Knowledge and Information Systems
https://doi.org/10.1007/s10115-024-02179-3

REGULAR PAPER

Cooperative coati optimization algorithm with transfer functions for feature selection and knapsack problems

Rui Zhong[1] · Chao Zhang[2] · Jun Yu[3]

Received: 3 April 2024 / Revised: 14 June 2024 / Accepted: 29 June 2024

Abstract
Coatis optimization algorithm (COA) has recently emerged as an innovative meta-heuristic algorithm (MA) for global optimization, garnering considerable attention from scholars and researchers. In this paper, we introduce three techniques to enhance COA: (1) the cooperative mechanism, (2) the fitness-based division, and (3) the optional base vector strategy. Collectively, we refer to our improved method as cooperative COA (CCOA). In addition, we introduce the incorporation of the S-shaped Sigmoid transfer function and the V-shaped Tanh transfer function into CCOA, leading to the development of SCCOA and VCCOA. These adaptations effectively address the challenges posed by feature selection tasks and the 0/1 knapsack problem. To comprehensively evaluate the performance of the continuous version of CCOA, as well as the binary versions of SCCOA and VCCOA, we conducted two distinct categories of numerical experiments. Firstly, we compared CCOA with nine representative MAs, including the original COA, on CEC2020 benchmark functions and six engineering optimization problems. Secondly, SCCOA and VCCOA are compared with six famous binary MAs on 13 feature selection datasets and 18 standard 0/1 knapsack problems. Experimental and statistical results show the competitiveness of CCOA and its binary versions, and it is promising to extend CCOA to various real-world application scenarios.

Keywords Coati optimization algorithm (COA) · Cooperative mechanism · Fitness-based division · Optional base vector strategy · Feature selection · 0/1 Knapsack problem

✉ Jun Yu
yujun@ie.niigata-u.ac.jp

Rui Zhong
rui.zhong.u5@elms.hokudai.ac.jp

Chao Zhang
zhang@eng.u-toyama.ac.jp

[1] Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan

[2] Faculty of Engineering, University of Toyama, Toyama, Japan

[3] Institute of Science and Technology, Niigata University, Niigata, Japan

Published online: 15 July 2024

1 Introduction

Optimization is an unavoidable topic involved in computer science and operations research. Traditional methods such as linear programming, nonlinear programming, integer programming, Newton's method, conjugate gradient method, and gradient descent method have achieved satisfactory performance for certain problem types [1–3]. However, these approaches often necessitate a convex feasible domain, a continuously differentiable objective function, or additional constraints [4, 5]. Nevertheless, complexities abound in real-world applications and complex systems, rendering problems non-differentiable, non-convex, multimodal, and often black-box in nature, which is a significant challenge for traditional optimization methods [6–8]. In addition, as the scale of the problem increases, traditional methods frequently struggle to deliver acceptable solutions within constrained computational resources for large-scale NP-hard optimization problems [9, 10]. Hence, the development of efficient and readily implementable algorithms becomes imperative for tackling these issues.

Fortunately, meta-heuristic algorithms (MAs) offer a potential avenue for addressing these challenges. As a kind of stochastic optimization technique, MAs are usually inspired by natural phenomena or behaviors of organisms to design search operators. Their advantages, which encompass robustness, scalability, applicability, and derivative-free characteristics, have garnered significant attention from scholars [11–13]. Over the past few decades, evolutionary computation (EC) and swarm intelligence (SI) have experienced remarkable growth and have achieved considerable success across various domains. A plethora of MAs have been introduced to tackle diverse optimization problems. In Table 1, we present an overview of the most recent MAs, and in Fig. 1, we summarize the number of published papers related to novel MAs. These visuals serve to underscore the vibrancy of the EC research community.

As one of the latest MAs, the coati optimization algorithm (COA) [44] imitates two key behaviors of coatis: (1) the hunting and attacking iguanas and (2) escaping from predators. In

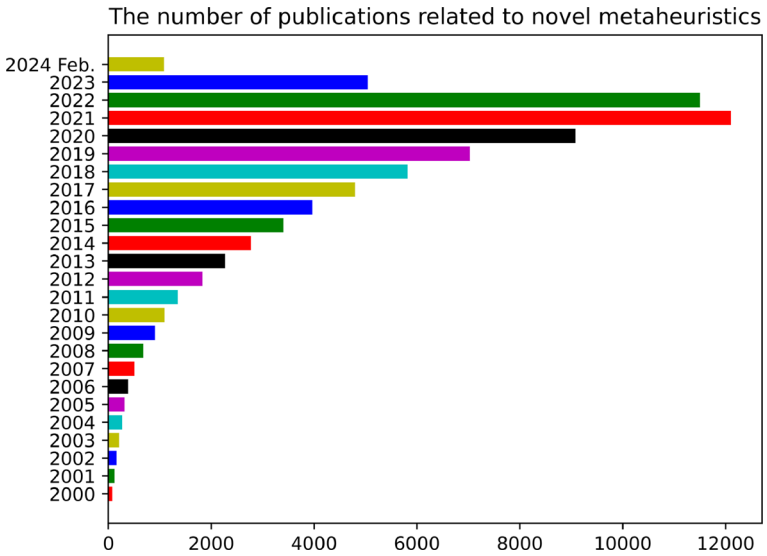


Fig. 1 The number of published papers related to novel meta-heuristics, and the data is counted by Google Scholar with keywords: ("novel meta-heuristic" OR "novel metaheuristic" OR "new meta-heuristic" OR "new metaheuristic")

Table 1 The latest MAs in literature

Algorithm	Year	Inspiration
Fire hawk optimizer (FHO) [14]	2023	The foraging behavior of whistling kites, black kites, and brown falcons
Energy valley optimizer (EVO) [15]	2023	Physics principles regarding stability and different modes of particle decay
Squid game optimizer (SGO) [16]	2023	The primary rules of a traditional Korean game named squid game
Waterwheel plant algorithm (WWPA) [17]	2023	The hunting behaviors of waterwheel plants
Spider wasp optimizer (SWO) [18]	2023	The hunting, nesting, and mating behaviors of the female spider wasps in nature
Nutcracker optimizer (NOA) [19]	2023	Two distinct behaviors of nutcracker in different seasons
Great wall construction algorithm (GWCA) [20]	2023	The competition and elimination mechanisms observed among workers
Walrus optimization algorithm (WaOA) [21]	2023	The feeding, migrating, escaping, and fighting behaviors of walrus
Mountaineering team-based optimization (MTBO) [22]	2023	The social behavior and human cooperation to reach a mountaintop
Growth optimizer (GO) [23]	2023	Learning and reflection mechanisms of individuals in their growth processes
Mother optimization algorithm (MOA) [24]	2023	The mother's care of children in three phases
RIME [25]	2023	Soft-rime and hard-rime growth process of rime-ice
Weighted-leader search (WLS) [26]	2023	The trade-off between exploration and exploitation in different stage
Drawer algorithm (DA) [27]	2023	The selection of objects from different drawers to create optimal combinations
Coronavirus metamorphosis optimization algorithm (CMOA) [28]	2023	Metabolism and transformation under various conditions

Table 1 continued

Algorithm	Year	Inspiration
Snow ablation optimizer (SAO) [29]	2023	The sublimation and melting behavior of snow
Dujiangyan irrigation system optimization (DISO) [30]	2023	The DIS's water separation and sand discharge principle
Special forces algorithm (SFA) [31]	2023	The missions of modern special forces in real life
tree optimization algorithm (TOA) [28]	2023	The growth of trees
Dark forest algorithm (DFA) [32]	2023	The development of civilization
Chernobyl disaster optimizer (CDO) [33]	2023	The nuclear reactor core explosion of Chernobyl
American zebra optimization algorithm (AZOA) [34]	2023	The social behavior of American zebras in the wild
Group learning algorithm (GLO) [35]	2023	The way individuals inside a group affect each other
Fick's law algorithm (FLA) [36]	2023	Fick's first rule of diffusion
Gazelle optimization algorithm (GOA) [37]	2023	The gazelles' survival ability in their predator-dominated environment
Vegetation evolution (VEGE) [38]	2022	The growth and maturity periods of plants
Special relativity search (SRS) [39]	2022	The interaction of particles in an electromagnetic field
GOZDE [40]	2022	The information sharing between the zones
Snake optimizer (SO) [41]	2022	The special mating behavior of snakes
Plant competition optimization (PCO) [42]	2022	Plant competition processes
Water optimization algorithm (WAO) [43]	2022	The chemical and physical properties of water molecules

the hunting and attacking phase, the coati swarm is divided into two equal-sized sub-swarms in a sequential manner. The first sub-swarm of coatis engages in attacking iguanas, while the other half continues to search for iguanas on the ground. These behaviors constitute the exploration stage. During the escaping phase, coatis move away from the predator, seeking the nearest safe location. COA characterizes this behavior as the exploitation stage. The excellent performance of the original COA can be observed from comparison experiments with eleven famous MAs, such as white shark optimizer (WSO), reptile search algorithm (RSA), and marine predators algorithm (MPA), on CEC2011 real-world applications and CEC2017 benchmark functions. However, certain issues remain unresolved in the original COA, including an imbalance in exploration and exploitation, the slow convergence speed, and the weak ability to escape from local optima. To address these challenges, this paper proposes a cooperative coati optimization algorithm (CCOA), which introduces more powerful and efficient search strategies. The competitive performance of CCOA is observed from the numerical experiments on the CEC2020 benchmark and engineering optimization tasks. Therefore, we infer that the binary versions of COA may also have great potential in solving combinatorial optimization tasks. We equip CCOA with two distinct transfer functions named the S-shaped Sigmoid and the V-shaped Tanh transfer functions to allow SCCOA and VCCOA to tackle combinatorial optimization problems effectively. In summary, the contributions of this paper are as follows:

- We introduce a cooperative mechanism into the original COA and modify the search strategy during the exploration stage.
- We incorporate an optional base vector strategy in the escaping phase to boost the exploitative capability of CCOA.
- We integrate both the S-shaped Sigmoid transfer function and the V-shaped Tanh transfer function into CCOA, resulting in two binary versions: SCCOA and VCCOA.
- Numerical experiments on 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions, as well as six engineering problems, are conducted to evaluate the performance of CCOA.
- Numerical experiments on 13 feature selection datasets and 18 standard knapsack problems are performed to further evaluate the performance of SCCOA and VCCOA.

The remainder of this paper is structured as follows. Section 2 introduces related works, encompassing the wrapper-based feature selection and the 0/1 knapsack problem. Section 3 introduces the original COA. Section 4 provides a detailed introduction to our proposal: cooperative COA and its two binary versions. Numerical experiments and analysis are presented in Sect. 5. Finally, Sect. 6 concludes this paper.

2 Related works

2.1 Wrapper-based feature selection

The wrapper-based feature selection approach regards the feature selection task as a combinatorial optimization problem, with the goal of identifying the optimal feature subset. This solution representation is illustrated in Fig. 2, where $x_i = 1$ means the corresponding feature is selected, and vice versa. In brief, the wrapper-based feature selection process can be divided into four key steps:

1. Initialize binary solution(s).
2. Evaluate solution(s) using the classifier.

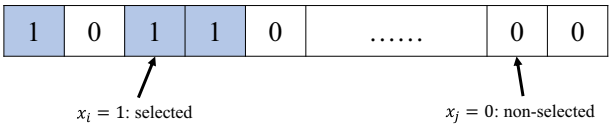


Fig. 2 The encoding of solutions in wrapper-based feature selection approaches [45]

3. Apply search operator to generate offspring solution(s).
4. Repeat step (2) to step (3) until the optimal feature subset is found or the computational budget is exhausted.

In the wrapper-based feature selection, the pivotal step is the search operator in step (3). Given a dataset with N features, the potential combinations of the feature subset are $2^N - 1$. Since brute-force search is computationally expensive, many MAs have been adopted as search engines: Hammouri et al. [46] proposed three improved versions of the dragonfly algorithm (DA): linear DA, quadratic DA, and sinusoidal DA. These variants incorporate the V-shaped Tanh function, which maps the continuous search space to discrete domains for feature selection tasks. Khurma et al. [47] proposed an enhanced moth flame optimization (EMFO) that includes the Lévy flight operator. They combined EMFO with eight transfer functions to create eight binary variants of BEMFO, applied to 23 medical datasets. Mohammad et al. [48] proposed an enhanced whale optimization algorithm (E-WOA), featuring a unique pooling mechanism and effective search strategies. E-WOA utilizes a uniform random transfer function to handle COVID-19 medical datasets. Chen et al. [49] proposed a chaotic antlion algorithm (CAA) with a quasi-opposition learning mechanism and chaotic strategies, and the robustness of CAA has been evaluated on various learning models, including decision tree, Naive Bayes, and support vector machine (SVM), in Chinese text feature selection tasks. Alzaqebah et al. [50] proposed a memory-based cuckoo search algorithm for gene expression datasets feature selection, where the memory-based mechanism saves the most informative features identified by the best solutions. Kundu et al. [51] proposed an altruistic whale optimization algorithm (AltWOA) for feature selection on microarray datasets. AltWOA introduces an altruistic mechanism to WOA, enhancing the search operator's efficiency in propagating candidate solutions and achieving convergence. Rahul et al. [52] proposed an improved spider monkey optimization with an oscillating mechanism (OSMO) for the soil image feature selection. The novel OSMO balances exploration and exploitation using an oscillating perturbation rate, enhancing the precision of classification and optimization convergence.

2.2 0/1 Knapsack problem

The 0/1 knapsack problem is a classic NP-complete combinatorial optimization problem [53]. It revolves around three important factors: items, knapsack capacity, and constraints. Each item is defined by two attributes: weight and value. The knapsack capacity restricts the maximum weight of the selected items, and the constraint dictates that each item either be selected once or not at all, which is the origin of the name 0/1 knapsack problem. Moreover, the mathematical model for this problem is formulated as follows:

$$\begin{aligned} &\text{maximize} \sum_{i=1}^{N} v_i \cdot x_i \\ &\text{subject to} \sum_{i=1}^{N} w_i \cdot x_i \leq C \\ &\text{where } x_i \in \{0, 1\} \end{aligned} \tag{1}$$

where x is a binary vector with the same dimension as the number of items, where $x_i = 1$ indicates that the i^{th} item is selected, and vice versa. The variables v_i and w_i are the value and the weight of the i^{th} item, respectively, and C is the knapsack capacity. Simply, the objective of the 0/1 knapsack problem is to maximize the total value of the selected items while ensuring that the total weight of the selected items does not exceed the knapsack's capacity. In addition, many real-world applications can be regarded as the variants of the knapsack problem, including water resource engineering and flood management [54], investments and portfolios [55], and distribution and allocation of resources [56]. Thus, research on the knapsack problem holds substantial importance and relevance.

3 Coati optimization algorithm (COA)

As a kind of animal-inspired MA, COA imitates two behaviors of coatis: (1) The behavior of hunting and attacking iguanas corresponds to the exploration phase in MA design. (2) The behavior of escaping from predators corresponds to the exploitation phase in MA design. In the following subsections, we will introduce the components of the COA sequentially.

3.1 Coati swarm initialization

COA adopts a random swarm initialization with a uniform distribution. Equation (2) outlines the structure of the coati swarm, consisting of n individuals and m dimensions.

$$P = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ \vdots \\ X_n \end{bmatrix}_{n \times m} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ x_{31} & x_{32} & \cdots & x_{3m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}_{n \times m} \tag{2}$$
$$x_{ij} = r \cdot (UB_j - LB_j) + LB_j$$

where P is the coati swarm, X_i is the i^{th} coati. LB_j and UB_j are the lower and the upper bound of the j^{th} search space, respectively, and r is a random number in (0, 1).

3.2 Hunting and attacking iguanas: exploration phase

During this phase, the coati swarm is split into two sub-swarms, and the division scheme is defined in Eq. (3).

$$P = \begin{cases} P_1 = \{X_1, X_2, ..., X_{\mathbf{int}(\frac{n}{2})}\} \\ P_2 = \{X_{\mathbf{int}(\frac{n}{2})+1}, X_{\mathbf{int}(\frac{n}{2})+2}, ..., X_n\} \end{cases} \tag{3}$$

where **int**(·) is the floor function. COA divides the coati swarm into two roughly equal-sized sub-swarms sequentially, and each of these sub-swarms performs distinct operations.

The first coati sub-swarm ascends the tree and intimidates the iguanas. To depict this behavior, the search operator in Eq. (4) is utilized, considering the iguana's position as that of the best coati in the current iteration.

$$X_{i,j}^{t+1} = X_{i,j}^{t} + r \cdot (X_{best,j} - I \cdot X_{i,j}^{t}) \tag{4}$$

Once the iguana has descended to the ground, the second coati sub-swarm advances toward it, as described by Eq. (5)

$$\begin{aligned} X_{rand,j} =& r \cdot (UB_j - LB_j) + LB_j \\ X_{i,j}^{t+1} =& \begin{cases} X_{i,j}^{t} + r \cdot (X_{rand,j} - I \cdot X_{i,j}^{t}), \ if \ f(X_{rand}) < f(X_i^t) \\ X_{i,j}^{t} + r \cdot (X_{i,j}^{t} - X_{rand,j}), \ otherwise \end{cases} \end{aligned} \tag{5}$$

where r is a random number within the interval [0, 1], I is randomly selected from the set $\{1, 2\}$, $X_{i,j}^{best}$ is the best solution in the current iteration, and X_{rand} represents the position after the iguana has descended to the ground.

3.3 Escaping from predators: exploitation phase

When the coati swarm is under attack by predators, coatis swiftly retreat from their current location and relocate to the nearest safe place. This behavior is described by Eq. (6).

$$X_{i,j}^{t+1} = X_{i,j}^{t} + \theta \cdot (r \cdot (ub_j - lb_j) + lb_j) \tag{6}$$

where θ is a random number within the range [-1, 1], and $ub_j = \frac{UB_j}{t}, lb_j = \frac{LB_j}{t}$ are two adaptive parameters that vary with the number of iteration t.

3.4 Selection mechanism

COA employs a greedy selection mechanism to determine the surviving coati individual. Equation (7) illustrates this selection scheme in the context of a minimization problem.

$$X_i^{t+1} = \begin{cases} X_i^{t+1}, \ if \ f(X_i^{t+1}) < f(X_i^t) \\ X_i^t, \ otherwise \end{cases} \tag{7}$$

In summary, the pseudocode of COA is shown in Alg. 1.

4 Cooperative COA and its binary versions

This section introduces our proposed cooperative COA (CCOA) and its binary versions designed for feature selection tasks and knapsack problems. We commence with an overview of CCOA, as depicted in Fig. 3

Our proposed CCOA inherits the main skeleton of COA while making adjustments to certain search operators. Following the initialization of the coati swarm initialization, CCOA enters the exploration phase. During this phase, the coati swarm is sorted based on fitness value. If a coati individual X_i possesses a fitness value higher than that of 50% coati individuals, it uses Eq. (4) to generate offspring individuals. Otherwise, Eq. (8) is adopted.

Algorithm 1 COA [44]

Require: Population size: N, Dimension: D, Maximum iteration: T_{max}
Ensure: Optimum: X_{best}
1: Initialize coati swarm P by Eq. (2)
2: $t = 1$
3: **while** $t \leq T_{max}$ **do**
4: Separate coati swarm P into two equal-sized sub-swarm P_1 and P_2 by Eq. (3)
5: ► **Exploration phase: Hunting and attacking iguanas**
6: Implement the search operator to P_1 with Eq. (4)
7: Implement the search operator to P_2 with Eq. (5)
8: Survive better coati individuals with selection operator by Eq. (7) % Selection
9: ► **Exploitation phase: Escaping from predators**
10: Implement the search operator to coati swarm P with Eq. (6)
11: Survive better coati individuals with selection operator by Eq. (7) % Selection
12: Update the optimum X_{best} found so far
13: **end while**
14: **return** X_{best}

Subsequently, the greedy selection mechanism, as described in Eq. (7), determines the survival of the individual from each pair of parent and offspring individuals, favoring the one with the superior fitness value.

Following the exploration phase, CCOA proceeds to the exploitation phase. During this stage, the exploitative search is executed using Eq. (9). If the count variable i surpasses the population size N, the exploitation phase concludes. The cycle of the exploration and exploitation phases continues until the global optimum is found or computational resources are depleted.

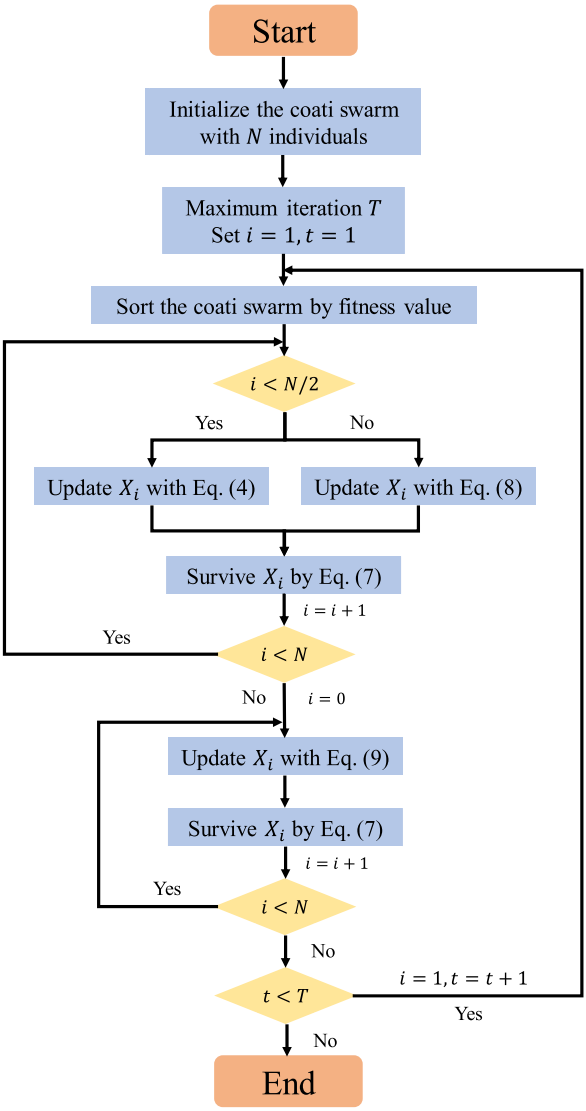
With this framework in mind, we introduce a variety of novel strategies incorporated into CCOA. Additionally, we provide binary adaptations of CCOA to address specific problem instances.

4.1 Fitness-based division and cooperative mechanism

In the original COA, the coati swarm is divided into two approximately equal-sized sub-swarms sequentially, with the first sub-swarm focusing on tree climbing and iguana scaring, while the second sub-swarm engages in iguana chasing after they fall to the ground. In our proposed CCOA, we advocate for a fitness-based division to replace the previous sequential division. In this new scheme, coati individuals with top-50% fitness values are assigned to the first sub-swarm, while the remaining individuals populate the second sub-swarm. The benefit of this division is that the individual with better fitness can exploit a better offspring individual around its position using Eq. (4), while the individual with worse fitness is driven to approach a better place using the cooperative search operator. This division of responsibilities can accelerate the optimization convergence since the search preference is specifically designed. Moreover, we simplify the search operator for the second sub-swarm of coatis and introduce a cooperative mechanism. These adjustments aim to enhance the collaborative behavior of the coati swarm during optimization tasks, fostering more efficient and effective optimization processes.

Our initial focus centers on the search operator in Eq. (5), where a randomly generated position corresponds to the fallen iguana's location. It is worth noting that this strategy consumes an additional fitness evaluation to determine the fitness value of this solution, while the purpose of this design is to introduce randomness and uncertainty to the search

Fig. 3 The flowchart of CCOA



operator, it is computationally expensive and unnecessary. Thus, we replace Eq. (5) from the original COA with Eq. (8).

$$X_{i,j}^{t+1} = X_{i,j}^{t} + r \cdot (X_{P_1,j}^{t} - X_{i,j}^{t}) \tag{8}$$

where X_{P_1} is a random coati individual sampled from the other coati sub-swarm P_1. A visual demonstration is shown in Fig. 4.

In this example, the coati individuals selected from P_1 and P_2 are represented by the blue and green points enclosed within the red-dotted cycle, respectively. Since this search operator is applied to the P_2, the two targeted coati individuals originate from different coati sub-swarms. The blue line with the arrow represents the direction of movement for the selected coati individual, and the yellow line denotes the corresponding movement vector.

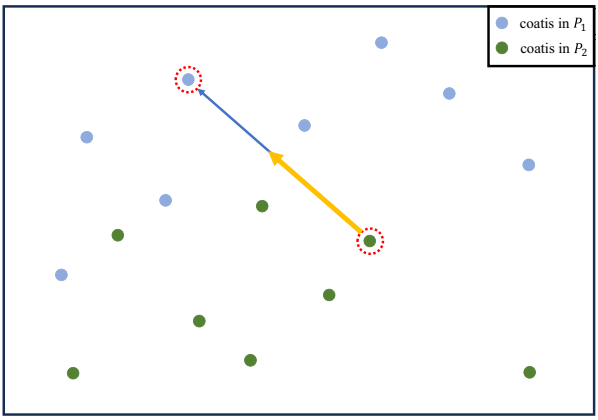


Fig. 4 A visual demonstration of the cooperative search operator

This search operator shifts coati individuals from P_2 toward P_1, aiming to potentially achieve improved fitness values. This mechanism simulates the inter-class communication and interaction between the two coati sub-swarms, effectively embodying the swarm intelligence within coatis.

4.2 Optional base vector strategy

In the exploitation phase, the search around the individual is primarily focused on navigating through promising regions to refine the solution space for incremental improvements. Building upon this concept, we suggest incorporating an optional base vector strategy into Eq. (6). This strategy seeks to bolster the exploitative capacity of the optimizer by placing greater emphasis on the exploitation of valuable regions within the search domain.

$$X_{i,j}^{t+1} = X_{base,j}^{t} + \theta \cdot (r \cdot (ub_j - lb_j) + lb_j) \tag{9}$$

where X_{base}^{t} is randomly selected from the set $\{X_{best}, X_{2^{nd}best}, X_{3^{rd}best}, X_i\}$. As the statement in the proximate optimality principle (POP) [57], individuals who are close to each other in the search space tend to have similar quality or fitness. On the other words, the exploitative search around elite individuals may be more promising and potential. Thus, we frequently reuse the elites' information with the optional base vector strategy. This strategy enhances both the exploitative capability and the population diversity during the optimization process. It effectively speeds up the convergence of optimization by involving elite individuals in the exploitation operator.

In summary, the pseudocode of CCOA is presented in Alg. 2.

The main differences between COA and CCOA are demonstrated in Algorithm 2 Lines 4, 7, and 10, which corresponds to the fitness-based division, cooperative search, and optional base vector strategy, respectively. These simple yet effective modifications in CCOA are expected to accelerate the optimization convergence and improve the optimization accuracy significantly.

Algorithm 2 CCOA

Require: Population size: N, Dimension: D, Maximum iteration: T_{max}
Ensure: Optimum: X_{best}
Initialize coati swarm P by Eq. (2)
$t = 1$
while $t \leq T_{max}$ **do**
  Separate coati swarm P into two equal-sized sub-swarm P_1 and P_2 based on fitness
  ► **Exploration phase: Hunting and attacking iguanas**
  Implement the search operator to P_1 with Eq. (4)
  Implement the search operator to P_2 with Eq. (8)
  Survive better coati individuals with selection operator by Eq. (7) % Selection
  ► **Exploitation phase: Escaping from predators**
  Implement the search operator to coati swarm P with Eq. (9)
  Survive better coati individuals with selection operator by Eq. (7) % Selection
  Update the optimum X_{best} found so far
end while
return X_{best}

4.3 Two binary versions of CCOA: SCCOA and VCCOA

The incorporation of transfer functions is a well-established and effective approach for mapping the continuous space into the discrete one. The essence of this transformation process is expressed in Eq. (10).

$$X_{i,j} = \begin{cases} 0, & if\ r < TF(X_{i,j}) \\ 1, & otherwise \end{cases} \tag{10}$$

where r is a random number within the range [0, 1] and $TF(\cdot)$ denotes a specific transfer function. Moreover, two common categories of transfer functions are the S-shaped Sigmoid transfer functions and the V-shaped Tanh transfer functions, which are formulated in Eq. (11).

$$\begin{aligned} S1(x) &= \frac{1}{1+e^{-2x}}, \quad S2(x) = \frac{1}{1+e^{-x}} \\ S3(x) &= \frac{1}{1+e^{-x/2}}, \quad S4(x) = \frac{1}{1+e^{-x/3}} \\ V1(x) &= |\tanh(x)|, \quad V2(x) = \left| erf(\frac{\sqrt{\pi}}{2}x) \right| \\ V3(x) &= \left| \frac{x}{\sqrt{1+x^2}} \right|, \quad V4(x) = \left| \frac{2}{\pi} \arctan(\frac{\pi}{2}x) \right| \end{aligned} \tag{11}$$

Each category comprises four variants, and Fig. 5 provides a visual representation of these transfer functions.

In this research, we integrate the transfer functions $S2(\cdot)$ and $V1(\cdot)$ as suggested in [55, 58] into the framework of CCOA to promote the development of SCCOA and VCCOA. These specialized adaptations are customized to tackle wrapper-based feature selection tasks and 0/1 knapsack problems.

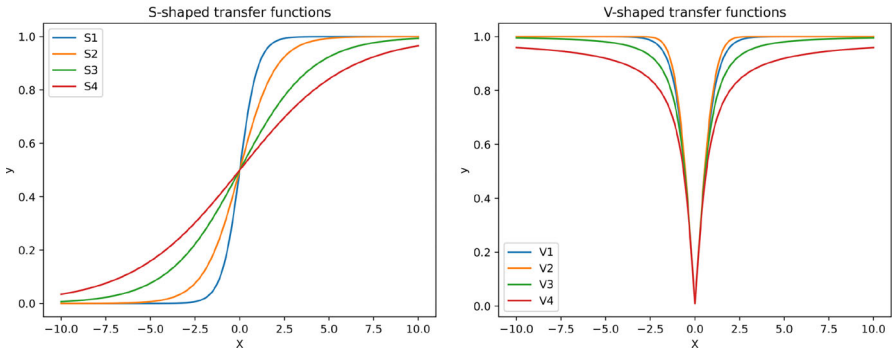


Fig. 5 Four S-shaped Sigmoid transfer functions and four V-shaped Tanh transfer functions

4.4 Computational complexity of CCOA

For the sake of simplicity, we analyze components consisting of CCOA independently. We assume that the population size is represented as N, the dimension of the problem is denoted as D, and the maximum iteration is defined as T.

- coati swarm initialization: $O(N \cdot D)$.
- coati swarm sort: $O(N \cdot \log N)$.
- two search operators in the exploration phase for the whole coati swarm: $O(N \cdot D)$.
- coati swarm selection: $O(N)$.
- search operator in the exploitation phase for the whole coati swarm: $O(N \cdot D)$.
- coati swarm selection: $O(N)$.

In summary, the computational complexity of CCOA can be calculated by Eq. (12).

$$\begin{aligned} & O(N \cdot D + T \cdot (N \cdot \log N + N \cdot D + N + N \cdot \log N + N \cdot D + N)) \\ & \quad = O(N \cdot D + T \cdot (2 \cdot (N \cdot \log N + N \cdot D + N)) \\ & \quad := O(N \cdot D + T \cdot (N \cdot \log N + N \cdot D)) \end{aligned} \tag{12}$$

Indeed, CCOA introduces a sorting operator, which adds some computational overhead compared to the original COA. However, the improvements in performance observed in numerical experiments justify the slightly higher computational complexity of CCOA, making it an acceptable trade-off for better optimization results.

5 Numerical experiments

This section provides an overview of the numerical experiments conducted in this study. Specifically, Sect. 5.1 outlines the details of the experimental settings, while Sect. 5.2 presents and analyzes the results obtained from these experiments.

Table 2 Summary of the CEC2020 benchmark functions: Uni.=Unimodal function, Multi.=Multimodal function, Hybrid.=Hybrid function, Comp.=Composition function

Fun	Description	Feature	Optimum
f_1	Shifted and Rotated Bent Cigar Function	Uni	100
f_2	Shifted and Rotated Schwefel's function	Multi.	1100
f_3	Shifted and Rotated Lunacek bi-Rastrigin function		700
f_4	Expanded Rosenbrock's plus Griewangk's function		1900
f_5	Hybrid function 1 (N = 3)	Hybrid.	1700
f_6	Hybrid function 2 (N = 4)		1600
f_7	Hybrid function 3 (N = 5)		2100
f_8	Composition function 1 (N = 3)	Comp.	2200
f_9	Composition function 2 (N = 4)		2400
f_{10}	Composition function 3 (N = 5)		2500

5.1 Experimental settings

5.1.1 Experimental environments and implementation

All numerical experiments described in this paper were performed using Python 3.11. The computations were conducted on a Lenovo Legion R9000P device running on Windows 11. The hardware is equipped with an AMD Ryzen 7 5800 H with Radeon Graphics, clocked at 3.20 GHz, and is supported by 16GB of RAM.

For the purpose of comparison, all the meta-heuristic algorithms (MAs) were sourced from the MEALPY library [59]. The CEC2020 benchmark functions utilized in the experiments were accessed from the OpFuNu library [60], while the six engineering optimization problems were obtained from the ENOPPY library [61]. Datasets integral to the feature selection tasks were obtained from the MAFESE library [62], and 18 knapsack problems employed in the study are detailed described in [55] and can be downloaded from https://people.sc.fsu.edu/~jburkardt/datasets/knapsack_01/knapsack_01.html.

5.1.2 Benchmark functions

This research encompasses four benchmark suites: 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions [63] and six engineering problems are employed to evaluate the performance of CCOA, the detailed description and visualization of engineering problems can be found in [64, 65]. Thirteen feature selection datasets [66] and 18 standard knapsack problems [55] are adopted to evaluate the performance of SCCOA and VCCOA. Tables 2, 3, 4, and 5 outline the essential information of these test problems.

5.1.3 Competitor algorithms and parameters

The primary objective of the first experiment is to evaluate the performance of CCOA across CEC2020 benchmark functions and engineering problems. We conduct a comparative analysis with nine other MAs including particle swarm optimization (PSO) [67], differential evolution (DE) [68], artificial ecosystem-based optimization (AEO) [69], bald eagle search (BES) [70], Pareto-like sequential sampling (PSS) [71], INFO [72], pelican optimization

Table 3 Summary of six engineering optimization problems

Name	Abbr	Dim	# of constraints
Compression Spring Problem	CSP	4	4
Three Bar Truss Problem	TBTD	2	3
Gear Train Problem	GTD	4	0
Tubular Column Problem	TCD	2	6
Corrugated Bulkhead Problem	CBHD	4	6
Reinforced Concrete Beam Problem	RCB	3	2

Table 4 Summary of 13 classification datasets

Datasets	# of instances	# of attributes	# of classes
BreastCancer	699	10	2
BreastEW	568	30	2
CongressEW	434	16	2
Glass	214	10	6
Heart	270	13	2
HeartEW	270	13	2
Horse	368	28	2
Ionosphere	351	34	2
Sonar	208	60	2
Soybean-small	47	35	4
SpectEW	267	22	2
wdbc	569	30	2
Wine	178	13	3

algorithm (POA) [73], osprey optimization algorithm (OOA) [74], and the original COA [44].

In this evaluation, the population size for all algorithms is fixed at 100, and the maximum number of function evaluations (FEs) for CEC2020 functions and engineering problems are $1000 \times D$ (D=dimension) and 20,000, respectively. To alleviate the impact of randomness during optimization, each MA is independently run 30 trial times. The specific parameter settings for the compared methods are provided in Table 6.

Furthermore, it is important to highlight that none of the competitor algorithms were originally designed to handle constrained engineering problems. To address this limitation, we equip them with a static penalty function [75], defined as shown in Eq. (13).

$$F(X_i) = f(X_i) + w \cdot \sum_{i=1}^{m} (\max(0, g_i(X_i))) \tag{13}$$

$F(\cdot)$ is the fitness function, while $f(\cdot)$ and $g_i(\cdot)$ are the objective function and constraint function, respectively. w is a constant, typically set to $10e^7$ by default.

The second experiment is designed to evaluate the performance of SCCOA and VCCOA in feature selection tasks and knapsack problems. We compare our proposal with six MA-based binary optimization approaches, including novel binary PSO (NBPSO) [76], binary Jaya (BJaya) [77], S-shaped and V-shaped whale optimization algorithm (SWOA and VWOA)

Table 5 Summary of 18 0/1 knapsack problems

Func	Dimension	x^*	$f(x^*)$
f_1	10	{0, 1, 1, 1, 0, 0, 0, 1, 1, 1}	295
f_2	20	{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1}	1024
f_3	4	{1, 1, 0, 1}	35
f_4	4	{0, 1, 0, 1}	23
f_5	15	{0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1}	481.0694
f_6	10	{0, 0, 1, 0, 1, 1, 1, 1, 1, 1}	52
f_7	7	{1, 0, 0, 1, 0, 0, 0}	107
f_8	23	{1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0}	9767
f_9	5	{1, 1, 1, 1, 0}	130
f_{10}	20	{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1}	1025
$KP01$	10	{1, 1, 1, 1, 0, 1, 0, 0, 0, 0}	165
$KP02$	5	{0, 1, 1, 1, 0}	26
$KP03$	6	{1, 1, 0, 0, 1, 0}	190
$KP04$	7	{1, 0, 0, 1, 0, 0, 0}	50
$KP05$	8	{1, 0, 1, 1, 1, 0, 1, 1}	104
$KP06$	7	{0, 1, 0, 1, 0, 0, 1}	169
$KP07$	15	{1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1}	1458
$KP08$	24	{1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1}	13,549,094

Table 6 Parameters of competitor algorithms for evaluating CCOA

EAs	Parameters	Value
PSO (1995)	Inertia factor w	1
	Acceleration coefficients c_1 and c_2	2.05
	Max. and min. speed	2, -2
DE (1996)	Mutation strategy	DE/cur-to-rand/1/bin
	Scaling factor F	0.8
	Crossover rate Cr	0.9
AEO (2020)	Parameter-free	
BES (2020)	Change controller α	2
	Acceleration coefficients c_1 and c_2	2
	Corner determination a	10
	Search cycle R	1.5
PSS (2021)	Acceptance rate α	3
	Sampling fashion	LHS
INFO (2022)	Constant number c and d	2 and 4
POA (2022)	Parameter-free	
OOA (2023)	Parameter-free	
COA (2023)	Parameter-free	
CCOA	Parameter-free	

Table 7 Parameters of competitor algorithms for evaluating CCOA

EAs	Parameters	Value
NBPSO (2017)	Constant i_m, i_p, and i_g	0.25, 0.25, and 0.5
BJaya (2018)	Parameter-free	
SWOA and VWOA (2018)	Parameter-free	
NBGOA (2022)	Constant C_{min} and C_{max}	0.00001 and 1
BHGSO-S (2022)	Constant P_{CR}, β_{min}, and β_{max}	0.8, 0.2, and 0.8
SCCOA and VCCOA	Parameter-free	

[78, 79], new binary grasshopper optimization algorithm (NBGOA) [80], and binary variants of a hunger games search optimization based on S-shaped transfer function (BHGSO-S) [81].

For a fair comparison, the population size for all competitor algorithms is set to 20, and the maximum number of iterations is $10 \times D$, where D is the number of features in the feature selection task or the number of items in the knapsack problem. Similarly, each approach is independently run 30 times. The detailed parameter settings for the competitor algorithms are listed in Table 7.

Since both the feature selection task and the knapsack problem are formulated as maximization problems, we can unify the evaluation framework by transforming the objective function using Eq. (14). This conversion ensures a consistent basis for evaluating and comparing the performance of the optimization algorithms across these diverse problem domains.

$$\max f(x) \Leftrightarrow \min -f(x) \tag{14}$$

Especially for the feature selection task, we adopt KNN ($k = 5$) with fivefold cross-validation as the classifier, and the classification accuracy [45] serves as the objective function, defined as shown in Eq. (15).

$$fit = \frac{T_p + T_N}{T_p + T_N + F_p + F_N} \cdot 100\% \tag{15}$$

where T_P and T_N are the numbers of positive and negative instances that the classifier has correctly classified, and F_P and F_N are the numbers of positive and negative instances that the classifier has wrongly classified.

5.2 Experimental results

5.2.1 Performance on CEC2020 benchmark functions

Tables 8, 9, 10, and 11 summarize the experimental results and statistical analysis for optimization across 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions, comparing ten MAs. To determine the significant difference between our proposed CCOA and other competitor algorithms, we first employ the Mann–Whitney U test for each pair of algorithms, and then, the Holm multiple comparison test [82] is employed to correct the p values obtained from the Mann–Whitney U test. The markers $+$, $\approx$, and $-$ denote that our proposed CCOA is significantly better, shows no significance, or is significantly worse than the respective compared method. The best value is in bold. Due to the limitation of space, we provide convergence curves of representative functions (i.e., f_1: Unimodal function, f_4: Multimodal function, f_6: Hybrid function, f_{10}: Composition function) in Fig. 6.

Table 8 Experimental and statistical results on 10-D CEC2020 benchmark functions

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_1										
Mean	4.72e+09 +	7.64e+08 +	1.19e+08 +	1.16e+09 +	7.94e+07 +	1.08e+06 ≈	6.77e+08 +	2.04e+08 +	1.39e+09 +	**4.03e+05**
Std	1.97e+09	1.61e+08	1.41e+08	9.13e+08	4.48e+07	3.41e+06	7.16e+08	2.44e+08	6.57e+08	**2.32e+05**
f_2										
Mean	3.52e+11 +	7.50e+10 +	9.04e+09 +	7.81e+10 +	6.14e+09 +	6.96e+08 ≈	3.07e+10 +	1.05e+10 +	1.39e+11 +	**1.42e+07**
Std	2.22e+11	2.17e+10	8.35e+09	5.50e+10	2.90e+09	1.96e+09	4.02e+10	9.18e+09	5.65e+10	**7.39e+06**
f_3										
Mean	1.25e+11 +	3.49e+10 +	3.59e+09 +	3.37e+10 +	1.55e+09 +	1.11e+08 +	1.25e+10 +	5.12e+09 +	4.11e+10 +	**9.73e+06**
Std	9.21e+10	9.08e+09	4.42e+09	2.92e+10	1.30e+09	1.32e+08	1.92e+10	6.20e+09	1.99e+10	**6.59e+06**
f_4										
Mean	3.31e+03 +	1.93e+03 +	1.91e+03 +	1.92e+03 +	1.91e+03 +	1.91e+03 +	1.92e+03 +	1.91e+03 +	1.95e+03 +	**1.90e+03**
Std	2.86e+03	1.66e+01	6.97e+00	3.18e+01	1.10e+00	4.63e+00	5.73e+01	1.53e+01	7.27e+01	**5.62e−01**
f_5										
Mean	7.77e+05 +	1.23e+04 ≈	1.68e+04 ≈	4.88e+04 +	3.16e+04 +	2.81e+04 +	**1.18e+04** ≈	2.12e+04 +	4.48e+04 +	1.56e+04
Std	1.05e+06	3.46e+03	7.92e+03	9.93e+04	1.30e+04	1.88e+04	**7.51e+03**	1.27e+04	2.78e+04	8.02e+03
f_6										
Mean	1.58e+05 +	2.88e+03 ≈	2.38e+03 ≈	3.31e+03 +	3.71e+03 +	6.77e+03 +	2.28e+03 ≈	**2.11e+03** ≈	3.71e+03 +	3.04e+03
Std	2.61e+05	3.93e+02	5.16e+02	1.83e+03	2.14e+03	3.59e+03	3.95e+02	**3.06e+02**	1.13e+03	3.27e+03

Table 8 continued

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_7										
Mean	1.20e+06 +	5.72e+04 +	1.28e+04 ≈	1.73e+04 ≈	4.38e+04 +	2.66e+04 ≈	**9.77e+03** −	1.39e+04 ≈	3.38e+04 +	1.75e+04
Std	2.11e+06	2.27e+04	7.75e+03	1.23e+04	3.14e+04	1.76e+04	**7.08e+03**	9.18e+03	2.31e+04	8.40e+03
f_8										
Mean	2.33e+03 +	2.32e+03 +	2.31e+03 +	2.31e+03 +	2.31e+03 +	2.31e+03 +	2.31e+03 +	2.31e+03 +	2.32e+03 +	**2.30e+03**
Std	1.29e+01	1.48e+00	3.39e+00	3.33e+00	1.48e+00	7.42e+00	4.98e+00	3.38e+00	4.67e+00	**1.40e+00**
f_9										
Mean	5.19e+03 +	3.82e+03 +	3.04e+03 +	4.05e+03 +	2.93e+03 +	2.71e+03 +	3.06e+03 +	3.11e+03 +	4.42e+03 +	**2.61e+03**
Std	8.01e+02	1.50e+02	1.99e+02	7.49e+02	1.31e+02	1.51e+02	3.15e+02	3.28e+02	4.01e+02	**6.04e+01**
f_{10}										
Mean	3.13e+03 +	3.09e+03 +	3.07e+03 +	3.13e+03 +	3.01e+03 +	3.04e+03 +	3.03e+03 +	3.06e+03 +	3.10e+03 +	**3.00e+03**
Std	1.40e+02	3.37e+01	5.37e+01	5.78e+01	2.67e+01	4.35e+01	3.59e+01	5.33e+01	3.50e+01	**2.91e+01**
+/≈/− summary	10/0/0	8/2/0	7/3/0	9/1/0	10/0/0	7/3/0	7/2/1	8/2/0	10/0/0	

Bold values indicate the best results among all the compared methods, such as the average optimal solution (Mean) found in multiple runs, and their standard deviation (std)
f_1: Unimodal function; $f_2 - f_4$: Multimodal functions; $f_5 - f_7$: Hybrid functions; $f_8 - f_{10}$: Composition functions; mean and std: the mean and the standard deviation of 30 trial runs

Table 9 Experimental and statistical results on 30-D CEC2020 benchmark functions

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_1										
Mean	4.42e+10 +	3.87e+10 +	7.20e+09 +	3.50e+10 +	2.77e+09 +	2.93e+09 +	1.49e+10 +	2.45e+10 +	3.18e+10 +	**1.19e+06**
Std	1.53e+10	5.06e+09	2.16e+09	6.75e+09	1.18e+09	2.00e+09	6.14e+09	6.46e+09	6.50e+09	**3.15e+05**
f_2										
Mean	4.73e+12 +	3.79e+12 +	8.91e+11 +	4.03e+12 +	2.98e+11 +	2.91e+11 +	1.51e+12 +	2.74e+12 +	3.77e+12 +	**1.25e+08**
Std	1.89e+12	3.83e+11	3.63e+11	9.16e+11	1.23e+11	2.44e+11	4.90e+11	6.88e+11	6.98e+11	**2.95e+07**
f_3										
Mean	1.40e+12 +	1.24e+12 +	2.15e+11 +	1.21e+12 +	1.18e+11 +	1.14e+11 +	4.49e+11 +	8.55e+11 +	1.10e+12 +	**4.95e+07**
Std	6.43e+11	1.56e+11	6.58e+10	2.56e+11	3.84e+10	8.10e+10	1.33e+11	2.14e+11	2.08e+11	**1.37e+07**
f_4										
Mean	7.07e+05 +	9.84e+04 +	8.85e+03 +	2.48e+05 +	2.01e+03 +	2.64e+03 +	1.06e+04 +	9.91e+04 +	1.61e+05 +	**1.91e+03**
Std	7.91e+05	3.84e+04	1.00e+04	2.54e+05	1.04e+02	1.06e+03	6.02e+03	9.20e+04	1.33e+05	**3.57e+00**
f_5										
Mean	5.74e+07 +	1.51e+07 +	5.06e+06 +	1.66e+07 +	3.42e+06 +	7.35e+05 ≈	9.81e+05 +	8.48e+06 +	1.37e+07 +	**4.42e+05**
Std	7.35e+07	5.77e+06	5.57e+06	1.29e+07	1.61e+06	5.03e+05	8.90e+05	8.94e+06	6.46e+06	**2.31e+05**
f_6										
Mean	1.53e+08 +	2.02e+05 +	9.62e+04 +	1.71e+07 +	1.89e+05 +	2.40e+04 ≈	5.37e+04 +	1.63e+05 +	1.60e+06 +	**2.31e+04**
Std	3.05e+08	4.71e+05	1.16e+05	3.55e+07	1.59e+05	9.01e+03	5.37e+04	3.71e+05	1.67e+06	**1.15e+04**
f_7										
Mean	3.41e+08 +	3.56e+07 +	1.36e+07 +	5.28e+07 +	5.45e+06 +	1.10e+06 +	1.76e+06 +	2.86e+07 +	5.52e+07 +	**5.30e+05**
Std	2.44e+08	1.06e+07	1.17e+07	4.56e+07	3.41e+06	8.04e+05	1.70e+06	2.98e+07	3.64e+07	**4.21e+05**

Table 9 continued

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_8										
Mean	3.56e+03 +	2.65e+03 +	2.91e+03 +	3.13e+03 +	2.46e+03 +	2.82e+03 +	2.72e+03 +	3.00e+03 +	3.01e+03 +	**2.44e+03**
Std	5.78e+02	3.37e+01	2.30e+02	4.92e+02	2.32e+01	2.12e+02	1.59e+02	2.50e+02	2.32e+02	**2.63e+01**
f_9										
Mean	2.68e+04 +	1.30e+04 +	8.46e+03 +	2.66e+04 +	6.73e+03 +	7.29e+03 +	1.74e+04 +	2.05e+04 +	1.55e+04 +	**2.69e+03**
Std	8.62e+03	6.30e+02	9.54e+02	4.83e+03	1.21e+03	3.31e+03	3.56e+03	3.75e+03	3.64e+03	**3.89e+01**
f_{10}										
Mean	6.58e+03 +	5.11e+03 +	3.53e+03 +	5.30e+03 +	3.12e+03 +	3.09e+03 +	3.59e+03 +	4.44e+03 +	4.84e+03 +	**2.94e+03**
Std	2.08e+03	4.66e+02	1.73e+02	5.82e+02	8.35e+01	7.42e+01	3.30e+02	4.77e+02	5.90e+02	**2.49e+01**
+/≈/− summary	10/0/0	10/0/0	10/0/0	10/0/0	10/0/0	8/2/0	10/0/0	10/0/0	10/0/0	

Table 10 Experimental and statistical results on 50-D CEC2020 benchmark functions

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_1										
Mean	1.06e+11 +	1.07e+11 +	2.24e+10 +	9.00e+10 +	1.43e+10 +	1.27e+10 +	4.57e+10 +	7.19e+10 +	8.79e+10 +	**1.73e+06**
Std	2.51e+10	1.04e+10	4.87e+09	1.21e+10	2.46e+09	4.87e+09	1.05e+10	9.46e+09	1.28e+10	**3.24e+05**
f_2										
Mean	1.14e+13 +	1.15e+13 +	2.48e+12 +	9.67e+12 +	1.61e+12 +	1.16e+12 +	4.63e+12 +	8.13e+12 +	9.96e+12 +	**1.99e+08**
Std	2.81e+12	1.39e+12	5.73e+11	1.73e+12	2.42e+11	4.81e+11	1.19e+12	9.81e+11	1.46e+12	**4.29e+07**
f_3										
Mean	4.31e+12 +	4.13e+12 +	8.26e+11 +	3.10e+12 +	5.20e+11 +	3.97e+11 +	1.38e+12 +	2.69e+12 +	3.34e+12 +	**7.40e+07**
Std	1.16e+12	4.60e+11	1.99e+11	6.26e+11	8.47e+10	1.80e+11	3.25e+11	4.35e+11	4.12e+11	**1.30e+07**
f_4										
Mean	1.94e+06 +	1.40e+06 +	5.59e+04 +	1.28e+06 +	6.34e+03 +	6.02e+03 +	8.36e+04 +	6.23e+05 +	1.25e+06 +	**1.95e+03**
Std	1.26e+06	6.32e+05	5.80e+04	5.88e+05	4.78e+03	4.68e+03	6.04e+04	3.66e+05	5.10e+05	**9.73e+00**
f_5										
Mean	2.09e+08 +	6.23e+07 +	2.18e+07 +	1.08e+08 +	1.96e+07 +	4.31e+06 +	7.54e+06 +	4.46e+07 +	5.63e+07 +	**1.14e+06**
Std	1.27e+08	2.18e+07	1.13e+07	6.15e+07	7.22e+06	2.73e+06	9.47e+06	4.08e+07	2.10e+07	**7.44e+05**

Table 10 continued

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_6										
Mean	4.30e+09 +	9.22e+07 +	7.34e+06 +	1.59e+09 +	3.27e+06 +	1.30e+06 +	5.74e+07 +	1.52e+08 +	6.36e+08 +	**1.46e+04**
Std	3.52e+09	3.22e+07	8.34e+06	2.20e+09	2.46e+06	5.04e+06	1.42e+08	1.74e+08	4.09e+08	**6.53e+03**
f_7										
Mean	3.61e+09 +	7.53e+08 +	1.46e+08 +	2.27e+09 +	8.50e+07 +	7.24e+06 +	4.35e+07 +	7.86e+08 +	9.33e+08 +	**2.38e+06**
Std	2.59e+09	1.81e+08	1.10e+08	1.58e+09	6.24e+07	5.51e+06	3.69e+07	5.07e+08	4.07e+08	**1.39e+06**
f_8										
mean	7.70e+03 +	3.19e+03 +	6.30e+03 +	8.81e+03 +	**2.73e+03** ≈	6.16e+03 +	5.51e+03 +	7.00e+03 +	5.44e+03 +	3.02e+03
std	3.20e+03	1.06e+02	2.33e+03	2.49e+03	**4.12e+01**	1.69e+03	1.97e+03	1.69e+03	1.15e+03	9.66e+02
f_9										
Mean	5.29e+04 +	2.74e+04 +	1.85e+04 +	6.30e+04 +	1.51e+04 +	2.36e+04 +	4.28e+04 +	5.45e+04 +	5.49e+04 +	**2.98e+03**
Std	1.41e+04	1.91e+03	5.36e+03	4.95e+03	1.04e+03	9.94e+03	5.94e+03	3.83e+03	8.19e+03	**4.08e+02**
f_{10}										
Mean	1.93e+04 +	1.23e+04 +	7.03e+03 +	1.76e+04 +	5.45e+03 +	4.49e+03 +	7.87e+03 +	1.37e+04 +	1.58e+04 +	**3.56e+03**
Std	7.12e+03	1.78e+03	7.70e+02	3.37e+03	4.78e+02	4.61e+02	1.28e+03	2.48e+03	2.41e+03	**1.15e+02**
+/≈/− summary	10/0/0	10/0/0	10/0/0	10/0/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0	

Table 11 Experimental and statistical results on 100-D CEC2020 benchmark functions

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_1										
Mean	3.28e+11 +	3.33e+11 +	6.96e+10 +	2.47e+11 +	8.30e+10 +	4.72e+10 +	1.28e+11 +	2.14e+11 +	2.46e+11 +	**2.27e+06**
Std	4.54e+10	2.41e+10	1.35e+10	1.94e+10	8.30e+09	1.42e+10	1.77e+10	1.50e+10	1.62e+10	**4.26e+05**
f_2										
Mean	3.19e+13 +	3.08e+13 +	6.93e+12 +	2.74e+13 +	8.45e+12 +	4.69e+12 +	1.36e+13 +	2.47e+13 +	2.77e+13 +	**2.22e+08**
Std	3.98e+12	2.17e+12	1.36e+12	2.33e+12	6.16e+11	1.36e+12	2.10e+12	1.86e+12	1.62e+12	**3.91e+07**
f_3										
Mean	1.20e+13 +	1.17e+13 +	2.36e+12 +	9.69e+12 +	2.91e+12 +	1.77e+12 +	4.68e+12 +	8.36e+12 +	9.49e+12 +	**8.26e+07**
Std	1.71e+12	8.57e+11	4.02e+11	7.00e+11	2.03e+11	4.85e+11	7.31e+11	5.51e+11	8.35e+11	**1.35e+07**
f_4										
Mean	9.32e+06 +	2.10e+07 +	1.41e+05 +	5.76e+06 +	1.55e+05 +	2.17e+04 +	2.47e+05 +	2.78e+06 +	4.93e+06 +	**2.21e+03**
Std	4.21e+06	6.64e+06	6.70e+04	2.12e+06	5.82e+04	1.36e+04	8.58e+04	8.17e+05	1.52e+06	**5.33e+01**
f_5										
Mean	1.18e+09 +	7.70e+08 +	1.76e+08 +	7.57e+08 +	2.25e+08 +	2.43e+07 +	7.08e+07 +	4.12e+08 +	5.00e+08 +	**5.14e+06**
Std	4.45e+08	1.04e+08	4.42e+07	2.39e+08	6.17e+07	1.10e+07	2.38e+07	1.54e+08	1.37e+08	**2.36e+06**
f_6										
Mean	2.77e+10 +	3.43e+09 +	8.17e+07 +	1.88e+10 +	1.30e+08 +	5.52e+07 +	1.61e+09 +	1.13e+10 +	1.17e+10 +	**4.05e+04**
Std	2.00e+10	1.24e+09	8.80e+07	8.68e+09	7.17e+07	8.84e+07	1.17e+09	5.26e+09	4.73e+09	**1.44e+04**

Table 11 continued

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
f_7										
Mean	2.11e+10 +	5.21e+09 +	8.66e+08 +	2.01e+10 +	8.68e+08 +	7.24e+07 +	9.31e+08 +	1.10e+10 +	7.91e+09 +	**2.73e+06**
Std	1.32e+10	1.48e+09	3.07e+08	6.26e+09	2.22e+08	4.82e+07	7.84e+08	3.53e+09	2.94e+09	**1.47e+06**
f_8										
Mean	2.54e+04 +	4.51e+03 ≈	1.86e+04 +	2.51e+04 +	**3.58e+03** −	1.38e+04 +	1.55e+04 +	2.08e+04 +	1.69e+04 +	5.26e+03
Std	8.22e+03	3.77e+02	5.17e+03	3.50e+03	**1.63e+02**	2.57e+03	5.14e+03	3.74e+03	4.03e+03	1.97e+03
f_9										
Mean	2.28e+05 +	1.16e+05 +	9.41e+04 +	1.78e+05 +	7.28e+04 +	8.26e+04 +	1.25e+05 +	1.65e+05 +	1.79e+05 +	**1.55e+04**
Std	4.26e+04	1.27e+04	1.98e+04	7.05e+03	1.18e+04	2.04e+04	1.25e+04	6.58e+03	4.60e+03	**1.46e+04**
f_{10}										
Mean	4.39e+04 +	4.19e+04 +	9.26e+03 +	3.22e+04 +	8.78e+03 +	6.21e+03 +	1.21e+04 +	2.52e+04 +	3.16e+04 +	**4.05e+03**
Std	8.89e+03	7.87e+03	1.05e+03	4.94e+03	6.32e+02	8.13e+02	1.89e+03	3.04e+03	4.83e+03	**1.30e+02**
+/≈/− summary	10/0/0	9/1/0	10/0/0	10/0/0	9/0/1	10/0/0	10/0/0	10/0/0	10/0/0	

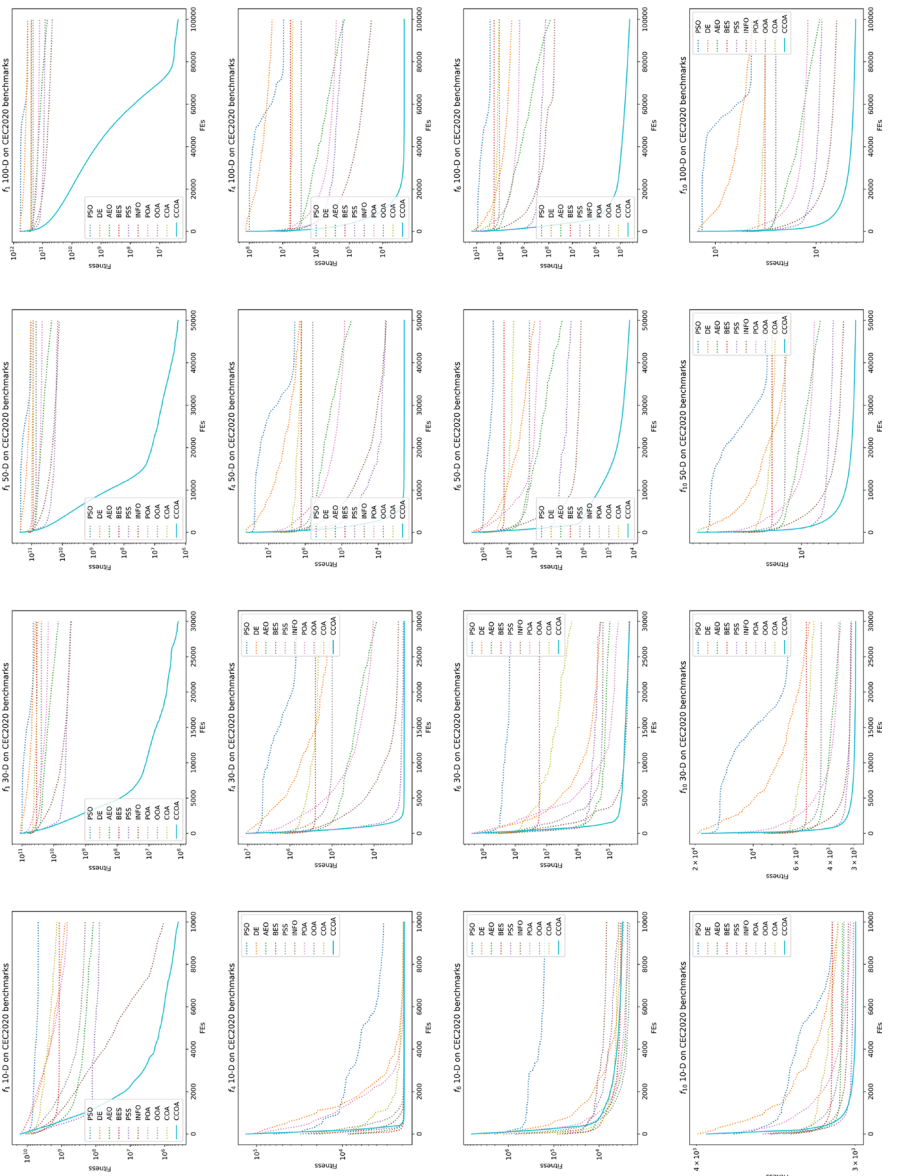


Fig. 6 Convergence curves of ten MAs on representative CEC2020 benchmark functions

The comprehensive evaluation of CCOA across the 10 various benchmark functions in the CEC2020 suite, spanning 10-D, 30-D, 50-D, and 100-D with a total of 40 benchmark functions, demonstrates that CCOA performs exceptionally well. Specifically, our proposed CCOA is only significantly worse than POA in 10-D f_7 and PSS in 100-D f_8. In most cases, CCOA is significantly better than the competitor algorithms. Notably, when compared to the original COA, CCOA shows significant improvements in all cases. These experiments provide strong evidence that the introduction of the fitness-based division, cooperative mechanism, and optional base vector strategy can significantly accelerate convergence and improve convergence accuracy.

In addition, convergence curves of representative functions in Fig. 6 clearly illustrate the excellent convergence speed and the remarkable capacity of CCOA to escape local optima. While many competitor algorithms stagnate or struggle to make progress, CCOA continues to approach convergence and find better solutions.

However, for 10-D $f_5 - f_7$ and 100-D f_7, these functions exhibit hybrid characteristics characterized by complex fitness landscapes and multimodal characteristics. In the context of the 10-D $f_5 - f_7$, algorithms such as DE, AEO, POA, and OOA are competitive with CCOA, Nevertheless, as the dimension of the problem increases, the superiority of CCOA over these competitor algorithms becomes more pronounced. Concurrently, in the case of the 100-D f_7 function, PSS emerges as the leading performer among the ten MAs considered. This observation leads us to infer that while CCOA excels in handling medium-dimensional hybrid functions, it may not be as effective in addressing hybrid functions with low or high dimensions.

5.2.2 Performance on engineering problems

Table 12 summarizes the experimental results of optimization across six engineering optimization problems, comparing ten EAs. Additionally, Fig. 7 provides the corresponding convergence curves for these problems.

These widely recognized engineering optimization problems serve as a valuable benchmark to evaluate the performance of CCOA in real-world applications. Among the six problems considered, CCOA emerges as the victor in the TBTD and RCB cases, demonstrating its effectiveness and efficiency. However, an examination of the convergence curves provided in Fig. 7 reveals that CCOA's performance, especially in the GTD and TCD cases, falls short of expectations. This observed performance variability can be attributed to the presence of the No Free Lunch Theorem. No Free Lunch Theorem [83] states that for every pair of optimization algorithms, the average performance remains identical across all possible problem instances. Therefore, if an algorithm performs better on a certain category of problems, it must inherently perform less effectively on the remaining problem types to maintain an overall consistent average performance. Thus, the determination of the structure and topology of well-performed problems for a specific MA is a potential research topic. Furthermore, the ongoing development of new MAs is meaningful.

5.2.3 Performance on feature selection tasks

For the feature selection task, we focus on two primary metrics: the classification accuracy of the classifier and the number of selected features. Table 13 summarizes the experimental results for these two metrics across 13 feature selection datasets. In addition, the rank information sorted by the average classification accuracy is also provided for reference.

Table 12 Experimental results on six engineering problems. Mean and std: the mean and the standard deviation of 30 trial runs; best and worst: the best and the worst final solution found among 30 trial runs

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
CSP										
Mean	6.0775e−03	**6.0761e−03**	**6.0761e−03**	**6.0761e−03**	6.2310e−03	6.2088e−03	**6.0761e−03**	6.0761e−03	6.1319e−03	6.0773e−03
Std	1.5168e−06	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	6.6584e−05	4.9744e−04	**0.0000e+00**	1.9186e−10	5.7358e−05	9.0284e−07
Best	**6.0761e−03**	**6.0761e−03**	**6.0761e−03**	**6.0761e−03**	6.1206e−03	**6.0761e−03**	**6.0761e−03**	**6.0761e−03**	6.0778e−03	6.0762e−03
Worst	6.0811e−03	**6.0761e−03**	**6.0761e−03**	**6.0761e−03**	6.3983e−03	8.1944e−03	**6.0761e−03**	**6.0761e−03**	6.3257e−03	6.0795e−03
TBTD										
Mean	2.6419e+02	2.6390e+02	2.6390e+02	2.6447e+02	2.6392e+02	2.6431e+02	2.6390e+02	2.6390e+02	2.6395e+02	**2.6390e+02**
Std	1.8476e−01	4.0728e−11	1.0714e−03	1.3830e+00	2.1333e−02	1.1909e+00	4.8009e−13	4.6300e−03	4.4176e−02	**1.4677e−14**
Best	2.6395e+02	**2.6390e+02**	**2.6390e+02**	**2.6390e+02**	**2.6390e+02**	**2.6390e+02**	**2.6390e+02**	**2.6390e+02**	**2.6390e+02**	**2.6390e+02**
Worst	2.6475e+02	**2.6390e+02**	**2.6390e+02**	2.7157e+02	2.6400e+02	2.7053e+02	**2.6390e+02**	2.6392e+02	2.6406e+02	**2.6390e+02**
GTD										
Mean	1.8409e−11	1.7176e−11	**0.0000e+00**	2.5852e−16	8.7187e−11	3.9217e−14	4.6480e−17	**0.0000e+00**	1.2664e−09	1.0621e−10
Std	4.0821e−11	2.2881e−11	**0.0000e+00**	1.1282e−15	1.6175e−10	5.3288e−14	9.5174e−17	**0.0000e+00**	2.0688e−09	3.6399e−10
Best	1.7883e−17	1.9768e−15	**0.0000e+00**	6.5204e−30	6.4672e−13	8.1302e−17	7.9018e−23	**0.0000e+00**	1.6841e−15	**0.0000e+00**
Worst	1.5576e−10	7.9739e−11	**0.0000e+00**	6.2673e−15	7.6443e−10	2.0882e−13	3.9763e−16	**0.0000e+00**	7.6997e−09	1.4760e−09
TCD										
Mean	3.0542e+01	3.0150e+01	3.0150e+01	**3.0150e+01**	3.0179e+01	3.0151e+01	3.0150e+01	3.0150e+01	3.0653e+01	3.0428e+01
Std	2.3954e−01	7.1466e−09	6.0611e−12	**1.0658e−14**	2.0633e−02	7.4814e−04	1.0040e−10	3.0954e−06	3.1695e−01	3.4094e−01
Best	3.0192e+01	**3.0150e+01**	**3.0150e+01**	**3.0150e+01**	**3.0150e+01**	**3.0150e+01**	**3.0150e+01**	**3.0150e+01**	3.0170e+01	3.0167e+01
Worst	3.1181e+01	**3.0150e+01**	**3.0150e+01**	**3.0150e+01**	3.0222e+01	3.0153e+01	**3.0150e+01**	**3.0150e+01**	3.1369e+01	3.1364e+01

Table 12 continued

Func.	PSO	DE	AEO	BES	PSS	INFO	POA	OOA	COA	CCOA
CBHD										
Mean	7.8378e+00	6.8482e+00	6.9846e+00	6.8617e+00	6.9361e+00	6.9112e+00	**6.8461e+00**	7.0855e+00	7.7918e+00	7.0335e+00
Std	7.1741e−01	2.2638e−03	9.0480e−02	3.2620e−02	3.0249e−02	2.0112e−01	**3.6986e−03**	2.0519e−01	3.7619e−01	1.5952e−01
Best	7.0410e+00	6.8448e+00	6.8465e+00	**6.8430e+00**	6.8771e+00	6.8433e+00	6.8431e+00	6.8487e+00	7.2704e+00	6.8485e+00
Worst	1.0352e+01	**6.8532e+00**	7.1372e+00	6.9819e+00	7.0059e+00	7.7278e+00	6.8586e+00	7.6234e+00	8.8080e+00	7.6444e+00
RCB										
Mean	1.6101e+02	1.6662e+02	1.6636e+02	1.6649e+02	1.6688e+02	1.6643e+02	1.6593e+02	1.6680e+02	1.6746e+02	**1.5954e+02**
Std	1.7280e+00	6.0548e−01	1.2671e+00	9.1381e−01	1.0672e−01	8.2091e−01	8.7079e−01	1.1679e+00	1.7349e+00	**1.6108e−01**
Best	1.5939e+02	1.6356e+02	1.6055e+02	1.6280e+02	1.6677e+02	1.6356e+02	1.6356e+02	1.6356e+02	1.6281e+02	**1.5936e+02**
Worst	1.6531e+02	1.6677e+02	1.6677e+02	1.6677e+02	1.6718e+02	1.6693e+02	1.6677e+02	1.7218e+02	1.7017e+02	**1.5993e+02**

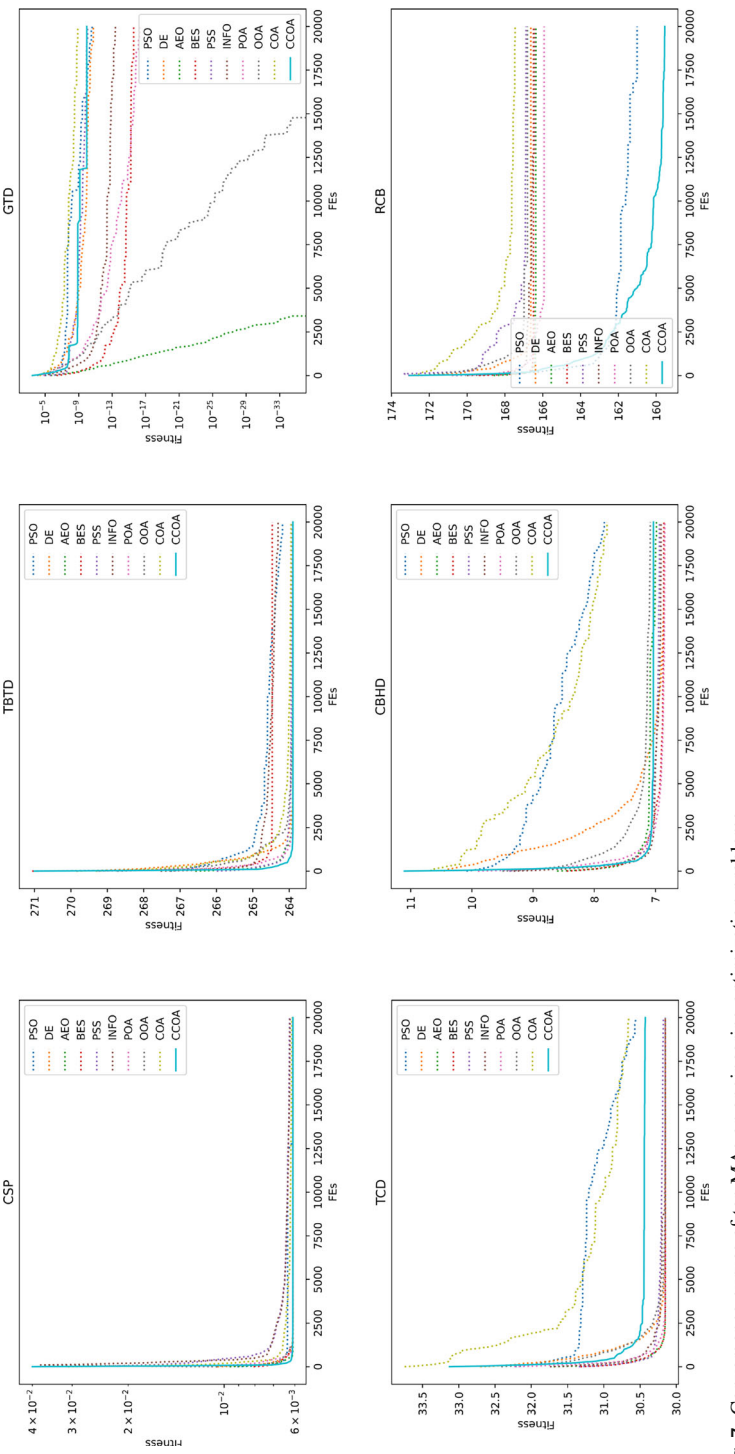


Fig. 7 Convergence curves of ten MAs on engineering optimization problems

Table 13 Experimental results on 13 feature selection datasets

Func.	NBPSO	BJaya	SWOA	VWOA	NBGOA	BHGSO-S	SCCOA	VCCOA
BreastCancer								
Mean accuracy	96.53	96.72	96.58	96.50	96.43	96.67	**96.80**	96.77
Mean # of features	5.97	5.93	5.77	**5.07**	5.97	5.90	5.73	5.63
BreastEW								
Mean accuracy	94.25	94.64	94.49	94.18	94.26	94.68	94.79	**94.88**
Mean # of features	15.53	16.10	14.30	**7.90**	15.20	14.77	14.00	12.17
CongressEW								
Mean accuracy	96.32	96.78	96.60	96.48	96.16	96.71	**96.87**	96.84
Mean # of features	7.10	6.70	6.23	**4.53**	6.90	6.10	7.33	5.40
Glass								
Mean accuracy	86.25	**86.32**	86.29	85.73	86.17	86.32	**86.32**	86.20
Mean # of features	7.43	7.80	7.93	**3.67**	7.23	7.80	7.37	5.93
Heart								
Mean accuracy	81.33	82.73	82.41	81.63	80.68	82.73	**83.53**	83.43
Mean # of features	5.60	6.23	6.20	**4.53**	5.47	5.60	5.33	5.67
HeartEW								
Mean accuracy	81.36	83.51	82.32	80.94	80.94	82.84	84.02	**84.25**
Mean # of features	5.83	5.80	6.13	**4.00**	5.67	5.90	5.57	5.13
Horse								
Mean accuracy	91.65	**100.00**	**100.00**	**100.00**	90.91	**100.00**	99.45	**100.00**
Mean # of features	8.97	2.87	2.60	**1.83**	10.13	2.27	5.47	2.03

Table 13 continued

Func.	NBPSO	BJaya	SWOA	VWOA	NBGOA	BHGSO-S	SCCOA	VCCOA
Ionosphere								
Mean accuracy	89.21	92.35	92.46	92.55	88.79	92.17	91.02	**93.14**
Mean # of features	12.90	5.07	4.57	**3.80**	13.60	4.83	10.13	4.47
Sonar								
Mean accuracy	68.02	77.68	76.33	77.39	66.76	76.38	73.21	**79.46**
Mean # of features	27.20	11.93	7.73	**6.13**	29.47	7.57	25.80	7.43
Soybean-small								
Mean accuracy	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Mean # of features	17.00	18.03	18.03	**4.50**	18.03	18.03	17.77	4.83
SpectEW								
Mean accuracy	82.49	83.15	83.02	82.22	82.11	83.16	**83.59**	83.44
Mean # of features	12.03	12.00	12.57	**7.53**	11.33	12.17	11.23	8.70
wdbc								
Mean accuracy	94.50	94.69	94.68	94.60	94.41	**94.73**	94.72	94.72
Mean # of features	14.43	15.13	13.50	**5.00**	15.80	14.80	16.53	10.40
Wine								
Mean accuracy	94.43	**95.30**	94.99	93.94	94.13	95.04	95.19	95.29
Mean # of features	6.57	7.63	7.43	**4.50**	6.73	7.53	7.17	6.67
Average rank of accuracy	5.92	4.69	5.07	5.30	6.53	4.23	2.76	**1.46**

The feature selection task aims to reduce the number of selected features while simultaneously improving the classification accuracy of the classifier under the limitation of computational budgets. In this research, we simply adopt classification accuracy as the objective function and compare our proposed SCCOA and VCCOA with six advanced MA-based techniques. The experimental results, as summarized in Table 13, show the competitiveness of SCCOA and VCCOA. SCCOA exhibits the best performance across six out of the 13 datasets considered, while VCCOA surpasses the compared algorithms in seven cases. From the metric of the average rank of classification accuracy, VCCOA emerges as the top performer among eight feature selection approaches, with SCCOA securing the second position. Therefore, we conclude that our proposed SCCOA and VCCOA provide novel and efficient approaches to addressing feature selection tasks.

Moreover, another metric that we are concerned about is the number of selected features. VWOA emerges as the top-performing approach based on this performance indicator, with VCCOA securing the second position. A noteworthy commonality between these two methods is their adoption of the V-shaped Tanh transfer function. Consequently, we hypothesize that the V-shaped Tanh transfer function may yield sparser solutions than the S-shaped Sigmoid transfer function, and this hypothesis will be proved practically and mathematically in our forthcoming research.

5.2.4 Performance on knapsack problems

Table 14 summarizes the experimental results of 30 trial runs on 0/1 knapsack problems in detail.

We also evaluated the performance of SCCOA and VCCOA on a set of 18 standard knapsack problems, with the experimental results summarized in Table 14, SCCOA emerges as the winner in five instances, while VCCOA excels on eight problems. When considering the average rank of SCCOA and VCCOA, SCCOA secures the second position, whereas VCCOA attains the top rank. These outcomes substantiate the practical superiority of SCCOA and VCCOA, particularly in solving 0/1 knapsack problems.

In summary, our proposed CCOA and its binary variants of SCCOA and VCCOA have demonstrated significant success in tackling both continuous and discrete optimization problems, as evidenced by the results obtained in the aforementioned experiments. We hold high hopes for the scalability and robustness of CCOA when confronted with unknown problems. Consequently, we list some open topics for future research.

6 Conclusion

This paper presents a novel optimization algorithm known as the cooperative coati optimization algorithm (CCOA). CCOA incorporates the fitness-based division and cooperative mechanism into the exploration phase of COA, while also integrating an optional base vector strategy into the exploitation phase. Numerical experiments, conducted on CEC2020 benchmark functions as well as six engineering optimization problems, show the competitiveness of CCOA adequately and sufficiently.

Additionally, we introduce two popular transfer functions into CCOA and propose SCCOA and VCCOA. The performance of these two binary versions of CCOA is evaluated across the feature selection task and the 0/1 knapsack problems. The experimental results clearly demonstrate the superior performance of SCCOA and VCCOA. Overall, our proposed CCOA

Table 14 Experimental results on 18 0/1 knapsack problems

Func.	NBPSO	BJaya	SWOA	VWOA	NBGOA	BHGSO-S	SCCOA	VCCOA
f_1								
Mean	278.4	292.2	286.8	278.5	279.6	292.9	**294.3**	294.1
Std	21.1	8.9	13.2	19.0	21.9	3.8	**1.4**	1.8
f_2								
Mean	923.0	1004.5	987.3	995.8	903.9	989.5	1013.2	**1020.4**
Std	47.5	16.7	27.8	26.2	55.4	34.4	13.0	**6.0**
f_3								
Mean	34.9	**35.0**	34.7	31.1	34.4	34.9	**35.0**	33.1
Std	0.4	**0.0**	0.7	3.4	1.4	0.4	**0.0**	2.9
f_4								
Mean	22.8	22.9	22.8	21.9	22.4	**23.0**	**23.0**	23.0
Std	0.9	0.7	0.4	2.3	1.2	**0.0**	**0.0**	0.2
f_5								
Mean	413.3	452.3	424.4	425.1	406.3	434.8	470.5	**474.2**
Std	29.4	32.3	38.1	37.8	30.1	25.9	17.4	**15.8**

Table 14 continued

Func.	NBPSO	BJaya	SWOA	VWOA	NBGOA	BHGSO-S	SCCOA	VCCOA
f_6								
Mean	50.7	**51.8**	51.2	48.2	49.3	51.7	51.7	51.5
Std	1.6	**0.7**	1.3	3.2	2.6	0.6	0.7	0.9
f_7								
Mean	103.0	105.7	104.8	105.8	98.9	106.4	105.2	**106.9**
Std	4.8	2.8	3.6	3.7	8.9	1.2	3.1	**0.4**
f_8								
Mean	9703.4	9727.7	9710.9	9729.8	9658.7	9734.7	**9746.8**	9746.7
Std	43.9	36.1	38.4	11.4	165.0	8.5	**6.2**	7.1
f_9								
Mean	128.1	**130.0**	126.6	119.2	125.0	**130.0**	129.6	**130.0**
Std	5.0	**0.0**	6.5	17.7	7.6	**0.0**	2.2	**0.0**
f_{10}								
Mean	904.2	1006.9	985.8	1004.7	915.6	993.2	1006.0	**1021.5**
Std	55.9	19.9	33.0	21.7	46.7	27.2	20.0	**8.4**
$KP01$								
Mean	275.3	**296.7**	275.1	262.9	257.7	291.0	294.6	295.2
Std	24.3	**16.8**	21.8	28.4	30.1	16.5	14.9	13.0
$KP02$								
Mean	48.5	**50.9**	49.1	46.4	48.8	**50.9**	50.6	49.9
Std	2.8	**0.7**	2.9	3.8	3.3	**0.7**	1.2	1.8
$KP03$								
Mean	138.0	142.5	141.2	132.3	135.9	**149.7**	142.8	142.1
Std	14.5	13.0	13.5	16.2	15.1	**1.0**	13.1	12.8

Table 14 continued

Func.	NBPSO	BJaya	SWOA	VWOA	NBGOA	BHGSO-S	SCCOA	VCCOA
*KP*04								
Mean	103.0	105.7	104.8	105.8	98.9	106.4	105.6	**106.9**
Std	4.8	2.8	3.6	3.7	8.9	1.2	2.5	**0.4**
*KP*05								
Mean	893.4	897.2	894.3	884.3	891.5	**899.6**	898.5	897.9
Std	7.0	4.7	6.3	20.1	7.8	**0.8**	3.6	4.1
*KP*06								
Mean	1717.8	1725.9	1719.0	1726.5	1706.2	1731.6	**1733.2**	1733.1
Std	21.1	13.6	19.1	20.0	25.1	9.4	**9.5**	4.8
*KP*07								
Mean	1439.9	1447.0	1437.4	1443.8	1434.8	1449.0	1449.2	**1450.8**
Std	9.6	5.9	11.7	9.4	14.9	4.2	5.2	**4.3**
*KP*08								
Mean	1.31e+07	1.33e+07	1.31e+07	1.32e+07	1.31e+07	1.32e+07	1.33e+07	**1.34e+07**
Std	2.0e+05	8.3e+04	1.3e+05	1.4e+05	1.5e+05	9.0e+04	9.1e+04	**9.0e+04**
Average rank	6.33	6.11	4.16	4.22	5.16	4.05	3.33	**2.61**

Bold values indicate the best results among all the compared methods, such as the average optimal solution (Mean) found in multiple runs, and their standard deviation (std)

exhibits exceptional scalability and robustness across a diverse range of optimization tasks. However, there are still many aspects that can be introduced to CCOA to further improve its robustness and scalability. Here, we list some open topics for future research.

Learning-based CCOA Numerous machine learning techniques have found successful applications within the field of MAs. These strategies encompass a wide array of approaches, including surrogate-assisted techniques [84, 85], integration of reinforcement learning [86, 87], and opposite-learning strategies [88, 89]. Similarly, these advancements hold the potential to further improve the performance of CCOA.

CCOA with its variants for different optimization tasks The scalability and robustness of CCOA have been validated across four distinct optimization tasks. It is reasonable to extend the applicability of CCOA to various other optimization domains. For instance, the integration of CCOA and non-dominated sorting [90] enables CCOA to effectively address multi-objective optimization problems. Additionally, the collaborative utilization of CCOA within a cooperative coevolution (CC) framework, following the approach introduced by Potter and De Jong [91], holds promise for solving complex optimization challenges. Furthermore, CCOA, SCCOA, and VCCOA can be extended to tackle prevalent problems within the realm of neural architecture search (NAS) problems [92, 93] and complex real-world applications [94, 95].

Finally, the extension of CCOA to diverse application scenarios remains a promising and fertile area for further investigation.

Acknowledgements This work was supported by JST SPRING Grant Number JPMJSP2119.

Author contributions **Rui Zhong** was involved in conceptualization, methodology, investigation, writing—original draft, writing—review & editing, and funding acquisition. **Chao Zhang** helped in investigation, methodology, formal analysis, and writing—review & editing. **Jun Yu** contributed to conceptualization and writing—review & editing, and project administration.

Data availability The source code of this research can be downloaded from https://github.com/RuiZhong961230/CCOA.

Declarations

Conflict of 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

1. Yin H, Lyu Y (2024) Gwo-based power allocation optimization algorithm for consumer iot networks. IEEE Trans Consum Electron 70(1):1294–1301. https://doi.org/10.1109/TCE.2023.3320661
2. Wu L, Huang X, Cui J, Liu C, Xiao W (2023) Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot. Expert Syst Appl 215:119410. https://doi.org/10.1016/j.eswa.2022.119410
3. Zhong R, Yu J (2024) Dea2h2: differential evolution architecture based adaptive hyper-heuristic algorithm for continuous optimization. Clust Comput, 1–28. https://doi.org/10.1007/s10586-024-04587-0
4. Cao B, Fan S, Zhao J, Tian S, Zheng Z, Yan Y, Yang P (2021) Large-scale many-objective deployment optimization of edge servers. IEEE Trans Intell Transp Syst 22(6):3841–3849. https://doi.org/10.1109/tits.2021.3059455
5. Lai Z, Li G, Feng X, Hu X, Jiang C (2024) A parallel chimp optimization algorithm based on tracking-learning and fuzzy opposition-learning behaviors for data classification. Appl Soft Comput 157:111547. https://doi.org/10.1016/j.asoc.2024.111547

6. Huang C, Zhou X, Ran X, Liu Y, Deng W, Deng W (2023) Co-evolutionary competitive swarm optimizer with three-phase for large-scale complex optimization problem. Inf Sci 619:2–18. https://doi.org/10.1016/j.ins.2022.11.019
7. Zhang Y (2023) Elite archives-driven particle swarm optimization for large scale numerical optimization and its engineering applications. Swarm Evol Comput 76:101212. https://doi.org/10.1016/j.swevo.2022.101212
8. Wei L, He J, Guo Z, Hu Z (2023) A multi-objective migrating birds optimization algorithm based on game theory for dynamic flexible job shop scheduling problem. Expert Syst Appl 227:120268. https://doi.org/10.1016/j.eswa.2023.120268
9. Zhong R, Zhang E, Munetomo M (2023) Cooperative coevolutionary surrogate ensemble-assisted differential evolution with efficient dual differential grouping for large-scale expensive optimization problems. Complex Intell Syst, pp 1–21. https://doi.org/10.1007/s40747-023-01262-6
10. Zhang E, Nie Z, Yang Q, Wang Y, Liu D, Jeon S-W, Zhang J (2023) Heterogeneous cognitive learning particle swarm optimization for large-scale optimization problems. Inf Sci 633:321–342. https://doi.org/10.1016/j.ins.2023.03.086
11. Lai Z, Feng X, Yu H, Luo F (2021) A parallel social spider optimization algorithm based on emotional learning. IEEE Trans Syst Man Cybern: Syst 51(2):797–808. https://doi.org/10.1109/TSMC.2018.2883329
12. Zhong R, Yu J, Zhang C, Munetomo M (2024) Srime: a strengthened rime with Latin hypercube sampling and embedded distance-based selection for engineering optimization problems. Neural Comput Appl 36:6721–6740. https://doi.org/10.1007/s00521-024-09424-4
13. Yuefeng X, Rui Z, Chao Z, Jun Y (2024) Multiplayer battle game-inspired optimizer for complex optimization problems. Clust Comput. https://doi.org/10.1007/s10586-024-04448-w
14. Azizi M, Talatahari S, Gandomi A (2023) Fire hawk optimizer: a novel metaheuristic algorithm. Artif Intell Rev 56:1–77. https://doi.org/10.1007/s10462-022-10173-w
15. Azizi M, Aickelin U, Khorshidi H, Baghalzadeh Shishehgarkhaneh M (2023) Energy valley optimizer: a novel metaheuristic algorithm for global and engineering optimization. Sci Rep 13:226. https://doi.org/10.1038/s41598-022-27344-y
16. Azizi M, Baghalzadeh Shishehgarkhaneh M, Basiri M, Moehler R (2023) Squid game optimizer (sgo): a novel metaheuristic algorithm. Sci Rep 13. https://doi.org/10.1038/s41598-023-32465-z
17. Abdelhamid AA, Towfek SK, Khodadadi N, Alhussan AA, Khafaga DS, Eid MM, Ibrahim A (2023) Waterwheel plant algorithm: a novel metaheuristic optimization method. Processes 11(5). https://doi.org/10.3390/pr11051502
18. Abdel-Basset M, Mohamed R, Jameel M, Abouhawwash M (2023) Spider wasp optimizer: a novel metaheuristic optimization algorithm. Artif Intell Rev, 1–64. https://doi.org/10.1007/s10462-023-10446-y
19. Abdel-Basset M, Mohamed R, Jameel M, Abouhawwash M (2023) Nutcracker optimizer: a novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems. Knowl-Based Syst 262:110248. https://doi.org/10.1016/j.knosys.2022.110248
20. Guan Z, Ren C, Niu J, Wang P, Shang Y (2023) Great wall construction algorithm: a novel meta-heuristic algorithm for engineer problems. Expert Syst Appl 233:120905. https://doi.org/10.1016/j.eswa.2023.120905
21. Trojovsky P, Dehghani M (2023) A new bio-inspired metaheuristic algorithm for solving optimization problems based on walruses behavior. Sci Rep 13. https://doi.org/10.1038/s41598-023-35863-5
22. Faridmehr I, Nehdi ML, Davoudkhani IF, Poolad A (2023) Mountaineering team-based optimization: A novel human-based metaheuristic algorithm. Mathematics **11**(5). https://doi.org/10.3390/math11051273
23. Zhang Q, Gao H, Zhan Z-H, Li J, Zhang H (2023) Growth optimizer: a powerful metaheuristic algorithm for solving continuous and discrete global optimization problems. Knowl-Based Syst 261:110206. https://doi.org/10.1016/j.knosys.2022.110206
24. Matoušová I, Trojovsky P, Dehghani M, Trojovska E, Kostra J (2023) Mother optimization algorithm: a new human-based metaheuristic approach for solving engineering optimization. Sci Rep 13. https://doi.org/10.1038/s41598-023-37537-8
25. Su H, Zhao D, Heidari AA, Liu L, Zhang X, Mafarja M, Chen H (2023) Rime: a physics-based optimization. Neurocomputing 532:183–214. https://doi.org/10.1016/j.neucom.2023.02.010
26. Wang X, Wu B, Xuan Y, Liang Y, Yang H (2023) Weighted-leader search: a new choice in metaheuristic and its application in real-world large-scale optimization. Adv Eng Softw 176:103405. https://doi.org/10.1016/j.advengsoft.2022.103405
27. Trojovská E, Dehghani M, Leiva V (2023) Drawer algorithm: a new metaheuristic approach for solving optimization problems in engineering. Biomimetics 8(2) . https://doi.org/10.3390/biomimetics8020239

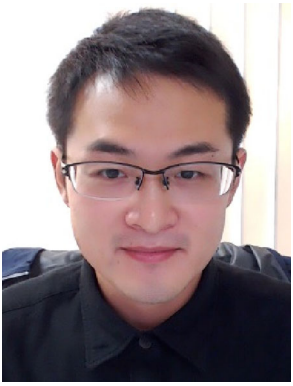
28. Mohammadi S, Nazarpour D, Beiraghi M (2023) A novel metaheuristic algorithm inspired by covid-19 for real-parameter optimization. Neural Comput Appl 35:1–50. https://doi.org/10.1007/s00521-023-08229-1
29. Deng L, Liu S (2023) Snow ablation optimizer: a novel metaheuristic technique for numerical optimization and engineering design. Expert Syst Appl 225:120069. https://doi.org/10.1016/j.eswa.2023.120069
30. Niu J, Ren C, Guan Z, Cao Z (2023) Dujiangyan irrigation system optimization (diso): a novel metaheuristic algorithm for dam safety monitoring. Structures 54:399–419. https://doi.org/10.1016/j.istruc.2023.04.102
31. Zhang W, Pan K, Li S, Wang Y (2023) Special forces algorithm: a novel meta-heuristic method for global optimization. Math Comput Simul 213:394–417. https://doi.org/10.1016/j.matcom.2023.06.015
32. Li D, Du S, Zhang Y, Zhao M (2023) Dark forest algorithm: a novel metaheuristic algorithm for global optimization problems. Comput Mater Continua 75:1–29. https://doi.org/10.32604/cmc.2023.035911
33. Shehadeh H (2023) Chernobyl disaster optimizer (cdo): a novel meta-heuristic method for global optimization. Neural Comput Appl. https://doi.org/10.1007/s00521-023-08261-1
34. Mohapatra S, Mohapatra P (2023) American zebra optimization algorithm for global optimization problems. Sci Rep 13. https://doi.org/10.1038/s41598-023-31876-2
35. Rahman C (2023) Group learning algorithm: a new metaheuristic algorithm. Neural Comput Appl 35:1–16. https://doi.org/10.1007/s00521-023-08465-5
36. Hashim FA, Mostafa RR, Hussien AG, Mirjalili S, Sallam KM (2023) Fick's law algorithm: a physical law-based algorithm for numerical optimization. Knowl-Based Syst 260:110146. https://doi.org/10.1016/j.knosys.2022.110146
37. Agushaka O, Ezugwu A, Abualigah L (2023) Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer. Neural Comput Appl 35:1–33. https://doi.org/10.1007/s00521-022-07854-6
38. Yu J (2022) Vegetation evolution: an optimization algorithm inspired by the life cycle of plants. Int J Comput Intell Appl 21. https://doi.org/10.1142/S1469026822500109
39. Goodarzimehr V, Shojaee S, Hamzehei-Javaran S, Talatahari S (2022) Special relativity search: a novel metaheuristic method based on special relativity physics. Knowl-Based Syst 257:109484. https://doi.org/10.1016/j.knosys.2022.109484
40. Kuyu YC, Vatansever F (2022) Gozde: a novel metaheuristic algorithm for global optimization. Future Gener Comput Syst 136(C):128–152
41. Hashim FA, Hussien AG (2022) Snake optimizer: a novel meta-heuristic optimization algorithm. Know-Based Syst **242**(C). https://doi.org/10.1016/j.knosys.2022.108320
42. Rahmani AM, AliAbdi I (2022) Plant competition optimization: a novel metaheuristic algorithm. Expert Syst 39(6):12956. https://doi.org/10.1111/exsy.12956
43. Daliri A, Asghari A, Azgomi H, Alimoradi M (2022) The water optimization algorithm: a novel metaheuristic for solving optimization problems. Appl Intell 52. https://doi.org/10.1007/s10489-022-03397-4
44. Dehghani M, Montazeri Z, Trojovská E, Trojovský P (2023) Coati optimization algorithm: a new bio-inspired metaheuristic algorithm for solving optimization problems. Knowl-Based Syst 259:110011. https://doi.org/10.1016/j.knosys.2022.110011
45. Ahadzadeh B, Abdar M, Safara F, Khosravi A, Menhaj MB, Suganthan PN (2023) Sfe: a simple, fast and efficient feature selection algorithm for high-dimensional data. IEEE Trans Evol Comput, 1 (2023). https://doi.org/10.1109/TEVC.2023.3238420
46. Hammouri AI, Mafarja M, Al-Betar MA, Awadallah MA, Abu-Doush I (2020) An improved dragonfly algorithm for feature selection. Knowl-Based Syst 203:106131. https://doi.org/10.1016/j.knosys.2020.106131
47. Khurma RA, Aljarah I, Sharieh AA-A (2020) An intelligent feature selection approach based on moth flame optimization for medical diagnosis. Neural Comput Appl 33:7165–7204. https://doi.org/10.1007/s00521-020-05483-5
48. Nadimi-Shahraki MH, Zamani H, Mirjalili S (2022) Enhanced whale optimization algorithm for medical feature selection: a covid-19 case study. Comput Biol Med 148:105858. https://doi.org/10.1016/j.compbiomed.2022.105858
49. Chen H, Zhou X, Shi D (2022) A chaotic antlion optimization algorithm for text feature selection. Int J Comput Intell Syst 15. https://doi.org/10.1007/s44196-022-00094-5
50. Alzaqebah M, Briki K, Alrefai N, Brini S, Jawarneh S, Alsmadi MK, Mohammad RMA, ALmarashdeh I, Alghamdi FA, Aldhafferi N, Alqahtani A (2021) Memory based cuckoo search algorithm for feature selection of gene expression dataset. Inform Med Unlocked 24:100572 (2021). https://doi.org/10.1016/j.imu.2021.100572

51. Kundu R, Chattopadhyay S, Cuevas E, Sarkar R (2022) Altwoa: altruistic whale optimization algorithm for feature selection on microarray datasets. Comput Biol Med 144:105349. https://doi.org/10.1016/j.compbiomed.2022.105349
52. Agarwal R, Shekhawat NS, Kumar S, Nayyar A, Qureshi B (2021) Improved feature selection method for the identification of soil images using oscillating spider monkey optimization. IEEE Access 9:167128–167139. https://doi.org/10.1109/ACCESS.2021.3135536
53. Du D-Z, Ko K-I, Hu X (2012) Design and analysis of approximation algorithms vol. 62. Springer, Berlin. https://doi.org/10.1007/978-1-4614-1701-9
54. Yaseen ZM, Sulaiman SO, Deo RC, Chau K-W (2019) An enhanced extreme learning machine model for river flow forecasting: state-of-the-art, practical applications in water resource engineering area and future research direction. J Hydrol 569:387–408. https://doi.org/10.1016/j.jhydrol.2018.11.069
55. Ali IM, Essam D, Kasmarik K (2021) Novel binary differential evolution algorithm for knapsack problems. Inf Sci 542:177–194. https://doi.org/10.1016/j.ins.2020.07.013
56. Vanderster DC, Dimopoulos NJ, Parra-Hernandez R, Sobie RJ (2009) Resource allocation on computational grids using a utility model and the knapsack problem. Future Gener Comput Syst 25(1):35–50. https://doi.org/10.1016/j.future.2008.07.006
57. Yaguchi K, Tamura K, Yasuda K, Ishigame A (2011) Basic study of proximate optimality principle based combinatorial optimization method. In: 2011 IEEE International conference on systems, man, and cybernetics, pp 1753–1758. https://doi.org/10.1109/ICSMC.2011.6083925
58. Faris H, Mafarja MM, Heidari AA, Aljarah I, Al-Zoubi AM, Mirjalili S, Fujita H (2018) An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. Knowl-Based Syst 154:43–67. https://doi.org/10.1016/j.knosys.2018.05.009
59. Van Thieu N, Mirjalili S (2023) Mealpy: an open-source library for latest meta-heuristic algorithms in python. J Syst Architect 139:102871. https://doi.org/10.1016/j.sysarc.2023.102871
60. Nguyen T (2020) A framework of optimization functions using Numpy (OpFuNu) for optimization problems. Zenodo. https://doi.org/10.5281/zenodo.3620960
61. Thieu NV (2023) ENOPPY: a python library for engineering optimization problems. Zenodo. https://doi.org/10.5281/zenodo.7953206
62. Van Thieu N, Nguyen NH, Heidari AA (2023) Feature selection using metaheuristics made easy: open source MAFESE library in python. Zenodo. https://doi.org/10.5281/zenodo.7969042
63. Yue CT, Price PNS KV (2020) Problem definitions and evaluation criteria for the cec 2020 special session and competition on single objective bound constrained numerical optimization. In: Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore
64. Ezugwu A, Agushaka O, Abualigah L, Mirjalili S, Gandomi A (2022) Prairie dog optimization algorithm. Neural Comput Appl 34:20017–20065. https://doi.org/10.1007/s00521-022-07530-9
65. Zhong R, Yu J (2024) A novel evolutionary status guided hyper-heuristic algorithm for continuous optimization. Clust Comput, pp 1–30 . https://doi.org/10.1007/s10586-024-04593-2
66. Kaur S, Kumar Y, Koul A, Kamboj S (2022) A systematic review on metaheuristic optimization techniques for feature selections in disease diagnosis: open issues and challenges. Arch Comput Methods Eng 30. https://doi.org/10.1007/s11831-022-09853-1
67. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95—international conference on neural networks, vol. 4, pp 1942–19484. https://doi.org/10.1109/ICNN.1995.488968
68. Storn R (1996) On the usage of differential evolution for function optimization. In: Proceedings of North American fuzzy information processing, pp 519–523. https://doi.org/10.1109/NAFIPS.1996.534789
69. Zhao W, Wang L, Zhang Z (2020) Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm. Neural Comput Appl 32:1–43. https://doi.org/10.1007/s00521-019-04452-x
70. Alsattar H, Zaidan A, Bahaa B (2020) Novel meta-heuristic bald eagle search optimisation algorithm. Artif Intell Rev 53. https://doi.org/10.1007/s10462-019-09732-5
71. Shaqfa M, Beyer K (2021) Pareto-like sequential sampling heuristic for global optimisation. Soft Comput 25:9077–9096. https://doi.org/10.1007/s00500-021-05853-8
72. Ahmadianfar I, Heidari AA, Noshadian S, Chen H, Gandomi AH (2022) Info: an efficient optimization algorithm based on weighted mean of vectors. Expert Syst Appl 195:116516. https://doi.org/10.1016/j.eswa.2022.116516
73. Trojovský P, Dehghani M (2022) Pelican optimization algorithm: a novel nature-inspired algorithm for engineering applications. Sensors 22(3). https://doi.org/10.3390/s22030855
74. Dehghani M, Trojovsky P (2023) Osprey optimization algorithm: a new bio-inspired metaheuristic algorithm for solving engineering optimization problems. Front Mech Eng 8:1126450. https://doi.org/10.3389/fmech.2022.1126450

75. Coello Coello CA (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191(11):1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1
76. Nguyen BH, Xue B, Andreae P (2017) A novel binary particle swarm optimization algorithm and its applications on knapsack and feature selection problems. In: Intelligent and evolutionary systems, pp 319–332. Springer, Cham. https://doi.org/10.1007/978-3-319-49049-6_23
77. Yang Z, Guo Y, Niu Q, Ma H, Zhou Y, Zhang L (2018) A novel binary jaya optimization for economic/emission unit commitment. In: 2018 IEEE congress on evolutionary computation (CEC), pp 1–6 (2018).https://doi.org/10.1109/CEC.2018.8477660
78. Mafarja M, Jaber I, Ahmed S (2018) Whale optimization algorithm for high-dimensional small-instance feature selection. In: 2018 Fifth International symposium on innovation in information and communication technology (ISIICT), pp 1–6. https://doi.org/10.1109/ISIICT.2018.8613293
79. Hussien AG, Hassanien AE, Houssein EH, Bhattacharyya S, Amin M (2019) S-shaped binary whale optimization algorithm for feature selection. In: Recent trends in signal and image processing, pp 79–87. Springer, Singapore. https://doi.org/10.1007/978-981-10-8863-6_9
80. Hichem H, Elkamel M, Rafik M, Mesaaoud MT, Ouahiba C (2022) A new binary grasshopper optimization algorithm for feature selection problem. J King Saud Univ - Comput Inf Sci 34(2):316–328. https://doi.org/10.1016/j.jksuci.2019.11.007
81. Manjula Devi R, Premkumar PJM (2022) Bhgso: binary hunger games search optimization algorithm for feature selection problem. Comput Mater Continua 70(1):557–579. https://doi.org/10.32604/cmc.2022.019611
82. Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Stat 6(2):65–70
83. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893
84. Wang Y, Yin D-Q, Yang S, Sun G (2019) Global and local surrogate-assisted differential evolution for expensive constrained optimization problems with inequality constraints. IEEE Trans Cybern 49(5):1642–1656. https://doi.org/10.1109/TCYB.2018.2809430
85. Dong H, Dong Z (2020) Surrogate-assisted grey wolf optimization for high-dimensional, computationally expensive black-box problems. Swarm Evol Comput 57:100713. https://doi.org/10.1016/j.swevo.2020.100713
86. Tapia D, Crawford B, Soto R, Palma W, Lemus-Romani J, Cisternas-Caneo F, Castillo M, Becerra-Rozas M, Paredes F, Misra S (2021) Embedding q-learning in the selection of metaheuristic operators: the enhanced binary grey wolf optimizer case. In: 2021 IEEE international conference on automation/XXIV congress of the Chilean association of automatic control (ICA-ACCA), pp 1–6. https://doi.org/10.1109/ICAACCA51523.2021.9465259
87. Huynh TN, Do DTT, Lee J (2021) Q-learning-based parameter control in differential evolution for structural optimization. Appl Soft Comput 107:107464. https://doi.org/10.1016/j.asoc.2021.107464
88. Xu Y, Yang Z, Li X, Kang H, Yang X (2020) Dynamic opposite learning enhanced teaching-learning-based optimization. Knowl-Based Syst 188:104966. https://doi.org/10.1016/j.knosys.2019.104966
89. Dong H, Xu Y, Li X, Yang Z, Zou C (2021) An improved antlion optimizer with dynamic random walk and dynamic opposite learning. Knowl-Based Syst 216:106752. https://doi.org/10.1016/j.knosys.2021.106752
90. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197. https://doi.org/10.1109/4235.996017
91. Potter MA, De Jong KA (1994) A cooperative coevolutionary approach to function optimization. In: Parallel problem solving from nature—PPSN III. Springer, Berlin, pp 249–257
92. Awad N, Mallik N, Hutter F (2021) Differential evolution for neural architecture search
93. Liu Y, Sun Y, Xue B, Zhang M, Yen GG, Tan KC (2023) A survey on evolutionary neural architecture search. IEEE Trans Neural Netw Learn Syst 34(2):550–570. https://doi.org/10.1109/TNNLS.2021.3100554
94. Zhong R, Fan Q, Zhang C, Yu J (2024) Hybrid remora crayfish optimization for engineering and wireless sensor network coverage optimization. Clust Comput, pp 1–28. https://doi.org/10.1007/s10586-024-04508-1
95. Zhang Y, Li S, Wang Y, Yan Y, Zhao J, Gao Z (2024) Self-adaptive enhanced learning differential evolution with surprisingly efficient decomposition approach for parameter identification of photovoltaic models. Energy Convers Manage 308:118387. https://doi.org/10.1016/j.enconman.2024.118387

Rui Zhong received a Bachelor degree from Huazhong Agricultural University, China, in 2019, and a Master degree from Kyushu University, Japan, in 2022. He is currently pursuing a PhD at Hokkaido University, Japan. His research interests include evolutionary computation, large-scale global optimization, and meta/hyper-heuristics.

Chao Zhang received his Ph.D. at Iwate University (Japan) in March 2017. He is now a specially appointed professor at the Faculty of Engineering, University of Toyama (Japan). His research interests include computer vision and graphics, mainly focused on vision-based optimization problems. He is a member of the IEEE, IEEJ, ITE, JSAI, and IEICE. More information can be found at http://www.labzhang.com/



Jun Yu received his Bachelor's degree from Northeastern University, China, in 2014, and his Master's and Doctorate degrees from Kyushu University, Japan, in 2017 and 2019, respectively. He is currently an Assistant Professor at Niigata University, Japan. His research interests include soft computing, deep learning, and machine learning.