10^{-06}
	500	1.7344×10^{-06}	2.8786×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	0.000261343	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}
F6	30	1.7344×10^{-06}	2.35342×10^{-06}	0.544006208	1.7344×10^{-06}	1.7344×10^{-06}	0.075213312	1.7344×10^{-06}	0.036826128	1.7344×10^{-06}
	500	1.7344×10^{-06}	2.12664×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	5.30699×10^{-05}	0.78126371	1.7344×10^{-06}	1.7344×10^{-06}
F7	30	1.7344×10^{-06}	0.318490603	0.7035637	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	0.926255087	1.7344×10^{-06}	0.014795424
	500	1.7344×10^{-06}	0.007730944	0.042766688	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	0.020671114	2.35342×10^{-06}	0.271155198
F8	30	1.92092×10^{-06}	0.7035637	1.7344×10^{-06}	4.28569×10^{-06}	1.49356×10^{-05}	1.7344×10^{-06}	1.7344×10^{-06}	1.92092×10^{-06}	4.7292×10^{-06}
	500	1.7344×10^{-06}	2.35342×10^{-06}	1.7344×10^{-06}	0.002765274	2.84342×10^{-05}	1.7344×10^{-06}	2.60333×10^{-06}	1.7344×10^{-06}	6.33914×10^{-06}
F9	30	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	1	1
	500	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	1	0.000196437
F10	30	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	1.52034×10^{-05}	1
	500	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	2.47127×10^{-06}	1.7344×10^{-06}
F11	30	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	1	1.7344×10^{-06}
	500	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	1	1	1	1.7344×10^{-06}
F12	30	1.7344×10^{-06}	1.7344×10^{-06}	0.452806505	1.7344×10^{-06}	1.7344×10^{-06}	0.002765274	2.12664×10^{-06}	0.158855499	1.7344×10^{-06}
	500	1.7344×10^{-06}	0.245190309	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	0.004114031	1.7344×10^{-06}	1.7344×10^{-06}
F13	30	1.7344×10^{-06}	0.673279807	1.7344×10^{-06}	1.7344×10^{-06}	2.8786×10^{-06}	1.36011×10^{-05}	3.11232×10^{-05}	1.7344×10^{-06}	2.12664×10^{-06}
	500	1.7344×10^{-06}	0.152860694	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	3.51524×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}

Table 8 (continued)

F	*dim*	COA vs SCA	COA vs SCSO	COA vs ROA	COA vs GA	COA vs STOA	COA vs BES	COA vs PDO	COA vs WOA	COA vs AOA
F14	2	0.001113801	0.191522107	0.271155198	0.000419551	0.006835856	0.184621877	0.271155198	0.135907785	0.000283079
F15	4	0.000331726	0.0003065	0.205888223	3.51524×10^{-06}	0.000114992	2.59671×10^{-05}	8.18775×10^{-05}	0.158855499	0.027029157
F16	2	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	0.002957462	7.69086×10^{-06}	1.7344×10^{-06}
F17	2	1.7344×10^{-06}	0.749871156	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	2.35342×10^{-06}	2.12664×10^{-06}	7.69086×10^{-06}
F18	5	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	3.11232×10^{-05}	3.88218×10^{-06}	0.000419551
F19	3	3.11232×10^{-05}	3.11232×10^{-05}	3.11232×10^{-05}	3.11232×10^{-05}	3.11232×10^{-05}	3.11232×10^{-05}	6.31976×10^{-05}	3.11232×10^{-05}	3.11232×10^{-05}
F20	6	1.92092×10^{-06}	0.000663921	1.7344×10^{-06}	0.530440091	1.7344×10^{-06}	1.7344×10^{-06}	6.98378×10^{-06}	0.008729668	1.7344×10^{-06}
F21	4	3.88218×10^{-06}	0.000331726	0.015658483	1.7344×10^{-06}	0.00024118	0.019569215	0.030009891	0.338856155	2.84342×10^{-05}
F22	4	1.12654×10^{-05}	0.006424212	0.002255124	1.7344×10^{-06}	0.228880422	0.416533807	0.006035006	0.452806505	5.30699×10^{-05}
F23	4	2.35342×10^{-06}	0.00089443	0.975387164	2.8786×10^{-06}	0.004114031	9.31566×10^{-06}	3.72426×10^{-05}	0.044918904	1.23808×10^{-05}

Table 9 Statistical results of each algorithm in CEC2014 test function

CEC2014	Metric	COA	SCA	SCSO	ROA	GA	STOA	BES	PDO	WOA	AOA
CEC14_1	Mean	**1242131.13**	15757361.28	12599836.7	63938517.44	109829424.4	8135157.515	91329102.67	141210009.9	17712960.4	100911007.4
	Std	**960811.6654**	8644093.715	10568971.99	52442866.71	155293271.1	5826166.069	44405584.22	73741178.51	12297576.86	236744990.3
CEC14_2	Mean	**13606.05102**	1216461611	231823767.6	3364342803	5481193785	584693110.1	3376754268	6125833660	88214411.61	7366749750
	Std	**9611.846976**	544129930.4	530095190.2	2675666001	2345074234	624847436.8	2752406517	2056593008	189520942.8	2149881130
CEC14_3	Mean	10208.15719	11144.29607	**8165.797739**	14476.56078	2016928.225	15509.50511	67693.26062	18975.4685	60053.81167	20783.24106
	Std	4604.416005	6928.55754	4401.84492	**3371.993269**	7945520.397	10130.37128	52228.18152	12521.77463	36474.98453	7796.508657
CEC14_4	Mean	**421.1578012**	488.1551792	454.3108605	874.659568	1134.557798	455.2445612	1077.963824	1591.12175	471.6177277	2103.049332
	Std	**16.79201846**	39.74541433	35.56545543	579.9905312	749.8137196	32.42431426	757.9648303	757.0555054	43.57957188	1197.022031
CEC14_5	Mean	**520.1317061**	520.518131	520.1570677	520.383777	520.4157647	520.4295823	520.4332578	520.3908711	520.2977484	520.1663804
	Std	**0.080238512**	0.104097374	0.109751443	0.129395738	0.196043463	0.11564057	0.10063077	0.169305066	0.170645251	0.102654536
CEC14_6	Mean	**605.4358868**	608.1413634	606.1263299	609.3394719	610.1036722	608.2914161	609.7496914	609.8418493	609.1104654	609.9716411
	Std	2.410633548	1.220305538	1.798683467	1.598876486	1.490940909	1.745188965	1.660891224	1.22850643	1.395314937	**0.970817258**
CEC14_7	Mean	**700.3129386**	715.0079601	705.1602691	737.4291326	781.0941015	705.4453797	772.0984686	822.8142181	702.0385575	845.0230397
	Std	**0.14891203**	8.105889371	12.25952581	28.3723307	34.81038143	3.93563149	42.96331972	47.48996643	1.195080593	67.05345412
CEC14_8	Mean	**812.9018921**	849.2586892	836.8525886	861.5849038	879.1695349	830.1583362	862.8649284	877.3396228	846.2936782	854.9588578
	Std	**7.161989081**	9.219544923	15.20290786	18.63644479	18.39909893	10.99247964	16.49703874	12.28195298	18.69814332	13.52477922
CEC14_9	Mean	942.1081812	952.6363621	939.1550525	963.1153151	972.5172097	**933.1358992**	967.3095436	967.6341969	955.5339111	947.2012598
	Std	10.79327217	**7.332715465**	13.61776926	16.33295355	17.73060957	13.50095137	15.37877226	12.66989793	21.74712278	10.52754419
CEC14_10	Mean	**1605.942713**	2215.931264	1758.724935	2189.544095	2018.662856	1824.280282	2392.555972	2446.963789	1768.773392	1741.440128
	Std	288.390891	**148.3648435**	306.1502764	362.883502	289.8153611	286.472467	270.8256748	272.5753428	336.5291022	283.578181
CEC14_11	Mean	**2109.059898**	2664.267261	2142.306361	2577.423996	3032.083769	2336.617558	2668.79402	2512.37107	2378.359473	2195.296981
	Std	279.533642	**215.8966286**	322.4270627	373.0016893	316.7175144	309.7996462	331.8385627	298.6868875	367.534523	278.3906314
CEC14_12	Mean	1200.569001	1201.553275	**1200.526615**	1201.122874	1202.030752	1201.40733	1201.326015	1201.379369	1201.076207	1200.662869
	Std	**0.238952878**	0.304118728	0.346879808	0.330616217	0.781807479	0.396771181	0.370591041	0.451079201	0.433122662	0.287028786
CEC14_13	Mean	**1300.280762**	1300.787209	1300.483413	1301.991024	1302.56339	1300.558525	1302.582943	1303.304503	1300.581408	1303.521236
	Std	**0.120920198**	0.176792812	0.175398921	1.17665455	0.984057583	0.187364766	1.097262506	1.113404773	0.198485679	0.90252271

Table 9 (continued)

CEC2014	Metric	COA	SCA	SCSO	ROA	GA	STOA	BES	PDO	WOA	AOA
CEC14_14	Mean	**1400.319189**	1401.503501	1400.728748	1412.239868	1415.857389	1400.889456	1417.347546	1420.151738	1400.410425	1428.616872
	Std	**0.099539426**	1.237008361	1.148617136	9.421216663	8.52672895	1.07723926	8.952842424	10.51418606	0.319187677	9.141046866
CEC14_15	Mean	**1502.551609**	1551.644774	1513.548232	2544.982525	12714.74252	1523.513567	3328.285633	4998.974278	1512.734656	5299.580914
	Std	**0.660272145**	203.8344907	48.01763667	2067.432987	47007.2827	99.29173341	4099.730451	5183.875775	9.232933555	6070.037252
CEC14_16	Mean	**1602.75276**	1603.587881	1603.3441	1603.426173	1604.049276	1603.566786	1603.563635	1603.845214	1603.627802	1603.687179
	Std	0.275316724	0.272182532	0.351584965	**0.239704737**	0.340331804	0.373638043	0.302142503	0.256919857	0.402739763	0.256966903
CEC14_17	Mean	**29495.88872**	106011.0701	103725.5557	456418.6522	9898023.624	198340.6813	1007000.145	552498.9631	573482.5887	449966.0315
	Std	**30126.36093**	208378.4788	188776.7549	338180.1561	13796403.34	224649.1362	2416306.179	260476.3576	932550.2914	388705.307
CEC14_18	Mean	**7831.802732**	62754.8821	13137.6413	122171.3319	48050844.35	19858.95211	1941147.252	122594.3026	17044.5385	14570.80924
	Std	**4563.455614**	94962.64826	11693.17026	608629.1185	52524778.26	15154.89142	4878704.586	508456.5886	16112.17739	11930.90853
CEC14_19	Mean	**1902.619793**	1906.488784	1904.432496	1916.805252	1946.071628	1904.694873	1914.279365	1914.693186	1907.182574	1943.860323
	Std	**0.872156649**	1.245229998	1.71074009	20.26807737	41.01787825	1.179687101	11.72181333	9.783484183	1.916509365	38.20611312
CEC14_20	Mean	**4252.282964**	11850.22669	9482.636426	47675.13771	28427084.05	14020.32397	144454.579	262032.039	15739.88107	12289.77495
	Std	**2249.269052**	10063.93655	4354.204705	101496.5808	67739244.61	11814.49189	330305.6183	832091.1657	17421.03104	8074.936046
CEC14_21	Mean	**7669.384591**	18893.38352	15670.25278	548281.5689	6445667.96	18218.27064	370564.6965	676884.2945	981720.2138	1521593.933
	Std	**5470.470587**	13721.67409	35082.2206	1278934.412	8804900.753	24859.85623	820518.9997	903995.8254	3030605.358	2601446.133
CEC14_22	Mean	**2249.26847**	2296.507351	2320.739063	2354.3033	2633.450389	2304.0583	2402.261786	2464.620364	2351.764125	2428.515799
	Std	**29.83164283**	46.98111233	70.9196507	101.5193152	177.1749613	84.59273207	122.708075	101.1180944	101.1087001	133.6883973
CEC14_23	Mean	**2500**	2650.402116	**2500**	**2500**	2752.777708	2643.762744	2591.000737	**2500**	2626.462344	2504.683399
	Std	**0**	11.88598654	**0**	**0**	123.8354761	9.992360419	108.6023125	**0**	43.42445221	25.65203329
CEC14_24	Mean	2596.378881	2562.45581	2595.871388	2595.751428	2600.424874	**2549.183139**	2588.688687	2597.308262	2589.973159	2593.233841
	Std	13.99125414	11.2964448	16.06123802	13.28136991	19.76354226	23.90516679	18.2633279	**8.952060518**	29.88542565	14.02622276
CEC14_25	Mean	2700	2702.027713	2697.975469	2699.810405	2707.007496	2701.139895	2697.359433	2698.942963	**2693.798545**	2699.346245
	Std	**0**	3.896822449	11.08881158	1.03845218	10.54474438	4.59749724	10.30910029	5.894294254	13.84432183	3.580763626

Table 9 (continued)

CEC2014	Metric	COA	SCA	SCSO	ROA	GA	STOA	BES	PDO	WOA	AOA
CEC14_26	Mean	**2700.23236**	2700.868919	2703.646832	2708.316173	2716.23242	2700.430556	2701.892234	2709.566552	2703.807194	2714.590419
	Std	**0.081426403**	0.212904106	18.19846474	24.94365063	35.4707312	0.121097045	1.141455328	24.60000443	18.52264954	29.63834882
CEC14_27	Mean	2893.834046	3069.697833	**2881.013055**	2894.610368	3205.492915	3123.419462	3160.14058	3216.844549	3111.207384	2901.764979
	Std	33.77232155	117.9773067	57.93637497	29.52023248	137.6300591	128.5000769	139.0253328	118.2142202	147.0379941	**24.57912931**
CEC14_28	Mean	**3000**	3299.454889	**3000**	**3000**	4008.25318	3187.064818	3433.308496	3267.336628	3436.715209	3038.577983
	Std	**0**	73.09121344	**0**	**0**	318.2811283	12.42791747	272.2772491	60.68911427	203.3544543	192.7269247
CEC14_29	Mean	**3227.435134**	29058.90525	191879.3602	452708.0634	12208857.84	7078.601086	1254136.256	455031.5301	608048.6096	2912440.704
	Std	**300.7255991**	29917.13269	742246.1742	1068970.365	13919954.81	5930.391207	2263651.106	580648.8298	1295171.754	11559290.67
CEC14_30	Mean	**4137.128012**	5525.534714	4956.676961	17916.81508	72701.57347	4250.132877	25186.98868	23229.27046	8154.493991	116788.3915
	Std	**578.9626601**	1662.764563	1182.443905	29443.13289	133114.1438	602.3945904	35159.38769	19437.0261	12912.4995	324796.0985

The bold part represents the best solution obtained from 10 comparative algorithms

of fitness value. The bold part is the best result obtained by 10 algorithms in the same function.

According to the results in Table 9, COA has achieved good results in most functions. In Unimodal Functions, COA achieved the best mean and std in CEC14_1 and CEC14_2. Other algorithms obtain larger fitness values and are not as good as COA. In CEC14_3, COA did not obtain better mean and std. The SCSO algorithm achieves the best mean. ROA achieves the best std. In Simple Multimodal Functions, COA has achieved excellent results, and most functions have achieved the best fitness value. In CEC14_4, CEC14_5, CEC14_7, CEC14_8, CEC14_12, CEC14_13, CEC14_14 and CEC14_15, COA has obtained the best mean and std. Compared with other comparison algorithms, COA has greater advantages. In CEC14_6, CEC14_10, CEC14_11 and CEC14_16, COA did not obtain the best std, but still obtained the best mean, indicating that the stability of COA in these functions is insufficient. At CEC14_6, the stability of AOA is better, resulting in smaller std, but larger fitness values are obtained. The fitness values obtained by ROA at CEC14_4, CEC14_7, and CEC14_8 are insufficient. In CEC14_9, the mean of COA is not as good as STOA and std is not as good as SCA. However, compared with other comparison algorithms, COA still has great advantages. In Hybrid Function 1, COA obtained the best fitness value in CEC14_17 to CEC14_22. Compared with SCA and AOA, COA has obvious advantages and good optimization effect in Hybrid Function 1. In Composition Functions, the fitness values obtained by COA in CEC14_23, CEC14_26, CEC14_28, CEC14_29 and CEC14_30 are better than other comparison algorithms, and COA is more stable. In CEC14_24, the mean of COA is not as good as other comparison algorithms, and the stability of COA is poor. In CEC14_25, BES algorithm and WOA have achieved better results, but COA is more stable. In CEC14_27, SCSO algorithm achieved better mean and ROA achieved more stable value. However, the fitness value obtained by COA is still better than other comparison algorithms. To sum up, COA has achieved excellent results in CEC2014 test function, which proves that COA has strong optimization ability.

3.2.2 Convergence curve analysis of CEC2014 test function

Figure 15 is the convergence curve of COA and 9 comparison algorithms. It can be seen from CEC14_1, CEC14_2 and CEC14_3 that COA has a strong convergence ability and can continuously converge and obtain the best fitness value. However, SCA, ROA, AOA and BES algorithms cannot converge, resulting in poor results. In CEC14_4 and CEC14_5, COA can continue to converge and get a outstand result. The SCSO algorithm has good convergence performance in CEC14_4, but its convergence ability is insufficient in CEC14_5. In CEC14_6, CEC14_10, CEC14_11 and CEC14_12, COA has obvious advantages, can converge quickly, and its convergence ability is obviously superior to other comparison algorithms. GA, SCA, ROA, and PDO algorithms are trapped in local optima and unable to obtain better fitness values. In CEC14_7, CEC14_8 and CEC14_9, GA, AOA, SCA and ROA are all trapped in local optimization, which leads to the algorithm's inability to converge better, while COA can continue to converge and find better fitness values. In CEC14_13, CEC14_14 and CEC14_15, the advantages of COA are not particularly obvious. However, compared with GA, PDO and BES algorithms, it can still get better fitness value. In CEC14_16, CEC14_17 and CEC14_18, COA can continuously converge in the iterative process, and the final fitness value is the first rate. Other comparative algorithms are difficult to converge in the later stages of iteration and cannot obtain better fitness values. In CEC14_19, most algorithms can find a better fitness value, and COA can also

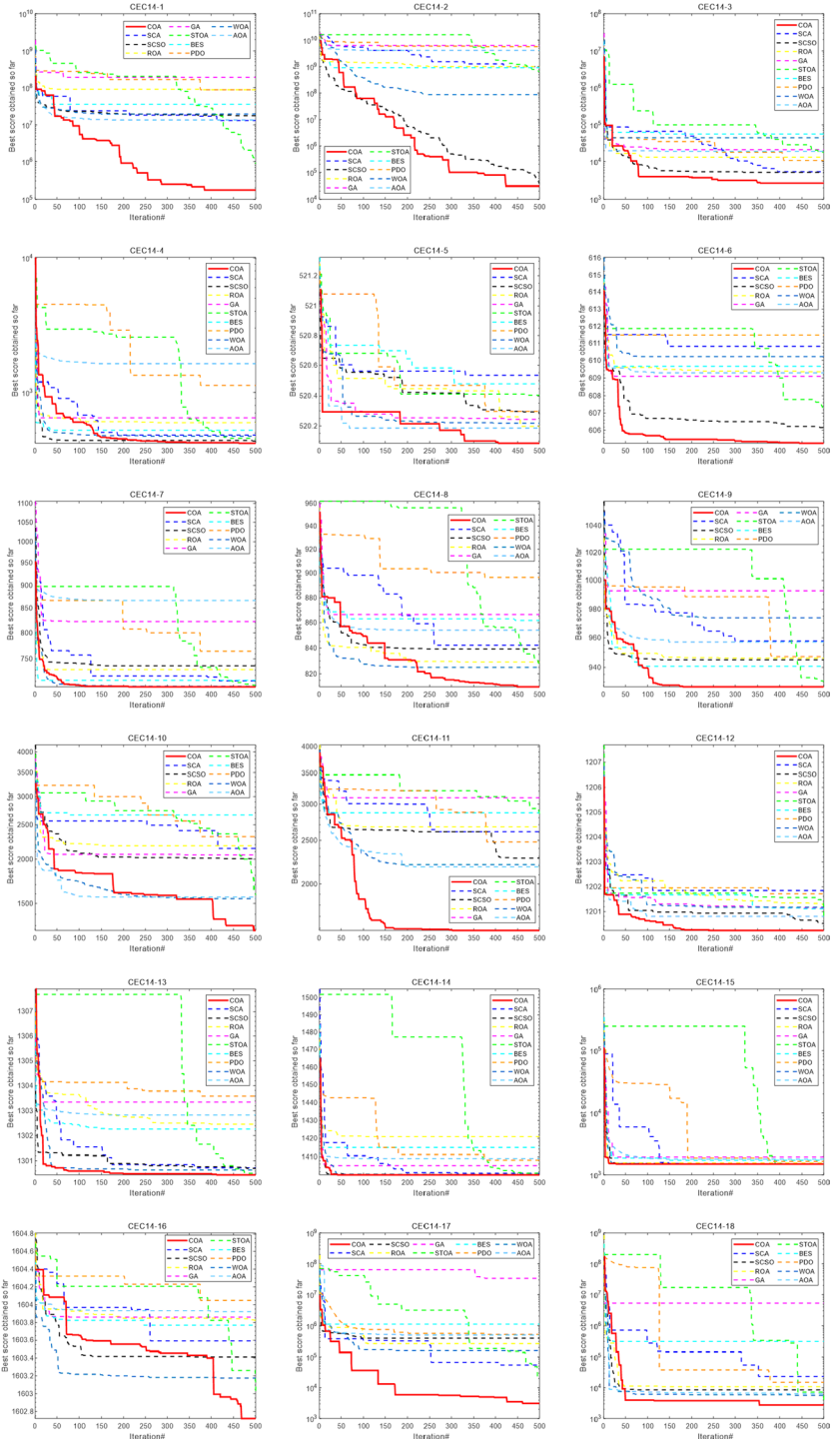

Fig. 15 Convergence curve of each algorithm in CEC2014

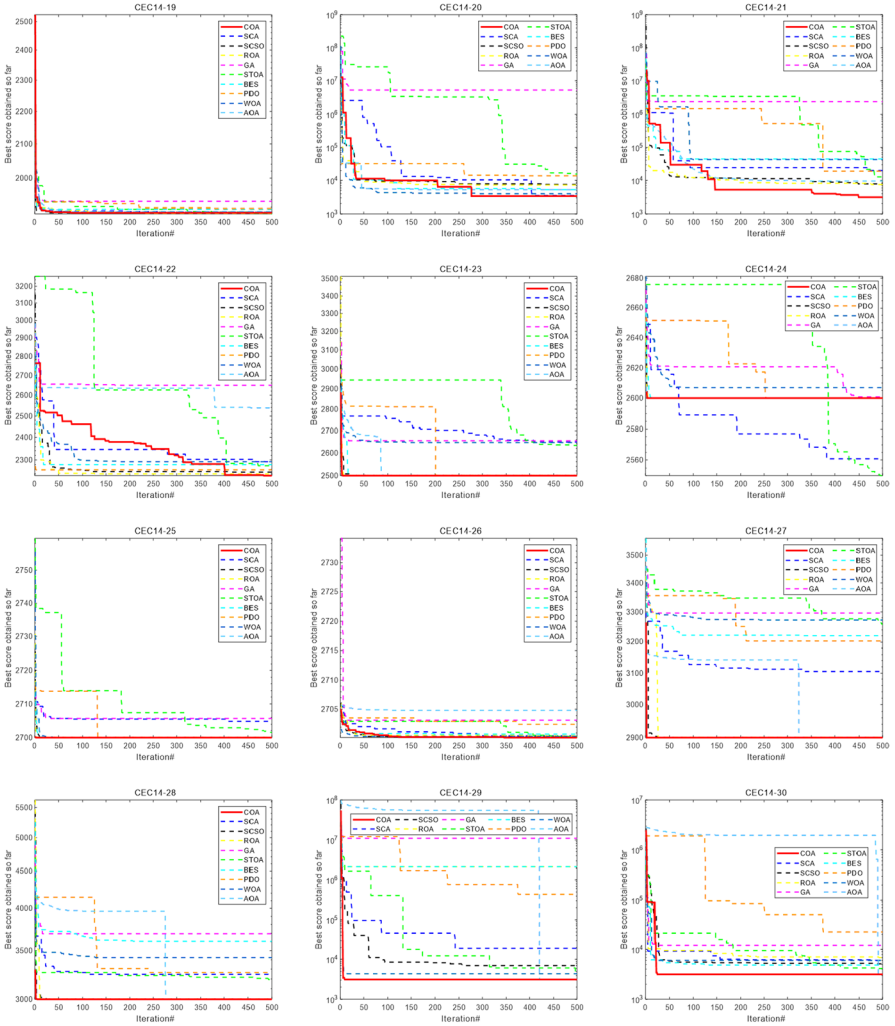

Fig. 15 (continued)

get a better fitness value. In CEC14_20, CEC14_21, and CEC14_22, COA can continue to converge and obtain better fitness values. In CEC14_24, SCA and STOA's results are better. COA and other algorithms can only reach about 2600. In CEC14_25, CEC14_26, CEC14_27, CEC14_28, CEC14_29 and CEC14_30, the fitness values obtained by STOA, SCA, PDO and BES algorithms are poor, but COA can find better fitness values. To sum up, COA has better optimization effect and convergence ability in CEC2014.

3.2.3 Analysis of box plot results

The box chart is a statistical chart (Wu et al. 2022). Figure 16 is a box diagram obtained by running 10 algorithms for 30 times. It is clear from the figure that COA has achieved

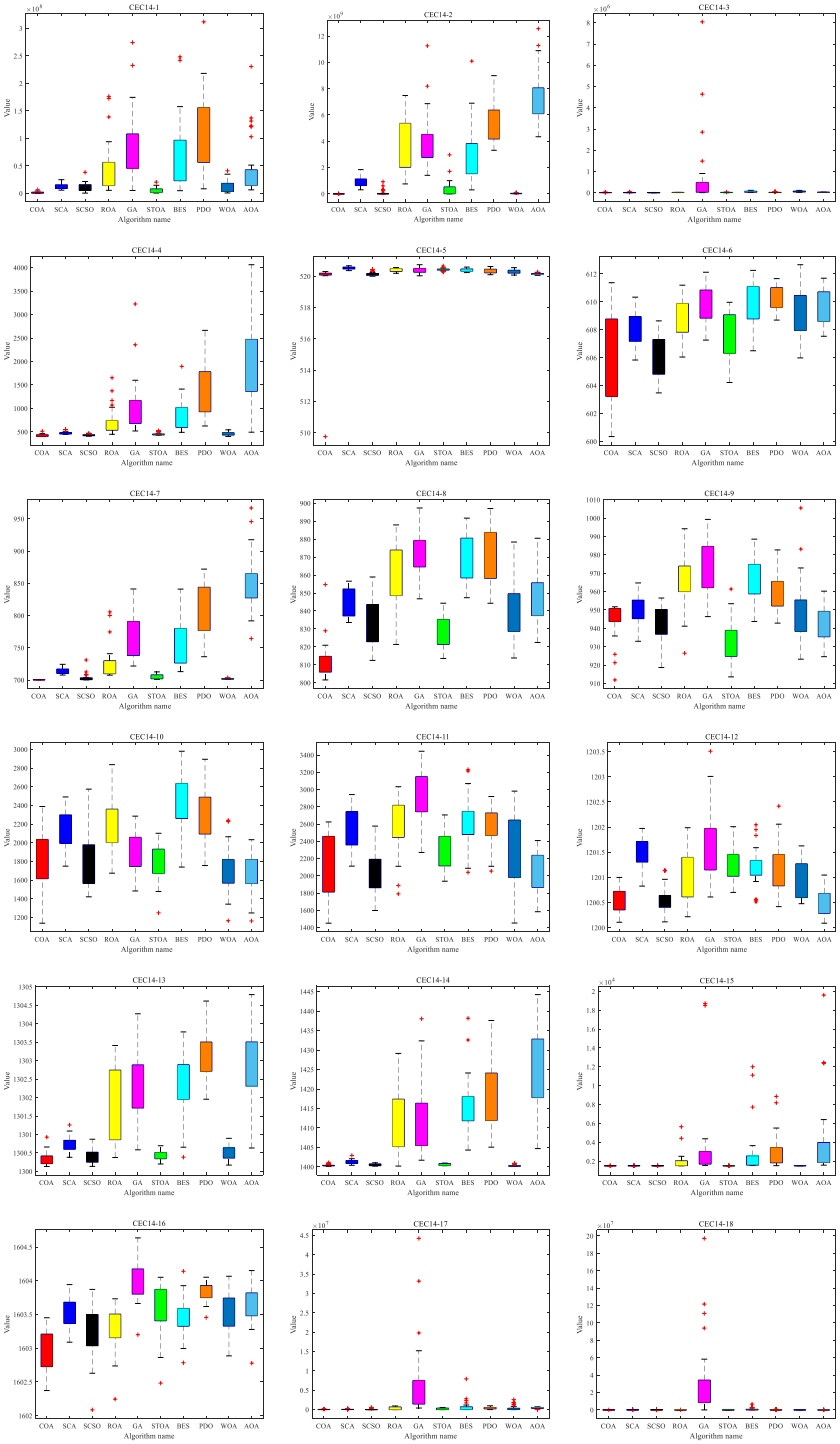

Fig. 16 Box diagram of each algorithm in CEC2014

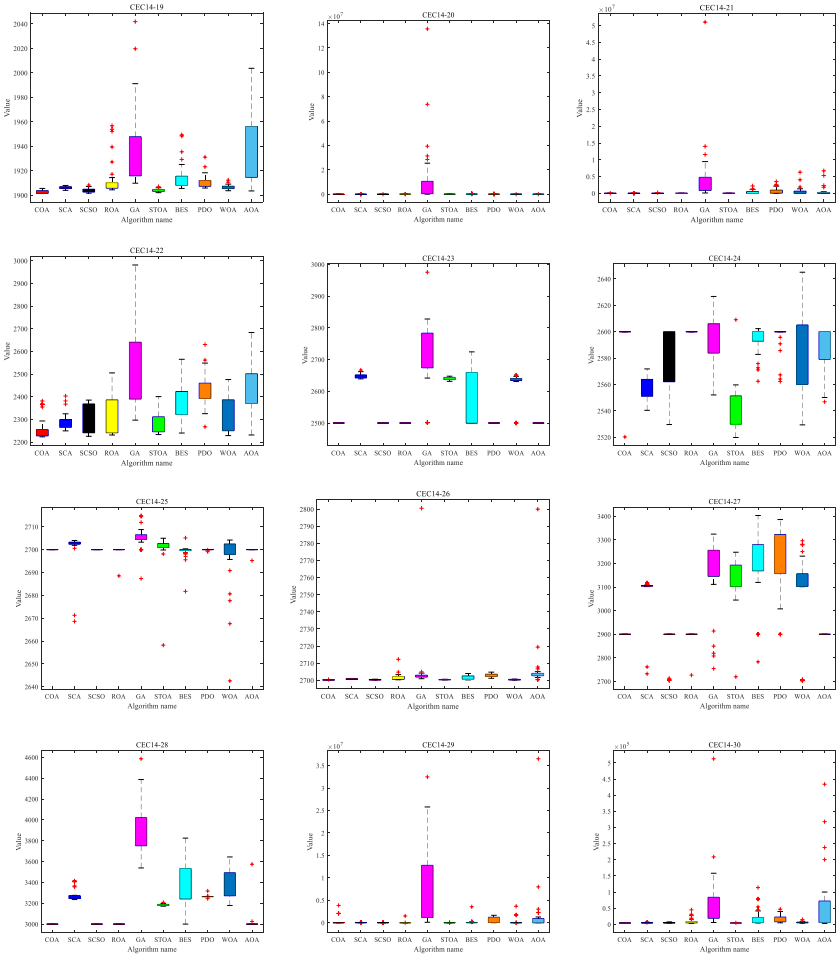

Fig. 16 (continued)

good results in many functions. In most functions, the box graph obtained by COA is narrow and bottom, which shows that the fitness value obtained by COA is not only very good but also very stable. However, in some functions, the optimization effect of COA can still be improved. In CEC14_6, CEC14_10 and CEC14_11, the box diagram of COA is significantly wider than other algorithms, indicating that COA is not stable in this function. However, the box graph obtained by COA is lower than that of other algorithms, indicating that COA can obtain better fitness values. In CEC14_9, the overall optimization effect of COA is not as good as SCA, SCSO algorithm and STOA. In CEC14_12, the box graph obtained by COA is very similar to the SCSO algorithm, which shows that in this function, the effect of the two algorithms is relatively small. Among other functions, the box graph obtained by COA is narrow and bottom, which is better than other comparison algorithms. For other comparative algorithms, the PDO algorithm obtained poor box plots at CEC14_1, CEC14_2, CEC14_4, CEC14_7, CEC14_8, CEC14_10, CEC14_13 and CEC14_17. The box plots obtained by AOA at CEC14_2, CEC14_4, CEC14_7, CEC14_8,

CEC14_13, CEC14_14 and CEC14_19 are long and high, indicating that the optimal fitness values obtained by the algorithm are large and unstable.The box plot obtained by the SCSO algorithm is better, but not as good as COA. The stability and effectiveness of GA in CEC2014 are insufficient. WOA has good performance in many functions, but has some shortcomings in CEC14_24, CEC14_27, and CEC14_28. Through these analyses, it has been proven that COA has a good overall optimization effect on CEC2014.

3.2.4 Analysis of Wilcoxon rank sum test results of CEC2014 test function

Table 10 shows the statistical results of COA and nine comparison algorithms running for 30 times. It can be seen from the table data that most of the results are below 5 %, which is at a significant level. However, some results are higher than 5 %, which shows that in these functions, the optimization effect gap between COA and the comparison algorithm is not obvious. In CEC14_23, the result of COA and SCSO algorithm, ROA, PDO algorithm and AOA is 1. In CEC14_25, the value of COA, SCSO and ROA is 1. In CEC14_28, the result of COA and SCSO algorithm, ROA and AOA is 1. These results are because the fitness values obtained by COA and these algorithms are the same, and the final statistical result is 1. In other functions, there are some results that are more than 5 %, but very few and not much different from 5 %. Most of the statistical results are at a significant level, indicating that the optimization effect of COA is significantly different from that of the comparison algorithm.

The above experimental results and analysis fully prove that COA algorithm has good optimization ability.

3.3 Parameter sensitivity analysis of food factor C_3

In order to better prove the influence of food factor on COA, we selected different values of food factor C_3 for comparative experiments. Table 11 shows the statistical results of different food factors independently run for 30 times in CEC2014. The bold part represents the best results of different food factors in CEC2014. It can be seen from the table that the results obtained by $C_3=3$ are significantly better than the other values. Only in some functions, the effect is not good. In CEC14_3 and CEC14_9, food factor $C_3=3$ did not get the best mean and std. In CEC14_4 and CEC14_10, although the result of $C_3=3$ does not get the best result in mean, it get the best std. In CEC14_12 and CEC14_27, the std obtained by food factor $C_3=3$ is not the best, but the mean obtained is the best. It shows that $C_3=3$ can get a better fitness value, but the obtained fitness value is not stable. In CEC14_24, the result of food factor C$C_3=3$ did not get the best std and mean. Although the result of food factor $C_3=3$ in CEC14_25 is not as good as the result of $C_3=4$, it is the same as the value of other numerical food factors. Among other functions, the result of food factor $C_3=3$ is better than other numerical food factors. Through the above analysis, it is proved that the food factor $C_3=3$ has better optimization effect.

3.4 Constrained engineering design problems

The upper part tests the optimization ability of COA through 23 standard test functions and CEC2014 test functions, which proves that COA has excellent optimization effect. In engineering problems, the range of all variables is known and controllable. Each problem can be constructed as a mathematical model. However, these problems are usually difficult to solve,

Table 10 CEC2014 experimental results of the Wilcoxon rank sum test on test functions

CEC2014	COA vs SCA	COA vs SCSO	COA vs ROA	COA vs GA	COA vs STOA	COA vs BES	COA vs PDO	COA vs WOA
CEC14_1	2.12664×10^{-06}	0.000125057	1.7344×10^{-06}	1.7344×10^{-06}	0.0003065	1.7344×10^{-06}	1.7344×10^{-06}	2.35342×10^{-06}
CEC14_2	1.7344×10^{-06}	0.001113801	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}
CEC14_3	0.106394173	0.01319417	0.047161747	3.88218×10^{-06}	0.025637124	6.98378×10^{-06}	0.125438239	2.8786×10^{-06}
CEC14_4	1.7344×10^{-06}	0.003378854	1.7344×10^{-06}	1.7344×10^{-06}	0.000283079	1.7344×10^{-06}	1.7344×10^{-06}	0.004991554
CEC14_5	1.7344×10^{-06}	0.893644387	2.35342×10^{-06}	2.35342×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.92092×10^{-06}	6.31976×10^{-05}
CEC14_6	0.000125057	0.002957462	0.002255124	0.000205153	8.18775×10^{-05}	0.000105695	5.21649×10^{-06}	0.001708773
CEC14_7	1.7344×10^{-06}	3.51524×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}
CEC14_8	3.51524×10^{-06}	3.11232×10^{-05}	1.92092×10^{-06}	1.7344×10^{-06}	0.00048969	1.7344×10^{-06}	1.7344×10^{-06}	1.97295×10^{-05}
CEC14_9	0.530440091	0.036826128	0.000615641	2.12664×10^{-06}	0.001113801	6.98378×10^{-06}	0.000125057	0.017518394
CEC14_10	0.000528725	0.349345562	5.79245×10^{-05}	0.085895826	0.416533807	6.33914×10^{-06}	1.7344×10^{-06}	0.477947439
CEC14_11	0.000205153	0.033268897	0.000261343	1.92092×10^{-06}	0.452806505	1.23808×10^{-05}	0.000160464	0.042766688
CEC14_12	2.12664×10^{-06}	0.005667173	0.000261343	1.92092×10^{-06}	4.7292×10^{-06}	6.33914×10^{-06}	1.49356×10^{-05}	0.001286631
CEC14_13	1.92092×10^{-06}	0.000419551	1.7344×10^{-06}	1.7344×10^{-06}	2.35342×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.63945×10^{-05}
CEC14_14	1.7344×10^{-06}	0.298943752	1.92092×10^{-06}	1.7344×10^{-06}	0.002957462	1.7344×10^{-06}	1.7344×10^{-06}	0.016565527
CEC14_15	1.7344×10^{-06}	0.004389618	1.7344×10^{-06}	1.7344×10^{-06}	5.75165×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.92092×10^{-06}
CEC14_16	1.7344×10^{-06}	0.000615641	0.000715703	1.7344×10^{-06}	1.97295×10^{-05}	1.7344×10^{-06}	1.7344×10^{-06}	1.36011×10^{-05}
CEC14_17	0.557742686	0.002957462	5.30699×10^{-05}	1.7344×10^{-06}	0.813016851	2.37045×10^{-05}	3.88218×10^{-06}	0.003854236
CEC14_18	2.12664×10^{-06}	0.102010695	0.054462504	1.7344×10^{-06}	0.001382036	3.51524×10^{-06}	0.001483928	0.011748106
CEC14_19	1.7344×10^{-06}	0.017518394	3.88218×10^{-06}	1.7344×10^{-06}	0.023038145	1.7344×10^{-06}	1.7344×10^{-06}	6.98378×10^{-06}
CEC14_20	0.020671114	0.006835856	1.36011×10^{-05}	1.7344×10^{-06}	0.009271025	0.000174228	4.44934×10^{-05}	8.18775×10^{-05}
CEC14_21	0.006835856	0.416533807	0.002584559	1.7344×10^{-06}	0.008216736	6.98378×10^{-06}	1.7344×10^{-06}	2.12664×10^{-06}
CEC14_22	0.027029157	0.000830707	0.028485956	1.7344×10^{-06}	0.033268897	0.0003065	2.35342×10^{-06}	0.428430029
CEC14_23	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	0.000488281	1	3.78962×10^{-06}
CEC14_24	1.7344×10^{-06}	0.243200971	0.5	0.004681835	1.92092×10^{-06}	0.05339578	0.009765625	0.015658483
CEC14_25	3.72426×10^{-05}	1	1	2.35342×10^{-06}	3.88218×10^{-06}	0.064453125	0.5	0.400417985
CEC14_26	1.7344×10^{-06}	0.585711569	1.7344×10^{-06}	1.7344×10^{-06}	0.038723026	1.7344×10^{-06}	1.7344×10^{-06}	0.023038145

Table 10 (continued)

CEC2014	COA vs SCA	COA vs SCSO	COA vs ROA	COA vs GA	COA vs STOA	COA vs BES	COA vs PDO	COA vs WOA
CEC14_27	1.23808×10^{-05}	0.375	1	3.88218×10^{-06}	3.88218×10^{-06}	3.16517×10^{-06}	1.92092×10^{-06}	3.18168×10^{-06}
CEC14_28	1.7344×10^{-06}	1	1	1.7344×10^{-06}	1.7344×10^{-06}	3.78962×10^{-06}	1.73331×10^{-06}	1.73331×10^{-06}
CEC14_29	0.000358884	0.241819658	0.014888682	1.7344×10^{-06}	0.001197338	0.000345249	0.000358884	0.005667173
CEC14_30	0.000114992	0.314666327	3.88218×10^{-06}	1.7344×10^{-06}	1.7344×10^{-06}	1.92092×10^{-06}	2.12664×10^{-06}	0.000125057

Table 11 Statistical results of different food factors in CEC2014

CEC2014	metric	$C_3=2$	$C_3=3$	$C_3=4$	$C_3=5$	$C_3=6$	$C_3=7$	$C_3=8$	$C_3=9$
CEC14_1	Mean	2623786.26	**977833.54**	1856654.82	1931641.78	2655741.25	2599028.62	2153939.36	1906899.22
	Std	4027445.20	**898131.19**	2574802.58	3769464.93	4682140.80	4473875.14	2692549.68	2487001.80
CEC14_2	Mean	13658068.08	**10018.31**	23573.63	26677.64	21692.68	24841.75	22481.65	22452.10
	Std	74687104.20	**6954.96**	38108.33	58296.76	18627.43	21793.36	49052.57	30678.85
CEC14_3	Mean	13862.07	11881.27	13447.77	14834.10	11639.79	12247.51	12035.97	**11601.86**
	Std	8174.43	5069.21	6094.12	8105.67	7425.11	7766.77	**4714.62**	6733.68
CEC14_4	Mean	432.88	429.19	435.34	432.63	427.13	424.88	**423.97**	428.13
	Std	25.32	**17.44**	18.78	27.03	18.91	17.66	21.98	20.62
CEC14_5	Mean	520.01	520.12	**519.89**	520.13	520.13	520.13	520.14	520.15
	Std	1.07	**0.07**	1.34	0.07	0.07	0.09	0.08	0.08
CEC14_6	Mean	606.61	**605.34**	606.18	606.08	606.01	606.60	605.89	606.00
	Std	2.33	**1.98**	2.43	2.52	2.32	2.15	2.59	2.32
CEC14_7	Mean	700.54	700.32	700.35	700.33	**700.30**	700.34	700.34	700.38
	Std	0.21	**0.13**	0.28	0.16	0.16	0.17	0.16	0.31
CEC14_8	Mean	820.12	**813.97**	817.19	814.61	818.83	816.85	815.65	816.27
	Std	13.61	**4.80**	11.19	11.78	9.69	12.66	9.31	12.19
CEC14_9	Mean	946.61	941.90	944.73	**938.32**	944.17	945.03	943.27	940.24
	Std	**9.53**	13.17	10.67	14.23	11.22	11.56	13.61	13.89
CEC14_10	Mean	1921.64	1748.58	1784.99	1779.62	1805.82	**1684.20**	1729.85	1751.87
	Std	396.22	**279.32**	358.88	403.29	368.33	390.56	349.80	346.95
CEC14_11	Mean	2156.82	**2070.25**	2163.59	2145.05	2136.95	2141.92	2146.80	2161.67
	Std	401.94	**289.39**	366.63	410.26	377.36	422.71	350.25	414.78
CEC14_12	Mean	1200.71	1200.59	1200.63	1200.68	1200.59	**1200.52**	1200.55	1200.53
	Std	0.38	0.28	0.28	0.30	0.28	0.28	0.34	**0.23**
CEC14_13	Mean	1300.33	**1300.29**	1300.33	1300.33	1300.35	1300.36	1300.34	1300.35
	Std	0.17	**0.11**	0.19	0.18	0.14	0.20	0.14	0.14

Table 11 (continued)

CEC2014	metric	$C_3=2$	$C_3=3$	$C_3=4$	$C_3=5$	$C_3=6$	$C_3=7$	$C_3=8$	$C_3=9$
CEC14_14	Mean	1400.44	**1400.36**	1400.45	1400.42	1400.44	1400.48	1400.42	1400.43
	Std	0.28	**0.18**	0.29	0.27	0.29	0.27	0.22	0.22
CEC14_15	Mean	1504.59	1503.09	1503.41	1503.45	1503.14	**1502.85**	1502.86	1502.94
	Std	2.75	**0.99**	1.77	1.39	2.08	1.21	2.00	1.24
CEC14_16	Mean	1603.11	1602.84	**1602.78**	1603.04	1602.94	1602.98	1602.92	1602.87
	Std	0.35	**0.33**	0.49	0.44	0.40	0.40	0.43	0.54
CEC14_17	Mean	51781.74	**18240.06**	70715.55	54177.03	63126.58	52986.72	73720.05	59985.30
	Std	86461.47	**18878.46**	115806.91	69029.61	72050.78	104716.54	115515.84	116670.48
CEC14_18	Mean	10031.67	**7718.38**	9603.21	10512.54	11485.91	10382.59	10876.58	11997.23
	Std	6822.73	**5016.23**	7213.84	7121.08	7375.07	8175.20	6445.33	8924.99
CEC14_19	Mean	1903.37	**1902.80**	1903.14	1903.35	1903.44	1903.35	1903.45	1903.33
	Std	1.55	**1.00**	1.35	1.27	1.44	1.38	1.47	1.51
CEC14_20	Mean	7358.99	**4721.36**	6367.91	6873.78	7481.29	6078.28	6924.00	6157.85
	Std	4121.77	**2850.83**	4710.88	8549.15	5854.74	5116.89	5218.16	4358.78
CEC14_21	Mean	10529.78	**8211.71**	8931.24	10107.06	10709.70	10551.32	12049.04	10645.27
	Std	8581.14	**5347.26**	7738.79	7758.18	7865.10	8431.27	8073.18	8517.17
CEC14_22	Mean	2281.60	**2242.16**	2263.20	2261.68	2265.13	2260.14	2266.54	2255.95
	Std	60.58	**28.29**	59.88	53.81	48.33	50.77	51.27	50.40
CEC14_23	Mean	**2500.00**	**2500.00**	**2500.00**	**2500.00**	**2500.00**	**2500.00**	**2500.00**	**2500.00**
	Std	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
CEC14_24	Mean	2600.00	2597.78	**2595.59**	2598.37	2600.00	2597.90	2598.14	2597.13
	Std	**0.00**	12.18	17.23	8.93	**0.00**	11.50	10.21	15.70
CEC14_25	Mean	**2700.00**	**2700.00**	**2697.84**	**2700.00**	**2700.00**	**2700.00**	**2700.00**	**2700.00**
	Std	**0.00**	**0.00**	11.84	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**

Table 11 (continued)

CEC2014	metric	$C_3=2$	$C_3=3$	$C_3=4$	$C_3=5$	$C_3=6$	$C_3=7$	$C_3=8$	$C_3=9$
CEC14_26	Mean	2713.57	**2700.26**	2713.53	2716.87	2710.26	2706.92	2713.59	2713.58
	Std	34.48	**0.11**	34.50	37.81	30.43	25.30	34.47	34.48
CEC14_27	Mean	2900.00	**2893.54**	2900.00	**2887.17**	2894.20	2900.00	2900.00	2900.00
	Std	**0.00**	35.37	**0.00**	48.84	31.76	**0.00**	**0.00**	**0.00**
CEC14_28	Mean	**3000.00**	**3000.00**	**3000.00**	**3000.00**	**3000.00**	**3000.00**	**3000.00**	**3000.00**
	Std	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
CEC14_29	Mean	587444.37	**3352.30**	210582.89	178482.19	344603.28	301243.13	269169.04	356142.53
	Std	1881485.40	**627.62**	638512.46	678521.23	1080557.97	988438.90	891208.74	1133130.28
CEC14_30	Mean	4671.14	**4281.76**	4947.39	4768.21	4495.49	4614.47	4595.26	4577.08
	Std	966.45	**556.61**	1265.79	1037.72	830.00	1087.09	940.11	961.75

The bold part represents the optimal solution obtained for COA under different food factors

with large amount of calculation and many variables to be processed. In order to verify the effect of COA algorithm in solving engineering problems, this paper selects five engineering problems for test experiments to verify that COA solves practical problems, as shown below.

3.4.1 Tension/compression spring design problem

The purpose of tension/compression spring design problem is to obtain the minimum spring mass through three variables and four constraints. The schematic diagram of the spring is shown in Fig. 17. The variables in the figure are coil diameter d, average coil diameter D, and effective coil number N. The constraints include minimum deviation (g_1), shear stress (g_2), impact frequency (g_3) and outer diameter limit (g_4). Each variable is brought into the constraint condition to obtain the minimum spring mass $f(x)$.

The problem's mathematical model is as follows:

Set:

$$x = \left[x_1\ x_2\ x_3\right] = [d\ D\ N]. \tag{19}$$

Objective function:

$$f(x) = \left(x_3 + 2\right) \times x_2 \times x_1^2. \tag{20}$$

Subject to:

$$g_1(x) = 1 - \frac{x_3 \times x_2^3}{71785 \times x_1^4} \le 0, \tag{21}$$

$$g_2(x) = \frac{4 \times x_2^2 - x_1 \times x_2}{12566 \times x_1^4} + \frac{1}{5108 \times x_1^2} - 1 \le 0, \tag{22}$$

$$g_3(x) = 1 - \frac{140.45 \times x_1}{x_2^2 \times x_3} \le 0, \tag{23}$$

$$g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \le 0. \tag{24}$$

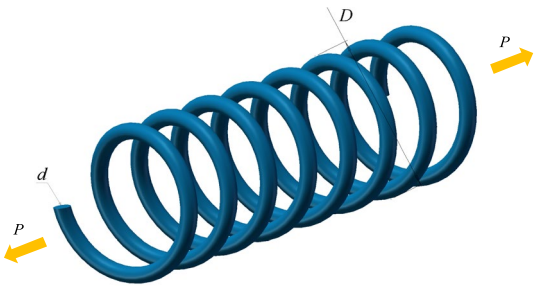

Fig. 17 Schematic diagram of the tension/compression spring design

Table 12 Experimental results of the tension/compression spring design

Algorithm	*d*	*D*	*N*	Best weight
COA	0.05	0.37442972	8.547782301	0.009873508
HHO Heidari et al. (2019)	0.051796	0.359305	11.13886	0.012665
EROA Wang et al. (2022)	0.053799	0.46951	5.811	0.010614
RCGO Ma et al. (2023)	0.052866	0.385549	9.787463	0.012701
CPSO He and Wang (2007)	0.051728	0.357644	11.244543	0.0126747
AGWO Ma et al. (2022)	0.051082	0.34226	12.19919	0.012681
SMA Precup et al. (2021)	0.058439	0.541892	5.261369	0.013438
EHO Hashim et al. (2019)	0.058	0.5278	5.582	0.0135

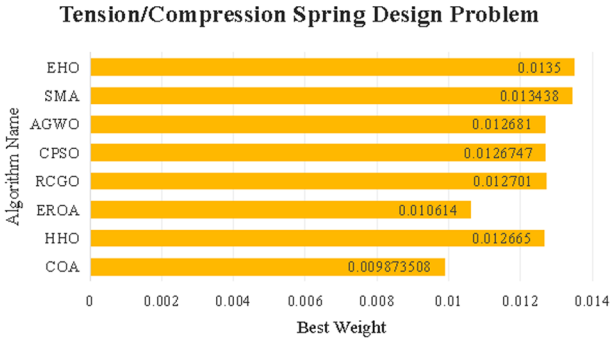

Fig. 18 Comparison of the best results of the tension/compression spring design

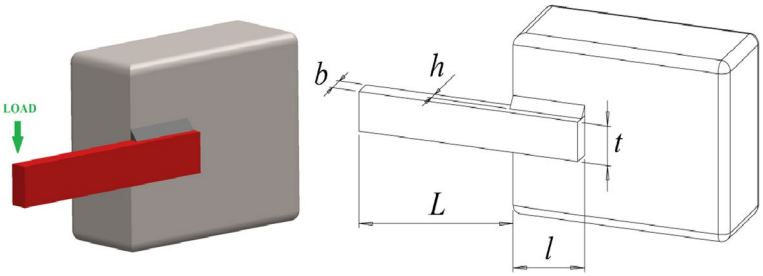

Fig. 19 Schematic diagram of the welded beam design

$$0.05 \le x_1 \le 2.0; 0.25 \le x_2 \le 1.3; \\ 2.0 \le x_3 \le 15.0. \tag{25}$$

Table 12 shows the results of COA and comparison algorithm in the tension/compression spring design problem. From the final data, it can be seen that COA has obtained the best Best Weight, and other algorithms are not as good as COA. Figure 18 is a comparison diagram of Best Weight among various algorithms. It can be seen from Fig. 18 that the length of COA is the shortest, indicating that the weight of the result obtained by COA is the smallest, which proves that COA has excellent effect in solving the tension/compression spring design problem.

3.4.2 Welded beam design problem

The purpose of the welded beam design problem is to obtain the minimum weight under four constraint conditions: weld width h, connecting beam length l, beam height t and connecting beam thickness b. The specific representation of each variable is shown in Fig. 19. The constraint conditions of the welded beam design problem are affected by shear stress (τ), bending stress (θ), beam bending load (P_c) and beam deflection (δ), as shown in the following formula.

The problem's mathematical model is as follows:

Set:

$$x = \left[x_1\ x_2\ x_3\ x_4\right] = [h\ l\ t\ b]. \tag{26}$$

Objective function:

$$f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4\left(14.0 + x_2\right). \tag{27}$$

Subject to:

$$g_1\left(\vec{x}\right) = \tau\left(\vec{x}\right) - \tau_{\max} \le 0, \tag{28}$$

$$g_2\left(\vec{x}\right) = \sigma\left(\vec{x}\right) - \sigma_{\max} \le 0, \tag{29}$$

$$g_3\left(\vec{x}\right) = \delta\left(\vec{x}\right) - \delta_{\max} \le 0, \tag{30}$$

$$g_4\left(\vec{x}\right) = x_1 - x_4 \le 0, \tag{31}$$

$$g_5\left(\vec{x}\right) = P - P_c\left(\vec{x}\right) \le 0, \tag{32}$$

$$g_6\left(\vec{x}\right) = 0.125 - x_1 \le 0, \tag{33}$$

$$g_7\left(\vec{x}\right) = 1.10471x_1^2 + 0.04811x_3x_4\left(14.0 + x_2\right) - 0.5 \le 0, \tag{34}$$

where:

$$\tau\left(\vec{x}\right) = \sqrt{\left(\tau'\right)^2 + 2\tau'\tau''\frac{x_2}{2R} + \left(\tau''\right)},\ \tau' = \frac{P}{\sqrt{2x_1x_2}},\ \tau'' = \frac{MR}{J}, \tag{35}$$

$$M = P\left(L + \frac{x_2}{2}\right),\ R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},\ \sigma\left(\vec{x}\right) = \frac{6PL}{x_4x_3^2}, \tag{36}$$

$$J = 2\left\{\sqrt{2x_1x_2}\left[\frac{x_x^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\},\ \delta\left(\vec{x}\right) = \frac{6PL^3}{Ex_4x_3^2}, \tag{37}$$

Table 13 Experimental results of the welded beam design

Algorithm	h	l	t	b	Best weight
COA	0.205557662	3.25636618	9.04034118	0.20575381	1.696251144
TSA Babalik et al. (2018)	0.244157	6.223066	8.29555	0.244405	2.38241101
SOA Dhiman and Kumar (2019)	0.205408	3.472316	9.035208	0.201141	1.723485
SSA Hashim and Hussien (2022)	0.2057	3.4714	9.0366	0.2057	1.72491
STOA Dhiman and Kaur (2019)	0.205415	3.472346	9.03522	0.20116	1.72359
ChOA Khishe and Mosavi (2020)	0.19963	3.7462	9.2175	0.21343	1.8445
ALO Mirjalili (2015)	0.189442	3.859115	9.036624	0.20573	1.7503

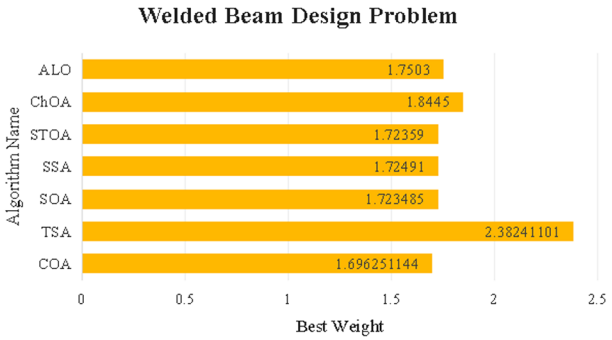

Fig. 20 Comparison of the best results of the welded beam design

$$P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{0}}}{L^2}, \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \tag{38}$$

$$P = 6000\,lb, L = 14\ \text{in}, \delta_{\max} = 0.25\ \text{in}, E = 30 \times 10^6\ psi, \tag{39}$$

$$\tau_{\max} = 13600\ psi, \text{and}\ \sigma_{\max} = 30000\ psi, \tag{40}$$

Boundaries:

$$0.1 \le x_i \le 2,\ i = 1, 4; 0.1 \le x_i \le 10,\ i = 2.3. \tag{41}$$

Table 13 shows the results obtained by COA and eight algorithms in the Welded Beam Design Problem. From the results, we can see that COA obtained the minimum Best Weight. The Best Weight obtained by TSA is significantly greater than that of COA. The value obtained by other algorithms is also inferior to COA. As can be seen from Fig. 20, the quality obtained by COA is significantly shorter. Therefore, COA has a good optimization effect in solving this engineering problem.

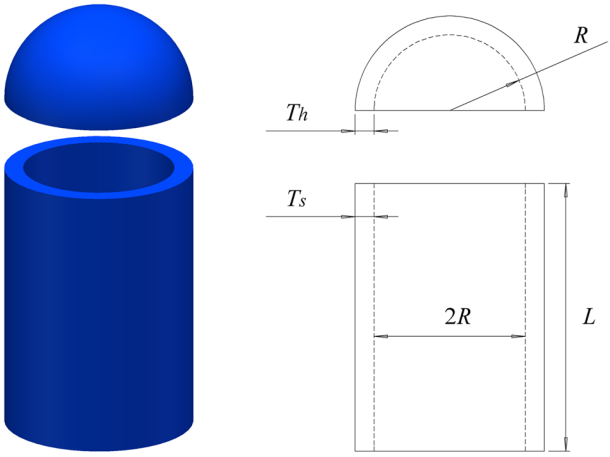

Fig. 21 Schematic diagram of the pressure vessel design

3.4.3 Pressure vessel design problem

The purpose of the pressure vessel design problem is to minimize the total cost of cylindrical pressure vessels. The variables of container include shell thickness T_s, head thickness T_h, internal radius R and container length L. The total cost is obtained when the four constraints are met. The schematic diagram of the pressure vessel is shown in Fig. 21.

The problem's mathematical model is as follows:

Set:

$$\vec{x} = \left[x_1\ x_2\ x_3\ x_4\right] = \left[T_s\ T_h\ R\ L\right]. \tag{42}$$

Objective function:

$$f\left(\vec{x}\right) = 0.6224x_1x_2x_3 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3. \tag{43}$$

Subject to:

$$g_1\left(\vec{x}\right) = -x_1 + 0.0193x_3 \le 0, \tag{44}$$

$$\vec{x} = \left[x_1\ x_2\ x_3\ x_4\right] = \left[T_s\ T_h\ R\ L\right], \tag{45}$$

$$g_3\left(\vec{x}\right) = -\pi x_3^2x_4 + \frac{4}{3}\pi x_{_3}^3 + 1296000 \le 0, \tag{46}$$

$$g_4\left(\vec{x}\right) = -x_4 - 240 \le 0. \tag{47}$$

Variable range:

$$0 \le x_1 \le 99,\ 0 \le x_2 \le 99,\ 10 \le x_3 \le 200,\ 10 \le x_4 \le 200. \tag{48}$$

Table 14 shows the statistical results obtained by COA in pressure vessel design problem. It can be seen from the results that the total cost of COA is the smallest, and it is obviously superior to other algorithms. It can be seen from Fig. 22 that the HS

Table 14 Experimental results of the pressure vessel design

Algorithm	T_s	T_h	R	L	Best cost
COA	0.74373884	0.370509119	40.32387722	199.9414282	5735.248836
AO Abualigah et al. (2021a, b)	1.054	0.182806	59.6219	38.805	5949.2258
HS Dhiman and Kumar (2017)	1.099523	0.906579	44.456397	179.65887	6550.023
DO Zhao et al. (2022)	0.7784	0.3848	40.331	199.8421	5885.7766
TSO Xie et al. (2021)	0.7952	0.4122	41.0635	190.628	6006.5711
TLBO Rao et al. (2012)	0.7963	0.4226	41.2309	188.0681	6016.6774
DA Mirjalili (2016a, b)	0.7784	0.3855	40.3279	199.9235	5888.7013

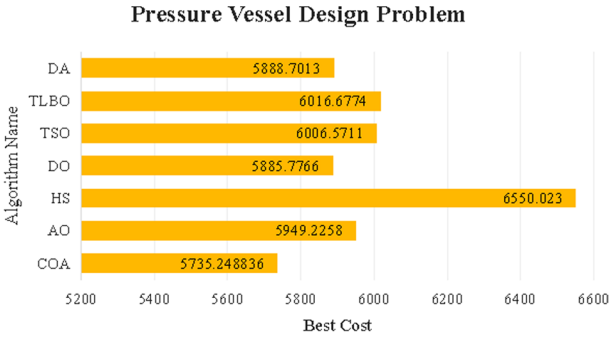

Fig. 22 Comparison of the best results of the pressure vessel design

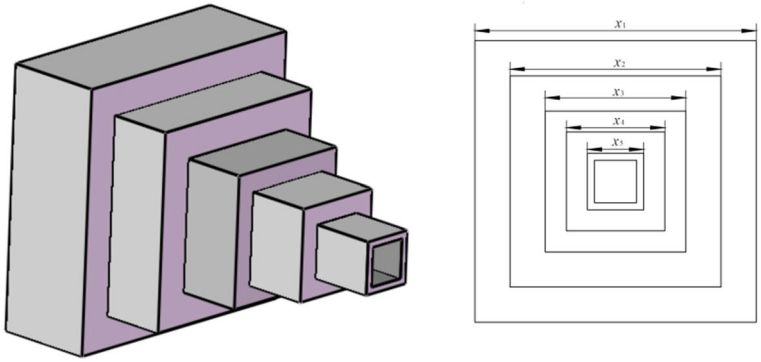

Fig. 23 Schematic diagram of the cantilever beam design

algorithm has the longest result and the largest total cost. The result of COA is the shortest, the cost is the smallest, and it is significantly shorter compared with other algorithms. It shows that COA has better optimization effect in this problem.

3.4.4 Cantilever beam design problem

The cantilever beam design problem is optimized to minimize the weight of the cantilever beam. The problem is to minimize the weight of the cantilever beam by setting five hollow blocks x_1, x_2, x_3, x_4 and x_5 with constant thickness. See Fig. 23 for details.

The problem's mathematical model is as follows:

Set:

$$x = \left[x_1\ x_2\ x_3\ x_4\ x_5\right]. \tag{49}$$

Objective function:

$$f(x) = 0.0624\left(x_1 + x_2 + x_3 + x_4 + x_5\right). \tag{50}$$

Subject to:

$$g(x) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0. \tag{51}$$

Boundaries:

$$0.01 \le x_i \le 100(i = 1, 2, \cdots 5). \tag{52}$$

The statistical results of cantilever beam design problem are shown in Table 15. From the Best Weight of each algorithm, it can be seen that the results obtained by each algorithm have a small difference, which requires more detailed optimization ability of the algorithm. In the table, the result obtained by COA is the best, indicating that COA also has outstanding ability in solving delicate problems. In Fig. 24, the results obtained by BWO algorithm are insufficient, and the rest of the algorithms are similar, with good optimization results.

3.4.5 Speed reducer design problem

The speed reducer design problem has seven variables. These seven variables are tooth surface width x_1, gear module x_2, number of teeth on pinion x_3, length of the first shaft between bearings x_4, length of the second shaft between bearings x_5, diameter of the first shaft x_6 and diameter of the second shaft xs_7. As shown in Fig. 25. The purpose of this problem is to obtain the minimum mass of the reducer under four design constraints. These four constraints are the bending stress of gear teeth, the covering stress, the lateral deflection of the shaft and the stress in the shaft.

The problem's mathematical model is as follows:

Set:

$$x = [x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7]. \tag{53}$$

Objective function:

$$\begin{aligned} f\left(\vec{x}\right) = {} & 07854 \times x_1 \times x_2^2 \times (3.3333 \times x_3^2 + 14.9334 \times x_3 - \\ & 43.0934) - 1.508 \times x_1 \times (x_6^2 + x_7^2) + 7.4777 \times x_6^3 + x_7^3 + \\ & 0.7854 \times x_4 \times x_6^2 + x_5 \times x_7^2. \end{aligned} \tag{54}$$

Subject to:

Table 15 Experimental results of the cantilever beam design

Algorithm	x_1	x_2	x_3	x_4	x_5	Best weight
COA	6.017257314	5.307150983	4.491255551	3.508156789	2.149913022	1.33996
MMA Chickermane et al. (1996)	6.01	5.3	4.49	3.49	2.15	1.34
ERHHO Song et al. (2022)	6.0509	5.2639	4.514	3.4605	2.1878	1.3402
GOA Ma et al. (2023)	6.011674	5.31297	4.48307	3.50279	2.16333	1.33996
GCA Chickermane et al. (1996)	6.01	5.3	4.49	3.49	2.15	1.34
BWO Hayyolalam and Kazem (2020)	6.2094	6.2094	6.2094	6.2094	6.2094	1.937
GSA Rashedi et al. (2009)	5.6052	4.9553	5.6619	3.1959	3.2026	1.41

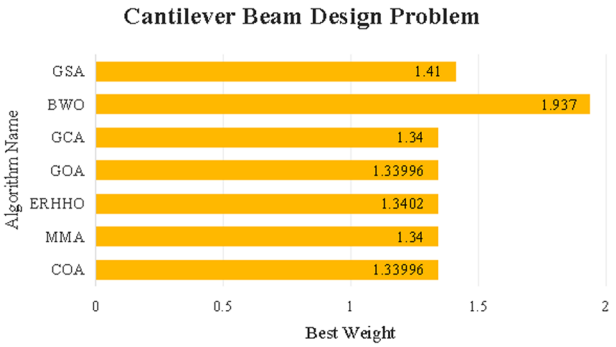

Fig. 24 Comparison of the best results of the cantilever beam design

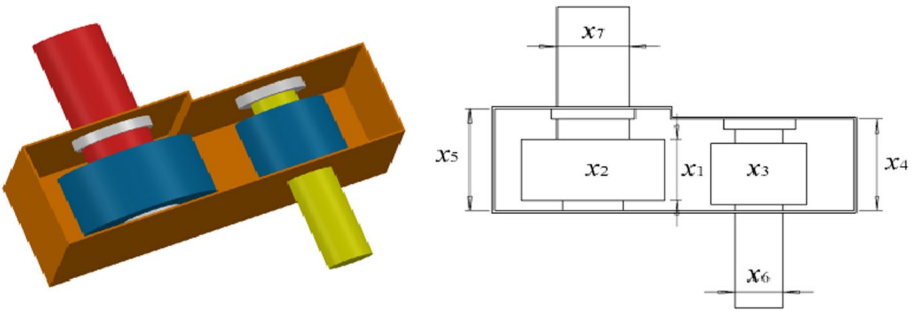

Fig. 25 Schematic diagram of the speed reducer design

$$g_1(\vec{x}) = \frac{27}{x_1 \times x_2^2 \times x_3} - 1 \le 0, \tag{55}$$

$$g_2(\vec{x}) = \frac{397.5}{x_1 \times x_2^2 \times x_3^2} - 1 \le 0, \tag{56}$$

$$g_3(\vec{x}) = \frac{1.93 \times x_4^3}{x_2 \times x_3 \times x_6^4} - 1 \le 0, \tag{57}$$

$$g_4(\vec{x}) = \frac{1.93 \times x_5^3}{x_2 \times x_3 \times x_7^4} - 1 \le 0, \tag{58}$$

$$g_5(\vec{x}) = \frac{1}{110 \times x_6^3} \times \sqrt{\left(\frac{745 \times x_4}{x_2 \times x_3}\right)^2 + 16.9 \times 10^6} - 1 \le 0, \tag{59}$$

$$g_6(\vec{x}) = \frac{1}{85 \times x_7^3} \times \sqrt{\left(\frac{745 \times x_5}{x_2 \times x_3}\right)^2 + 16.9 \times 10^6} - 1 \le 0, \tag{60}$$

$$g_7(\vec{x}) = \frac{x_2 \times x_3}{40} - 1 \leq 0, \tag{61}$$

$$g_8(\vec{x}) = \frac{5 \times x_2}{x_1} - 1 \leq 0, \tag{62}$$

$$g_9(\vec{x}) = \frac{x_1}{12 \times x_2} - 1 \leq 0, \tag{63}$$

$$g_{11}(\vec{x}) = \frac{1.1 \times x_7 + 1.9}{x_5} - 1 \leq 0. \tag{64}$$

Boundaries:

$$\begin{aligned} &2.6 \leq x_1 \leq 3.6, 0.7 \leq x_2 \leq 0.8, 17 \leq x_3 \leq 28, 7.3 \leq x_4 \leq 8.3, \\ &7.3 \leq x_5 \leq 8.3, 2.9 \leq x_6 \leq 3.9, 5 \leq x_7 \leq 5.5. \end{aligned} \tag{65}$$

Table 16 shows the statistical results of the speed reducer design problem. From the final result data, it can be concluded that COA can achieve good results. The results of RSA, AOA and WSA are close to those of COA. Other comparison algorithms differ greatly from the data obtained by COA. As can be seen in Fig. 26, the data obtained by the hHHO-SCA, FA and EHO algorithms are significantly larger. The data obtained by COA is better. It is proved that COA has good effect in this engineering problem.

4 Analyze the characteristics and optimization capability of COA

4.1 Characteristics of COA

- In COA, the algorithm enters different stages through temperature changes. This method better balances the exploration and exploitation capabilities of COA.
- In the summer resort stage, crawfish are approaching the cave. Caves represent the best solution, which brings each population closer to the best solution.
- In the competition stage, two crayfish compete with each other and adjust their positions, increasing the search range of the population. The exploration ability of the algorithm is improved.
- In the foraging stage, crayfish can be divided into two feeding modes. Crayfish will eat according to their position and food position. If the food is large, the crayfish will break down the larger food before eating. The exploitation capability of COA are greatly excellent. Benefiting from this stage, more detailed results can be obtained when solving problems.

4.2 Optimization capability of COA

According to the above experimental results and analysis, it can be concluded that COA has good optimization effects in 23 standard reference functions, CEC2014 and 5 engineering problems. In the convergence curves of many test functions, COA does not obtain a good fitness value at first. However, in the process of iteration, COA can continuously

Table 16 Experimental results of the speed reducer design

Algorithm	x_1	x_2	x_3	x_4	x_5	x_6	x_7	Best weight
COA	3.498064153	0.7	17.00009041	7.3	7.8	3.35003002	5.285458663	2995.472941
RSA Abualigah et al. (2022)	3.50279	0.7	17	7.30812	7.74715	3.35067	5.28675	2996.5157
AOA Abualigah et al. (2021a, b)	3.50384	0.7	17	7.3	7.72933	3.35649	5.2867	2997.9157
hHHO-SCA Kamboj et al. (2020)	3.506119	0.7	17	7.3	7.99141	3.452569	5.286749	3029.873076
WSA Baykasoğlu and Akpinar (2015)	3.5	0.7	17	7.3	7.8	3.350215	5.286683	2996.348225
FA Baykasoğlu and Ozsoydan (2015)	3.507495	0.7001	17	7.719674	8.080854	3.351512	5.287051	3010.137492
EHO Hashim et al. (2019)	2.9	0.7	17	7.3	7.8	3.1	5.2	3019.01
HGSO Hashim et al. (2019)	3.498	0.71	17.02	7.67	7.81	3.36	5.289	2997.1

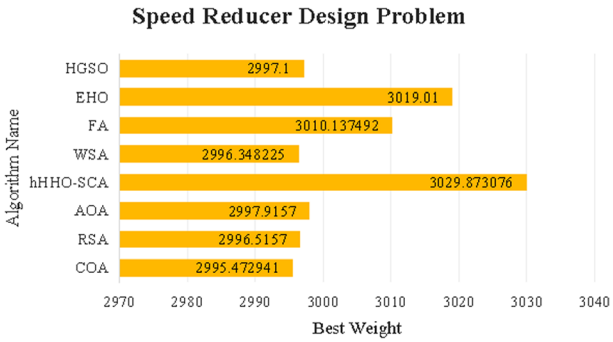

Fig. 26 Comparison of the best results of the speed reducer design

converge and obtain a good fitness value. This is because in the summer resort stage, the crayfish approach the cave differently, which means that the algorithm has strong convergence ability. In the foraging stage, crayfish is also approaching food, enhancing the exploitation ability of the algorithm. Many comparison algorithms are prone to fall into local optimum at the end of the iteration, which leads to the convergence difficulties. However, although COA also has the problem of falling into local optimum in the later stage. The exploration and exploitation of COA is regulated by temperature, which can better balance the exploration and exploitation capabilities, so that the algorithm still has a good convergence ability in the later iteration period, and finally gets a good fitness value.

4.3 Deficiencies and limitations of COA

COA defines temperature to balance the exploration and exploitation capabilities of algorithms. Although COA has good optimization ability, there is a problem of falling into local optima in the later stage. The algorithm has two exploitation stages, which weakens its exploration ability. In the early stages of iteration, COA has strong exploitation capabilities, enabling the algorithm to converge quickly. However, in the later stages of iteration, COA's exploration ability is weak, making it difficult for the algorithm to find better locations. But from the convergence curves of the 23 standard benchmark functions and the CEC2014 test function, it can be seen that if COA jumps out of local optimum, COA can quickly converge and find a good solution. In the later research, we will strengthen the exploration ability of COA and apply it to engineering problems such as Feature selection, image processing, multi threshold image segmentation, etc.

5 Conclusions

This article proposes a meta heuristic optimization algorithm based on the summer avoidance, competition, and foraging behavior of crayfish, known as the Crayfish Optimization Algorithm. By defining temperature, the COA enters different stages, balancing its exploration and exploitation capabilities. Among them, Summer resort stage represents the exploration stage of COA. Composition stage and foraging stage represent the exploitation stage of COA. The characteristic of this algorithm is through the exploration and exploitation process of temperature control algorithms. During the exploration phase, a cave was defined, and crayfish would enter the cave to escape the heat. The exploitation stage is

divided into the foraging stage and the competition stage. During the foraging stage, COA defines the size of the food to choose whether to shred it, and the feeding of crayfish is controlled by the foraging amount. In the competition stage, crayfish will compete in caves and enter them to evade. The algorithm has good optimization effect.

In the experimental section, 23 standard benchmark functions and CEC2014 are used to verify the optimization effect of COA, and 9 comparative algorithms are selected for comparative experiments. The result analysis shows that the overall performance of COA is good, but its exploration ability is weak, which leads to the algorithm being prone to falling into local optimum in the later stage. However, in most test functions, COA still has good convergence ability. After jumping out of local optima, COA can quickly converge and find a good solution. For the comparison algorithms SCA, SCSO, ROA, GA, STOA, BES, PDO, and WOA, all of these algorithms have good results in the test function, but there is a lack of convergence ability due to the inability to jump out of local optimum in the later stage, resulting in the inability to obtain better fitness values. The data shows that COA can obtain a good fitness value in these five engineering problems. This indicates that COA has a good effect in solving practical problems.

This study validated the effectiveness of COA in solving optimization problems using 23 standard benchmark functions, CEC2014 test functions, and engineering problems. In future work, we will further improve the exploration ability of COA and apply the algorithm to solve some practical engineering problems, such as 3D path planning of UAV, feature selection, image processing, multi threshold image segmentation, etc.

Author contributions JH: conceptualization, methodology, investigation, funding acquisition, writing—review and editing, writing—original draft; RH: conceptualization, methodology, software, data curation, writing—original draft; WC: validation, visualization; SM: supervision, writing—review and editing.

Declarations

Competing interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abedinpourshotorban H, Shamsuddin SM, Beheshti Z, Jawawi DN (2016) Electromagnetic field optimization: a physics-inspired metaheuristic optimization algorithm. Swarm Evol Comput 26:8–22. https://doi.org/10.1016/j.swevo.2015.07.002

Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021a) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609. https://doi.org/10.1016/j.cma.2020.113609

Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-Qaness MA, Gandomi AH (2021b) Aquila optimizer: a novel meta-heuristic optimization algorithm. Comput Ind Eng 157:107250. https://doi.org/10.1016/j.cie.2021.107250

Abualigah L, Abd Elaziz M, Sumari P, Geem ZW, Gandomi AH (2022) Reptile search algorithm (RSA): a nature-inspired meta-heuristic optimizer. Expert Syst Appl 191:116158. https://doi.org/10.1016/j.eswa.2021.116158

Allan EL, Froneman PW, Hodgson AN (2006) Effects of temperature and salinity on the standard metabolic rate (SMR) of the caridean shrimp *Palaemon peringueyi*. J Exp Mar Biol Ecol 337(1):103–108. https://doi.org/10.1016/j.jembe.2006.06.006

Alsattar HA, Zaidan AA, Zaidan BB (2020) Novel meta-heuristic bald eagle search optimisation algorithm. Artif Intell Rev 53:2237–2264. https://doi.org/10.1007/s10462-019-09732-5

Babalik A, Cinar AC, Kiran MS (2018) A modification of tree-seed algorithm using Deb's rules for constrained optimization. Appl Soft Comput 63:289–305. https://doi.org/10.1016/j.asoc.2017.10.013

Banzhaf W, Koza JR, Ryan C, Spector L, Jacob C (2000) Genetic programming. IEEE Intell Syst their Appl 15(3):74–84. https://ieeexplore.ieee.org/abstract/document/846288

Baykasoğlu A, Akpinar Ş (2015) Weighted superposition attraction (WSA): a swarm intelligence algorithm for optimization problems–Part 2: constrained optimization. Appl Soft Comput 37:396–415. https://doi.org/10.1016/j.asoc.2015.08.052

Baykasoğlu A, Ozsoydan FB (2015) Adaptive firefly algorithm with chaos for mechanical design optimization problems. Appl Soft Comput 36:152–164. https://doi.org/10.1016/j.asoc.2015.06

Bellman KL, Krasne FB (1983) Adaptive complexity of interactions between feeding and escape in crayfish. Science 221(4612):779–781

Berrill M, Chenoweth B (1982) The burrowing ability of nonburrowing crayfish. Am Midl Nat. https://doi.org/10.2307/2425310

Beyer HG, Schwefel HP (2002) Evolution strategies–a comprehensive introduction. Nat Comput 1:3–52. https://doi.org/10.1023/A:1015059928466

Braik M, Hammouri A, Atwan J, Al-Betar MA, Awadallah MA (2022) White shark optimizer: a novel bio-inspired meta-heuristic algorithm for global optimization problems. Knowl Based Syst 243:108457. https://doi.org/10.1016/j.knosys.2022.108457

Chen H, Chen L, Zhang G (2022) Block-structured integer programming: can we parameterize without the largest coefficient? Discrete Optim 46:100743. https://doi.org/10.1016/j.disopt.2022.100743

Cheng S, Qin Q, Chen J, Shi Y (2016) Brain storm optimization algorithm: a review. Artif Intell Rev 46:445–458. https://doi.org/10.1007/s10462-016-9471-0

Chickermane HE, M. I. A. N. T, Gea HC (1996) Structural optimization using a new local approximation method. Int J Numer Methods Eng 39(5):829–846.

Crandall KA, De Grave S (2017) An updated classification of the freshwater crayfishes (*Decapoda*: *Astacidea*) of the world, with a complete species list. J Crustac Biol 37(5):615–653. https://doi.org/10.1093/jcbiol/rux070

Dantzig GB (2002) Linear programming. Oper Res 50(1):42–47. https://doi.org/10.1287/opre.50.1.42.17798

Daryalal M, Bodur M, Luedtke JR (2022) Lagrangian dual decision rules for multistage stochastic mixed-integer programming. Operations Res. https://doi.org/10.1287/opre.2022.2366

Das M, Roy A, Maity S, Kar S, Sengupta S (2022) Solving fuzzy dynamic ship routing and scheduling problem through new genetic algorithm. Decis Making: Appl Manage Eng 5(2):329–361. https://doi.org/10.31181/dmame181221030d

Dhiman G, Kumar V (2017) Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 114:48–70. https://doi.org/10.1016/j.advengsoft.2017.05.014

Dhiman G, Kaur A (2019) STOA: a bio-inspired based optimization algorithm for industrial engineering problems. Eng Appl Artif Intell 82:148–174. https://doi.org/10.1016/j.engappai.2019.03.021

Dhiman G, Kumar V (2019) Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems. Knowl Based Syst 165:169–196. https://doi.org/10.1016/j.knosys.2018.11.024

Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Mag 1(4):28–39. https://ieeexplore.ieee.org/abstract/document/4129846

Ezugwu AE, Shukla AK, Nath R, Akinyelu AA, Agushaka JO, Chiroma H, Muhuri PK (2021) Metaheuristics: a comprehensive overview and classification along with bibliometric analysis. Artif Intell Rev 54:4237–4316. https://doi.org/10.1007/s10462-020-09952-0

Ezugwu AE, Agushaka JO, Abualigah L, Mirjalili S, Gandomi AH (2022) Prairie dog optimization algorithm. Neural Comput Appl 34(22):20017–20065. https://doi.org/10.1007/s00521-022-07530-9

Florey CL, Moore PA (2019) Analysis and description of burrow structure in four species of freshwater crayfishes (*Decapoda*: *Astacoidea*: *Cambaridae*) using photogrammetry to recreate casts as 3D models. J Crustacean Biology 39(6):711–719. https://doi.org/10.1093/jcbiol/ruz075

García-Guerrero M, Hernández-Sandoval P, Orduña-Rojas J, Cortés-Jacinto E (2013) Effect of temperature on weight increase, survival, and thermal preference of juvenile redclaw crayfish Cherax quadricarinatus. Hidrobiológica 23(1):73–81

Gandomi AH, Yang XS, Alavi AH (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng With Comput 29:17–35. https://doi.org/10.1007/s00366-011-0241-y

Gautier A, Granot F (1994) On the equivalence of constrained and unconstrained flows. Discrete Appl Math 55(2):113–132. https://doi.org/10.1016/0166-218X(94)90003-5
Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68. https://doi.org/10.1177/003754970107600201
Graham ZA, Stubbs MB, Loughman ZJ (2022) Digging ability and digging performance in a hyporheic gravel-dwelling crayfish, the hairy crayfish Cambarus friaufi (Hobbs 1953)(*Decapoda*: *Astacidae*: *Cambaridae*). J Crustac Biol 42(1):ruac002. https://doi.org/10.1093/jcbiol/ruac002
Hatamlou A (2013) Black hole: a new heuristic optimization approach for data clustering. Inf Sci 222:175–184. https://doi.org/10.1016/j.ins.2012.08.023
Hashim FA, Hussien AG (2022) Snake optimizer: a novel meta-heuristic optimization algorithm. Knowl Based Syst 242:108320. https://doi.org/10.1016/j.knosys.2022.108320
Hashim FA, Houssein EH, Mabrouk MS, Al-Atabany W, Mirjalili S (2019) Henry gas solubility optimization: a novel physics-based algorithm. Future Gener Computer Syst 101:646–667. https://doi.org/10.1016/j.future.2019.07.015
Hashim FA, Houssein EH, Hussain K, Mabrouk MS, Al-Atabany W (2022) Honey badger algorithm: new metaheuristic algorithm for solving optimization problems. Math Comput Simul 192:84–110. https://doi.org/10.1016/j.matcom.2021.08.013
Hayyolalam V, Kazem AAP (2020) Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems. Eng Appl Artif Intell 87:103249. https://doi.org/10.1016/j.engappai.2019.103249
He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20(1):89–99. https://doi.org/10.1016/j.engappai.2006.03.003
Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Future Gener Computer Syst 97:849–872. https://doi.org/10.1016/j.future.2019.02.028
Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–73. https://www.jstor.org/stable/24939139
Jaderyan M, Khotanlou H (2016) Virulence optimization algorithm. Appl Soft Comput 43:596–618. https://doi.org/10.1016/j.asoc.2016.02.038
Jia H, Peng X, Lang C (2021) Remora optimization algorithm. Expert Syst Appl 185:115665. https://doi.org/10.1016/j.eswa.2021.115665
Jia H, Sun K, Li Y, Cao N (2022a) Improved marine predators algorithm for feature selection and SVM optimization. KSII Trans Internet Inform Syst (TIIS) 16(4):1128–1145. https://doi.org/10.3837/tiis.2022.04.003
Jia H, Zhang W, Zheng R, Wang S, Leng X, Cao N (2022b) Ensemble mutation slime mould algorithm with restart mechanism for feature selection. Int J Intell Syst 37(3):2335–2370. https://doi.org/10.1002/int.22776
Jones CM, Ruscoe IM (2001) Assessment of five shelter types in the production of redclaw crayfish *Cherax quadricarinatus* (*Decapoda*: *Parastacidae*) under earthen pond conditions. J World Aquaculture Soc 32(1):41–52. https://doi.org/10.1111/j.1749-7345.2001.tb00920.x
Kamboj VK, Nandi A, Bhadoria A, Sehgal S (2020) An intensify Harris Hawks optimizer for numerical and engineering optimization problems. Appl Soft Comput 89:106018. https://doi.org/10.1016/j.asoc.2019.106018
Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim 39:459–471. https://doi.org/10.1007/s10898-007-9149-x
Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112:283–294. https://doi.org/10.1016/j.compstruc.2012.09.003
Kaveh A, Dadras A (2017) A novel meta-heuristic optimization algorithm: thermal exchange optimization. Adv Eng Softw 110:69–84. https://doi.org/10.1016/j.advengsoft.2017.03.014
Kennedy J, Eberhart R (1995) Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (vol 4, pp 1942–1948). IEEE. https://ieeexplore.ieee.org/abstract/document/488968
Khishe M, Mosavi MR (2020) Chimp optimization algorithm. Expert Syst Appl 149:113338. https://doi.org/10.1016/j.eswa.2020.113338
Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
Kouba A, Petrusek A, Kozák P (2014) Continental-wide distribution of crayfish species in Europe: update and maps. Knowl Manage Aquat Ecosyst. https://doi.org/10.1051/kmae/2014007
Larson ER, Olden JD (2011) The state of crayfish in the Pacific Northwest. Fisheries 36(2):60–73. https://doi.org/10.1577/03632415.2011.10389069

Liu Q, Li N, Jia H, Qi Q, Abualigah L (2022) Modified remora optimization algorithm for global optimization and multilevel thresholding image segmentation. Mathematics 10(7):1014. https://doi.org/10.3390/math10071014

Ma C, Huang H, Fan Q, Wei J, Du Y, Gao W (2022) Grey wolf optimizer based on aquila exploration method. Expert Syst Appl 205:117629. https://doi.org/10.1016/j.eswa.2022.117629

Ma B, Hu Y, Lu P, Liu Y (2023) Running City game optimizer: a game-based metaheuristic optimization algorithm for global optimization. J Comput Des Eng 10(1):65–107. https://doi.org/10.1093/jcde/qwac131

Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98. https://doi.org/10.1016/j.advengsoft.2015.01.010

Mirjalili S (2016a) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27:1053–1073. https://doi.org/10.1007/s00521-015-1920-1

Mirjalili S (2016b) SCA: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133. https://doi.org/10.1016/j.knosys.2015.12.022

Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27:495–513. https://doi.org/10.1007/s00521-015-1870-7

Mzili T, Riffi ME, Mzili I, Dhiman G (2022) A novel discrete rat swarm optimization (DRSO) algorithm for solving the traveling salesman problem. Decis making: Appl Manage Eng 5(2):287–299. https://doi.org/10.31181/dmame0318062022m

Mzili I, Mzili T, Riffi ME (2023) Efficient routing optimization with discrete penguins search algorithm for MTSP. Decis Making: Appl Manage Eng 6(1):730–743. https://doi.org/10.31181/dmame04092023m

Payette AL, McGaw IJ (2003) Thermoregulatory behavior of the crayfish *Procambarus clarki* in a burrow environment. Comp Biochem Physiol A: Mol Integr Physiol 136(3):539–556. https://doi.org/10.1016/S1095-6433(03)00203-4

Precup RE, David RC, Roman RC, Petriu EM, Szedlak-Stinean AI (2021) Slime mould algorithm-based tuning of cost-effective fuzzy controllers for servo systems. Int J Comput Intell Syst 14(1):1042–1052. https://www.atlantis-press.com/journals/ijcis/125954163

Qi H, Zhang G, Jia H, Xing Z (2021) A hybrid equilibrium optimizer algorithm for multi-level image segmentation. Math Biosci Eng 18:4648–4678

Rao RV, Savsani VJ, Vakharia DP (2012) Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. Inf Sci 183(1):1–15. https://doi.org/10.1016/j.ins.2011.08.006

Rao H, Jia H, Wu D, Wen C, Li S, Liu Q, Abualigah L (2022) A modified group teaching optimization algorithm for solving constrained engineering optimization problems. Mathematics 10(20):3765. https://doi.org/10.3390/math10203765

Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248. https://doi.org/10.1016/j.ins.2009.03.004

Satapathy S, Naik A (2016) Social group optimization (SGO): a new population evolutionary optimization technique. Complex & Intell Syst 2(3):173–203. https://doi.org/10.1007/s40747-016-0022-8

Seyyedabbasi A, Kiani F (2022) Sand cat swarm optimization: a nature-inspired algorithm to solve global optimization problems. Eng With Comput. https://doi.org/10.1007/s00366-022-01604-x

Sinha N, Chakrabarti R, Chattopadhyay PK (2003) Evolutionary programming techniques for economic load dispatch. IEEE Trans Evol Comput 7(1):83–94. https://ieeexplore.ieee.org/abstract/document/1179910

Song M, Jia H, Abualigah L, Liu Q, Lin Z, Wu D, Altalhi M (2022) Modified harris hawks optimization algorithm with exploration factor and random walk strategy. Comput Intell Neurosci. https://doi.org/10.1155/2022/4673665

Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341. https://doi.org/10.1023/A:1008202821328

Wang S, Hussien AG, Jia H, Abualigah L, Zheng R (2022) Enhanced remora optimization algorithm for solving constrained engineering optimization problems. Mathematics 10(10):1696. https://doi.org/10.3390/math10101696

Wen C, Jia H, Wu D, Rao H, Li S, Liu Q, Abualigah L (2022) Modified remora optimization algorithm with multistrategies for global optimization problem. Mathematics 10(19):3604. https://doi.org/10.3390/math10193604

Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://ieeexplore.ieee.org/abstract/document/585893

Wu D, Rao H, Wen C, Jia H, Liu Q, Abualigah L (2022) Modified sand cat swarm optimization algorithm for solving constrained engineering optimization problems. Mathematics 10(22):4350. https://doi.org/10.3390/math10224350

Xie L, Han T, Zhou H, Zhang ZR, Han B, Tang A (2021) Tuna swarm optimization: a novel swarm-based metaheuristic algorithm for global optimization. Comput Intell Neurosci 2021:1–22. https://doi.org/10.1155/2021/9210050

Xing B, Gao WJ, Xing B, Gao WJ (2014) Imperialist competitive algorithm. In: Kacprzyk J, Jain LC (eds) Innovative computational intelligence: a rough guide to 134 clever algorithms. Springer, Berlin. https://doi.org/10.1007/978-3-319-03404-1_15

Zhang Y, Jin Z (2020) Group teaching optimization algorithm: a novel metaheuristic method for solving global optimization problems. Expert Syst Appl 148:113246. https://doi.org/10.1016/j.eswa.2020.113246

Zhao S, Zhang T, Ma S, Chen M (2022) Dandelion optimizer: a nature-inspired metaheuristic algorithm for engineering applications. Eng Appl Artif Intell 114:105075. https://doi.org/10.1016/j.engappai.2022.105075