Cluster Computing
https://doi.org/10.1007/s10586-024-04508-1

Hybrid remora crayfish optimization for engineering and wireless sensor network coverage optimization

Rui Zhong[1] · Qinqin Fan[2] · Chao Zhang[3] · Jun Yu[4]

Received: 18 January 2024 / Revised: 3 April 2024 / Accepted: 5 April 2024

Abstract
This paper proposes a novel hybrid metaheuristic algorithm called the remora crayfish optimization algorithm (HRCOA) designed for solving continuous optimization problems. The crayfish optimization algorithm (COA), recently proposed as a meta-heuristic algorithm (MA), exhibits certain limitations such as imbalanced exploration and exploitation capacities, susceptibility to premature optimization, and a propensity for stagnation. To address these shortcomings, we incorporate the exploitation operators from the remora optimization algorithm (ROA) to enhance the exploitative behaviors of COA. In addition, we simplify the summer resort operator in the original COA to streamline the search operator design, thus avoiding unnecessary complexity. Furthermore, numerical experiments on 10-dimensional (D) and 20-D CEC2022 benchmark functions, 50-D and 100-D CEC2020 benchmark functions, engineering optimization problems, and wireless sensor networks (WSNs) coverage optimization problems are conducted to investigate the performance of our proposed HRCOA comprehensively. We compare the proposed HRCOA against eight well-known state-of-the-art MAs, including CMAES and the original COA, as competitor algorithms. The experimental and statistical results confirm the effectiveness, competitiveness, and scalability of our proposal. Finally, we conclude that the proposed HRCOA possesses significant potential for addressing diverse optimization challenges in real-world scenarios.

Keywords Crayfish optimization algorithm (COA) · Remora optimization algorithm (ROA) · Hybridization · Engineering problem · Wireless sensor networks (WSNs) coverage optimization

1 Introduction

The rapid development of the artificial intelligence (AI) field has given rise to increasingly complex and high-dimensional optimization problems in both academic research and industrial scenarios. The traditional techniques, such as linear/nonlinear programming, Newton's method, conjugate gradient method, and gradient descent method, are not suitable to deal with these intricate optimization challenges. This inadequacy stems from the requirement of convexity in the feasible domain and the need for continuous differentiability in the objective function [1], which these conventional methods may not satisfy. Therefore, many conventional optimization approaches face a formidable challenge, necessitating the development of an efficient and robust optimization algorithm [2, 3].

Fortunately, metaheuristic algorithms (MAs) provide a feasible methodology to address such complex

✉ Jun Yu
yujun@ie.niigata-u.ac.jp

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

Qinqin Fan
forever123fan@163.com

Chao Zhang
zhang@u-fukui.ac.jp

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

[2] Logistics Research Center, Shanghai Maritime University, Shanghai, China

[3] Department of Engineering, University of Fukui, Fukui, Japan

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

Published online: 04 May 2024

optimization problems. MAs belong to a class of stochastic optimization techniques that draw inspiration from natural phenomena or organism behaviors, employing iterative searches to find acceptable solutions. Thanks to their outstanding attributes, including gradient-free optimization, ease of implementation, robustness, and flexibility, MAs have attracted widespread attention among scholars and have experienced remarkable growth. Over the past few decades, numerous novel MAs have been proposed and successfully applied across various domains. Here, we list some representative MAs in Table 1 and categorize them into four groups: evolution-based, swarm intelligence, human-based, and law-based (utilizing principles from fields like mathematics, physics, and chemistry).

Meanwhile, some researchers have identified limitations inherent in the single MA, including imbalanced exploration and exploitation capacities, susceptibility to premature optimization, and a propensity for stagnation. They suggest that combining the strengths of various MAs through hybridization could potentially mitigate these issues: Qu et al. [31] introduced a novel approach by hybridizing the simplified grey wolf optimizer (SGWO) with the commensalism phase of the modified symbiotic organisms search (MSOS), resulting in the HSGWO-MSOS algorithm tailored for solving unmanned aerial vehicle (UAV) path planning tasks. Similarly, Zhang et al. [32] proposed the hybridization of particle swarm and grey wolf optimizer (HGWOP), which leverages the simplified GWO with a differential perturbation strategy alongside PSO employing a stochastic mean example learning strategy. This hybridization has proven to be a highly competitive optimizer, particularly excelling in CEC2013 and CEC2015 benchmark functions. Che et al. [33] recognized a resemblance between the spiral attack prey behavior observed in seagulls and the predation strategy known as the whale bubble net. Leveraging this insight, they introduced whale optimization into the seagull algorithm, resulting in a hybrid approach termed the whale

Table 1 A selection of representative MAs from the literature

Category	Algorithm	Inspiration
Evolution-based	Genetic algorithm (GA) [4]	Evolutionary concepts
	Differential evolution (DE) [5]	Darwin's theory of evolution
	Genetic programming (GP) [6]	Biological evolution
	Evolutionary programming (EP) [7]	Finite state machine
Swarm intelligence	Particle swarm optimization (PSO) [8]	Social behaviors of bird flock
	Salp swarm algorithm (SSA) [9]	The navigating and foraging behaviors of salp
	Marine predators algorithm (MPA) [10]	The widespread foraging strategy of ocean predators
	Snake optimizer (SO) [11]	Special mating behavior of snakes
	Vegetation evolution (VEGE) [12]	Growth mechanism of tomatoes
	Honey badger algorithm (HBA) [13]	Intelligent foraging behavior of honey badgers
	Sand cat swarm optimization (SCSO) [14]	The sand cat behavior that tries to survive in nature
	Coati optimization algorithm (COA) [15]	Natural behaviors of coatis
Human-based	Group teaching optimization algorithm (GTOA) [16]	The group teaching mechanism
	Search and rescue optimization (SARO) [17]	Human behaviors during search and rescue operations
	Child drawing development optimization (CDDO) [18]	Child's learning behavior and cognitive development
	Sewing training-based optimization (STBO) [19]	Process of sewing to beginner tailors
	Multiplayer battle game optimizer (MBGO) [20]	Mechanisms of multiplayer battle royale games
	Mountaineering team-based optimization (MTBO) [21]	Cooperative human behaviors in mountaineering
	Mother optimization algorithm (MOA) [22]	The interaction between a mother and her children
Law-based	Material generation algorithm (MGA) [23]	Chemical compounds and chemical reactions
	RUN [24]	Runge Kutta method
	Pareto-like sequential sampling (PSS) [25]	Pareto's principle
	INFO [26]	Weighted mean of the vector
	RIME [27]	The soft-rime and hard-rime growth process
	Snow ablation optimizer (SAO) [28]	The sublimation and melting behavior of snow
	Optical microscope algorithm (OMA) [29]	Microscope magnification
	Chernobyl disaster optimizer (CDO) [30]	The nuclear reactor core explosion of Chernobyl

optimization and seagull algorithm (WSOA). By combining the surrounding mechanism of the WOA with the spiral attack behavior of the SOA, they aimed to enhance the convergence accuracy of SOA. Additionally, they incorporated levy flight into the search formula of SOA to mitigate premature convergence. Numerical experiments conducted on a standard set of 25 benchmark functions confirmed the efficiency and effectiveness of the proposed WSOA. Furthermore, Han et al. [34] introduced a multi-stage search strategy from the marine predator algorithm (MPA) into PSO, yielding the hybrid PSO for marine predators (HMPPSO). The optimization process of HMPPSO is segmented into three stages: the initial stage employs Brownian motion for exploration, followed by a middle stage employing a hybrid strategy between Brownian motion and random wandering strategy, and finally, the learning strategy is activated in the third stage. HMPPSO has demonstrated superior performance over state-of-the-art optimizers in various scenarios, including CEC2017 benchmark test functions and multi-dimensional non-linear structural design optimization problems. These studies underscore the potential of hybridizing various MAs as a promising avenue for enhancing their performance.

The crayfish optimization algorithm (COA) [35], one of the latest additions to MAs, mimics three behaviors of crayfish (namely summer resort behavior, competition behavior, and foraging behavior) to realize global optimization. Through practical experimentation and statistical analysis involving 23 standard benchmark functions and the CEC2014 benchmark function, the efficiency and robustness of COA have been substantiated. However, akin to many newly proposed MAs, COA exhibits certain limitations, including an imbalanced exploration and exploitation capacity, susceptibility to stagnation, and premature optimization. In addition, issues have been identified within the original COA framework. These issues encompass the neglect to provide a comprehensive explanation of the selection process and the presence of overlapping elements in the designed search operators of COA.

In light of the mentioned challenges, we propose a novel hybrid remora crayfish optimization algorithm (HRCOA). Our approach begins with the simplification of the COA to address issues related to the overlap in the design of search operators. Additionally, we draw inspiration from the remora optimization algorithm (ROA) [35], a well-established and efficient MA in the domain of global optimization. We propose incorporating the exploitation operators of ROA into COA to further enhance its exploitative search capabilities. Moreover, numerical experiments across various scenarios, including low-, median-, and high-dimensional IEEE-CEC benchmark functions, engineering optimization problems, and wireless sensor networks (WSNs) coverage problems are implemented, and eight famous and state-of-the-art competitor algorithms (i.e., PSO [8], covariance matrix adaptation evolution strategy (CMAES) [36], circle search algorithm (CSA) [37], Siberian tiger optimization (STO) [38], pelican optimization algorithm (POA) [39], serval optimization algorithm (SOA) [40], energy valley optimizer (EVO) [41], and the original COA [35]) are employed to comprehensively investigate the performance of our proposed HRCOA. Specifically, the contributions of this paper are as follows:

- We state the shortage of the COA and propose a novel hybrid remora crayfish optimization algorithm (HRCOA) for continuous optimization.
- The search operators in ROA are embedded into HRCOA to strengthen the exploitation ability.
- Low-dimensional CEC2022 benchmark functions and median- and high-dimensional CEC2020 benchmark functions are employed to evaluate the overall performance of HRCOA, and engineering problems and wireless sensor networks (WSNs) coverage problems are adopted to investigate the performance of HRCOA in real-world scenarios.
- Ablation experiments are conducted to investigate the contribution of two proposed strategies independently.
- Sensitivity analysis experiments are conducted to investigate the sensitivity of the proposed HRCOA with the crucial hyper-parameter temperature T.
- Numerical experiments are conducted in accordance with the standard experimental settings, employing eight state-of-the-art MAs for comparison. Experimental and statistical results confirm the effectiveness and competitiveness of HRCOA.

The structure of the remaining sections in this paper is as follows. Section 2 introduces the operations of ROA and COA. Section 3 provides a detailed presentation of our proposed HRCOA. Section 4 introduces the numerical experiments. Section 5 analyzes the performance of HRCOA and lists some open topics for future research. Finally, Sect. 6 concludes the paper.

2 Related works

2.1 Remora optimization algorithm (ROA)

Jia et al. noticed an interesting behavior of the remora where upon spotting a swordfish or a white shark passing by, the remora swiftly advances and firmly attaches itself to the swordfish or white shark [42, 43]. Inspired by this unique behavior, the classic ROA was developed, featuring

two distinct search phases: free travel and thoughtful eating.

2.1.1 Free travel: exploration stage

During the free travel stage, ROA employs two search operators: swordfish optimization (SFO) [44] strategy and experience attack.

SFO strategy: When the remora attaches itself to the swordfish, it mimics the movements of the swordfish. Therefore, this phase of ROA employs the elite-based search operator as described in SFO [44].

$$X_i^{t+1} = X_{best}^t - \left(r \cdot \left(\frac{X_{best}^t + X_{rand}^t}{2} \right) - X_{rand}^t \right) \tag{1}$$

where X_i^{t+1} is the i^{th} remora individual in the $t+1$ iteration, X_{best}^t is the best remora individual in the t iteration, X_{rand}^t is the randomly selected remora individual from the t iteration, and r is a random value which follows the uniform distribution in (0, 1).

Experience attack: Similar to the experience accumulation, the remora individual makes slight perturbations in its movements around the host to assess whether any changes to its attachment are warranted.

$$X_{att} = X_i^t + r^{'} \cdot (X_i^t - X_{pre}) \tag{2}$$

where X_{att} is a tentative movement of the i^{th} remora, X_{pre} is the position of the corresponding remora in the previous iteration, and $r^{'}$ is a random value which follows the normal distribution with a mean of 0 and a standard deviation of 1. And only if the X_{att} has a better fitness, the remora will adjust its position accordingly by moving toward the new position.

2.1.2 Thoughtful eating: exploitation stage

Similarly, ROA employs two search operators to facilitate the exploitative search: whale optimization algorithm (WOA) [45] strategy and host feeding.

WOA strategy: When the host of a remora is the whale, it adheres to the feeding strategy of the whale and incorporates the bubble-net attacking behavior expressed by Eq. (3).

$$\begin{aligned} X_i^{t+1} &= D \cdot e^{\alpha} \cdot \cos(2\pi\alpha) + X_i^t \\ \alpha &= r \cdot (a-1) + 1 \\ a &= -(1 + t/T) \\ D &= |X_{best}^t - X_i^t| \end{aligned} \tag{3}$$

where D represents the distance of i^{th} remora to the best remora found so far, α is a random number in [-1, 1], t and T are the current iteration and maximum iteration, respectively, and a decreases from -1 to -2 linearly.

Host feeding: During the host feeding phase, the remora actively searches for food in the vicinity of its host, and the mathematical model describing this behavior is as follows:

$$\begin{aligned} X_i^{t+1} &= X_i^t + A \\ A &= B \cdot (X_i^t - C \cdot X_{best}^t) \\ B &= V \cdot (2 \cdot r - 1) \\ V &= 2 \cdot (1 - t/T) \end{aligned} \tag{4}$$

where A indicates the small movement around the host, $C = 0.1$ is a constant recommended used in the ROA, and V is a parameter linearly decreases from 2 to 0.

In addition, an auxiliary binary vector denoted as H is employed to keep track of the host for the remora swarm. If $H_i = 0$, the corresponding i^{th} remora determines the whale as its host and if $H_i = 1$, the sailfish is selected as the host. Equation (5) is applied to randomly determine whether a 0 or 1 should be assigned in the binary vector H.

$$H_i = \mathbf{rand}(\{0, 1\}) \tag{5}$$

In summary, the pseudocode of ROA is shown in Algorithm 1.

Algorithm 1 ROA [46]

Require: Population size: N, Dimension: D, Maximum iteration: T
Ensure: Optimum: X_{best}

```
Initialize the remora swarm X randomly
O ← best(P)
t ← 0
while t < T do
    Generate the binary vector H
    for i = 0 to N do
        if H_i == 0 then
            Update the X_i with Eq. (3) # WOA strategy
        else
            Update the X_i with Eq. (1) # SFO strategy
        end if
        Experience attack with Eq. (2) # Experience attack
        if X_att has better fitness value then
            Use Eq. (5) to shift the host
        else
            Update the X_i with Eq. (4) # Host feeding
        end if
    end for
    Update the remora swarm X and current optimum X_best
    t ← t + 1
end while
return X_best
```

2.2 Crayfish optimization algorithm (COA)

COA draws inspiration from the specific behaviors of crayfish, such as the summer resort, competition, and foraging, to realize iterative optimization. The transition between these three phases is governed by the temperature parameter *temp*, as defined in Eq. (6).

$$temp = 15 \cdot r + 20 \tag{6}$$

2.2.1 Summer resort: exploration stage

When the temperature is excessively high (i.e., $temp > 30$), the crayfish initiates the summer resort behavior, and its mathematical model is represented by Eq. (7).

$$\begin{aligned} X_i^{t+1} &= X_i^t + C_2 \cdot r \cdot (X_{shade} - X_i^t) \\ C_2 &= 2 - (t/T) \\ X_{shade} &= (X_G + X_L)/2 \end{aligned} \tag{7}$$

where X_{shade} denotes the position of the shaded cave, which is determined as the mean of the current best solution X_G and the optimum within the current crayfish swarm X_L. t and T are the current iteration and maximum iteration, respectively.

2.2.2 Competition: exploitation stage

While the crayfish are situated within the shaded cave, they engage in competition to secure the superior cave by employing Eq. (8).

$$X_i^{t+1} = X_i^t - X_i^{rand} + X_{shade} \tag{8}$$

2.2.3 Foraging: exploitation stage

As the temperature decreases, the environment becomes more favorable for the crayfish to engage in feeding. Additionally, the parameter Q, representing the size of food, plays a pivotal role in determining the specific foraging strategies. The definition of Q is outlined in Eq. (9)

$$Q = C_3 \cdot r \cdot (F_i/F_{best}) \tag{9}$$

where C_3 is the food factor and, it is recommended to be set at 3 [35], F_i and F_{best} are the fitness value of the i^{th} crayfish individual and the best crayfish individual, respectively.

When the food size Q exceeds a certain threshold, specifically when $Q > (C_3 + 1)/2$, indicating that the food size is too big, the crayfish will proceed to tear the food into smaller pieces. The new position of food can be calculated by Eq. (10)

$$X_{food}^{'} = \exp\left(-\frac{1}{Q}\right) \cdot X_{food} \tag{10}$$

where X_{food} is the optimal position of crayfish found so far, and the foraging operator is formulated by Eq. (11)

$$\begin{aligned} X_i^{t+1} =& X_i^t + X_{food}^{'} \cdot p \cdot (\cos(2\pi r) - \sin(2\pi r)) \\ p =& C_1 \cdot \left(\frac{1}{\sqrt{2\pi}\sigma} \cdot \exp\left(-\frac{(temp - \mu)^2}{2\sigma^2}\right)\right) \end{aligned} \tag{11}$$

where $C_1 = 0.2$ is suggested as a control parameter, p is the food intake factor determined by normal distribution, μ and σ are set to 25 and 3, respectively.

When the food size Q is less than or equal to $(C_3 + 1)/2$, it indicates that the food is small enough for the crayfish to consume directly.

$$X_i^{t+1} = ((1 + r) \cdot X_i^t - X_{food}) \cdot p \tag{12}$$

In summary, the pseudocode of COA is shown in Algorithm 2.

2.3 Wireless sensor networks (WSNs) coverage optimization

Wireless Sensor Networks (WSNs) encompass an array of small, battery-powered sensor nodes with capabilities for sensing, processing, and wireless data transmission [47]. This technique is widely used in real-world applications, including environmental monitoring, surveillance, healthcare, and industrial automation. A pivotal aim in the design and deployment of WSNs is to guarantee that the area of interest is effectively and comprehensively monitored by the sensors, a task commonly referred to as WSNs coverage optimization [48].

Suppose the number of sensor nodes in the monitoring area is N, the coordinate of the i^{th} sensor s_i is (x_{s_i}, y_{s_i}), and the sensing radius is R. For the j^{th} monitoring node n_j, which has coordinate (x_{n_j}, y_{n_j}), the Euclidean distance between the i^{th} sensor s_i and the j^{th} monitoring node n_j can be calculated using Eq. (13)

$$dis(s_i, n_j) = \sqrt{(x_{s_i} - x_{n_j})^2 + (y_{s_i} - y_{n_j})^2} \tag{13}$$

Algorithm 2 COA [35]

Require: Population size: N, Dimension: D, Maximum iteration: T
Ensure: Optimum: X_{best}

Initialize the crayfish swarm X randomly
$t \leftarrow 0$
while $t < T$ **do**
  Determine the temperature $temp$ by Eq. (6)
  for $i = 0$ *to* N **do**
    Calculate the position of X_{shade} by Eq. (7)
    if $temp > 30$ **then**
      Generate a random number r
      if $r < 0.5$ **then**
        Update the X_i with Eq. (7) # Summer resort strategy
      else
        Update the X_i with Eq. (8) # Competition strategy
      end if
    else
      Calculate the p and Q by Eq. (11) and Eq. (9)
      if $Q > 2$ **then**
        Update the X_i with Eq. (11) # Large food foraging strategy
      else
        Update the X_i with Eq. (12) # Small food foraging strategy
      end if
    end if
  end for
  Update the crayfish swarm X and current optimum X_{best}
  $t \leftarrow t + 1$
end while
return X_{best}

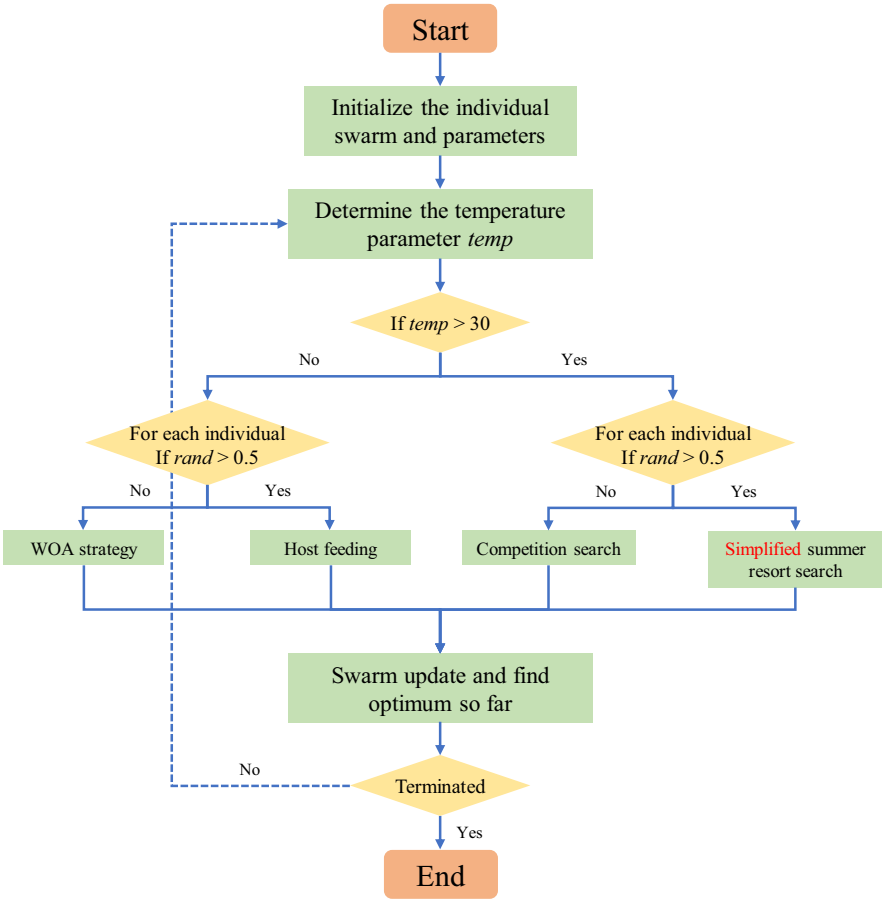


Fig. 1 The flowchart of our proposed HRCOA

In this paper, a binary perception model [49] is employed. This model defines the perceived probability of the monitoring node n_j with respect to the sensor s_i, and it can be expressed using Eq. (14)

$$p(s_i, n_j) = \begin{cases} 1, & if\ dis(s_i, n_j) \leq R \\ 0, & otherwise \end{cases} \tag{14}$$

When $p(s_i, n_j) = 1$, it indicates that the monitoring node n_j is successfully covered by the sensor s_i. Considering a single monitoring node can be perceived by multiple sensors simultaneously, thus, the probability that the monitoring node n_j is perceived by the sensor set S is calculated as follows:

$$p(S, n_j) = 1 - \prod_{s_i \in S} (1 - p(s_i, n_j)) \tag{15}$$

Here, the coverage rate of the node set N over the entire monitoring area is:

$$R_{cov} = \frac{\sum_{n_j \in N} p(S, n_j)}{Area} \tag{16}$$

where $Area$ is the area of the whole monitoring space. Equation (16) is the objective function waiting for maximizing in the WSNs coverage optimization problem.

3 Our proposal: HRCOA

We first provide the overview of our proposal HRCOA in Fig. 1. HRCOA inherits the temperature parameter *temp* to determine the switching between exploitation and exploration modes. Additionally, a random generator is employed to select the specific search operator for each individual. When $temp \leq 30$, the embedded ROA exploitative search operators are activated to enhance the exploitation ability of HRCOA. Moreover, to prevent an overemphasis on the design of the summer resort search operator, we simplify it when $temp > 30$ and random value $rand > 0.5$.

The original summer resort operator calculates the position of the shaded cave, represented as X_{shade}, by averaging the current optimum X_G and the best solution in the latest swarm X_L. This averaging approach proves effective when X_G and X_L exhibit distinct values. However, the original paper on COA does not detail the selection scheme. Upon examining the publicly available COA code, it became apparent that the algorithm employs a greedy selection mechanism. Under this mechanism, if the fitness value of the offspring individual X_i^{t+1} surpasses that of the parent individual X_i^t, the offspring individual replaces the

corresponding parent individual and persists into the subsequent iteration. Equation (17) elucidates this scheme within the context of minimization problems.

$$X_i^{t+1} = \begin{cases} X_i^{t+1}, & if\ f(X_i^{t+1}) < f(X_i^t) \\ X_i^t, & otherwise \end{cases} \tag{17}$$

This straightforward and enhancement-driven selection mechanism mirrors the concept of "survival of the fittest" and has been extensively utilized in the development of MAs [50]. Nonetheless, this strategy, centered on prioritizing elites, ensures the survival of the optimum, thereby leading to the consistency of the current optimum X_G with the best solution in the latest swarm X_L in COA. To mitigate this concern, we have streamlined the summer resort operator to the form delineated in Equation (18).

$$\begin{aligned} X_i^{t+1} &= X_i^t + C_2 \cdot r \cdot (X_{shade} - X_i^t) \\ C_2 &= 2 - (t/T) \\ X_{shade} &= X_{best} \end{aligned} \tag{18}$$

where the mean of X_G and X_L is directly replaced by the current optimum X_{best}.

Additionally, we have integrated both the WOA strategy and the host feeding strategy from the ROA into the exploitation phase of HRCOA. To dynamically alternate between these two strategies, we employ a random generator, ensuring a uniform switch. This straightforward adjustment yields several key advantages:

- It obviates the necessity for complex hyper-parameter determination, such as Q in Eq. (9) and C_1, μ, and σ in Eq. (11), simplifying the optimization process.
- By granting each search operator an equal probability of activation, this modification maintains swarm diversity, fostering a more comprehensive exploration of the solution space.
- Leveraging the superior exploitative behavior inherited from ROA, our approach enhances the algorithm's capacity for locating optimal solutions.

In summary, the pseudocode of HRCOA is shown in Algorithm 3.

Algorithm 3 HRCOA

Require: Population size: N, Dimension: D, Maximum iteration: T
Ensure: Optimum: X_{best}

```
Initialize the swarm X randomly
t ← 0
while t < T do
    Determine the temperature temp by Eq. (6)
    for i = 0 to N do
        Calculate the position of X_shade by Eq. (18)
        if temp > 30 then
            Generate a random number r
            if r < 0.5 then
                Update the X_i with Eq. (18) # Simplified summer resort strategy
            else
                Update the X_i with Eq. (8) # Competition strategy
            end if
        else
            Generate a random number r
            if r > 0.5 then
                Update the X_i with Eq. (3) # WOA strategy
            else
                Update the X_i with Eq. (4) # Host feeding
            end if
        end if
    end for
    Update the swarm X and current optimum X_best
    t ← t + 1
end while
return X_best
```

Table 2 Summary of the CEC2022 benchmark functions: Uni.=Unimodal function, Basic.=Basic function, Hybrid.=Hybrid function, Comp.=Composition function

No	Func	Feature	Optimum
f_1	Shifted and full Rotated Zakharov Function	Uni	300
f_2	Shifted and full Rotated Rosenbrock's Function	Basic.	400
f_3	Shifted and full Rotated Expanded Schaffer's f_6 Function		600
f_4	Shifted and full Rotated Non-Continuous Rastrigin's Function		800
f_5	Shifted and full Rotated Levy Function		900
f_6	Hybrid function 1 (N = 3)	Hybrid.	1800
f_7	Hybrid function 2 (N = 6)		2000
f_8	Hybrid function 3 (N = 5)		2200
f_9	Composition function 1 (N = 5)	Comp.	2300
f_{10}	Composition function 2 (N = 4)		2400
f_{11}	Composition function 3 (N = 5)		2600
f_{12}	Composition function 3 (N = 6)		2700
Search range: $[-100, 100]^D$			

Table 3 Summary of the CEC2020 benchmark functions: *Uni.* Unimodal function, *Multi.* Multimodal function, *Hybrid.* Hybrid function, *Comp.* Composition function

No	Func	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
Search range: $[-100, 100]^D$			

4 Numerical experiments

This section presents a comprehensive set of numerical experiments designed to evaluate the performance of our proposed HRCOA. Section 4.1 provides an overview of the experimental settings, including details about the experimental environments, benchmark functions, and competitor algorithms. Section 4.2 shows the experimental results and provides statistical analysis.

4.1 Experiment settings

4.1.1 Experimental environments and implementation

We conducted our experiments using Python 3.11 as the main experimental environment. The testing and experimentation phase was carried out on a Lenovo Legion R9000P equipped with an AMD Ryzen 7 5800 H processor running at a clock speed of 3.20 GHz, and it has 16GB of RAM. This well-equipped setup was chosen to ensure the fairness of our computational experiments in this study.

4.1.2 Benchmark functions

Our numerical experiments employ four categories of benchmark functions to analyze the performance of HRCOA comprehensively.

- 10-D and 20-D CEC2022 benchmark functions [51], which are listed in Table 2 and are made available through the OpFuNu library [52].
- 50-D and 100-D CEC2020 benchmark functions [53], which are listed in Table 3 and are made available through the OpFuNu library [52].
- Six engineering optimization problems [54] are provided by the ENOPPY library [55].

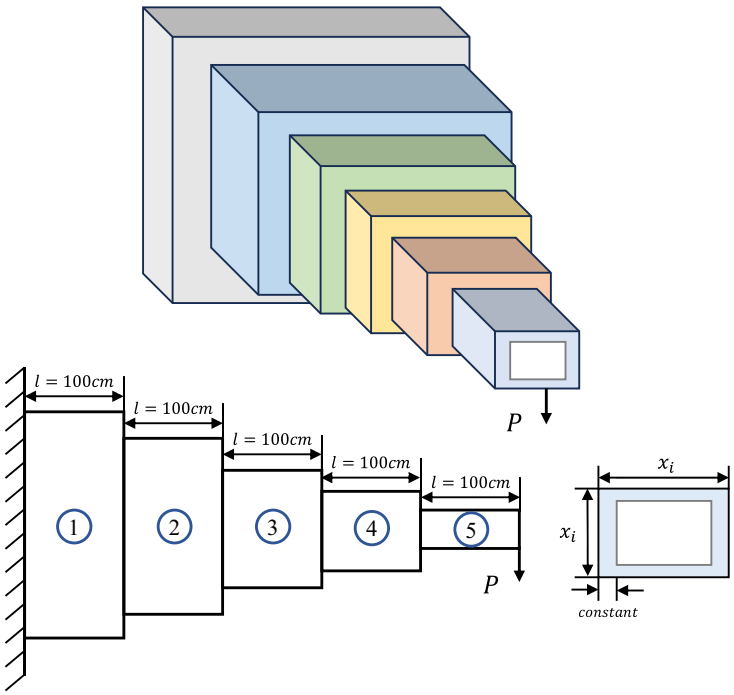


Fig. 2 The demonstration of the cantilever beam problem

- WSNs coverage optimization problem [56].

Here, we considered six engineering problems, namely, the cantilever beam problem (CBD), corrugated bulkhead problem (CBHD), I-beam problem (IBD), piston lever problem (PLD), pressure vessel problem (PVP), and speed reducer problem (SRD). Note that $f(x)$ is the objective function and $g_i(x)$ denotes the i^{th} constraint function.

cantilever beam problem (CBD): The mathematical model and the visualized demonstration of the CBD are shown in Eq. (19) and Fig. 2, respectively.

$$\begin{aligned} &\textbf{min}\ f(X) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5) \\ &\textbf{s.t.}\ g(X) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0 \\ &\textbf{where}\ 0.01 \le x_i \le 100,\ i \in \{1,2,3,4,5\} \end{aligned} \tag{19}$$

The objective of CBD is to minimize the total mass of the cantilever beam while ensuring that the bearing capacity is met, as described in [57].

corrugated bulkhead problem (CBHD): The mathematical model of the CBHD is presented in Eq. (20).

$$\begin{aligned} &\textbf{min}\ f(X) = \frac{5.885x_4(x_1 + x_3)}{x_1 + \sqrt{|x_3^2 - x_2^2|}} \\ &\textbf{s.t.}\ g_1(X) = -x_4x_2\left(0.4x_1 + \frac{x_3}{6}\right) + 8.94\left(x_1 + \sqrt{|x_3^2 - x_2^2|}\right) \le 0 \\ &\qquad g_2(X) = -x_4x_2^2\left(0.2x_1 + \frac{x_3}{12}\right) \\ &\qquad\qquad + 2.2\left(8.94\left(x_1 + \sqrt{|x_3^2 - x_2^2|}\right)\right)^{4/3} \le 0 \\ &\qquad g_3(X) = -x_4 + 0.0156x_1 + 0.15 \le 0 \\ &\qquad g_4(X) = -x_4 + 0.0156x_3 + 0.15 \le 0 \\ &\qquad g_5(X) = -x_4 + 1.05 \le 0 \\ &\qquad g_6(X) = -x_3 + x_2 \le 0 \\ &\textbf{where}\ 0 \le x_1, x_2, x_3 \le 100 \\ &\qquad 0 \le x_4 \le 5 \end{aligned} \tag{20}$$

The objective of CBHD is to minimize the weight of a corrugated bulkhead, where the design variables are the width (x_1), depth (x_2), length (x_3), and plate thickness (x_4) [58].

I-beam problem (IBD): The mathematical model and the visualized demonstration of the IBD are shown in Eq. (21) and Fig. 3, respectively.

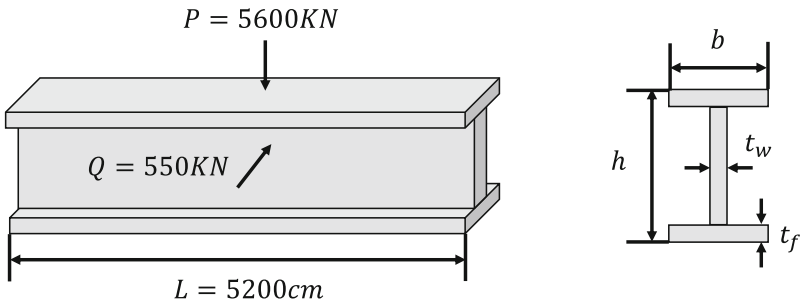


Fig. 3 The demonstration of the I-beam problem

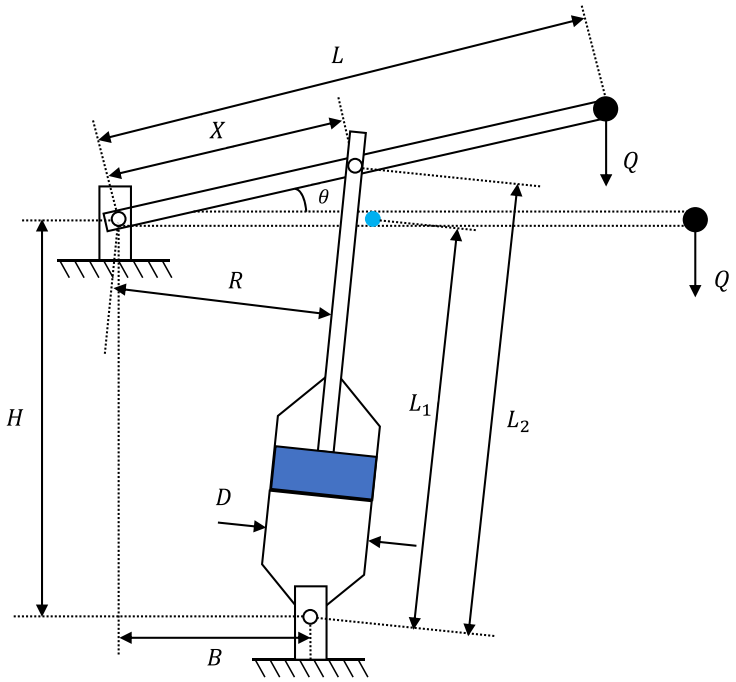


Fig. 4 The demonstration of the piston lever problem

$$
\begin{aligned}
\textbf{min}\ f(X) =& \frac{5000}{x_3(x_2-2x_4)^3/12 + (x_1x_4^3/6) + 2x_1x_4(x_2-x_4/2)^2} \\
\textbf{s.t.}\ g_1(X) =& 2x_1x_3 + x_3(x_2-2x_4) - 300 \le 0 \\
g_2(X) =& \frac{180000x_2}{x_3(x_2-2x_4)^3 + 2x_1x_3(4x_4^2 + 3x_2(x_2-2x_4))} + \\
& \frac{15000x_1}{(x_2-2x_4)x_3^2 + 2x_3x_1^3} - 56 \le 0 \\
\textbf{where}\ & 10 \le x_1 \le 50 \\
& 10 \le x_2 \le 80 \\
& 0.9 \le x_3, x_4 \le 5
\end{aligned}
\tag{21}
$$

This problem aims to minimize the vertical deflection during the design phase [27]. The structural components under consideration include the width of flange $b(=x_1)$, the height of section $h(=x_2)$, the thickness of the web $t_w(=x_3)$ and the thickness of the flange $t_f(=x_4)$.

piston lever problem (PLD): The mathematical model and the visualized demonstration of the PLD are shown in Eq. (22) and Fig. 4, respectively.

$$
\begin{aligned}
\textbf{min}\ & f(X) = \frac{1}{4}\pi x_3^2(L_2 - L_1) \\
\textbf{s.t.}\ & g_1(X) = QL\cos(\theta) - RF \le 0 \\
& g_2(X) = Q(L - x_4) - M_{max} \le 0 \\
& g_3(X) = 1.2(L_2 - L_1) - L_1 \le 0 \\
& g_4(X) = \frac{x_3}{2} - x_2 \le 0 \\
\textbf{where}\ & R = \frac{|-x_4(x_4\sin\theta + x_1) + x_1(x_2 - x_4\cos\theta)|}{\sqrt{(x_4 - x_2)^2 + x_1^2}} \\
& F = \frac{\pi P x_3^2}{4} \\
& L_1 = \sqrt{(x_4 - x_2)^2 + x_1^2} \\
& L_2 = \sqrt{(x_4\sin\theta + x_1)^2 + (x_2 - x_4\cos\theta)^2} \\
& \theta = \pi/4 \\
& Q = 10,000 \\
& L = 240 \\
& M_{max} = 1,800,000 \\
& P = 1,500 \\
& 0.05 \le x_1, x_2, x_4 \le 500 \\
& 0.05 \le x_3 \le 120
\end{aligned}
\tag{22}
$$

The objective of the PLD, as described in [58], is to minimize the oil volume by determining the values of the decision variables $H(= x_1)$, $B(= x_2)$, $D(= x_3)$, and $X(= x_4)$, while the lever of the piston is lifted up from 0 to $\pi/4$.

pressure vessel problem (PVP): The mathematical model and the visualized demonstration of the PVP are shown in Eq. (23) and Fig. 5, respectively.

$$
\begin{aligned}
\textbf{min}\ & f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 \\
& \quad + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
\textbf{s.t.}\ & g_1(X) = -x_1 + 0.0193x_3 \le 0 \\
& g_2(X) = -x_2 + 0.00954x_3 \le 0 \\
& g_3(X) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \le 0 \\
& g_4(X) = x_4 - 240 \le 0 \\
\textbf{where}\ & 0 \le x_1, x_2 \le 99 \\
& 10 \le x_3, x_4 \le 200
\end{aligned}
\tag{23}
$$

The objective of the PVP is to minimize the total cost, which encompasses material, forming, and welding [59]. This problem is characterized by four decision variables: the thickness of the shell $T_s(= x1)$, the thickness of the head $T_h(= x2)$, the inner radius $R(= x3)$, and the length of the cylindrical section of the vessel, not including the head $L(= x4)$.

speed reducer problem (SRD): The mathematical model of the SRD is shown in Eq. (24).

$$
\begin{aligned}
\textbf{min}\ & f(X) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0924) \\
& \quad - 1.508x_1(x_6^2 + x_7^2) + \\
& \quad 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \\
\textbf{s.t.}\ & g_1(X) = \frac{27}{x_1x_2^2x_3} - 1 \le 0 \\
& g_2(X) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0 \\
& g_3(X) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \le 0 \\
& g_4(X) = \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \le 0 \\
& g_5(X) = \frac{\sqrt{(745x_4/x_2x_3)^2 + 16900000}}{110x_6^3} - 1 \le 0 \\
& g_6(X) = \frac{\sqrt{(745x_5/x_2x_3)^2 + 157500000}}{85x_7^3} - 1 \le 0 \\
& g_7(X) = \frac{x_2x_3}{40} - 1 \le 0 \\
& g_8(X) = \frac{5x_2}{x_1} - 1 \le 0 \\
& g_9(X) = \frac{x_1}{12x_2} - 1 \le 0 \\
& g_{10}(X) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0 \\
& g_{11}(X) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0 \\
\text{where}\ & 2.6 \le x_1 \le 2.6 \\
& 0.7 \le x_2 \le 0.8 \\
& 17 \le x_3 \le 28 \\
& 7.3 \le x_4x_5 \le 8.3 \\
& 2.9 \le x_6 \le 3.9 \\
& 5 \le x_7 \le 5.5
\end{aligned}
\tag{24}
$$

The objective of the SRD is to minimize the weight of the speed reducer while adhering to 11 constraints [60]. This problem is characterized by seven variables: face width x_1,

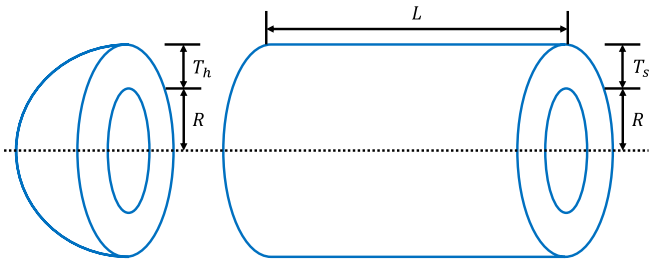


Fig. 5 The demonstration of the pressure vessel problem

the module of teeth x_2, the number of teeth in the pinion x_3, the length of the first shaft between bearings x_4, length of the second shaft between bearings x_5, the diameter of first shafts x_6, and the diameter of second shafts x_7.

For the WSNs coverage problem, we use the experiment setting as described in [56]. Specifically, the sensing radius is set to 5, the monitoring range is a two-dimensional space within $[0, 50]^2$, and the numbers of sensors are configured as $\{32, 42, 54\}$, respectively.

4.1.3 Competitor algorithms and parameters

In our numerical experiments, we compare our proposal with eight MAs, namely the PSO [8], covariance matrix adaptation evolution strategy (CMAES) [36], circle search algorithm (CSA) [37], Siberian tiger optimization (STO) [38], pelican optimization algorithm (POA) [39], serval optimization algorithm (SOA) [40], energy valley optimizer (EVO) [41], and the original COA [35]). All of these MAs are provided by MEALPY library [61], and the summarized parameter settings are listed in Table 4.

For CEC2020 and CEC2022 benchmark functions, the population size for all algorithms is set to 100, and the maximum function evaluation (FE) is 1,000 $\times D$ (D=dimension). In the case of engineering problems, the population size is also 100 with a maximum of 20,000 FEs. For the WSNs coverage problems, the population size is set to 30 and the maximum iteration is 100. To mitigate the impact of randomness during optimization, each MA is independently run 30 trial times.

In addition, we have incorporated the static penalty function [62] into MAs to address constrained engineering optimization problems, and the formulation is presented in Eq. (25).

$$F(X_i) = f(X_i) + w \cdot \sum_{i=1}^{m} (\max(0, g_i(X_i))) \tag{25}$$

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

Table 4 Parameters of competitor algorithms for evaluating HRCOA

EAs	Parameters	Value
PSO (1995)	Inertia factor w	1
	Acceleration coefficients c_1 and c_2	2.05
	Max. and min. speed	2, -2
CMAES (2001)	Parameter-free	
CSA (2022)	Parameter-free	
STO (2022)	Parameter-free	
POA (2022)	Parameter-free	
SOA (2022)	Parameter-free	
EVO (2023)	Parameter-free	
COA (2023)	Temperature threshold	30
	C_1 and C_3	0.2 and 3
	μ and σ	25 and 3
HRCOA	Temperature threshold	30
	C	0.1

4.2 Experimental results

4.2.1 Case study 1: performance on CEC2022 benchmark functions

This section summarizes the experimental and statistical results among nine competitor MAs on 10-D and 20-D CEC2022 benchmark functions. Tables 5 and 6 provide the detailed mean, std, and statistical results. Figures 6 and 7 visualize the convergence curves.

Here, we have collected the optimal fitness values from 30 trial runs of each optimization algorithm. We then applied the Friedman test to assess the significance of the results. If statistical significance is observed, the Mann–Whitney U test is employed to check the p-value of every pair of algorithms, followed by correction using the Holm multiple comparison test [63] to identify statistical significance. Symbols such as $+$, $\approx$, and $-$ are used to indicate whether our proposed HRCOA is significantly better, shows no significance, or is significantly worse compared to the specific competitor algorithm, with the best-performing algorithm highlighted in bold.

4.2.2 Case study 2: performance on CEC2020 benchmark functions

Tables 7 and 8 summarize the experimental and statistical results on 50-D and 100-D CEC2020 benchmark functions, and the convergence curves are provided in Fig. 8.

4.2.3 Case study 3: performance on engineering problems

Table 9 summarizes the experimental and statistical results on six engineering problems, and the corresponding convergence curves are provided in Fig. 9.

4.2.4 Case study 4: performance on WSNs coverage problems

Table 10 summarizes the experimental results on WSNs coverage optimization problems, and we draw the optimal distribution of sensors in Figs. 10 and 11.

4.2.5 Case study 5: ablation experiments on CEC2020 and CEC2022

To evaluate the efficacy of our proposed strategies, namely the simplified summer resort operator and the hybridization with ROA, we conduct ablation experiments. These experiments are conducted on both 50-D CEC2020 and

Table 5 Experimental and statistical results on 10-D CEC2022 benchmark functions. f_1: Unimodal function; $f_2 - f_5$: Basic functions; $f_6 - f_8$: Hybrid functions; $f_9 - f_{12}$: Composition functions; mean and std: the mean and the standard deviation of 30 trial runs

Func.		PSO	CMAES	CSA	STO	POA	SOA	EVO	COA	HRCOA
f_1	Mean	1.66e+03 +	4.22e+02 +	8.40e+02 +	5.52e+03 +	9.50e+02 +	4.83e+03 +	2.61e+03 +	3.03e+02 ≈	**3.02e+02**
	Std	1.56e+03	4.01e+01	1.30e+03	2.30e+03	6.60e+02	1.34e+03	1.63e+03	4.21e+00	2.83e+00
f_2	Mean	5.16e+02 +	4.17e+02 +	5.43e+02 +	8.99e+02 +	4.36e+02 +	5.84e+02 +	4.73e+02 +	4.20e+02 +	**4.10e+02**
	Std	4.47e+01	3.04e+00	1.62e+02	2.45e+02	3.74e+01	7.73e+01	3.30e+01	2.75e+01	1.26e+01
f_3	Mean	6.00e+02 +	6.00e+02 +	6.00e+02 +	6.00e+02 +	6.00e+02 +	6.00e+02 +	6.00e+02 +	6.00e+02 +	**6.00e+02**
	Std	6.02e−03	6.56e−04	3.12e−02	9.05e−02	2.56e−02	4.19e−02	1.28e−02	1.28e−04	1.43e−06
f_4	Mean	8.01e+02 +	8.01e+02 +	8.01e+02 ≈	8.01e+02 +	**8.00e+02** −	8.01e+02 +	8.00e+02 ≈	8.00e+02 ≈	8.00e+02
	Std	1.92e−01	1.57e−01	5.11e−01	3.11e−01	4.58e−02	1.52e−01	1.30e−01	2.09e−01	2.21e−01
f_5	Mean	9.01e+02 +	9.00e+02 +	9.02e+02 +	9.02e+02 +	9.01e+02 +	9.01e+02 +	9.00e+02 +	9.01e+02 +	**9.00e+02**
	Std	6.04e−01	6.38e−02	2.21e+00	8.11e−01	3.35e−01	2.69e−01	2.10e−01	1.47e+00	2.65e−01
f_6	Mean	2.02e+06 +	**4.10e+03** −	5.32e+04 +	2.88e+07 +	2.20e+04 ≈	7.44e+07 +	9.92e+05 +	3.88e+04 ≈	2.73e+04
	Std	4.47e+06	6.09e+02	1.86e+04	3.74e+07	9.23e+03	7.16e+07	1.89e+06	2.34e+04	1.51e+04
f_7	Mean	2.12e+03 +	2.08e+03 +	2.22e+03 +	2.18e+03 +	2.05e+03 ≈	2.31e+03 +	2.05e+03 ≈	2.06e+03 ≈	**2.04e+03**
	Std	5.24e+01	1.48e+01	1.58e+02	1.30e+02	2.21e+01	1.41e+02	2.41e+01	4.38e+01	1.08e+01
f_8	Mean	2.50e+03 +	2.23e+03 ≈	3.86e+03 +	2.66e+03 +	2.25e+03 +	2.30e+03 ≈	5.27e+05 +	**2.23e+03** ≈	2.25e+03
	Std	4.84e+02	1.94e+00	1.94e+03	2.64e+02	7.33e+01	9.30e+01	1.66e+06	3.54e+00	7.36e+01
f_9	Mean	2.63e+03 ≈	2.60e+03 ≈	2.79e+03 +	2.96e+03 +	**2.52e+03** ≈	2.97e+03 +	2.65e+03 +	2.62e+03 ≈	2.56e+03
	Std	1.48e+02	1.11e+02	1.02e+02	2.30e+02	1.76e+02	1.40e+02	1.73e+02	1.81e+02	1.69e+02
f_{10}	Mean	2.62e+03 +	**2.61e+03** ≈	2.70e+03 +	2.70e+03 +	2.62e+03 +	2.70e+03 +	2.65e+03 +	2.65e+03 +	2.61e+03
	Std	8.24e+00	1.07e+00	1.16e+02	7.84e+01	2.12e+01	5.76e+01	6.02e+01	7.13e+01	3.03e+01
f_{11}	Mean	2.63e+03 ≈	2.74e+03 +	2.79e+03 +	2.75e+03 +	**2.62e+03** ≈	2.71e+03 +	2.66e+03 +	2.69e+03 +	2.63e+03
	Std	2.07e+01	3.01e+02	3.43e+02	1.04e+02	2.73e+01	5.56e+01	1.57e+02	2.63e+02	1.57e+02
f_{12}	Mean	2.89e+03 +	2.87e+03 +	2.90e+03 +	3.05e+03 +	2.87e+03 +	3.10e+03 +	2.87e+03 +	2.87e+03 +	**2.87e+03**
	Std	1.03e+01	5.35e−01	3.76e+01	7.29e+01	4.62e+00	6.20e+01	6.37e+00	8.79e+00	1.09e+00
+/≈/− Summary		10/2/0	8/3/1	11/1/0	12/0/0	7/4/1	11/1/0	10/2/0	6/6/0	–

Table 6 Experimental and statistical results on 20-D CEC2022 benchmark functions

Func.		PSO	CMAES	CSA	STO	POA	SOA	EVO	COA	HRCOA
f_1	Mean	1.36e+04 +	2.55e+03 +	2.51e+03 +	2.27e+04 +	6.39e+03 +	1.73e+04 +	1.08e+04 +	3.41e+02 +	**3.29e+02**
	Std	5.98e+03	4.33e+02	1.28e+03	4.76e+03	2.53e+03	2.82e+03	6.00e+03	3.91e+01	5.65e+01
f_2	Mean	7.14e+02 +	5.10e+02 +	5.65e+02 +	1.82e+03 +	6.33e+02 +	1.89e+03 +	6.42e+02 +	4.70e+02 ≈	**4.60e+02**
	Std	1.99e+02	1.54e+01	6.27e+01	4.49e+02	7.46e+01	3.99e+02	6.78e+01	1.81e+01	1.69e+01
f_3	Mean	6.00e+02 +	6.00e+02 +	6.00e+02 +	6.01e+02 +	6.00e+02 +	6.01e+02 +	6.00e+02 +	6.00e+02 +	**6.00e+02**
	Std	6.54e−02	7.24e−03	1.56e−01	1.61e−01	1.65e−01	8.12e−02	3.72e−02	5.05e−05	7.03e−07
f_4	Mean	8.04e+02 +	8.04e+02 +	8.03e+02 +	8.04e+02 +	**8.01e+02** −	8.03e+02 +	8.01e+02 ≈	8.01e+02 ≈	8.01e+02
	Std	4.04e−01	3.57e−01	1.36e+00	6.27e−01	1.87e−01	4.69e−01	5.42e−01	4.38e−01	5.54e−01
f_5	Mean	9.05e+02 +	9.03e+02 +	9.06e+02 +	9.07e+02 +	9.03e+02 +	9.06e+02 +	9.02e+02 ≈	9.02e+02 ≈	**9.02e+02**
	Std	3.33e+00	4.79e−01	2.89e+00	2.05e+00	1.15e+00	1.05e+00	1.04e+00	9.05e−01	1.09e+00
f_6	Mean	2.71e+08 +	5.28e+06 +	2.80e+08 +	2.13e+09 +	1.20e+05 ≈	1.59e+09 +	1.76e+07 +	**7.61e+04** ≈	7.96e+04
	Std	2.11e+08	2.08e+06	1.07e+09	9.29e+08	1.32e+05	8.33e+08	3.36e+07	3.15e+04	3.23e+04
f_7	Mean	2.96e+03 +	2.29e+03 +	2.76e+03 +	3.16e+03 +	2.27e+03 +	2.35e+03 +	2.24e+03 +	2.18e+03 +	**2.10e+03**
	Std	4.13e+02	7.37e+01	5.31e+02	3.71e+02	1.90e+02	8.69e+01	1.68e+02	1.80e+02	5.03e+01
f_8	Mean	4.65e+07 +	**2.41e+03** −	4.08e+11 +	8.25e+08 +	3.22e+03 ≈	3.54e+03 ≈	3.83e+04 +	2.63e+03 −	3.36e+03
	Std	1.94e+08	5.48e+01	2.19e+12	3.39e+09	4.96e+02	2.83e+02	1.69e+05	3.70e+02	1.10e+03
f_9	Mean	3.18e+03 +	2.65e+03 +	3.57e+03 +	5.79e+03 +	2.82e+03 +	4.66e+03 +	2.88e+03 +	2.64e+03 ≈	**2.64e+03**
	Std	2.21e+02	3.92e+00	8.09e+02	1.27e+03	9.69e+01	4.12e+02	1.08e+02	1.12e+01	9.96e+00
f_{10}	Mean	**2.84e+03** ≈	3.14e+03 +	4.65e+03 +	4.33e+03 +	3.07e+03 ≈	2.89e+03 ≈	3.16e+03 +	3.59e+03 +	3.06e+03
	Std	3.90e+01	9.33e+02	1.30e+03	1.52e+03	5.50e+02	5.22e+01	6.84e+02	9.21e+02	7.65e+02
f_{11}	Mean	2.73e+03 −	**2.62e+03** −	3.52e+03 +	5.72e+03 +	2.74e+03 −	5.70e+03 +	2.81e+03 ≈	2.98e+03 +	2.91e+03
	Std	1.55e+02	3.11e+00	1.58e+03	1.78e+03	1.01e+02	1.29e+03	3.80e+02	6.27e+02	6.13e+02
f_{12}	Mean	3.23e+03 +	**2.96e+03** ≈	3.15e+03 +	3.68e+03 +	3.03e+03 +	3.44e+03 +	3.02e+03 +	3.01e+03 +	2.96e+03
	Std	6.97e+01	7.64e+00	1.53e+02	1.57e+02	5.56e+01	6.49e+01	3.62e+01	3.83e+01	1.43e+01
+/≈/− Summary		10/1/1	9/1/2	12/0/0	12/0/0	7/3/2	10/2/0	9/3/0	6/5/1	–

20-D CEC2022 benchmark functions. The aim is to assess the significance of each proposed strategy independently, allowing for a comprehensive understanding of their respective contributions to optimization performance.

4.2.6 Case study 6: sensitivity analysis experiments on CEC2020 and CEC2022

Temperature threshold *temp* plays an important role in switching from different search strategies. This section offers the experimental results and statistical analyses on sensitivity analysis experiments on CEC2020 and CEC2022 benchmark functions. The alternative temperature thresholds are $\{25, 27.5, 30, 32.5\}$.

5 Discussion

In this section, we conduct a computational complexity analysis of HRCOA and analyze its performance on a range of test problems, including CEC2022 and CEC2020 benchmark functions, engineering problems, and WSNs coverage problems, respectively.

5.1 Computational complexity analysis of HRCOA

Assuming the population size is N, the dimension of the problem is D, and the maximum iteration is T, the analysis of HRCOA can be divided into two key components: swarm initialization and iterative search.

Similar to most MAs, HRCOA utilizes random initialization, resulting in a computational complexity of $O(N \times D)$. During the iterative search phase, four operators have probabilities of being adopted: WOA strategy, host feeding strategy, competition strategy, and simplified summer resort strategy. For the sake of simplicity, we analyze the computational complexity of these strategies independently for a swarm of individuals.

- WOA strategy: $O(N \times D)$.
- host feeding strategy: $O(N \times D)$.
- competition strategy: $O(N \times D)$.

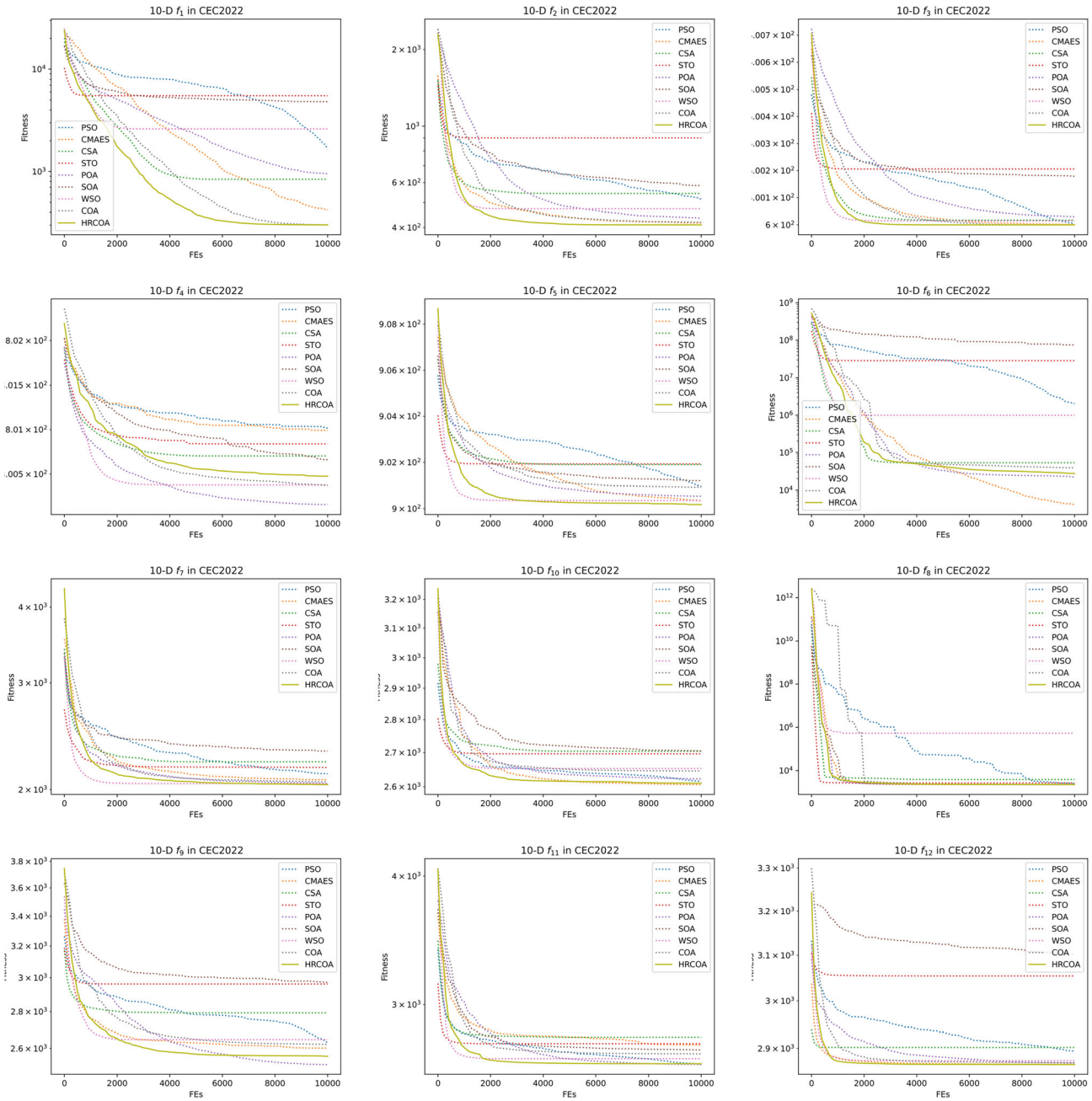


Fig. 6 Convergence curves of nine MAs on 10-D CEC2022 benchmark functions

- simplified summer resort strategy: $O(N \times D)$.

Since these search strategies do not involve the additional operator such as the sorting process, they theoretically possess identical computational complexity. Besides, the computational complexity of the selection and update processes is $O(N)$.

In summary, the total computational complexity of HRCOA is

$$\begin{aligned} O(HRCOA) &= O(N \times D) + O(T \times (N \times D + D)) \\ &:= O(T \times N \times D) \end{aligned} \tag{26}$$

5.2 Performance analysis on CEC2022 benchmark functions

We conducted numerical experiments on 10-D and 20-D CEC2022 benchmark functions to evaluate the performance of nine MAs in handling low-dimensional tasks.

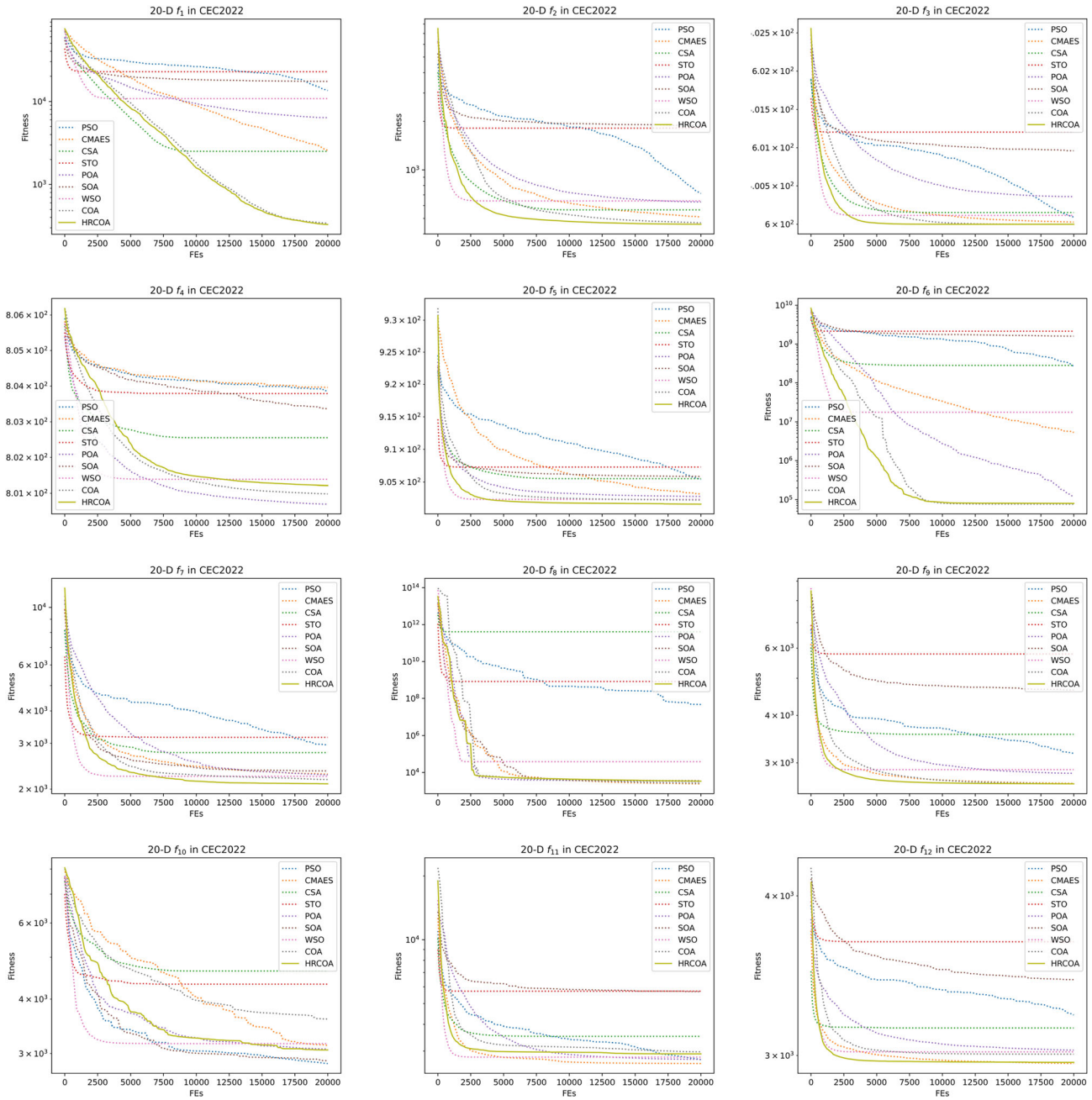


Fig. 7 Convergence curves of nine MAs on 20-D CEC2022 benchmark functions

The selected benchmark functions, ranging from unimodal to composite, collectively represent a diverse set of optimization challenges. The experimental and statistical results presented in Tables 5 and 6 demonstrate the effectiveness and efficiency of our proposed HRCOA. In most instances, HRCOA either outperforms or statistically equals the competitor algorithms. When focusing on the improvement over the original COA, HRCOA achieved statistical summaries of 6/6/0 and 6/5/1 on 10-D and 20-D CEC2022 benchmark functions, respectively. These results confirm that the simplified summer resort strategy and the integration of ROA exploitation operators can enhance the optimization capacity and accelerate the convergence of COA.

In addition, we notice that HRCOA exhibits significant performance inferiority to the competitor algorithms in some instances, such as POA in 10-D and 20-D f_4, CMAES in 10-D f_6, CMAES and COA in 20-D f_8, CMAES and POA in 20-D f_{11}. These performance deteriorations are, in fact, inevitable, and we turn to the No Free Lunch Theorem (NFLT) [64] to shed light on this phenomenon. NFLT states that, on average, every pair of MAs performs equally

Table 7 Experimental and statistical results on 50-D CEC2020 benchmark functions. f_1: Unimodal function; $f_2 - f_4$: Multimodal functions; $f_5 - f_7$: Hybrid functions; $f_8 - f_{10}$: Composition functions

Func.		PSO	CMAES	CSA	STO	POA	SOA	EVO	COA	HRCOA
f_1	Mean	1.06e+11 +	1.11e+10 +	9.80e+09 +	9.80e+10 +	4.57e+10 +	1.05e+11 +	3.95e+10 +	4.15e+08 +	**1.72e+06**
	Std	2.51e+10	3.09e+09	2.74e+09	8.00e+09	1.05e+10	8.83e+09	8.75e+09	7.52e+08	1.54e+06
f_2	Mean	1.14e+13 +	1.34e+12 +	1.23e+12 +	1.08e+13 +	4.63e+12 +	1.20e+13 +	4.21e+12 +	4.95e+10 +	**3.23e+08**
	Std	2.81e+12	4.36e+11	3.12e+11	1.12e+12	1.19e+12	1.23e+12	9.14e+11	7.61e+10	5.25e+08
f_3	Mean	4.31e+12 +	4.05e+11 +	3.96e+11 +	3.90e+12 +	1.38e+12 +	4.01e+12 +	1.22e+12 +	8.62e+09 +	**6.07e+07**
	Std	1.16e+12	8.35e+10	1.31e+11	3.97e+11	3.25e+11	3.59e+11	2.33e+11	1.09e+10	6.24e+07
f_4	Mean	1.94e+06 +	1.01e+04 +	1.14e+04 +	2.14e+06 +	8.36e+04 +	1.81e+06 +	1.07e+05 +	2.00e+03 +	**1.96e+03**
	Std	1.26e+06	6.69e+03	1.27e+04	7.87e+05	6.04e+04	4.42e+05	1.11e+05	3.00e+01	1.89e+01
f_5	Mean	2.09e+08 +	2.39e+06 ≈	3.34e+07 +	4.10e+08 +	7.54e+06 +	2.31e+08 +	2.18e+07 +	**7.36e+05** −	2.31e+06
	Std	1.27e+08	4.70e+05	7.67e+07	1.91e+08	9.47e+06	7.72e+07	1.64e+07	2.88e+05	2.04e+06
f_6	Mean	4.30e+09 +	3.51e+04 +	4.03e+07 +	2.19e+09 +	5.74e+07 +	6.70e+07 +	8.72e+07 +	1.13e+04 +	**6.75e+03**
	Std	3.52e+09	6.68e+03	1.56e+08	1.08e+09	1.42e+08	2.47e+07	6.85e+07	7.87e+03	2.78e+03
f_7	Mean	3.61e+09 +	1.27e+07 +	3.63e+08 +	1.15e+10 +	4.35e+07 +	1.04e+10 +	1.81e+08 +	1.32e+06 ≈	**1.32e+06**
	Std	2.59e+09	3.96e+06	1.08e+09	5.04e+09	3.69e+07	4.04e+09	1.95e+08	1.49e+06	1.22e+06
f_8	Mean	7.70e+03 +	2.66e+03 +	6.73e+03 +	1.09e+04 +	5.51e+03 +	9.56e+03 +	2.84e+03 +	3.84e+03 +	**2.53e+03**
	Std	3.20e+03	3.91e+01	2.89e+03	1.48e+03	1.97e+03	7.21e+02	9.98e+01	1.97e+03	3.64e+01
f_9	Mean	5.29e+04 +	1.12e+04 +	1.35e+04 +	6.32e+04 +	4.28e+04 +	6.32e+04 +	4.01e+04 +	4.32e+03 +	**3.09e+03**
	Std	1.41e+04	1.36e+03	3.07e+03	2.19e+03	5.94e+03	1.83e+03	8.14e+03	1.02e+03	4.40e+02
f_{10}	Mean	1.93e+04 +	4.58e+03 +	5.17e+03 +	2.22e+04 +	7.87e+03 +	2.10e+04 +	8.80e+03 +	3.50e+03 +	**3.41e+03**
	Std	7.12e+03	2.49e+02	5.53e+02	3.29e+03	1.28e+03	2.66e+03	1.00e+03	1.51e+02	1.18e+02
+/≈/− Summary		10/0/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0	10/0/0	8/1/1	–

across all conceivable black-box optimization problems. Thus, if an algorithm excels in a particular category of problems, it must, by necessity, deteriorate on the remaining problems to maintain an equilibrium in average performance.

5.3 Performance analysis on CEC2020 benchmark functions

The curse of dimensionality, as pointed out by Mario et al. [65], leads to an exponential growth of the search space as the number of decision variables increases. This phenomenon significantly slows down the convergence speed of MAs. As a result, high-dimensional optimization becomes a formidable challenge for algorithms like HRCOA.

The numerical experiments conducted on 50-D and 100-D CEC2020 benchmark functions serve as a challenge to the optimization capacities of HRCOA in both median- and high-dimensional optimization scenarios. In the case of the unimodal function f_1, as evidenced by the results in Tables 7 and 8, as well as the convergence curves depicted in Fig. 8, HRCOA shows remarkable exploitation abilities.

Furthermore, HRCOA demonstrates a well-balanced capacity between exploration and exploitation, along with an excellent ability to escape local optima. This characteristic is evident in the remaining experimental results and their corresponding convergence curves. In many cases, competing MAs tend to become trapped in local optima, resulting in premature convergence. This tendency can be observed from the convergence behaviors depicted in Fig. 8, where the competitor algorithms show flattened convergence curves in the late phases of optimization.

Notably, HRCOA demonstrates exceptional convergence characteristics and an impressive search capability. Even when compared to the original COA, HRCOA achieves remarkable statistical analysis summaries of 8/1/1 and 7/2/1 on 50-D and 100-D CEC2020 benchmark functions, respectively. These results underscore the superiority of HRCOA in addressing median- and high-dimensional optimization tasks. We attribute this success to the incorporation of the exploitation operators of ROA and the simplified summer resort strategy within HRCOA.

5.4 Performance analysis on engineering problems

Since the optimization performance on six engineering problems can reflect the capacity of MAs in real-world scenarios, we summarize the performance indicators,

Table 8 Experimental and statistical results on 100-D CEC2020 benchmark functions

Func.		PSO	CMAES	CSA	STO	POA	SOA	EVO	COA	HRCOA
f_1	Mean	3.28e+11 +	8.40e+10 +	4.07e+10 +	2.48e+11 +	1.28e+11 +	2.41e+11 +	1.48e+11 +	5.19e+09 +	**4.36e+07**
	Std	4.54e+10	1.42e+10	8.77e+09	1.25e+10	1.77e+10	4.62e+09	1.28e+10	2.91e+09	4.31e+07
f_2	Mean	3.19e+13 +	8.88e+12 +	4.22e+12 +	2.86e+13 +	1.36e+13 +	2.75e+13 +	1.40e+13 +	7.04e+11 +	**6.91e+09**
	Std	3.98e+12	1.66e+12	7.74e+11	1.56e+12	2.10e+12	6.53e+11	1.89e+12	5.65e+11	1.05e+10
f_3	Mean	1.20e+13 +	3.01e+12 +	1.43e+12 +	9.99e+12 +	4.68e+12 +	8.79e+12 +	5.05e+12 +	1.75e+11 +	**7.60e+09**
	Std	1.71e+12	6.10e+11	4.20e+11	4.61e+11	7.31e+11	9.46e+10	6.98e+11	9.12e+10	2.43e+10
f_4	Mean	9.32e+06 +	7.88e+04 +	3.53e+04 +	5.57e+06 +	2.47e+05 +	4.88e+06 +	6.52e+05 +	2.59e+03 +	**2.10e+03**
	Std	4.21e+06	3.02e+04	2.15e+04	1.47e+06	8.58e+04	1.04e+06	2.63e+05	2.66e+02	5.09e+01
f_5	Mean	1.18e+09 +	2.89e+07 +	5.18e+07 +	1.43e+09 +	7.08e+07 +	7.06e+08 +	1.83e+08 +	**5.06e+06** −	7.45e+06
	Std	4.45e+08	6.80e+06	3.64e+07	4.52e+08	2.38e+07	1.21e+08	5.64e+07	1.76e+06	3.61e+06
f_6	Mean	2.77e+10 +	3.24e+05 +	8.67e+06 +	3.36e+10 +	1.61e+09 +	3.80e+10 +	2.21e+09 +	**1.65e+04** ≈	1.80e+04
	Std	2.00e+10	1.15e+05	1.48e+07	1.15e+10	1.17e+09	8.68e+09	1.69e+09	7.54e+03	1.22e+04
f_7	Mean	2.11e+10 +	3.55e+07 +	3.17e+08 +	3.14e+10 +	9.31e+08 +	2.34e+10 +	1.90e+09 +	4.70e+06 ≈	**4.64e+06**
	Std	1.32e+10	1.10e+07	3.11e+08	6.78e+09	7.84e+08	4.68e+09	6.33e+08	2.97e+06	2.77e+06
f_8	Mean	2.54e+04 +	3.54e+03 +	1.82e+04 +	2.57e+04 +	1.55e+04 +	2.27e+04 +	4.19e+03 +	6.53e+03 +	**2.60e+03**
	Std	8.22e+03	2.52e+02	4.20e+03	1.93e+03	5.14e+03	1.11e+03	6.11e+02	5.08e+03	4.84e+01
f_9	Mean	2.28e+05 +	6.75e+04 +	6.23e+04 +	1.76e+05 +	1.25e+05 +	1.61e+05 +	1.41e+05 +	2.16e+04 +	**4.62e+03**
	Std	4.26e+04	8.73e+03	2.38e+04	4.62e+03	1.25e+04	7.00e+02	9.10e+03	4.75e+03	2.01e+03
f_{10}	Mean	4.39e+04 +	8.62e+03 +	6.71e+03 +	3.76e+04 +	1.21e+04 +	3.89e+04 +	1.46e+04 +	4.14e+03 +	**3.72e+03**
	Std	8.89e+03	1.11e+03	6.17e+02	4.84e+03	1.89e+03	4.18e+03	2.02e+03	2.17e+02	1.06e+02
+/≈/− Summary		10/0/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0	10/0/0	7/2/1	–

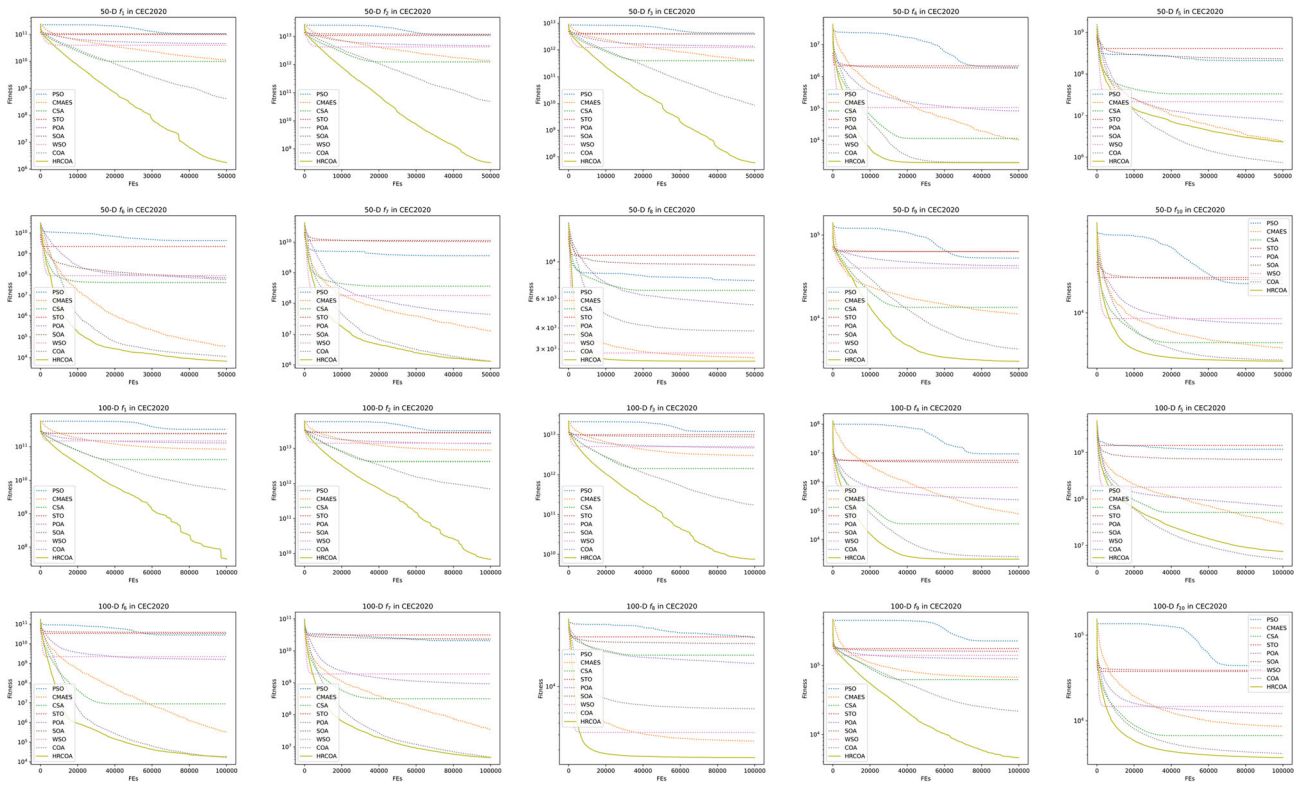


Fig. 8 Convergence curves of nine MAs on 50-D and 100-D CEC2020 benchmark functions

Table 9 Experimental and statistical results on engineering optimization problems; best and worst: the best fitness value and the worst fitness value of optima among 30 trial runs

Prob.		PSO	CMAES	CSA	STO	POA	SOA	EVO	COA	HRCOA
CBD	Mean	2.7914e+00 +	1.3420e+00 +	1.3428e+00 +	1.5026e+00 +	1.3400e+00 ≈	1.3445e+00 +	2.1672e+00 +	1.3401e+00 +	**1.3400e+00**
	Std	5.7027e−01	1.2834e−03	3.5289e−03	8.9963e−02	5.4051e−05	4.1262e−03	4.5218e−01	9.7286e−05	3.4104e−05
	best	2.1447e+00	1.3402e+00	1.3401e+00	1.3566e+00	1.3400e+00	1.3411e+00	1.4512e+00	1.3400e+00	1.3400e+00
	worst	4.2211e+00	1.3455e+00	1.3555e+00	1.6876e+00	1.3401e+00	1.3651e+00	3.2013e+00	1.3403e+00	1.3401e+00
CBHD	Mean	7.8378e+00 +	6.8437e+00 +	7.5636e+00 +	7.6100e+00 +	6.8461e+00 +	7.0706e+00 +	8.9145e+00 +	6.8436e+00 +	**6.8430e+00**
	Std	7.1741e−01	2.4452e−04	8.2400e−01	3.4239e−01	3.6986e−03	6.1314e−02	8.2938e−01	2.6682e−03	6.4072e−06
	best	7.0410e+00	6.8433e+00	6.8434e+00	7.0529e+00	6.8431e+00	6.9648e+00	7.2500e+00	6.8430e+00	6.8430e+00
	worst	1.0352e+01	6.8443e+00	1.0663e+01	8.8033e+00	6.8586e+00	7.1988e+00	1.0978e+01	6.8578e+00	6.8430e+00
IBD	Mean	1.7458e−04 +	**1.7458**e−04 ≈	1.7514e−04 +	2.2935e−04 +	**1.7458e** −04 ≈	1.7458e−04 +	1.9790e−04 +	1.7458e−04 ≈	**1.7458e** −04
	Std	6.4910e−10	1.3553e−19	7.3236e−07	3.8957e−05	1.3553e−19	5.6185e−11	1.9463e−05	1.7370e−18	1.3553e−19
	best	1.7458e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.7459e−04	1.7458e−04	1.7458e−04
	worst	1.7458e−04	1.7458e−04	1.7610e−04	3.2907e−04	1.7458e−04	1.7458e−04	2.6354e−04	1.7458e−04	1.7458e−04
PLD	Mean	1.0074e+02 +	1.0575e+00 +	1.2065e+02 +	4.0862e+02 +	1.1261e+00 ≈	7.1880e+00 +	7.6292e+02 +	2.3247e+01 +	**1.0574e+00**
	Std	1.2262e+02	6.4088e−05	1.5889e+02	3.0290e+02	3.6972e−01	2.8181e+00	7.1836e+02	5.6570e+01	2.3153e−05
	best	1.1589e+00	1.0574e+00	1.0574e+00	2.2567e+01	1.0574e+00	3.5240e+00	7.7311e+01	1.0574e+00	1.0574e+00
	worst	4.0705e+02	1.0576e+00	5.4180e+02	1.3212e+03	3.1170e+00	1.5478e+01	3.0660e+03	1.6747e+02	1.0575e+00
PVP	Mean	9.8111e+03 +	8.2540e+03 +	9.1216e+03 +	8.9073e+04 +	**2.9001e+03** −	2.9625e+03 −	5.1312e+04 +	6.3838e+03 +	4.5276e+03
	Std	2.6979e+03	4.2880e+02	4.6000e+03	6.0644e+04	1.7406e+01	4.1105e+01	3.5783e+04	2.5723e+03	2.4917e+03
	best	3.5495e+03	5.9448e+03	2.8924e+03	4.4793e+03	2.8924e+03	2.9165e+03	1.6398e+04	2.8924e+03	2.8924e+03
	worst	1.7224e+04	8.3336e+03	1.7861e+04	2.2700e+05	2.9665e+03	3.1601e+03	1.7929e+05	8.3336e+03	8.3336e+03
SRD	Mean	3.0695e+03 +	**2.9871e+03** ≈	3.2951e+03 +	5.3565e+05 +	2.9941e+03 +	3.0572e+03 +	3.4347e+03 +	2.9922e+03 +	2.9873e+03
	Std	5.3510e+01	4.9198e−02	4.8030e+02	8.6510e+05	4.6241e+00	1.1002e+01	3.3585e+02	6.9908e+00	2.3274e+00
	best	3.0298e+03	2.9870e+03	3.0332e+03	3.3528e+03	2.9874e+03	3.0341e+03	3.0100e+03	2.9869e+03	2.9869e+03
	worst	3.2223e+03	2.9872e+03	5.8243e+03	3.2910e+06	3.0059e+03	3.0769e+03	4.2829e+03	3.0090e+03	2.9998e+03
+/≈/− Summary		6/0/0	4/2/0	6/0/0	6/0/0	2/3/1	5/0/1	6/0/0	5/1/0	–

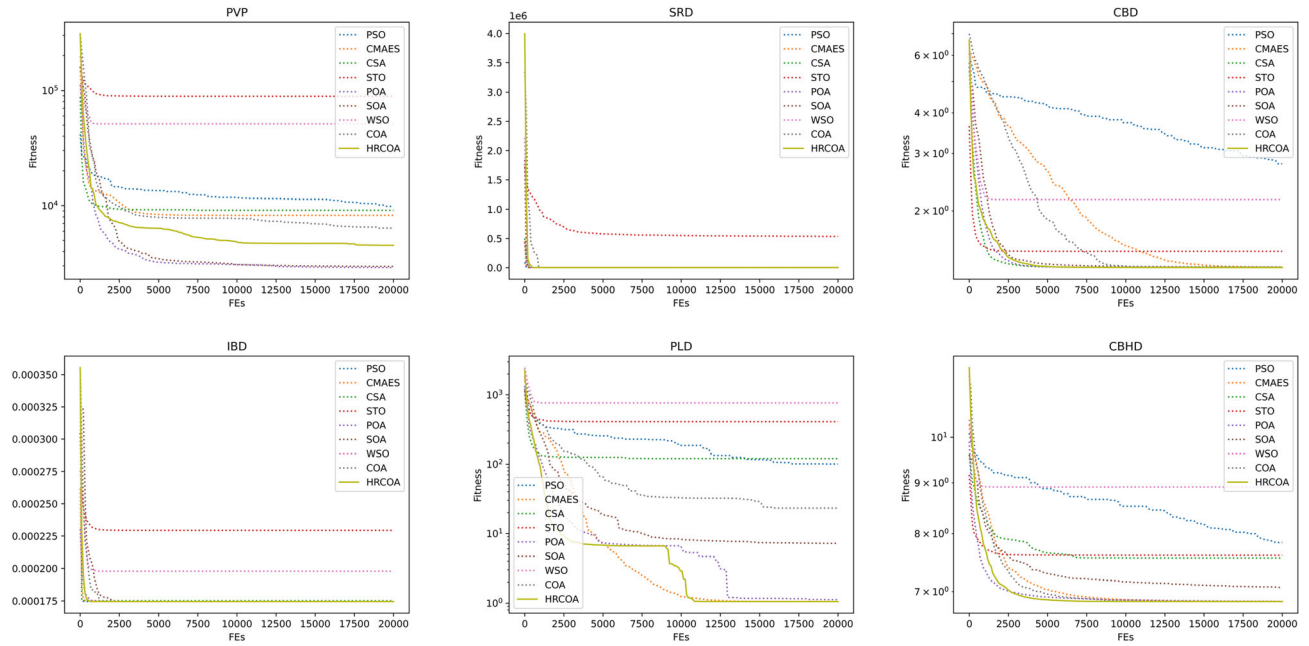


Fig. 9 Convergence curves of nine MAs on engineering optimization problems

Table 10 Experimental results on WSNs coverage optimization problems

# of sensors		PSO	CMAES	CSA	STO	POA	SOA	EVO	COA	HRCOA
32	Mean	71.2%	67.7%	74.3%	69.1%	79.1%	69.9%	66.8%	77.7%	**80.0%**
	Std	0.7%	1.4%	1.6%	1.6%	1.2%	1.0%	1.4%	2.1%	1.3%
42	Mean	80.9%	76.8%	83.9%	78.1%	87.8%	78.6%	76.0%	87.4%	**89.5%**
	Std	0.8%	0.8%	1.8%	1.2%	0.9%	1.0%	1.1%	2.3%	1.0%
54	Mean	88.4%	84.9%	90.6%	86.2%	94.0%	86.2%	84.6%	93.5%	**95.1%**
	Std	0.9%	1.1%	1.7%	1.3%	0.8%	1.4%	1.2%	1.5%	1.0%

including the mean, the standard deviation (std), the best, and the worst in Table 9 and visualize the convergence curves in Fig. 9. Except for the excellent average performance of our proposal, the stability and robustness of HRCOA are also noteworthy. Our proposed HRCOA exhibits the lowest std in 4 out of 6 instances, which collectively provides strong evidence of its efficiency in practical applications.

5.5 Performance analysis on WSNs coverage problems

Many real-world optimization challenges, such as vaccine supply chain design [66] and mobile base station position optimization [67], can be regarded as variants of the wireless sensor networks (WSNs) coverage problem. Therefore, research on WSN coverage problems holds significant practical importance. In this study, we have extended the application of HRCOA to address the boolean sensing model based WSNs coverage problems. The experimental results, as summarized in Table 10, are noteworthy. With a monitoring area ranging from 0 to 50 and a sensing radius of 5, our proposed HRCOA demonstrates satisfying performance with 32, 42, and 54 sensors, respectively. Moreover, visual representations of the optimal solutions obtained by each MA across 30 trial runs are presented in Figs. 10 and 11, where HRCOA also outperforms other competitor algorithms. In summary, HRCOA demonstrates significant potential and promising scalability in real-world applications.

5.6 Performance analysis on ablation and sensitivity analysis experiments

To meticulously investigate the efficacy of the two proposed strategies-namely, the simplified summer resort operator and the hybridization with ROA-we conduct ablation experiments. This comprehensive evaluation encompasses four variations: COA, SCOA (COA + the simplified summer resort operator), RCOA (COA + the hybridization with ROA), and HRCOA (COA + two proposed strategies). Through the experimental results and

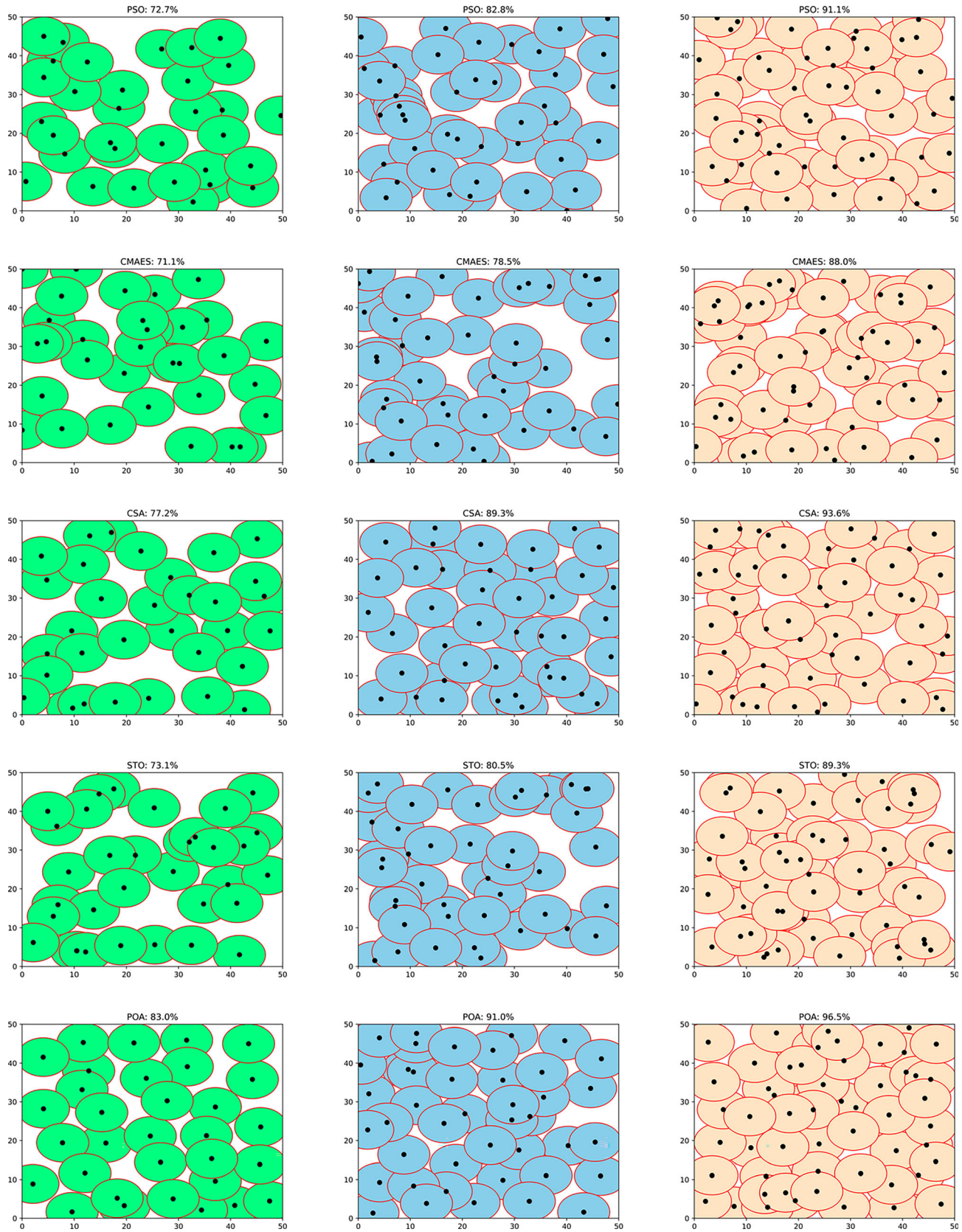


Fig. 10 Optimal distribution of sensors within nine MAs. The numbers of sensors are $\{32, 42, 54\}$

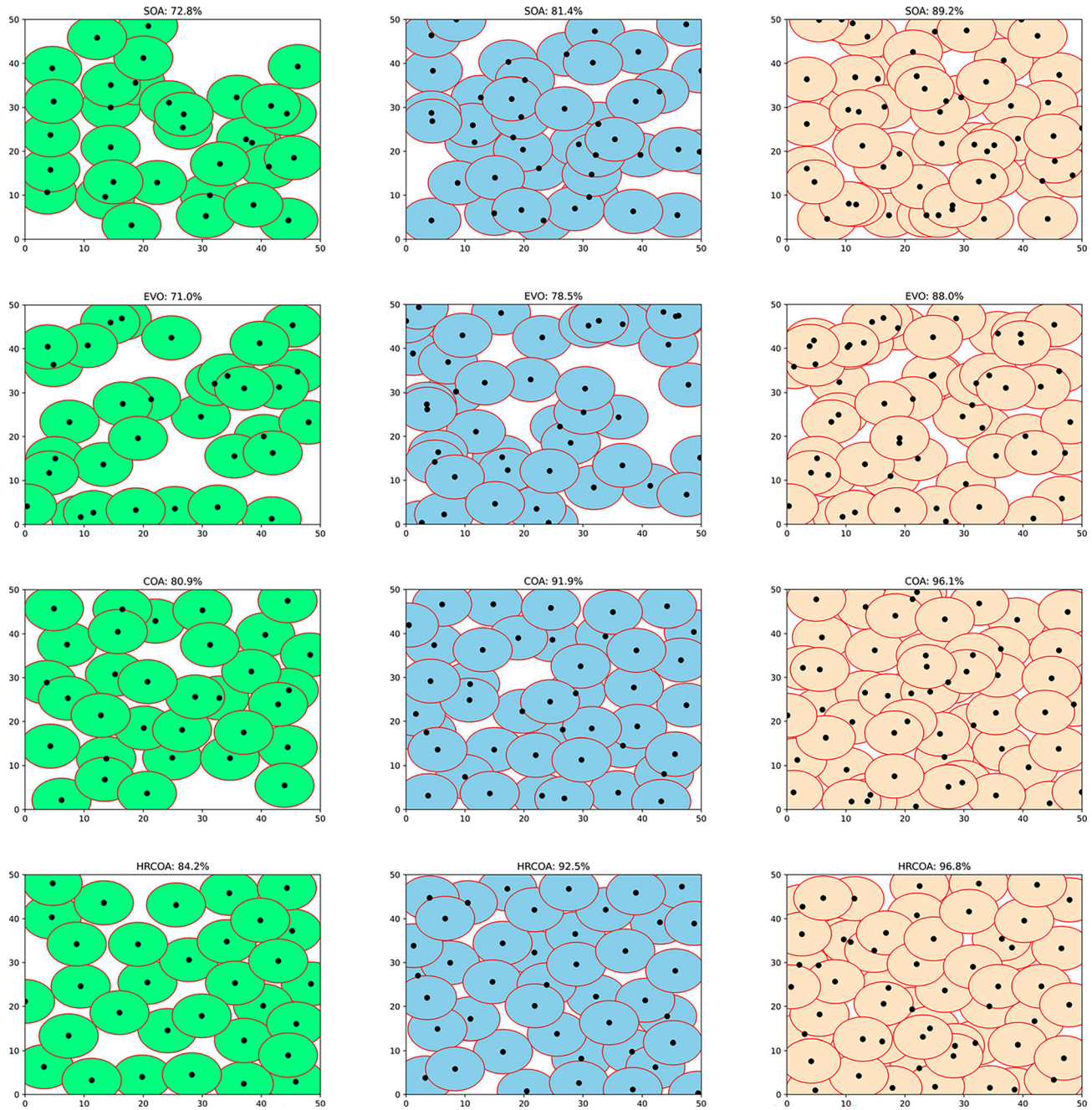


Fig. 11 Optimal distribution of sensors within nine MAs. The numbers of sensors are $\{32, 42, 54\}$ (Continued)

statistical analyses outlined in Table 11, we notice that the significant enhancement observed between COA and HRCOA predominantly stems from the integration of ROA. This finding is consistent with our initial motivation behind introducing the simplified summer resort operator and the hybridization with ROA. The previous strategy aims to mitigate potential exaggerations in the design of COA, and the latter strategy is geared towards augmenting the global search capability within HRCOA. Consequently, our proposed HRCOA emerges as a successful and improved variant of COA, demonstrating promising potential as an alternative approach for numerous optimization tasks.

Moreover, the pivotal hyper-parameter, temperature threshold *temp*, holds significance in transitioning between different search strategies. To investigate the sensitivity of HRCOA on the temperature threshold *temp*, we conduct sensitivity analysis experiments across four variants:

Table 11 Ablation experiments on CEC2020 and CEC2022 benchmark functions. SCOA: COA + simplified summer resort operator; RCOA: hybrid COA with ROA; HRCOA: COA + two proposed strategies

Func.		50-D CEC2020				20-D CEC2022			
		COA	SCOA	RCOA	HRCOA	COA	SCOA	RCOA	HRCOA
f_1	Mean	4.15e+08	3.53e+08 ≈	1.83e+06 −	**1.72e+06** −	3.41e+02	3.48e+02 ≈	3.62e+02 ≈	**3.29e+02** −
	Std	7.52e+08	4.82e+08	1.71e+06	1.54e+06	3.91e+01	8.52e+01	5.97e+01	5.65e+01
f_2	Mean	4.95e+10	4.13e+10 ≈	3.92e+08 −	**3.23e+08** −	4.70e+02	4.72e+02 ≈	4.61e+02 ≈	**4.60e+02** ≈
	Std	7.61e+10	9.63e+10	2.50e+08	5.25e+08	1.81e+01	2.85e+01	1.70e+01	1.69e+01
f_3	Mean	8.62e+09	9.41e+09 ≈	**6.04e+07** −	6.07e+07 −	6.00e+02	6.00e+02 ≈	6.00e+02 −	**6.00e+02** −
	Std	1.09e+10	1.13e+10	6.60e+07	6.24e+07	5.05e−05	2.03e−04	1.22e−06	7.03e−07
f_4	Mean	2.00e+03	2.01e+03 ≈	1.96e+03 −	**1.96e+03** −	8.01e+02	8.01e+02 ≈	8.01e+02 ≈	**8.01e+02** ≈
	Std	3.00e+01	3.30e+01	1.83e+01	1.89e+01	4.38e−01	3.76e−01	6.13e−01	5.54e−01
f_5	Mean	**7.36e+05**	7.71e+05 ≈	2.05e+06 +	2.31e+06 +	9.02e+02	9.02e+02 ≈	9.02e+02 ≈	**9.02e+02** ≈
	Std	2.88e+05	3.39e+05	2.00e+06	2.04e+06	9.05e−01	9.88e−01	1.12e+00	1.09e+00
f_6	Mean	1.13e+04	1.00e+04 ≈	6.85e+03 −	**6.75e+03** −	7.61e+04	7.78e+04 ≈	**7.45e+04** ≈	7.96e+04 ≈
	Std	7.87e+03	6.84e+03	3.25e+03	**2.78e+03**	3.15e+04	3.35e+04	3.09e+04	3.23e+04
f_7	Mean	1.32e+06	**1.02e+06** ≈	1.80e+06 ≈	1.32e+06 ≈	2.18e+03	2.14e+03 ≈	2.17e+03 ≈	**2.10e+03** −
	Std	1.49e+06	8.74e+05	1.58e+06	1.22e+06	1.80e+02	9.05e+01	2.04e+02	5.03e+01
f_8	Mean	3.84e+03	3.57e+03 ≈	2.53e+03 −	**2.53e+03** −	**2.63e+03**	2.69e+03 ≈	4.28e+03 +	3.36e+03 +
	Std	1.97e+03	1.50e+03	3.33e+01	3.64e+01	3.70e+02	4.53e+02	1.70e+03	1.10e+03
f_9	Mean	4.32e+03	4.34e+03 ≈	**2.92e+03** −	3.09e+03 −	2.64e+03	**2.63e+03** ≈	2.65e+03 ≈	2.64e+03 ≈
	Std	1.02e+03	1.25e+03	3.91e+02	4.40e+02	1.12e+01	6.20e+01	2.52e+01	9.96e+00
f_{10}	Mean	3.50e+03	3.51e+03 ≈	3.43e+03 −	**3.41e+03** −	3.59e+03	3.75e+03 ≈	3.17e+03 −	**3.06e+03** −
	Std	1.51e+02	1.19e+02	1.71e+02	1.18e+02	9.21e+02	1.15e+03	3.17e+02	7.65e+02
f_{11}	Mean	–	–	–	–	2.98e+03	2.87e+03 −	**2.74e+03** −	2.91e+03 −
	Std	–	–	–	–	6.27e+02	5.54e+02	4.32e+02	6.13e+02
f_{12}	Mean	–	–	–	–	3.01e+03	3.04e+03 ≈	2.97e+03 −	**2.96e+03** −
	Std	–	–	–	–	3.83e+01	7.07e+01	1.65e+01	1.43e+01
+/≈/−		–	0/10/0	1/1/8	1/1/8	–	0/11/1	1/7/4	1/5/6

HRCOA1 (HRCOA with $temp = 25$), HRCOA2 (HRCOA with $temp = 27.5$), HRCOA (HRCOA with $temp = 30$), and HRCOA3 (HRCOA with $temp = 32.5$). Our experimental results and statistical analyses, as detailed in Table 12, reveal that HRCOA displays minimal sensitivity to variations in the temperature threshold *temp*. Particularly noteworthy is the comparison between HRCOA2 and HRCOA, where no statistically significant differences are observed across all benchmark functions in CEC2020 and CEC2022. In essence, HRCOA exhibits superior robustness even when the temperature threshold *temp* changes.

5.7 Open topics for future research

Based on the experimental results and analysis presented above, our proposed HRCOA has demonstrated satisfactory performance. However, there still several unresolved problems remain, and we list some open topics for future exploration.

5.7.1 Advancing HRCOA for various optimization tasks

Our proposed HRCOA has performed competitiveness through the experimental results, thus, it is promising to extend HRCOA to other optimization tasks such as multi-objective optimization problems [68], multimodal optimization problems [69], discrete optimization problems [70], high-dimensional optimization problems [71], and real-world applications [72].

5.7.2 Hybridizing HRCOA with other superior MAs

In recent decades, the evolutionary computation (EC) community has witnessed remarkable advancements. Apart from notable algorithms like the WOA, SFO, ROA, and COA, numerous fish-inspired MAs have been proposed, such as salp swarm algorithm (SSA) [9], tuna swarm optimization (TSO) [73], and many others. Therefore, exploring the integration of advantages from other fish-

Table 12 Sensitivity analysis experiments on CEC2020 and CEC2022 benchmark functions. HRCOA1: *temp* = 25, HRCOA2: *temp* = 27.5, HRCOA: *temp* = 30, and HRCOA3: *temp* = 32.5

Func.		50-D CEC2020				20-D CEC2022			
		HRCOA1	HRCOA2	HRCOA	HRCOA3	HRCOA1	HRCOA2	HRCOA	HRCOA3
f_1	Mean	1.69e+06 ≈	**1.58e+06** ≈	1.72e+06	3.12e+07 +	1.03e+03 +	**3.18e+02** ≈	3.29e+02	1.35e+03 +
	Std	1.58e+06	1.01e+06	1.54e+06	1.71e+07	3.80e+02	1.82e+01	5.65e+01	5.96e+02
f_2	Mean	2.58e+08 ≈	**1.83e+08** ≈	3.23e+08	5.29e+09 +	**4.60e+02** ≈	4.61e+02 ≈	4.60e+02	4.78e+02 +
	Std	3.87e+08	1.30e+08	5.25e+08	2.31e+09	1.42e+01	1.25e+01	1.69e+01	2.07e+01
f_3	Mean	6.98e+07 ≈	7.15e+07 ≈	**6.07e+07**	1.07e+09 +	6.00e+02 +	6.00e+02 ≈	**6.00e+02**	6.00e+02 +
	Std	1.10e+08	4.94e+07	6.24e+07	4.02e+08	1.90e−05	1.59e−07	7.03e−07	6.92e−05
f_4	Mean	1.99e+03 ≈	1.97e+03 ≈	1.96e+03	**1.95e+03** ≈	8.02e+02 ≈	8.01e+02 ≈	**8.01e+02**	8.02e+02 ≈
	Std	3.18e+01	2.67e+01	1.89e+01	7.07e+00	8.49e−01	7.09e−01	5.54e−01	8.50e−01
f_5	Mean	**1.46e+06** ≈	1.93e+06 ≈	2.31e+06	4.39e+06 ≈	9.02e+02 ≈	9.02e+02 ≈	**9.02e+02**	9.02e+02 ≈
	Std	5.92e+05	1.69e+06	2.04e+06	4.59e+06	9.11e−01	1.10e+00	1.09e+00	6.57e−01
f_6	Mean	7.07e+03 ≈	**6.32e+03** ≈	6.75e+03	1.07e+04 ≈	8.02e+04 ≈	**7.60e+04** ≈	7.96e+04	1.03e+05 ≈
	Std	3.32e+03	2.69e+03	2.78e+03	8.83e+03	2.00e+04	1.94e+04	3.23e+04	3.11e+04
f_7	Mean	2.89e+06 ≈	2.42e+06 ≈	**1.32e+06**	1.95e+06 ≈	2.10e+03 ≈	2.13e+03 ≈	**2.10e+03**	2.16e+03 ≈
	Std	4.74e+06	1.89e+06	1.22e+06	1.84e+06	4.79e+01	8.18e+01	5.03e+01	1.09e+02
f_8	Mean	2.61e+03 ≈	2.56e+03 ≈	2.53e+03	**2.48e+03** −	4.85e+03 +	**3.15e+03** ≈	3.36e+03	4.55e+03 +
	Std	6.05e+01	3.20e+01	3.64e+01	2.72e+01	7.64e+02	8.55e+02	1.10e+03	1.74e+03
f_9	Mean	**2.74e+03** −	2.93e+03 ≈	3.09e+03	3.17e+03 ≈	2.65e+03 ≈	2.65e+03 ≈	**2.64e+03**	2.65e+03 ≈
	Std	2.80e+02	5.42e+02	4.40e+02	1.95e+02	1.67e+01	1.31e+01	9.96e+00	1.02e+01
f_{10}	Mean	**3.38e+03** ≈	3.40e+03 ≈	3.41e+03	3.49e+03 ≈	3.28e+03 ≈	2.95e+03 ≈	3.06e+03	**2.82e+03** ≈
	Std	8.72e+01	1.04e+02	1.18e+02	1.37e+02	1.04e+03	3.81e+02	7.65e+02	**7.97e+01**
f_{11}	Mean	–	–	–	–	2.91e+03 ≈	2.76e+03 ≈	2.91e+03	**2.61e+03** ≈
	Std	–	–	–	–	6.09e+02	4.66e+02	6.13e+02	7.43e+00
f_{12}	Mean	–	–	–	–	2.96e+03 ≈	2.98e+03 ≈	2.96e+03	**2.95e+03** ≈
	Std	–	–	–	–	1.76e+01	2.09e+01	1.43e+01	8.88e+00
+/≈/−		0/9/1	0/10/0	–	3/6/1	3/9/0	0/12/0	–	4/8/0

inspired MAs to further enhance the hybrid version of COA presents a promising avenue for research.

Furthermore, the hyper-heuristic (HH) framework presents a practical and versatile approach to realizing hybridization [74, 75]. Within the HH framework, an optimization algorithm is viewed as a sequence of diverse search strategies. For example, the genetic algorithm (GA) typically incorporates iterations of crossover and mutation operators, while particle swarm optimization (PSO) involves velocity and location updates. Crucially, within the HH paradigm, the sequence of these heuristics can be optimized. Consequently, integrating the HH framework into algorithms such as our proposed HRCOA holds promise for enhancing efficiency and effectiveness. This approach allows for the optimization of the sequence encompassing strategies like the WOA strategy, Host feeding, competition search, and simplified summer resort search.

6 Conclusion

This paper proposed a novel hybrid remora crayfish optimization algorithm (HRCOA), where the exploitation operators from the remora optimization algorithm (ROA) are integrated into the crayfish optimization algorithm (COA) to further strengthen its exploitative capabilities. Additionally, we incorporated a simplified summer resort strategy to avoid overemphasis in the search operator design.

To evaluate the performance of HRCOA, comprehensive numerical experiments are implemented covering various domains. These included 10-D and 20-D CEC2022 benchmark functions, 50-D and 100-D CEC2020 benchmark functions, six engineering problems, and WSNs coverage optimization problems. We compared HRCOA with eight famous and state-of-the-art MAs. The experimental, supported by statistical analysis, confirms the effectiveness and efficiency of our proposed HRCOA.

In the future, we will focus on developing HRCOA to tackle a wide range of various optimization tasks such as multi-objective optimization problems, feature selection tasks, and optimization problems in noisy environments.

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

Author Contributions RZ: conceptualization, methodology, investigation, writing—original draft, writing—review & editing, and funding acquisition. QF: investigation, methodology, formal analysis, and writing—review & editing. CZ: conceptualization and writing—review & editing. JY: writing—review & editing, and project administration.

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

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. Zhong, R., Zhang, E., Munetomo, M.: Cooperative coevolutionary differential evolution with linkage measurement minimization for large-scale optimization problems in noisy environments. Complex Intell. Syst. **9**, 4439–4456 (2023). https://doi.org/10.1007/s40747-022-00957-6
2. Lu, Z., Whalen, I., Boddeti, V., Dhebar, Y., Deb, K., Goodman, E., Banzhaf, W.: Nsga-net: Neural architecture search using multi-objective genetic algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 419–427. Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3321707.3321729
3. Zhong, R., Peng, F., Zhang, E., Yu, J., Munetomo, M.: Vegetation evolution with dynamic maturity strategy and diverse mutation strategy for solving optimization problems. Biomimetics (2023). https://doi.org/10.3390/biomimetics8060454
4. De Jong, K.: Learning with genetic algorithms: An overview. Mach. Learn. **3**, 121–138 (1988). https://doi.org/10.1007/BF00113894
5. Storn, R.: On the usage of differential evolution for function optimization. In: Proceedings of North American Fuzzy Information Processing, pp. 519–523 (1996). https://doi.org/10.1109/NAFIPS.1996.534789
6. Koza, J.R.: Genetic programming as a means for programming computers by natural selection. Stat. Comput. **4**, 87–112 (1994). https://doi.org/10.1007/BF00175355
7. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Trans. Evol. Comput. **3**(2), 82–102 (1999). https://doi.org/10.1109/4235.771163
8. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, pp. 1942–19484 (1995). https://doi.org/10.1109/ICNN.1995.488968
9. Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H., Mirjalili, S.M.: Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. Adv. Eng. Softw. **114**, 163–191 (2017). https://doi.org/10.1016/j.advengsoft.2017.07.002
10. Faramarzi, A., Heidarinejad, M., Mirjalili, S., Gandomi, A.H.: Marine predators algorithm: A nature-inspired metaheuristic. Expert Syst. Appl. **152**, 113377 (2020). https://doi.org/10.1016/j.eswa.2020.113377
11. Hashim, F.A., Hussien, A.G.: Snake optimizer: A novel metaheuristic optimization algorithm. Knowl.-Based Syst. (2022). https://doi.org/10.1016/j.knosys.2022.108320
12. Yu, J.: Vegetation evolution: An optimization algorithm inspired by the life cycle of plants. Int. J. Comput. Intell. Appl. (2022). https://doi.org/10.1142/S1469026822500109
13. Hashim, F.A., Houssein, E.H., Hussain, K., Mabrouk, M.S., Al-Atabany, W.: Honey badger algorithm: New metaheuristic algorithm for solving optimization problems. Math. Comput. Simul. **192**, 84–110 (2022). https://doi.org/10.1016/j.matcom.2021.08.013
14. Seyyedabbasi, A., Kiani, F.: Sand cat swarm optimization: a nature-inspired algorithm to solve global optimization problems. Eng. Comput. (2022). https://doi.org/10.1007/s00366-022-01604-x
15. Dehghani, M., Montazeri, Z., Trojovská, E., Trojovský, P.: Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. Knowl.-Based Syst. **259**, 110011 (2023). https://doi.org/10.1016/j.knosys.2022.110011
16. Zhang, Y., Jin, Z.: Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems. Expert Syst. Appl. **148**, 113246 (2020). https://doi.org/10.1016/j.eswa.2020.113246
17. Shabani, A., Asgarian, B., Salido, M., Asil Gharebaghi, S.: Search and rescue optimization algorithm: A new optimization method for solving constrained engineering optimization problems. Expert Syst. Appl. **161**, 113698 (2020). https://doi.org/10.1016/j.eswa.2020.113698
18. Abdulhameed, S., Rashid, T.: Child drawing development optimization algorithm based on child's cognitive development. Arab. J. Sci. Eng. (2021). https://doi.org/10.1007/s13369-021-05928-6
19. Dehghani, M., Trojovska, E., Zuščák, T.: A new human-inspired metaheuristic algorithm for solving optimization problems based on mimicking sewing training. Sci. Rep. (2022). https://doi.org/10.1038/s41598-022-22458-9
20. Xu, Y., Zhong, R., Zhang, C., Yu, J.: Multiplayer battle game-inspired optimizer for complex optimization problems (2023)
21. Faridmehr, I., Nehdi, M.L., Davoudkhani, I.F., Poolad, A.: Mountaineering team-based optimization: A novel human-based metaheuristic algorithm. Mathematics (2023). https://doi.org/10.3390/math11051273
22. Matoušová, I., Trojovsky, P., Dehghani, M., Trojovska, E., Kostra, J.: Mother optimization algorithm: a new human-based metaheuristic approach for solving engineering optimization. Sci. Rep. (2023). https://doi.org/10.1038/s41598-023-37537-8
23. Talatahari, S., Azizi, M., Gandomi, A.H.: Material generation algorithm: A novel metaheuristic algorithm for optimization of engineering problems. Processes (2021). https://doi.org/10.3390/pr9050859
24. Ahmadianfar, I., Heidari, A.A., Gandomi, A.H., Chu, X., Chen, H.: Run beyond the metaphor: An efficient optimization algorithm based on runge kutta method. Expert Syst. Appl. **181**, 115079 (2021). https://doi.org/10.1016/j.eswa.2021.115079
25. Shaqfa, M., Beyer, K.: Pareto-like sequential sampling heuristic for global optimisation. Soft. Comput. **25**, 9077–9096 (2021). https://doi.org/10.1007/s00500-021-05853-8
26. Ahmadianfar, I., Heidari, A.A., Noshadian, S., Chen, H., Gandomi, A.H.: Info: An efficient optimization algorithm based on weighted mean of vectors. Expert Syst. Appl. **195**, 116516 (2022). https://doi.org/10.1016/j.eswa.2022.116516

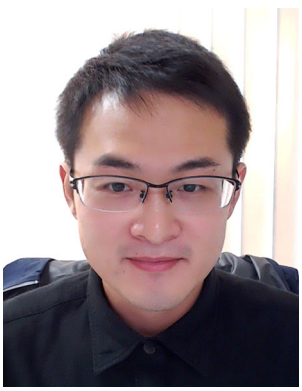
27. Su, H., Zhao, D., Heidari, A.A., Liu, L., Zhang, X., Mafarja, M., Chen, H.: Rime: A physics-based optimization. Neurocomputing **532**, 183–214 (2023). https://doi.org/10.1016/j.neucom.2023.02.010
28. Deng, L., Liu, S.: Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design. Expert Syst. Appl. **225**, 120069 (2023). https://doi.org/10.1016/j.eswa.2023.120069
29. Cheng, M.-Y., Sholeh, M.N.: Optical microscope algorithm: A new metaheuristic inspired by microscope magnification for solving engineering optimization problems. Knowl.-Based Syst. (2023). https://doi.org/10.1016/j.knosys.2023.110939
30. Shehadeh, H.: Chernobyl disaster optimizer (cdo): a novel meta-heuristic method for global optimization. Neural Comput. Appl. (2023). https://doi.org/10.1007/s00521-023-08261-1
31. Qu, C., Gai, W., Zhang, J., Zhong, M.: A novel hybrid grey wolf optimizer algorithm for unmanned aerial vehicle (uav) path planning. Knowl.-Based Syst. **194**, 105530 (2020). https://doi.org/10.1016/j.knosys.2020.105530
32. Zhang, X., Lin, Q., Mao, W., Liu, S., Dou, Z., Liu, G.: Hybrid particle swarm and grey wolf optimizer and its application to clustering optimization. Appl. Soft Comput. **101**, 107061 (2021). https://doi.org/10.1016/j.asoc.2020.107061
33. Che, Y., He, D.-X.: A hybrid whale optimization with seagull algorithm for global optimization problems. Math. Probl. Eng. (2021). https://doi.org/10.1155/2021/6639671
34. Han, B., Li, B., Qin, C.: A novel hybrid particle swarm optimization with marine predators. Swarm Evol. Comput. **83**, 101375 (2023). https://doi.org/10.1016/j.swevo.2023.101375
35. Jia, H., Rao, H., Wen, C., Mirjalili, S.: Crayfish optimization algorithm. Artif. Intell. Rev. **1**, 1 (2023). https://doi.org/10.1007/s10462-023-10567-4
36. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evol. Comput. **9**(2), 159–195 (2001). https://doi.org/10.1162/106365601750190398
37. Qais, M.H., Hasanien, H.M., Turky, R.A., Alghuwainem, S., Tostado-Véliz, M., Jurado, F.: Circle search algorithm: A geometry-based metaheuristic optimization algorithm. Mathematics (2022). https://doi.org/10.3390/math10101626
38. Trojovský, P., Dehghani, M., Hanuš, P.: Siberian tiger optimization: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems. IEEE Access **10**, 132396–132431 (2022). https://doi.org/10.1109/ACCESS.2022.3229964
39. Trojovský, P., Dehghani, M.: Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications. Sensors (2022). https://doi.org/10.3390/s22030855
40. Dehghani, M., Trojovský, P.: Serval optimization algorithm: A new bio-inspired approach for solving optimization problems. Biomimetics (2022). https://doi.org/10.3390/biomimetics7040204
41. Azizi, M., Aickelin, U., Khorshidi, H., Baghalzadeh Shishehgarkhaneh, M.: Energy valley optimizer: a novel metaheuristic algorithm for global and engineering optimization. Sci. Rep. **13**, 226 (2023). https://doi.org/10.1038/s41598-022-27344-y
42. Fertl, D., Landry, A.M., Jr.: Sharksucker (echeneis naucrates) on a bottlenose dolphin (tursiops truncatus) and a review of other cetacean-remora associations. Mar. Mamm. Sci. **15**(3), 859–863 (1999). https://doi.org/10.1111/j.1748-7692.1999.tb00849.x
43. Williams, E.H., Jr., Mignucci-Giannoni, A.A., Bunkley-Williams, L., Bonde, R.K., Self-Sullivan, C., Preen, A., Cockcroft, V.G.: Echeneid-sirenian associations, with information on sharksucker diet. J. Fish Biol. **63**(5), 1176–1183 (2003). https://doi.org/10.1046/j.1095-8649.2003.00236.x
44. Shadravan, S., Naji, H.R., Bardsiri, V.K.: The sailfish optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. Eng. Appl. Artif. Intell. **80**, 20–34 (2019). https://doi.org/10.1016/j.engappai.2019.01.001
45. Mirjalili, S., Lewis, A.: The whale optimization algorithm. Adv. Eng. Softw. **95**, 51–67 (2016). https://doi.org/10.1016/j.advengsoft.2016.01.008
46. Jia, H., Peng, X., Lang, C.: Remora optimization algorithm. Expert Syst. Appl. **185**, 115665 (2021). https://doi.org/10.1016/j.eswa.2021.115665
47. Deepa, R., Venkataraman, R.: Enhancing whale optimization algorithm with levy flight for coverage optimization in wireless sensor networks. Comput. Electr. Eng. **94**, 107359 (2021). https://doi.org/10.1016/j.compeleceng.2021.107359
48. Singh, A., Sharma, S., Singh, J.: Nature-inspired algorithms for wireless sensor networks: A comprehensive survey. Comput. Sci. Rev. **39**, 100342 (2021). https://doi.org/10.1016/j.cosrev.2020.100342
49. Wei, Y., Wei, X., Huang, H., Bi, J., Zhou, Y., Du, Y.: Ssma: simplified slime mould algorithm for optimization wireless sensor network coverage problem. Syst. Sci. Control Eng. **10**(1), 662–685 (2022). https://doi.org/10.1080/21642583.2022.2084650
50. Golalipour, K., Faraji Davoudkhani, I., Nasri, S., Naderipour, A., Mirjalili, S., Abdelaziz, A.Y., El-Shahat, A.: The corona virus search optimizer for solving global and engineering optimization problems. Alex. Eng. J. **78**, 614–642 (2023). https://doi.org/10.1016/j.aej.2023.07.066
51. Kumar, A., Price, K.V., Mohamed, A.W., Hadi, A.A., Suganthan, P.N.: Problem definitions and evaluation criteria for the cec 2022 special session and competition on single objective bound constrained numerical optimization. In: Technical Report (2022)
52. Nguyen, T.: A framework of optimization functions using Numpy (OpFuNu) for optimization problems. Zenodo (2020). https://doi.org/10.5281/zenodo.3620960
53. Yue, C.T., Price, P.N.S.K.V.: Problem definitions and evaluation criteria for the cec 2020 special session and competition on single objective bound constrained numerical optimization. In: Technical Report (2020)
54. Zhong, R., Peng, F., Yu, J., Munetomo, M.: Q-learning based vegetation evolution for numerical optimization and wireless sensor network coverage optimization. Alex. Eng. J. **87**, 148–163 (2024). https://doi.org/10.1016/j.aej.2023.12.028
55. Thieu, N.V.: ENOPPY: a python library for engineering optimization problems. Zenodo (2023). https://doi.org/10.5281/zenodo.7953206
56. Liang, J., Tian, M., Liu, Y., Zhou, J.: Coverage optimization of soil moisture wireless sensor networks based on adaptive cauchy variant butterfly optimization algorithm. Sci. Rep. (2022). https://doi.org/10.1038/s41598-022-15689-3
57. Li, S., Chen, H., Wang, M., Heidari, A.A., Mirjalili, S.: Slime mould algorithm: A new method for stochastic optimization. Futur. Gener. Comput. Syst. **111**, 300–323 (2020). https://doi.org/10.1016/j.future.2020.03.055
58. Bayzidi, H., Talatahari, S., Saraee, M., Lamarche, C.-P.: Social network search for solving engineering optimization problems. Comput. Intell. Neurosci. **2021**, 1–32 (2021). https://doi.org/10.1155/2021/8548639
59. Salgotra, R., Singh, U.: The naked mole-rat algorithm. Neural Comput. Appl. (2019). https://doi.org/10.1007/s00521-019-04464-7
60. Guan, Z., Ren, C., Niu, J., Wang, P., Shang, Y.: Great wall construction algorithm: A novel meta-heuristic algorithm for engineer problems. Expert Syst. Appl. **233**, 120905 (2023). https://doi.org/10.1016/j.eswa.2023.120905
61. Van Thieu, N., Mirjalili, S.: Mealpy: An open-source library for latest meta-heuristic algorithms in python. J. Syst. Architect. **139**, 102871 (2023). https://doi.org/10.1016/j.sysarc.2023.102871

62. Coello Coello, C.A.: 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 (2002). https://doi.org/10.1016/S0045-7825(01)00323-1
63. Zhong, R., Yu, J., Zhang, C., Munetomo, M.: Srime: a strengthened rime with latin hypercube sampling and embedded distance-based selection for engineering optimization problems. Neural Comput. Appl. (2024). https://doi.org/10.1007/s00521-024-09424-4
64. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997). https://doi.org/10.1109/4235.585893
65. Köppen, M.: The curse of dimensionality. In: 5th Online World Conference on Soft Computing in Industrial Applications (WSC5), vol. 1, pp. 4–8 (2000)
66. Valizadeh, J., Boloukifar, S., Soltani, S., Jabalbarezi Hookerd, E., Fouladi, F., Andreevna Rushchtc, A., Du, B., Shen, J.: Designing an optimization model for the vaccine supply chain during the covid-19 pandemic. Expert Syst. Appl. **214**, 119009 (2023). https://doi.org/10.1016/j.eswa.2022.119009
67. Lyu, J., Zeng, Y., Zhang, R., Lim, T.J.: Placement optimization of uav-mounted mobile base stations. IEEE Commun. Lett. **21**(3), 604–607 (2017). https://doi.org/10.1109/LCOMM.2016.2633248
68. Abdullah, J., Rashid, T., Maaroof, B., Mirjalili, S.: Multi-objective fitness-dependent optimizer algorithm. Neural Comput. Appl. **35**, 1–19 (2023). https://doi.org/10.1007/s00521-023-08332-3
69. Gupta, S., Su, R.: Diversity-enhanced modified sine cosine algorithm and its application in solving engineering design problems. J. Comput. Sci. **72**, 102105 (2023). https://doi.org/10.1016/j.jocs.2023.102105
70. Hichem, H., Elkamel, M., Rafik, M., Mesaaoud, M.T., Ouahiba, C.: A new binary grasshopper optimization algorithm for feature selection problem. J. King Saud Univ. Comput. Inf. Sci. **34**(2), 316–328 (2022). https://doi.org/10.1016/j.jksuci.2019.11.007
71. Zhong, R., Zhang, E., Munetomo, M.: Cooperative coevolutionary surrogate ensemble-assisted differential evolution with efficient dual differential grouping for large-scale expensive optimization problems. Complex Intell. Syst. (2023). https://doi.org/10.1007/s40747-023-01262-6
72. Obayya, M., Alhebri, A., Maashi, M., Salama, S., A., Mustafa Hilal, A., Alsaid, M.I., Osman, A.E., Alneil, A.A.: Henry gas solubility optimization algorithm based feature extraction in dermoscopic images analysis of skin cancer. Cancers (2023). https://doi.org/10.3390/cancers15072146
73. Xie, L., Han, T., Zhou, H., Zhang, Z.-R., Han, B., Tang, A., Khalil, A.M.: Tuna swarm optimization: A novel swarm-based metaheuristic algorithm for global optimization. Intell. Neurosci. (2021). https://doi.org/10.1155/2021/9210050
74. Zhong, R., Zhang, E., Munetomo, M.: Evolutionary multi-mode slime mold optimization: a hyper-heuristic algorithm inspired by slime mold foraging behaviors. J. Supercomput. (2024). https://doi.org/10.1007/s11227-024-05909-0
75. Dokeroglu, T., Kucukyilmaz, T., Talbi, E.-G.: Hyper-heuristics: A survey and taxonomy. Comput. Ind. Eng. **187**, 109815 (2024). https://doi.org/10.1016/j.cie.2023.109815

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 Ph.D. at Hokkaido University, Japan. His research interests include evolutionary computation, large-scale global optimization, and mete/hyper-heuristics.

Qinqin Fan received the Ph.D. degree in control science and engineering from the East China University of Science and Technology, Shanghai, China, in 2015. He is currently an Associate Professor with Shanghai Maritime University. His current research interests include evolutionary computation, hyper heuristic algorithm, multi-objective optimization, and their real-world applications.

Chao Zhang received his Ph.D. at Iwate University (Japan) in March 2017. He was a full-time lecturer (a.k.a. assistant professor) at the Faculty of Engineering, University of Fukui (Japan) from 2017 to 2024. Starting from April 2024, he is currently a Specially Appointed Professor at the University of Toyama. His research interests include computer vision and machine learning. He is a member of the IEEE, IEICE, ITE, and IEEJ. More information can be found at https://www.labzhang.com/.



Jun Yu received a Bachelor degree from Northeastern University, China in 2014, and a Master degree and a doctorate from Kyushu University, Japan in 2017 and 2019, respectively. He is currently an Assistant Professor at Niigata University, Japan. His research interests include evolutionary computation, artificial neural networks, and machine learning.

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com