Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.038670

ARTICLE

# Ensemble of Population-Based Metaheuristic Algorithms

**Hao Li, Jun Tang*, Qingtao Pan, Jianjun Zhan and Songyang Lao**

College of Systems Engineering, National University of Defense Technology, Changsha, 410000, China

*Corresponding Author: Jun Tang. Email: tangjun06@nudt.edu.cn

Received: 23 December 2022 Accepted: 10 April 2023 Published: 08 October 2023

**ABSTRACT**

No optimization algorithm can obtain satisfactory results in all optimization tasks. Thus, it is an effective way to deal with the problem by an ensemble of multiple algorithms. This paper proposes an ensemble of population-based metaheuristics (EPM) to solve single-objective optimization problems. The design of the EPM framework includes three stages: the initial stage, the update stage, and the final stage. The framework applies the transformation of the real and virtual population to balance the problem of exploration and exploitation at the population level and uses an elite strategy to communicate among virtual populations. The experiment tested two benchmark function sets with five metaheuristic algorithms and four ensemble algorithms. The ensemble algorithms are generally superior to the original algorithms by Friedman's average ranking and Wilcoxon signed ranking test results, demonstrating the ensemble framework's effect. By solving the iterative curves of different test functions, we can see that the ensemble algorithms have faster iterative optimization speed and better optimization results. The ensemble algorithms cannot fall into local optimum by virtual populations distribution map of several stages. The ensemble framework performs well from the effects of solving two practical engineering problems. Some results of ensemble algorithms are superior to those of metaheuristic algorithms not included in the ensemble framework, further demonstrating the ensemble method's potential and superiority.

**KEYWORDS**

Ensemble; population-based metaheuristics; real and virtual population; elite strategy; swarm intelligence

## 1 Introduction

Multiple optimization techniques have been proposed to address the challenges of various optimization problems, which include accurate and approximate methods [1]. Accurate methods obtain an exact solution to guarantee optimality. Approximate methods can generate a high-quality solution in a reasonable time to meet the actual requirements, but they cannot ensure a globally optimal solution. As one type of approximate method, metaheuristics are general for solving optimization problems. Metaheuristics include single-solution-based and population-based by the number of solutions processed in the optimization phase [2].

Although engineers apply meta-heuristic algorithms to various optimization problems, the solution to a specific optimization problem is only sometimes satisfactory. Therefore, researchers widely use integration strategies to design general algorithms for particular problem sets. Many practical

and available algorithms have been developed for single-objective optimization [3], constrained optimization [4], multi-objective optimization [5], niche [6], and others. Metaheuristic algorithms have significantly been developed and applied to solving optimization problems in many fields [7], such as scheduling [8], data mining [9], and unmanned system [10].

When engineers integrate various optimization algorithms, they have to design appropriate integration schemes to take advantage of their respective strengths. The paper proposes an ensemble of population-based metaheuristics (EPM) to solve the single-objective problem. We consider each population-based metaheuristic algorithm as a whole in the ensemble framework without understanding its characteristics and internal mechanisms. The ensemble framework puts forward the transformation between the real and virtual populations to coordinate exploration and exploitation at the population level. It also adopts an elite strategy to realize information exchanges among virtual populations to enhance the diversity of populations. The transformation between the real and virtual populations mainly solves the problem of falling into local optimum. We also apply an elite strategy to speed up the convergence. The practical issues that the ensemble framework can solve are related to the application field of its integrated meta-heuristic algorithms. If a meta-heuristic algorithm can be applied to solve a particular practical problem, then the algorithm can still solve the optimization problem in the ensemble framework.

The rest of this paper is structured as follows. Section 2 summarizes the ensemble methods of the population-based metaheuristic algorithms. Section 3 describes the design of the EPM framework and how to use it to integrate various metaheuristic algorithms. Section 4 conducts testing and analysis, where it tests the original and ensemble algorithms using two test function sets and two engineering application problems. Finally, in Section 5, the full text is summarized, pointing to future research.

## 2 Related Work

### *2.1 Population-Based Metaheuristics*

Population-based metaheuristics have the same paradigm, as shown in Fig. 1. A population represents a set of solutions, and each individual represents a feasible solution. Generation strategies are designed to generate a new population, and the original individual is replaced by the newly generated individual according to selection strategies so that the population is continuously optimized until the set stop condition is met. The various population-based metaheuristic algorithms differ in how they perform the generation and selection processes.

The design of population-based metaheuristics is inspired by nature and human society, including evolutionary processes, physics, biology, chemistry, mathematics, human bodies, society, and so on. Evolutionary algorithms simulate the process of biological evolution, using crossover, mutation, reproduction, and other operators to produce better solutions. Typical algorithms include differential evolution [11], genetic algorithm [12], and genetic programming [13]. Biology-based algorithms mimic the behaviors in biological groups, such as foraging, mating, and movement, including particle swarm optimization [14], artificial bee swarm optimization [15], cuckoo search [16], and sand cat swarm optimization [17]. Human-based algorithms mainly simulate human body structure, way of thinking, and social behavior, including tabu search [18], cooperation search [19], brainstorming algorithms [20], heap-based optimization [21], and war strategy optimization [22]. Physics-based algorithms draw lessons from the classical laws of physics. For example, the idea of gravity search comes from the law of gravity [23]. The big bang algorithm simulates the big bang effects and the big bang processes [24]. The electromagnetic mechanism-like algorithm simulates charged particles' attraction and repulsion mechanisms in the electromagnetic field [25]. The algorithm based on mathematics draws lessons from

mathematical formulas and mathematical theorems, such as the sine cosine algorithm based on the mathematical properties of the sine and cosine functions [26], the chaotic golden section algorithm based on chaotic motion and golden section [27] and the arithmetic optimization algorithm based on the main arithmetic operators in mathematics [28].

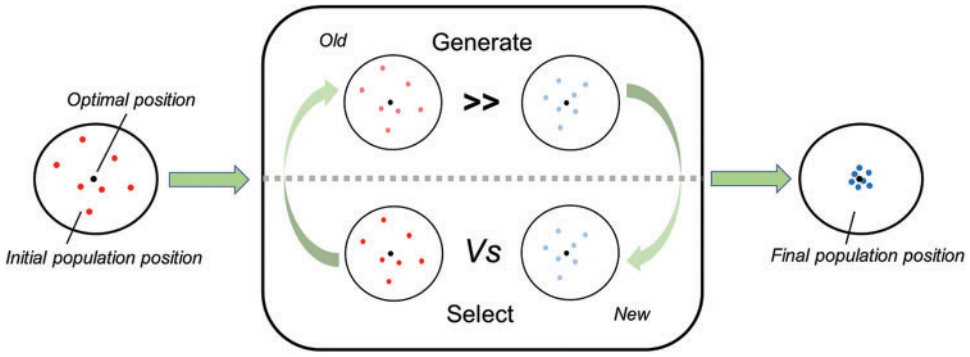


**Figure 1:** The typical process diagram of population-based metaheuristics

Engineers apply metaheuristic algorithms in many fields, such as engineering, material, robot, computer, home life, and medical treatment. Chopra et al. proposed a golden jackal optimization algorithm for the economic load dispatch problem in electrical engineering [29]. Zhang et al. designed a generalized normal distribution optimization to improve the accuracy of extracting the model parameters in the photovoltaic material industry [30]. Ibrahim et al. proposed a hybrid wind-driven-based fruit fly optimization algorithm for adjusting several model parameters in the double-diode cell material industry [31]. Tang et al. combined fuzzy C-means clustering and ant colony optimization algorithm for mission planning of unmanned aerial vehicles in emergent scenarios [32]. Tian applied a backtracking search optimization algorithm to adjust the algorithm parameters of the least square support vector machine in the computer field [33]. Jordehi designed a quadratic binary particle swarm optimization to schedule multiple interruptible and non-interruptible appliances in an intelligent home [34]. Nadimi-Shahraki et al. applied an enhanced whale optimization algorithm to detect coronavirus disease 2019 [35].

The search steps of these algorithms include exploration and exploitation phases [36]. In the exploration phase, it ensures the diversity of the population as much as possible so that the individuals in the population can fully explore each area of the feature space. If there is insufficient exploration, the solution is locally optimal. In the exploitation phase, the search should focus on areas with high-quality solutions, constantly searching the neighborhood. If underexploited, there is no way to converge to a global solution.

### *2.2 Ensemble Approaches of Population-Based Metaheuristics*

A single algorithm cannot handle all the optimization problems [37], so the ensemble of multiple meta-heuristic algorithms can increase the effect of optimization. The ensemble technology of the meta-heuristic algorithms includes low-level and high-level ensembles depending on the objects. The low-level ensemble integrates multiple search strategies and variable parameter values of a single algorithm. The high-level ensemble integrates multiple meta-heuristic algorithms and their variants.

An algorithm may be excellent at solving one type of optimization problem and not good at solving another. The search strategy of the algorithm mainly expresses this particularity. Therefore,

to enhance the effect of the algorithm on optimization problems that the algorithm is not good at, a variety of search strategies can be integrated to improve the algorithm to obtain a better solution. Qin et al. proposed a self-adaptive differential evolution algorithm that randomly assigned the experimental vector generation strategy and the control parameters to population individuals based on the characteristics of the historical solution [38]. Pan et al. designed an adaptive differential evolution algorithm based on the hybrid leapfrog strategy, which can maintain the overall diversity of the population during dynamic evolution [39]. Gong et al. applied a multi-operator search strategy based on a cheap agent model, using multiple sub-generation search operators and proxy models to select the best candidate solution [40]. Ali et al. divided the population into several subgroups, each adopting a modified mutation strategy based on differential evolution [41]. Rakhshani et al. proposed a new extension of the cuckoo algorithm, which combines the exploration capabilities of computer science with the development behavior provided by the covariance matrix adaptation evolution strategy through multiple search strategies [42]. Li et al. designed an adaptive learning framework for particle swarm optimization to select the optimal method from some strategies to deal with different search spaces [43].

When the set of optimization problems is vast, a single swarm intelligence optimization algorithm cannot handle it. So researchers ensemble multiple algorithms to solve optimization problems. Lynn et al. proposed an evolutionary particle swarm optimization algorithm that integrates different particle swarm optimization algorithms [44]. Zhang et al. proposed a multivariable coordination framework, which coordinates multiple improved differential evolution algorithms [45]. Thangavelu et al. designed an island-based dynamic data exchange algorithm mixing four classical differential evolution variants that fill each island with different dynamic data exchange variables as a possible way to implement a robust dynamic data exchange system [46]. Wu et al. proposed an algorithm based on an ensemble of multiple differential evolution variants, which divides the whole population into index and reward subpopulations and allocates the reward subpopulation dynamics to the well-performing differential evolution variant algorithm according to the performance of the index subpopulation [47]. Elsayed et al. applied a group of different particle swarm optimization variants to integrate and measure the performance of varying optimization variants according to the improvement index to adaptively determine the number of individuals allocated to the variant algorithm in each generation [48]. Vrugt et al. applied an adaptive learning strategy that integrates evolution strategy, genetic algorithm, and particle swarm optimization to dynamically adjust the influence of these three algorithms on individuals [49]. Xue et al. proposed an integrated evolutionary algorithm using an adaptive learning search technique incorporating three different algorithms [50]. Elsayed et al. designed a framework with multiple adaptive algorithms and operators, using two decisions to adaptively select the appropriate algorithm and search operator [51]. The existing ensemble frameworks are usually designed for specific algorithms and their variants, focusing on complementarity and neglecting generality.

In this paper, we design an ensemble framework based on common structural features of meta-heuristic algorithms rather than a specific algorithm. The ensemble framework belongs to the high-level ensemble, which could integrate various meta-heuristic and variant algorithms. These algorithms can have multiple search strategies, including low-level ensembles.

## 3 Ensemble of Population-Based Metaheuristics

This section introduces the EPM framework in detail. Section 3.1 explains the idea and features of the framework. Section 3.2 shows the stages of using the EPM framework to integrate the metaheuristic algorithm. It presents the implementation procedure and pseudocode in Section 3.3.

### *3.1 Basic Concepts*

#### *3.1.1 Algorithm Library*

After filling in multiple meta-heuristic algorithms, the ensemble framework can only solve optimization problems. These meta-heuristic algorithms are placed together in the algorithm library. It is hard to judge which population-based metaheuristic algorithm is the best for a particular optimization problem, especially when there is not enough information about the problem. While engineers can find better algorithms through trial and error, it would be time-consuming and labor-intensive. In addition, to solve some problems, a satisfactory solution cannot be obtained through a search strategy using a single algorithm. Neither the specific transition point from one algorithm to another nor which algorithms to apply is known. Each algorithm has its characteristics: local search, global search, optimization speed, and population diversity. The role of the EPM framework is to make all kinds of algorithms give full play to their respective strengths and provide a communication bridge for these algorithms. It requires that as many algorithms as possible be included in the algorithm library.

#### *3.1.2 Real and Virtual Populations*

The EPM framework proposes the setting and conversion of real and virtual populations. Virtual populations represent various virtual states of the population obtained by the evolution of the real population in different directions. All virtual populations come from the real population. Each virtual population applies a population-based metaheuristic algorithm, which updates the population by selection rules to obtain individuals with better fitness values. Each virtual population can exchange information to facilitate population optimization. When certain conditions are satisfied, it retains the optimal virtual population as the optimal population, which materializes into a real population. After that, it uses the same operation to virtualize and set up the next population that meets certain conditions, constantly optimizing the population until reaching the stop condition. The adjustment of parameters for each swarm intelligence algorithm is not considered, nor is the specific update strategy within the population. The setting of the real and virtual populations is a new idea for coordinated exploration and exploitation at the population level.

#### *3.1.3 Elite Strategy*

The exchange of information among virtual populations enables the population to obtain satisfactory solutions more quickly in the renewal process, reducing the incidences of falling into local optima. The algorithm framework designed in this study uses an elite strategy to communicate between virtual populations. To minimize the interference in the virtual population, the individual with the smallest fitness value is an elite in each virtual population. If multiple individuals have the same fitness value, it randomly selects one as the elite. It brings these elites together to form a group and selects the best elite according to their fitness. When meeting certain conditions, the best elite individual is added to each virtual population to complete the exchange of information between populations. In this way, it reduces the interference of virtual populations and realizes population optimization with the help of the optimal individual's influence on the population.

### *3.2 The Framework of EPM*

The EPM framework consists of three stages: the initial stage, the update stage, and the final stage. The update stage includes two sub-stages high-level generation and selection. Fig. 2 shows the entire process diagram of EPM. These three stages are described in detail below.

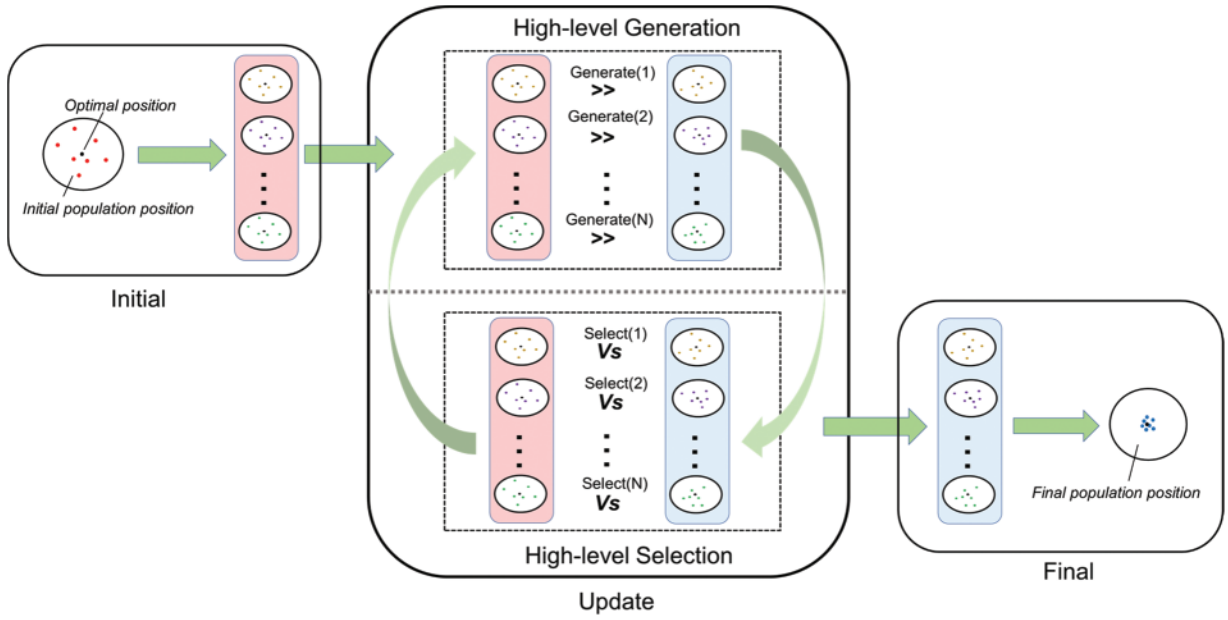


**Figure 2:** The entire process diagram of EPM

#### *3.2.1 The Initial Stage*

The search space dimension of the optimization problem is $D$. Its upper boundary $\mathbf{U}$ is $(u_1, u_2, \ldots, u_d, \ldots, u_D)$, and its lower boundary $\mathbf{L}$ is $(l_1, l_2, \ldots, l_d, \ldots, l_D)$. The number of individuals in the population is $N$, and the population set $\mathbf{X}$ is $\{\mathbf{X_1}, \mathbf{X_2}, \ldots, \mathbf{X_n}, \ldots, \mathbf{X_N}\}$. The position of the $n^{th}$ individual $\mathbf{X_n}$ is $(x_1^n, x_2^n, \ldots, x_d^n, \ldots, x_D^n)$, where $x_d^n$ is calculated according to Eq. (1), $r$ is a random number in the [0, 1].

It selects the required algorithm in the algorithm library, and the number of algorithms selected is $M$. The set of algorithms $\mathbf{A}$ is $\{A_1, A_2, \ldots A_m, \ldots, A_M\}$. $\mathbf{R}$ represents the real population, and $\mathbf{X^R(t)}$ represents the population at the $t^{th}$ iteration. The virtual population set $\mathbf{V}$ is $\{V_1, V_2, \ldots, V_m, \ldots, V_M\}$. $\mathbf{X^{Vm}(t)}$ represents the virtual population corresponding to the $m^{th}$ algorithm at the $t^{th}$ iteration. When $t = 0$, $\mathbf{X^R}(t)$ and $\mathbf{X^{Vm}}(t)$ are obtained from Eq. (2).

$$x_d^n = u_d + r(u_d - l_d) \tag{1}$$

$$\mathbf{X^{Vm}}(0) = \mathbf{X^R}(0) = \mathbf{X}(0), m = 1, 2, 3, \ldots M \tag{2}$$

#### *3.2.2 The Update Stage*

At the high-level generation process, each virtual population adopts the corresponding algorithm to update separately, where their internal update strategy set $\mathbf{Rw}$ is $\{Rw_1, Rw_2, \ldots, Rw_m, \ldots, Rw_M\}$. $Rw_m$ represents the internal update strategy of the algorithm $A_m$, and the update process of the virtual population position $X^{Vm}$ is shown in Fig. 3.

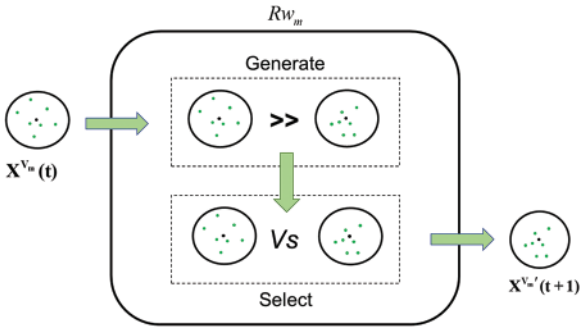


**Figure 3:** The updating process diagram of the virtual population position

At the $(t+1)^{th}$ iteration, $\mathbf{X}^{\mathbf{V_m}'}(\mathbf{t+1})$ is obtained from $\mathbf{X}^{\mathbf{V_m}}(\mathbf{t})$ through the generation and selection strategy within the algorithm $A_m$, as shown in Eq. (3). However, it is not the virtual population for the next update $\mathbf{X}^{\mathbf{V_m}}(\mathbf{t+1})$, which needs to be determined according to the high-level selection process.

$$\mathbf{X}^{\mathbf{V_m}'}(\mathbf{t+1}) = Rw_m(\mathbf{X}^{\mathbf{V_m}}(\mathbf{t})) \tag{3}$$

The entire high-level selection process is shown in Fig. 4. The elite swarm consists of individuals corresponding to the minimum fitness value in each virtual population. The set of these fitness values $\mathbf{E}$ is $\{E_1, E_2, \ldots, E_m, \ldots, E_M\}$, where $E_m$ represents the minimum fitness value in the $m^{th}$ virtual population. At the $(t+1)^{th}$ iteration, $E_m(t+1)$ is calculated by Eq. (4), where $f()$ represents calculating the fitness value. $B$ is the minimum in the set $\mathbf{E}$, and $B(t+1)$ is calculated by Eq. (5). Then, the individual with the minimum fitness value of the elite swarm $\mathbf{B}$ is considered as the best elite. Its corresponding virtual population is called the optimal virtual population $V_b$ and its corresponding individual position is $\mathbf{X}_{\mathbf{e}}^{\mathbf{V_b}}$. If there are multiple individuals corresponding to the minimum fitness value, random selection is performed.

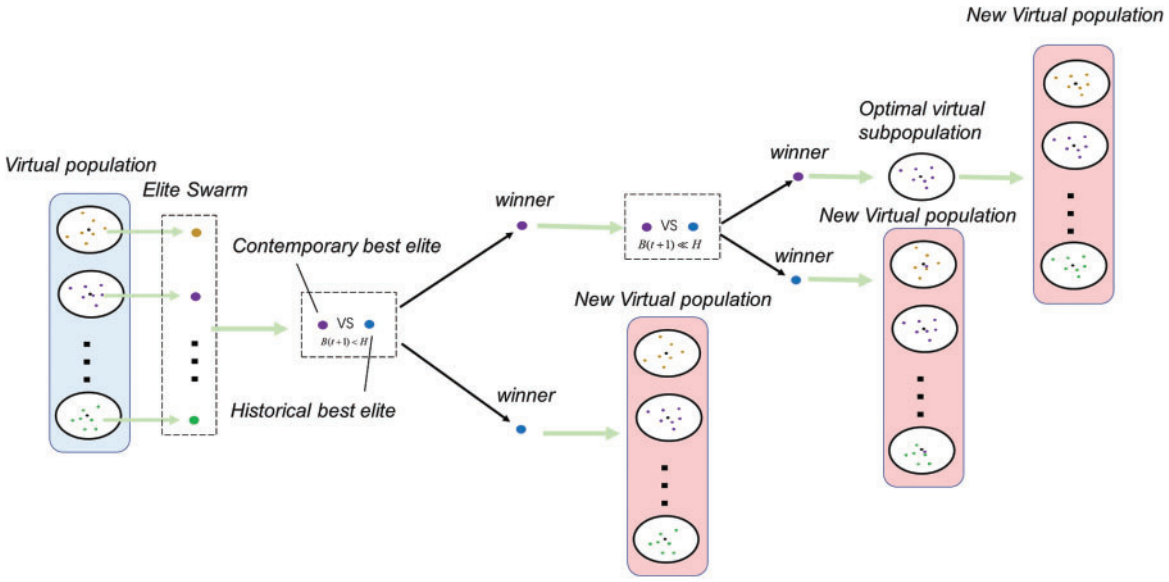


**Figure 4:** The process of a high-level selection

$$E_m(t+1) = \min f(\mathbf{X_i^{V_m'}(t+1)}), i = 1, 2, 3, \ldots, N \tag{4}$$

$$B(t+1) = \min\{E_1(t+1), \ldots, E_j(t+1), \ldots, E_M(t+1)\} \tag{5}$$

$H$ is the minimum fitness value of the best elite in history. When $B(t+1) \geq H$, all virtual populations are preserved, and no information exchange is carried out among them. The virtual populations to be updated next time is shown in Eq. (6).

$$\mathbf{X^{V_j}(t+1)} = \mathbf{X^{V_j'}(t+1)}, j = 1, 2, \ldots M \tag{6}$$

When $B(t+1) < H$, if $B(t+1) \ll H$, the optimal virtual population $V_b$ is materialized to the real population. The real population generates new virtual populations to be updated next time, as shown in Eq. (7). In this process, the parameters in the algorithms corresponding to the original virtual populations are retained. In this paper, it can be determined that the difference is large if one of the following three conditions is met: $\{B(t+1) \times H > 0, B(t+1) > 0, a \leq 0.1\}$, $\{B(t+1) \times H > 0, H < 0, a \geq 10\}$, $\left\{B(t+1) \times H < 0, e^{B(t+1)} < e^H\right\}$, where $a$ is the ratio of $B(t+1)$ to $H$, as shown in Eq. (8). If $B(t+1)$ is not much different from $H$, the best elite is copied and distributed to other virtual populations and an individual in each virtual population is randomly replaced in the process. For virtual population $V_m$, $q$ is a random integer in $[1, N]$, representing a random individual which will be replaced by the best elite. The improved population will be updated in the next time as the new virtual populations, and the process is shown in Eq. (9).

$$\mathbf{X^{V_j}(t+1)} = \mathbf{X^{R}(t+1)} = \mathbf{X^{V_b'}(t+1)}, j = 1, 2, \ldots, N \tag{7}$$

$$a = \frac{B(t+1)}{H}, H \neq 0 \tag{8}$$

$$\mathbf{X^{V_j}(t+1)} \overset{\mathbf{X_q^{V_j'}(t+1)=X_e^{V_b}(t+1)}}{\longleftarrow} \mathbf{X^{V_j'}(t+1)}, j = 1, 2, \ldots, M \tag{9}$$

### *3.2.3 The Final Stage*

The update stage repeats until reaching the maximum of iterations $T$, then the whole process ends. In the last iteration, the optimal virtual population $V_b$ is materialized to obtain the real population, as shown in Eq. (10). After the last iteration, the optimal virtual population is transformed into the real population.

$$\mathbf{X^R(T) = X^{V_b}(T)} \tag{10}$$

## *3.3 General Process of EPM*

The ensemble range covered by the EPM framework includes all population-based metaheuristic algorithms and their modified versions. The larger the number of algorithms in the algorithm library, the better the possibility of covering many aspects of the optimization problem. First, it selects an appropriate number of algorithms from the algorithm library for the ensemble. In this process, the number of algorithms can be set without prior knowledge and then selected randomly. In the case of specific prior knowledge, it chooses the appropriate algorithm according to the previous knowledge. The pseudocode of EPM is presented below:

***Input*** *N, T, M*
***Initialize***
*Initialize the population position X according to* Eq. (1)
*Get* $X^R$, $X^V$ *according to* Eq. (2)
***Update repeatedly***
***While*** $t < T$
  ***For*** $j = 1{:}M$
    *Get* $X^{Vj'}(t+1)$ *from* $X^{Vj}(t)$ *according to* Eq. (3)
  ***End***
  *Obtain* $B(t+1)$ *according to* Eqs. (4) *and* (5)
  ***For*** $j = 1{:}\ M$
    ***If*** $B(t+1) \geq H$
      *Get* $X^{Vj}(t+1)$ *from* $X^{Vj'}(t+1)$ *according to* Eq. (6)
    ***Else if*** $B(t+1) \ll H$
      *Get* $X^{Vj}(t+1)$ *from* $X^{Vb'}(t+1)$ *according to* Eq. (7)
    ***Else***
      *Get* $X^{Vj}(t+1)$ *from* $X^{Vj'}(t+1)$ *according to* Eq. (9)
    ***End***
  ***End***
  *Get the historically best elite H*
  $t = t+1$
***End while***
***Output*** *H*

## 4 Experiment Results

It applies ensemble algorithms to two test sets, 23 standard benchmark functions [52] and the 2017 congress on evolutionary computation (CEC2017) benchmark functions [53]. Section 4.1 describes these functions in detail. Section 4.2 presents the five algorithms in the algorithm library along with a comprehensive comparison between the integrated and original algorithms, introducing the parameters of each algorithm in the experiment. Section 4.3 illustrates the experimental results of the nine algorithms. It gives the iterative curves of all algorithms on some functions in Section 4.4 to directly observe the optimization effect and convergence of the algorithms. Section 4.5 shows the evolution of the real and virtual populations in the optimization process. Section 4.6 applies the ensemble framework to solve two engineering application problems.

### *4.1 Benchmark Functions*

We select two sets of benchmark functions. One set consists of the 23 standard benchmark functions of the experiment, which are described in detail in Table 1, including function type, function name, and function definition. D denotes the number of independent variables in the function, and the initialization range means the range of variables. The optimal value corresponding to each function is the global optimum. In these functions, F1–F13 are high-dimensional with a dimension of 30; F1–F7 are unimodal; F8–F13 are multi-peak; F14–F23 are low-dimensional with only a few local minima. The other set is the CEC2017 benchmark functions set, as described in Table 2. Among these functions,

C1–C2 are unimodal; C3–C9 are simple multi-peak; C10–C19 are mixed; C20–C29 are compound. The variable dimensions range from [−100, 100]. We set the dimensions uniformly to 30 and 50.

**Table 1:** 23 standard benchmark functions [52]

| Function type | No. | Function name | D | Initialization range | Global optimum |
|---|---|---|---|---|---|
| Unimodal test functions | F1 | Sphere model | 30 | $-100 \leq x_i \leq 100$ | 0 |
| | F2 | Schwefel's problem 2.22 | 30 | $-10 \leq x_i \leq 10$ | 0 |
| | F3 | Schwefel's problem 1.2 | 30 | $-100 \leq x_i \leq 100$ | 0 |
| | F4 | Schwefel's problem 2.21 | 30 | $-100 \leq x_i \leq 100$ | 0 |
| | F5 | Generalized Rosenbrock's function | 30 | $-30 \leq x_i \leq 30$ | 0 |
| | F6 | Step function | 30 | $-100 \leq x_i \leq 100$ | 0 |
| | F7 | Quartic function | 30 | $-1.28 \leq x_i \leq 1.28$ | 0 |
| Multimodal test functions | F8 | Generalized Schwefel's problem 2.26 | 30 | $-500 \leq x_i \leq 500$ | $-418.9829 \times d$ |
| | F9 | Generalized Rastrigin's function | 30 | $-5.12 \leq x_i \leq 5.12$ | 0 |
| | F10 | Ackley's function | 30 | $-32 \leq x_i \leq 32$ | 0 |
| | F11 | Generalized Griewank function | 30 | $-600 \leq x_i \leq 600$ | 0 |
| | F12 | Generalized Penalized functions | 30 | $-50 \leq x_i \leq 50$ | 0 |
| | F13 | Generalized Penalized functions | 30 | $-50 \leq x_i \leq 50$ | 0 |
| | F14 | Shekel's Foxholes function | 2 | $-65.53 \leq x_i \leq 65.53$ | 1 |
| | F15 | Kowalik's function | 4 | $-5 \leq x_i \leq 5$ | 0.0003 |
| | F16 | Six-hump camel-back function | 2 | $-5 \leq x_i \leq 5$ | −1.0316 |
| | F17 | Branin function | 2 | $-5 \leq x_i \leq 10$, $0 \leq x_2 \leq 15$ | 0.398 |

(Continued)

**Table 1 (continued)**

| Function type | No. | Function name | D | Initialization range | Global optimum |
|---|---|---|---|---|---|
| Multimodal test functions with fix dimension | F18 | Goldstein-price function | 2 | $-5 \leq x_i \leq 5$ | 3 |
| | F19 | Hartman's family | 3 | $0 \leq x_i \leq 1$ | −3.86 |
| | F20 | Hartman's family | 6 | $0 \leq x_i \leq 10$ | −3.32 |
| | F21 | Shekel's family | 4 | $0 \leq x_i \leq 10$ | −10.1532 |
| | F22 | Shekel's family | 4 | $0 \leq x_i \leq 10$ | −10.4028 |
| | F23 | Shekel's family | 4 | $0 \leq x_i \leq 10$ | −10.5363 |

**Table 2:** The CEC2017 benchmark functions [53]

| Function type | No. | Function name | Dim | Initialization range | Global optimum |
|---|---|---|---|---|---|
| Unimodal functions | C1 | Shifted and rotated bent cigar function | 30, 50 | $-100 \leq x_i \leq 100$ | 100 |
| | C2 | Shifted and rotated Zakharov function | 30, 50 | $-100 \leq x_i \leq 100$ | 200 |
| Simple multimodal functions | C3 | Shifted and rotated Rosenbrock's function | 30, 50 | $-100 \leq x_i \leq 100$ | 300 |
| | C4 | Shifted and rotated Rastrigin's function | 30, 50 | $-100 \leq x_i \leq 100$ | 400 |
| | C5 | Shifted and rotated expanded Scaffer's F6 function | 30, 50 | $-100 \leq x_i \leq 100$ | 500 |
| | C6 | Shifted and rotated Lunacek Bi_Rastrigin function | 30, 50 | $-100 \leq x_i \leq 100$ | 600 |
| | C7 | Shifted and rotated non-continuous Rastrigin's function | 30, 50 | $-100 \leq x_i \leq 100$ | 700 |
| | C8 | Shifted and rotated Levy function | 30, 50 | $-100 \leq x_i \leq 100$ | 800 |
| | C9 | Shifted and rotated Schwefel's function | 30, 50 | $-100 \leq x_i \leq 100$ | 900 |
| Hybrid functions | C10 | Hybrid function 1 | 30, 50 | $-100 \leq x_i \leq 100$ | 1000 |
| | C11 | Hybrid function 2 | 30, 50 | $-100 \leq x_i \leq 100$ | 1100 |
| | C12 | Hybrid function 3 | 30, 50 | $-100 \leq x_i \leq 100$ | 1200 |
| | C13 | Hybrid function 4 | 30, 50 | $-100 \leq x_i \leq 100$ | 1300 |
| | C14 | Hybrid function 5 | 30, 50 | $-100 \leq x_i \leq 100$ | 1400 |

(Continued)

**Table 2 (continued)**

| Function type | No. | Function name | Dim | Initialization range | Global optimum |
|---|---|---|---|---|---|
| Hybrid functions | C15 | Hybrid function 6 | 30, 50 | $-100 \leq x_i \leq 100$ | 1500 |
| | C16 | Hybrid function 7 | 30, 50 | $-100 \leq x_i \leq 100$ | 1600 |
| | C17 | Hybrid function 8 | 30, 50 | $-100 \leq x_i \leq 100$ | 1700 |
| | C18 | Hybrid function 9 | 30, 50 | $-100 \leq x_i \leq 100$ | 1800 |
| | C19 | Hybrid function 10 | 30, 50 | $-100 \leq x_i \leq 100$ | 1900 |
| Composition functions | C20 | Composition function 1 | 30, 50 | $-100 \leq x_i \leq 100$ | 2000 |
| | C21 | Composition function 2 | 30, 50 | $-100 \leq x_i \leq 100$ | 2100 |
| | C22 | Composition function 3 | 30, 50 | $-100 \leq x_i \leq 100$ | 2200 |
| | C23 | Composition function 4 | 30, 50 | $-100 \leq x_i \leq 100$ | 2300 |
| | C24 | Composition function 5 | 30, 50 | $-100 \leq x_i \leq 100$ | 2400 |
| | C25 | Composition function 6 | 30, 50 | $-100 \leq x_i \leq 100$ | 2500 |
| | C26 | Composition function 7 | 30, 50 | $-100 \leq x_i \leq 100$ | 2600 |
| | C27 | Composition function 8 | 30, 50 | $-100 \leq x_i \leq 100$ | 2700 |
| | C28 | Composition function 9 | 30, 50 | $-100 \leq x_i \leq 100$ | 2800 |
| | C29 | Composition function 10 | 30, 50 | $-100 \leq x_i \leq 100$ | 2900 |

### *4.2 Original Algorithms and Ensemble Algorithms*

In the experiment, we select some new representative algorithms for the ensemble, which are Harris hawks optimization (HHO) [54], seagull optimization algorithm (SOA) [55], slime mold algorithm (SMA) [56], salp swarm algorithm (SSA) [57], manta ray foraging optimization (MRFO) algorithm [58]. The EPM class algorithms represent the algorithms integrated under the EPM framework. This paper names the algorithms integrated by the EPM framework as "EPM + number", where "number" represents the number of filled algorithms in the ensemble framework. It obtains the EPM2 algorithm by integrating HHO and SOA, the EPM3 algorithm by integrating HHO, SOA, and SMA, the EPM4 algorithm by integrating HHO, SOA, SMA, and SSA, the EPM5 algorithm by integrating HHO, SOA, SMA, SSA, and MRFO. The ensemble framework does not interfere with the updated formulas of the integrated algorithms, so researchers can adjust their parameters according to the characteristics of the original algorithms. The parameters of the original algorithms are consistent with the papers that proposed these algorithms. The parameters in integrated algorithms are the same as the original algorithms' parameters. It lists the settings of the parameters in Table 3.

**Table 3:** The parameter settings of the algorithms

| Algorithm | Parameter | Introduction and settings |
|---|---|---|
| HHO | $r_1$, $r_2$, $r_3$, $r_4$, q, $E_0$, r, J, S, $\beta$ | $r_1$, $r_2$, $r_3$, $r_4$, q, $E_0$, r are all random numbers in [0, 1]; J is the following number within [0, 2]; S is a random vector of dimension D; the elements are random numbers in [0, 1]; and $\beta$ is set to 1.5. |

(Continued)

**Table 3 (continued)**

| Algorithm | Parameter | Introduction and settings |
|---|---|---|
| SOA | $f_c$, $r_d$, θ, u, v, e | $f_c$ decreases linearly from 2 to 0, $r_d$ is a random number in [0, 1]; θ is a random angle in [0, $2\pi$]; u = 1, v = 1; e is the base of the natural logarithm. |
| SMA | rand, z, $v_b$, $v_c$, r | rand is in [0, 1], and the location update parameter z = 0.03; $v_b$ is in [−a, a]; $v_c$ linearly decreases from 1 to 0, and r represents a random value in [0, 1]. |
| SSA | $c_1$, $c_2$, $c_3$ | $c_1$ decreases from 2 to 0, and $c_2$ and $c_3$ are random numbers in [0, 1]. |
| MRFO | r, $r_2$, $r_3$, rand | r, $r_1$, $r_2$, $r_3$ and rand represent random numbers in [0, 1]. |

### *4.3 Comparison Results*

The average (AVE) and standard deviation (STD) of the best solutions obtained by all test algorithms in 25 independent runs and 500 iterations in each dimension were recorded (see Tables 4–6) and compared.

**Table 4:** The comparison results of 23 standard benchmark functions by different optimization algorithms

| Function | Metric | EPM5 | EPM4 | EPM3 | EPM2 | HHO | SOA | SMA | SSA | MRFO |
|---|---|---|---|---|---|---|---|---|---|---|
| F1 | AVE | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3.09E−09 | 0 |
| | STD | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7.33E−10 | 0 |
| F2 | AVE | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.355149 | 0 |
| | STD | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.784676 | 0 |
| F3 | AVE | 0 | 0 | 0 | 0 | 0 | 2.45912 | 0 | 1.77E−08 | 0 |
| | STD | 0 | 0 | 0 | 0 | 0 | 5.722051 | 0 | 4.23E−09 | 0 |
| F4 | AVE | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4.48E−05 | 0 |
| | STD | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2.86E−05 | 0 |
| F5 | AVE | 4.07E−26 | 7.13E−10 | 1.62E−09 | 2.13E−09 | 1.03E−06 | 18.78447 | 0.000457 | 35.12469 | 3.85E−11 |
| | STD | 1.59E−25 | 1.78E−09 | 3.82E−09 | 5.7E−09 | 1.24E−06 | 13.60155 | 0.000292 | 44.13715 | 4.72E−11 |
| F6 | AVE | 4.44E−33 | 3.15E−11 | 4.46E−11 | 6.52E−11 | 1.34E−08 | 0.563822 | 1.61E−06 | 2.87E−09 | 7.4E−34 |
| | STD | 1.06E−32 | 5.12E−11 | 1.02E−10 | 1.51E−10 | 1.61E−08 | 0.542323 | 5.27E−07 | 5.49E−10 | 2.56E−33 |
| F7 | AVE | 1.81E−06 | 1.74E−06 | 1.77E−06 | 2.12E−06 | 2.11E−06 | 4.14E−06 | 2.7E−06 | 0.00271 | 2.−06 |
| | STD | 1.44E−06 | 1.5E−06 | 1.54E−06 | 1.87E−06 | 1.47E−06 | 4.02E−06 | 3.11E−06 | 0.001164 | 1.45E−06 |
| F8 | AVE | −12569.5 | −12569.5 | −12569.5 | −12569.5 | −12569.5 | −12569.5 | −12569.5 | −7895.55 | −8805.3 |
| | STD | 1.96E−12 | 8.4E−10 | 8.01E−09 | 7.67E−11 | 6.6E−05 | 2.81E−05 | 0.000112 | 720.5855 | 480.4721 |
| F9 | AVE | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 55.08084 | 0 |
| | STD | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14.91848 | 0 |
| F10 | AVE | 8.88E−16 | 8.88E−16 | 8.88E−16 | 8.88E−16 | 8.88E−16 | 8.88E−16 | 8.88E−16 | 1.743802 | 8.88E−16 |
| | STD | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.922858 | 0 |
| F11 | AVE | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.014162 | 0 |
| | STD | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.012837 | 0 |
| F12 | AVE | 1.59E−32 | 3.87E−12 | 8.57E−12 | 1.23E−11 | 1.05E−09 | 0.01385 | 4.64E−07 | 0.355485 | 1.58E−32 |
| | STD | 3.1E−34 | 4.38E−12 | 1.04E−11 | 2.02E−11 | 1.52E−09 | 0.011173 | 4.2E−07 | 0.615933 | 1.2E−34 |
| F13 | AVE | 1.38E−32 | 5.84E−11 | 9.1E−11 | 9.06E−11 | 6.77E−09 | 0.132341 | 8.44E−07 | 0.002637 | 1.355246 |
| | STD | 1.04E−33 | 8.07E−11 | 1.42E−10 | 1.91E−10 | 1.19E−08 | 0.12463 | 5.17E−07 | 0.004789 | 1.412608 |
| F14 | AVE | 0.998004 | 0.998004 | 0.998004 | 1.037765 | 1.037765 | 4.687508 | 0.998004 | 0.998004 | 0.998004 |

(Continued)

**Table 4 (continued)**

| Function | Metric | EPM5 | EPM4 | EPM3 | EPM2 | HHO | SOA | SMA | SSA | MRFO |
|---|---|---|---|---|---|---|---|---|---|---|
| | STD | 0 | 7.02E−15 | 5.48E−14 | 0.198805 | 0.198805 | 4.688223 | 3.01E−15 | 1.83E−16 | 0 |
| F15 | AVE | 0.000307 | 0.000307 | 0.000308 | 0.000308 | 0.000308 | 0.001109 | 0.000368 | 0.000815 | 0.000385 |
| | STD | 2.06E−19 | 1.21E−11 | 6.38E−08 | 4.96E−08 | 4.58E−08 | 0.00072 | 0.000192 | 0.000363 | 0.000253 |
| F16 | AVE | −1.03163 | −1.03163 | −1.03163 | −1.03163 | −1.03163 | −1.03163 | −1.03163 | −1.03163 | −1.03163 |
| | STD | 6.8E−16 | 1.33E−15 | 2.2E−14 | 5.87E−14 | 9.27E−15 | 3.21E−07 | 1.25E−12 | 1.38E−14 | 6.8E−16 |
| F17 | AVE | 0.397887 | 0.397887 | 0.397887 | 0.397887 | 0.397887 | 0.398059 | 0.397887 | 0.397887 | 0.397887 |
| | STD | 0 | 2.35E−14 | 3.56E−09 | 1.75E−08 | 1.14E−08 | 0.000401 | 1.96E−09 | 1.32E−14 | 0 |
| F18 | AVE | 3 | 3 | 3 | 4.08 | 5.16 | 12.20428 | 3 | 3 | 3 |
| | STD | 1.26E−15 | 2.91E−13 | 5.97E−12 | 5.400001 | 7.475962 | 13.71568 | 3.01E−14 | 2.46E−13 | 5.94E−16 |
| F19 | AVE | −3.86278 | −3.86278 | −3.86278 | −3.86278 | −3.86278 | −3.74259 | −3.86278 | −3.86278 | −3.86278 |
| | STD | 2.27E−15 | 1.81E−14 | 5.47E−08 | 1.39E−06 | 6.15E−07 | 0.170625 | 4.29E−09 | 5.74E−15 | 2.27E−15 |
| F20 | AVE | −3.29346 | −3.26493 | −3.25541 | −3.25539 | −3.25538 | −2.96169 | −3.21261 | −3.21261 | −3.26493 |
| | STD | 0.051824 | 0.060624 | 0.060235 | 0.060236 | 0.060235 | 0.170635 | 0.03292 | 0.03292 | 0.060624 |
| F21 | AVE | −10.1532 | −10.1532 | −10.1532 | −6.07478 | −5.05519 | −4.44062 | −10.1532 | −9.74901 | −9.94928 |
| | STD | 5.44E−15 | 1.94E−05 | 1.81E−05 | 2.081226 | 5.5E−06 | 1.754009 | 7.69E−06 | 1.398954 | 1.0196 |
| F22 | AVE | −10.4029 | −10.4029 | −10.4029 | −6.57593 | −5.08767 | −5.83812 | −10.4029 | −9.58101 | −10.1903 |
| | STD | 1.92E−15 | 5.69E−06 | 2.45E−05 | 2.435741 | 6.22E−06 | 2.508465 | 1.02E−05 | 2.305573 | 1.063054 |
| F23 | AVE | −10.5364 | −10.5364 | −10.5364 | −6.21004 | −5.5611 | −4.8448 | −10.5364 | −10.322 | −10.5364 |
| | STD | 1.92E−15 | 1.48E−05 | 2.86E−05 | 2.207733 | 1.497374 | 2.028815 | 7.76E−06 | 1.072153 | 1.81E−15 |

**Table 5:** The comparison results of the CEC2017 benchmark functions by different optimization algorithms (D = 30)

| | | EPM5 | EPM4 | EPM3 | EPM2 | HHO | SOA | SMA | SSA | MRFO |
|---|---|---|---|---|---|---|---|---|---|---|
| C1 | AVE | 902.7646 | 3290.374 | 2685.852 | 489759.6 | 5825.588 | 4.71E+10 | 7489.476 | 3354.55 | 3953.039 |
| | STD | 1118.24 | 4044.062 | 4243.236 | 2418277 | 3309.647 | 9.59E+09 | 7115.915 | 3783.203 | 4491.838 |
| C2 | AVE | 200 | 200.0002 | 200.0012 | 8.88E+33 | 4.77E+31 | 2.57E+49 | 200.0002 | 200.0001 | 200 |
| | STD | 1.76E−05 | 6.6E−05 | 0.00085 | 4.28E+34 | 1.64E+32 | 1.28E+50 | 0.000123 | 3.04E−05 | 9.63E−06 |
| C3 | AVE | 300 | 300 | 300.847 | 28223.06 | 32767.91 | 137237.7 | 300.001 | 300 | 300 |
| | STD | 0 | 7.04E−09 | 0.5768 | 5322.439 | 4405.763 | 45948.26 | 0.000584 | 3.77E−09 | 0 |
| C4 | AVE | 404.4552 | 436.6484 | 428.9964 | 662.9686 | 634.4399 | 11593.27 | 488.2914 | 479.8565 | 409.3811 |
| | STD | 11.44006 | 25.74595 | 23.56127 | 218.7391 | 108.0743 | 3366.472 | 5.52105 | 25.40835 | 21.0248 |
| C5 | AVE | 536.893 | 531.9988 | 559.8022 | 730.6306 | 746.4483 | 880.6916 | 575.2127 | 609.7334 | 649.9196 |
| | STD | 9.786207 | 12.33574 | 16.24438 | 33.09082 | 29.05142 | 43.61556 | 18.01149 | 29.2981 | 40.06929 |
| C6 | AVE | 600.8313 | 600.5959 | 601.5444 | 653.6156 | 657.7857 | 678.0107 | 600.3176 | 625.4053 | 619.4164 |
| | STD | 0.593362 | 0.375077 | 0.882561 | 7.245739 | 4.69095 | 9.483725 | 0.192387 | 11.44769 | 12.61308 |
| C7 | AVE | 766.9689 | 761.2482 | 792.5594 | 1199.308 | 1256.393 | 1418.424 | 811.1121 | 859.4338 | 989.8419 |
| | STD | 8.979743 | 10.43814 | 15.01733 | 63.29684 | 66.28971 | 58.77982 | 20.65983 | 35.38709 | 86.96931 |
| C8 | AVE | 836.8598 | 833.5277 | 850.9737 | 960.6261 | 968.3868 | 1106.081 | 877.8322 | 894.8789 | 932.9259 |
| | STD | 6.6184 | 9.776067 | 14.33767 | 23.16343 | 25.06896 | 28.21193 | 13.90581 | 31.50098 | 28.2599 |
| C9 | AVE | 900.3583 | 900.6909 | 900.5986 | 4955.688 | 5063.042 | 9446.444 | 1506.398 | 2220.573 | 3669.055 |
| | STD | 0.374084 | 0.724316 | 0.718643 | 698.6464 | 609.5586 | 1539.526 | 1027.192 | 879.4835 | 1097.606 |
| C10 | AVE | 2628.831 | 2540.615 | 2678.661 | 6063.822 | 5253.509 | 9541.206 | 3559.888 | 4575.969 | 4808.949 |
| | STD | 308.2059 | 345.6339 | 473.6041 | 994.8354 | 885.5511 | 529.2566 | 475.7757 | 705.6062 | 715.4212 |
| C11 | AVE | 1126.983 | 1132.647 | 1139.906 | 1276.364 | 1287.052 | 10240.64 | 1194.034 | 1256.286 | 1184.318 |
| | STD | 8.540598 | 10.10047 | 14.1604 | 75.00597 | 64.74324 | 3050.719 | 41.25983 | 44.91092 | 31.17383 |
| C12 | AVE | 15027.23 | 88204.73 | 100155.4 | 45934047 | 67881154 | 8.45E+09 | 238634.7 | 771305.1 | 19146.42 |
| | STD | 9080.098 | 56539.9 | 60453.42 | 72150740 | 1.1E+08 | 3.71E+09 | 145317.3 | 689276.8 | 12496.58 |
| C13 | AVE | 5477.185 | 27176.19 | 25679.45 | 173224 | 19554468 | 2.3E+09 | 26860.77 | 95078.71 | 16508.89 |
| | STD | 4442.871 | 20246.44 | 25013.57 | 194785 | 97111555 | 3.12E+09 | 25417.04 | 62176.79 | 12679.22 |
| C14 | AVE | 1567.886 | 3082.811 | 4749.069 | 339770.8 | 280555 | 1441032 | 9348.684 | 3512.879 | 1604.296 |
| | STD | 45.90245 | 1584.016 | 3683.581 | 329934.4 | 287152.8 | 1445004 | 6823.33 | 2262.68 | 71.7545 |
| C15 | AVE | 3623.123 | 8074.151 | 10605.32 | 33162.86 | 37733.69 | 4.18E+08 | 30796.05 | 50766.18 | 5584.899 |

(Continued)

**Table 5 (continued)**

| | | EPM5 | EPM4 | EPM3 | EPM2 | HHO | SOA | SMA | SSA | MRFO |
|---|---|---|---|---|---|---|---|---|---|---|
| | STD | 2810.202 | 6215.589 | 8942.196 | 29081.38 | 32078.31 | 5.53E+08 | 11458.03 | 40469.1 | 6645.642 |
| C16 | AVE | 1801.661 | 1949.337 | 2125.943 | 3153.129 | 3258.431 | 4787.399 | 2400.258 | 2407.28 | 2631.215 |
| | STD | 130.349 | 205.3241 | 214.2251 | 495.0602 | 448.0817 | 824.6578 | 303.5794 | 228.4946 | 279.2315 |
| C17 | AVE | 1832.242 | 1820.747 | 1955.324 | 2518.757 | 2620.966 | 3751.296 | 2063.12 | 1993.909 | 2129.348 |
| | STD | 57.19301 | 63.95016 | 131.0033 | 317.6166 | 327.4079 | 1679.326 | 186.789 | 164.0414 | 215.483 |
| C18 | AVE | 22870.17 | 60371.9 | 65056.1 | 1053220 | 924321.3 | 53180666 | 166444.2 | 86677.09 | 29835.96 |
| | STD | 19990.05 | 38430.41 | 48955.33 | 1342197 | 827943.6 | 75027162 | 76311.93 | 48923.04 | 17856.1 |
| C19 | AVE | 3824.338 | 13040.8 | 10533.51 | 50554.58 | 42888.47 | 4.7E+08 | 41470.4 | 108043.6 | 10004.24 |
| | STD | 3315.617 | 12306.69 | 10690.86 | 37080.57 | 56123.46 | 6.29E+08 | 17380.8 | 49528.17 | 10141.75 |
| C20 | AVE | 2119.088 | 2148.06 | 2169.274 | 2616.076 | 2650.025 | 2957.38 | 2313.325 | 2351.115 | 2380.781 |
| | STD | 65.23756 | 90.19009 | 116.4725 | 153.5199 | 168.6338 | 200.5934 | 129.0021 | 146.3238 | 209.8808 |
| C21 | AVE | 2338.545 | 2338.708 | 2354.376 | 2506.347 | 2548.73 | 2681.681 | 2388.157 | 2405.765 | 2412.039 |
| | STD | 9.689742 | 8.679737 | 14.91288 | 40.69084 | 46.96825 | 61.8029 | 23.58705 | 33.65642 | 27.48967 |
| C22 | AVE | 2300.121 | 2300.223 | 2301.24 | 6730.48 | 6983.713 | 9654.64 | 5344.452 | 3911.237 | 2607.776 |
| | STD | 0.57528 | 0.597588 | 1.233973 | 1600.559 | 678.1923 | 1098.703 | 530.8082 | 2053.508 | 1065.147 |
| C23 | AVE | 2685.296 | 2683.946 | 2703.451 | 3166.514 | 3206.271 | 3328.737 | 2731.09 | 2739.832 | 2790.808 |
| | STD | 9.284617 | 11.54514 | 13.65706 | 135.4268 | 150.4809 | 139.2706 | 19.91425 | 25.89812 | 44.39223 |
| C24 | AVE | 2856.568 | 2857.054 | 2888.855 | 3335.164 | 3359.277 | 3583.298 | 2929.555 | 2898.986 | 2979.212 |
| | STD | 12.73916 | 11.06584 | 14.77498 | 141.8446 | 141.0508 | 195.7917 | 21.97067 | 24.94498 | 66.27963 |
| C25 | AVE | 2885.649 | 2886.709 | 2886.745 | 2958.363 | 2952.888 | 4572.914 | 2886.848 | 2893.028 | 2895.376 |
| | STD | 1.601232 | 0.681923 | 0.685797 | 32.52224 | 39.78769 | 460.3713 | 1.030813 | 14.86022 | 16.38967 |
| C26 | AVE | 3725.473 | 3934.553 | 4036.657 | 7607.326 | 8304.568 | 10358.2 | 4472.035 | 4464.339 | 4870.579 |
| | STD | 464.9816 | 165.8759 | 275.9082 | 1334.879 | 1258.937 | 1119.672 | 235.3289 | 831.3642 | 1641.959 |
| C27 | AVE | 3166.58 | 3168.203 | 3168.97 | 3611.143 | 3741.738 | 3976.634 | 3206.848 | 3224.959 | 3267.911 |
| | STD | 10.76602 | 8.738246 | 8.745193 | 148.5094 | 197.485 | 322.848 | 12.5343 | 15.85913 | 24.35195 |
| C28 | AVE | 3100 | 3100.003 | 3100.089 | 3453.845 | 3394.916 | 6406.415 | 3224.79 | 3132.602 | 3126.623 |
| | STD | 0 | 0.015433 | 0.026477 | 160.969 | 169.3918 | 710.1894 | 66.59074 | 47.08401 | 55.40508 |
| C29 | AVE | 3390.706 | 3437.384 | 3431.83 | 5012.228 | 5167.606 | 6355.715 | 3697.814 | 3789.834 | 3942.537 |
| | STD | 79.95842 | 94.77418 | 108.0931 | 434.1358 | 712.0787 | 824.497 | 174.1942 | 173.2771 | 271.9866 |

**Table 6:** The comparison results of the CEC2017 benchmark functions by different optimization algorithms (D = 50)

| | | EPM5 | EPM4 | EPM3 | EPM2 | HHO | SOA | SMA | SSA | MRFO |
|---|---|---|---|---|---|---|---|---|---|---|
| C1 | AVE | 1564.423 | 3363.094 | 5868.441 | 15223.61 | 14185.43 | 1.01E+11 | 15409.67 | 4773.522 | 4792.013 |
| | STD | 2074.851 | 3416.559 | 6299.051 | 5317.337 | 4192.902 | 8.13E+09 | 11336.18 | 6920.711 | 6722.137 |
| C2 | AVE | 200 | 202.2101 | 340.291 | 1.86E+59 | 5.31E+59 | 6.26E+84 | 201.7396 | 200.8243 | 200 |
| | STD | 9.59E−06 | 7.80E+00 | 90.16485 | 8.38E+59 | 2.66E+60 | 3.12E+85 | 7.430594 | 4.12E+00 | 1.48E−05 |
| C3 | AVE | 300 | 300 | 303.7261 | 44216.94 | 47427.25 | 252790.3 | 300.0082 | 300 | 300 |
| | STD | 4.84E−13 | 2.31E−08 | 1.75844 | 7642.29 | 9387.97 | 90954.65 | 0.002057 | 9.08E−09 | 2.39E−13 |
| C4 | AVE | 417.1868 | 433.3292 | 430.2082 | 933.5133 | 945.979 | 30758.01 | 504.3023 | 544.9028 | 422.8925 |
| | STD | 14.18587 | 20.07223 | 1.988095 | 599.3669 | 483.4245 | 6286.819 | 59.34425 | 47.54378 | 28.53133 |
| C5 | AVE | 586.8017 | 570.0516 | 615.8567 | 852.495 | 849.6292 | 1150.826 | 687.0763 | 742.8122 | 815.8375 |
| | STD | 11.33574 | 19.6727 | 18.62161 | 33.7776 | 26.29409 | 36.83114 | 41.034 | 59.28636 | 51.54474 |
| C6 | AVE | 604.7959 | 603.0001 | 606.5931 | 661.39 | 663.1714 | 694.9141 | 601.0333 | 638.0651 | 640.2838 |
| | STD | 1.728911 | 1.287632 | 2.364182 | 5.401158 | 4.73241 | 7.585304 | 0.506592 | 9.609981 | 11.76814 |
| C7 | AVE | 826.695 | 834.6281 | 879.2088 | 1648.979 | 1693.569 | 2042.556 | 926.7816 | 1009.936 | 1367.78 |
| | STD | 15.39112 | 13.70104 | 28.97706 | 80.90631 | 53.91651 | 46.16475 | 45.60423 | 72.26719 | 178.9801 |
| C8 | AVE | 885.8617 | 870.7809 | 915.449 | 1144.098 | 1163.041 | 1441.307 | 963.9156 | 1038.168 | 1119.021 |
| | STD | 16.68351 | 17.65313 | 18.10443 | 28.4163 | 31.71651 | 33.01192 | 24.77768 | 57.34869 | 54.97388 |
| C9 | AVE | 937.6997 | 920.8308 | 1088.04 | 12716.74 | 12612.68 | 34561.62 | 5948.958 | 8208.887 | 9954.461 |
| | STD | 61.96252 | 26.5344 | 363.9745 | 1621.532 | 985.5352 | 5698.828 | 3876.126 | 2373.46 | 2379.059 |
| C10 | AVE | 3694.749 | 3761.504 | 3700.033 | 7626.868 | 8230.649 | 13892.13 | 6280.105 | 7617.053 | 7348.594 |

(Continued)

**Table 6 (continued)**

| | | EPM5 | EPM4 | EPM3 | EPM2 | HHO | SOA | SMA | SSA | MRFO |
|---|---|---|---|---|---|---|---|---|---|---|
| | STD | 558.9742 | 532.997 | 582.0079 | 935.1439 | 1204.979 | 758.1973 | 626.1485 | 783.3506 | 939.2402 |
| C11 | AVE | 1177.74 | 1166.028 | 1180.263 | 1396.777 | 1383.864 | 22591.94 | 1303.236 | 1322.368 | 1261.253 |
| | STD | 15.6531 | 9.219968 | 16.98053 | 71.8303 | 87.64393 | 4489.043 | 63.97801 | 37.66977 | 30.2221 |
| C12 | AVE | 37036.41 | 893674.8 | 991485.5 | 1.37E+08 | 1.35E+08 | 5.26E+10 | 1598430 | 4238014 | 83609.84 |
| | STD | 22014.37 | 627633 | 857632.9 | 2.04E+08 | 2.99E+08 | 1.81E+10 | 1074183 | 2520333 | 41626.2 |
| C13 | AVE | 4075.313 | 16996.81 | 14110.42 | 212873.9 | 125767.5 | 1.82E+10 | 29634.34 | 130309.7 | 4124.072 |
| | STD | 3396.271 | 12763.81 | 11479.36 | 507175.8 | 94114.22 | 9.25E+09 | 10151.04 | 75453.32 | 4256.024 |
| C14 | AVE | 2426.744 | 7447.321 | 15906.74 | 1680775 | 1705491 | 31389494 | 52348.57 | 15112.99 | 3072.003 |
| | STD | 658.6872 | 6164.037 | 11337.99 | 3798183 | 3885700 | 32006311 | 25969.55 | 12145.55 | 1094.581 |
| C15 | AVE | 5616.482 | 12308.82 | 13023.14 | 4694751 | 4537062 | 6.16E+09 | 29350.23 | 39421.45 | 9413.125 |
| | STD | 4499.291 | 7888.652 | 8786.184 | 23254860 | 22398016 | 3.85E+09 | 6731.168 | 22304.52 | 6287.582 |
| C16 | AVE | 2237.506 | 2332.344 | 2553.069 | 4028.273 | 4048.173 | 8094.206 | 3164.62 | 3063.408 | 3297.917 |
| | STD | 247.3496 | 249.5483 | 240.8204 | 513.0978 | 698.4317 | 1485.932 | 368.3294 | 326.5094 | 521.3009 |
| C17 | AVE | 2257.742 | 2274.431 | 2474.799 | 3639.855 | 3672.466 | 7247.96 | 2871.545 | 2960.49 | 3205.788 |
| | STD | 155.2337 | 193.8793 | 194.0557 | 452.6595 | 473.4265 | 3760.402 | 242.0486 | 268.2297 | 326.7872 |
| C18 | AVE | 19549.7 | 80794.79 | 90703.99 | 4242293 | 2935428 | 1.02E+08 | 207761.6 | 107742.9 | 33126 |
| | STD | 12822.92 | 27734.12 | 50939.99 | 6473234 | 3355806 | 80050046 | 126316.6 | 35916.97 | 22584.63 |
| C19 | AVE | 4774.262 | 10989.99 | 9610.29 | 62330.2 | 52244.67 | 2.15E+09 | 14225.18 | 201180.9 | 14846.17 |
| | STD | 3910.518 | 12388.52 | 8085.775 | 45900.33 | 27950.68 | 1.68E+09 | 17052.33 | 54726.35 | 11388.86 |
| C20 | AVE | 2287.938 | 2319.214 | 2533.119 | 3162.177 | 3257.845 | 4062.8 | 2818.477 | 2957.316 | 3179.556 |
| | STD | 108.3397 | 139.2058 | 213.9936 | 311.8407 | 258.3304 | 404.9723 | 288.1831 | 253.0242 | 349.4163 |
| C21 | AVE | 2388.669 | 2375.095 | 2420.159 | 2785.253 | 2812.891 | 3138.545 | 2492.834 | 2505.974 | 2563.45 |
| | STD | 14.39998 | 19.76868 | 21.43885 | 72.10099 | 82.51934 | 97.2633 | 36.10054 | 45.65499 | 65.47573 |
| C22 | AVE | 2300.425 | 2300.273 | 2302.295 | 10608.09 | 10422.03 | 16728.95 | 8257.222 | 9165.022 | 8619.539 |
| | STD | 0.791301 | 0.679075 | 1.070631 | 1533.443 | 1014.829 | 853.0564 | 988.791 | 952.0107 | 1623.496 |
| C23 | AVE | 2804.425 | 2796.67 | 2831.246 | 3801.936 | 3871.871 | 4141.234 | 2917.741 | 2938.457 | 3118.995 |
| | STD | 14.84545 | 21.28376 | 27.53321 | 184.6423 | 223.2331 | 194.9199 | 44.99394 | 40.59267 | 101.1374 |
| C24 | AVE | 2978.396 | 2970.417 | 3030.666 | 4096.05 | 4039.954 | 4620.13 | 3095.88 | 3078.31 | 3285.498 |
| | STD | 14.84988 | 23.40616 | 28.51908 | 215.5657 | 174.8984 | 271.4433 | 36.24266 | 40.7089 | 104.6463 |
| C25 | AVE | 2974.773 | 2971.509 | 2975.22 | 3249.563 | 3258.095 | 14143.84 | 3017.684 | 3023.769 | 3052.767 |
| | STD | 8.554417 | 17.51097 | 12.64854 | 105.7085 | 93.53723 | 1790.841 | 38.19031 | 45.47396 | 35.97161 |
| C26 | AVE | 3945.906 | 4502.135 | 4913.446 | 12205.31 | 12226.77 | 17256.95 | 5719.404 | 4752.921 | 6541.465 |
| | STD | 887.4889 | 197.8206 | 249.1071 | 904.1735 | 1051.926 | 960.7725 | 373.1578 | 2102.883 | 3501.497 |
| C27 | AVE | 3170.97 | 3170.389 | 3170.612 | 4830.036 | 4949.775 | 6216.204 | 3323.926 | 3368.092 | 3592.878 |
| | STD | 9.292116 | 9.521278 | 8.661707 | 446.8078 | 543.9687 | 712.5707 | 65.79624 | 77.34347 | 145.1447 |
| C28 | AVE | 3258.189 | 3259.178 | 3259.365 | 3757.093 | 3737.579 | 11912.49 | 3284.561 | 3301.465 | 3290.119 |
| | STD | 1.823178 | 2.67E−01 | 0.280697 | 311.0017 | 217.6791 | 1688.529 | 22.72083 | 16.07822 | 28.54801 |
| C29 | AVE | 3477.968 | 3495.964 | 3509.711 | 6852.671 | 6928.21 | 26533.06 | 4176.37 | 4571.589 | 4437.38 |
| | STD | 159.98 | 140.3959 | 225.5474 | 1164.126 | 1186.805 | 17826.29 | 325.5151 | 294.3504 | 444.2482 |

The population size is 70, and the number of iterations is $500 \times D$. D is the dimension of the test function. According to the characteristics of the EPM framework, the more algorithms are integrated into the framework, the more times the function will be evaluated. To compare the results of the integration algorithm and the original algorithm more fairly, we design a perfect ensemble (PE) that takes the optimal value from the effects of multiple original algorithms. In each independent run, it records the best result of several original algorithms as the result of this run, and after 25 runs, all the results are averaged. This paper names the results obtained by the PE framework as "PE + number", where "number" represents the number of integrated results in the framework. It expresses the results obtained by averaging the optimal values of HHO and SOA as PE2, the results obtained by averaging the running results of HHO, SOA, and SMA as PE3, the results obtained by averaging the optimal

values of HHO, SOA, SMA, and SSA as PE4, the results obtained by averaging the optimal values of HHO, SOA, SMA, SSA, and MRFO as PE5.

Table 7 displays the results of Friedman's mean rank test for the benchmark functions. On the 23 standard benchmark functions, the mean rankings of EPM5, EPM3, and EPM2 are higher than those of PE5, PE3, and PE2, respectively; the mean ranking of EPM4 is lower than that of PE4. On the CEC2017 benchmark functions with 30 dimensions, the mean rankings of EPM5, EPM4, EPM3, and EPM2 are higher than that of PE5, PE4, PE3, and PE2, respectively. On the CEC2017 benchmark functions with 50 dimensions, the mean rankings of EPM5, EPM4, EPM3, and EPM2 are higher than that of PE5, PE4, PE3, and PE2, respectively. In most cases, the algorithms in the ensemble framework get a better ranking. Increasing the number of integrated metaheuristic algorithms enhances the average optimization ability of the overall ensemble framework.

**Table 7:** Friedman's mean ranking test results of benchmark functions

| Benchmark functions | Comparison | EPM | PE |
|---|---|---|---|
| 23 standard benchmark functions | EPM2 *vs.* PE2 | 1.3478 | 1.6522 |
| | EPM3 *vs.* PE3 | 1.5435 | 1.4565 |
| | EPM4 *vs.* PE4 | 1.4348 | 1.5652 |
| | EPM5 *vs.* PE5 | 1.4348 | 1.5652 |
| CEC2017 benchmark functions (D = 30) | EPM2 *vs.* PE2 | 1.3103 | 1.6897 |
| | EPM3 *vs.* PE3 | 1.1034 | 1.8966 |
| | EPM4 *vs.* PE4 | 1.2069 | 1.7931 |
| | EPM5 *vs.* PE5 | 1.0862 | 1.9138 |
| CEC2017 benchmark functions (D = 50) | EPM2 *vs.* PE2 | 1.4138 | 1.5862 |
| | EPM3 *vs.* PE3 | 1.1034 | 1.8966 |
| | EPM4 *vs.* PE4 | 1.1379 | 1.8621 |
| | EPM5 *vs.* PE5 | 1.1207 | 1.8793 |

Table 8 displays the *p*-value test results on benchmark functions. On the 23 standard benchmark functions, the *p*-value resulting from the comparison between EPM2 and PE2 is less than 0.1 and more significant than 0.05, indicating that the results of the algorithms are different at the significance level of 0.1. The *p*-values resulting from the comparisons between EPM3 and PE3, between EPM4 and PE4, and between EPM5 and PE5 are greater than 0.1, indicating no significant difference between the results. On the CEC2017 benchmark functions with 30 dimensions, the *p*-value resulting from the comparison between EPM2 and PE2 is greater than 0.1, indicating no significant difference between the results. The *p*-values for the comparison between EPM3 and PE3, between EPM4 and PE4, and between EPM5 and PE5 are less than 0.01, indicating that the results of these algorithms are significantly different at the significance level of 0.01. On the CEC2017 benchmark functions with 50 dimensions, the *p*-value resulting from the comparison between EPM2 and PE2 is greater than 0.1, indicating no significant difference between the results. The *p*-values for the comparison between EPM3 and PE3, between EPM4 and PE4, and between EPM5 and PE5 are less than 0.01, indicating that the results of these algorithms are significantly different at the significance level of 0.01.

**Table 8:** Wilcoxon signed ranking test results of benchmark functions

| Benchmark functions | Comparison | R+ | R- | *p*-value | $\alpha$ |
|---|---|---|---|---|---|
| 23 standard benchmark functions | EPM2 *vs.* PE2 | 195 | 81 | 9.95E−02 | 0.1 |
| | EPM3 *vs.* PE3 | 131 | 145 | 1.00E+00 | >0.1 |
| | EPM4 *vs.* PE4 | 175 | 101 | 1.56E−01 | >0.1 |
| | EPM5 *vs.* PE5 | 171.5 | 104.5 | 1.39E−01 | >0.1 |
| CEC2017 benchmark functions (D = 30) | EPM2 *vs.* PE2 | 265 | 170 | 3.04E−01 | >0.1 |
| | EPM3 *vs.* PE3 | 427 | 8 | 5.90E−06 | 0.01 |
| | EPM4 *vs.* PE4 | 378 | 57 | 5.19E−04 | 0.01 |
| | EPM5 *vs.* PE5 | 414.5 | 20.5 | 2.52E−05 | 0.01 |
| CEC2017 benchmark functions (D = 50) | EPM2 *vs.* PE2 | 222 | 213 | 9.22E−01 | >0.1 |
| | EPM3 *vs.* PE3 | 419 | 16 | 1.32E−05 | 0.01 |
| | EPM4 *vs.* PE4 | 413 | 22 | 2.36E−05 | 0.01 |
| | EPM5 *vs.* PE5 | 398.5 | 36.5 | 1.08E−04 | 0.01 |

### *4.4 Iterative Curves*

We select some test functions, draw the iterative curve of the experimental algorithm, and visually observe the convergence of the algorithms. The iterative curves in Fig. 5 correspond to functions F3, F15, F20, C5, C16, and C20, representing the performance of EPM5 under different types of functions. It is observed that the EPM5 algorithm has a faster iterative optimization speed and achieves better results through the iterative curves.

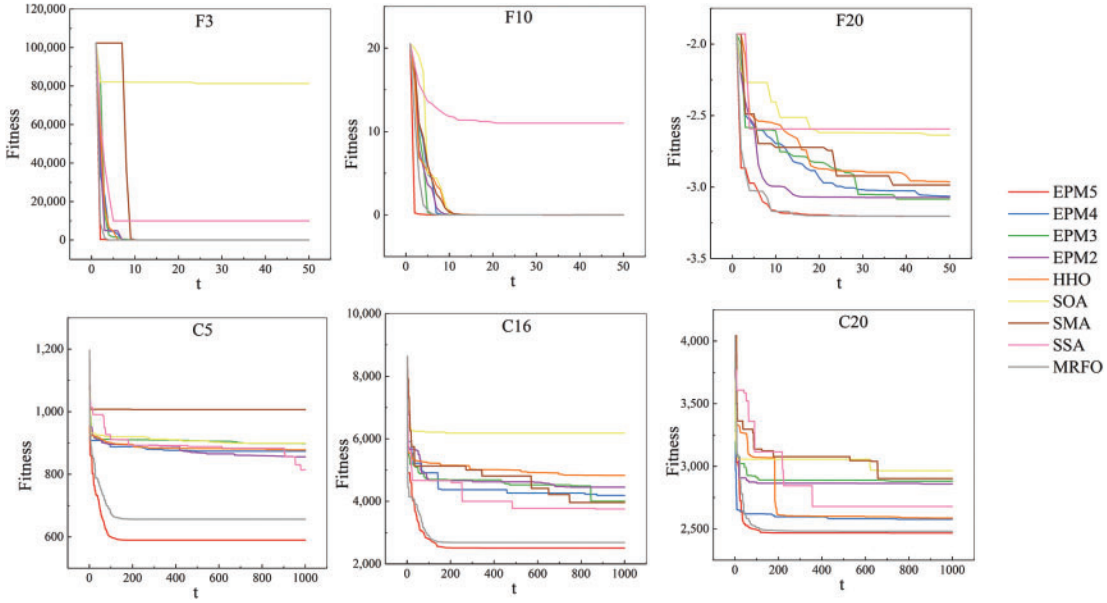


**Figure 5:** Iterative curves of partial test functions

### *4.5 Evolution Process of Real Population and Virtual Population*

We analyzed the process of solving function F3 by the EPM5 algorithm to observe the relationship between the real and virtual populations and follow the evolution process more intuitively during the operation of the ensemble algorithm. Fig. 6 shows the position changes of the virtual populations in the $x_1$ and $x_2$ dimensions, before and after 1, 2, 5, 6, 20, 21, 100, 101, 1000, 1001, 14999, and 15000 iterations, where 'a' represents the population before the iteration and 'b' represents the population after the iteration. It lists the historical optimal solutions corresponding to the number of partial iterations in Table 9.

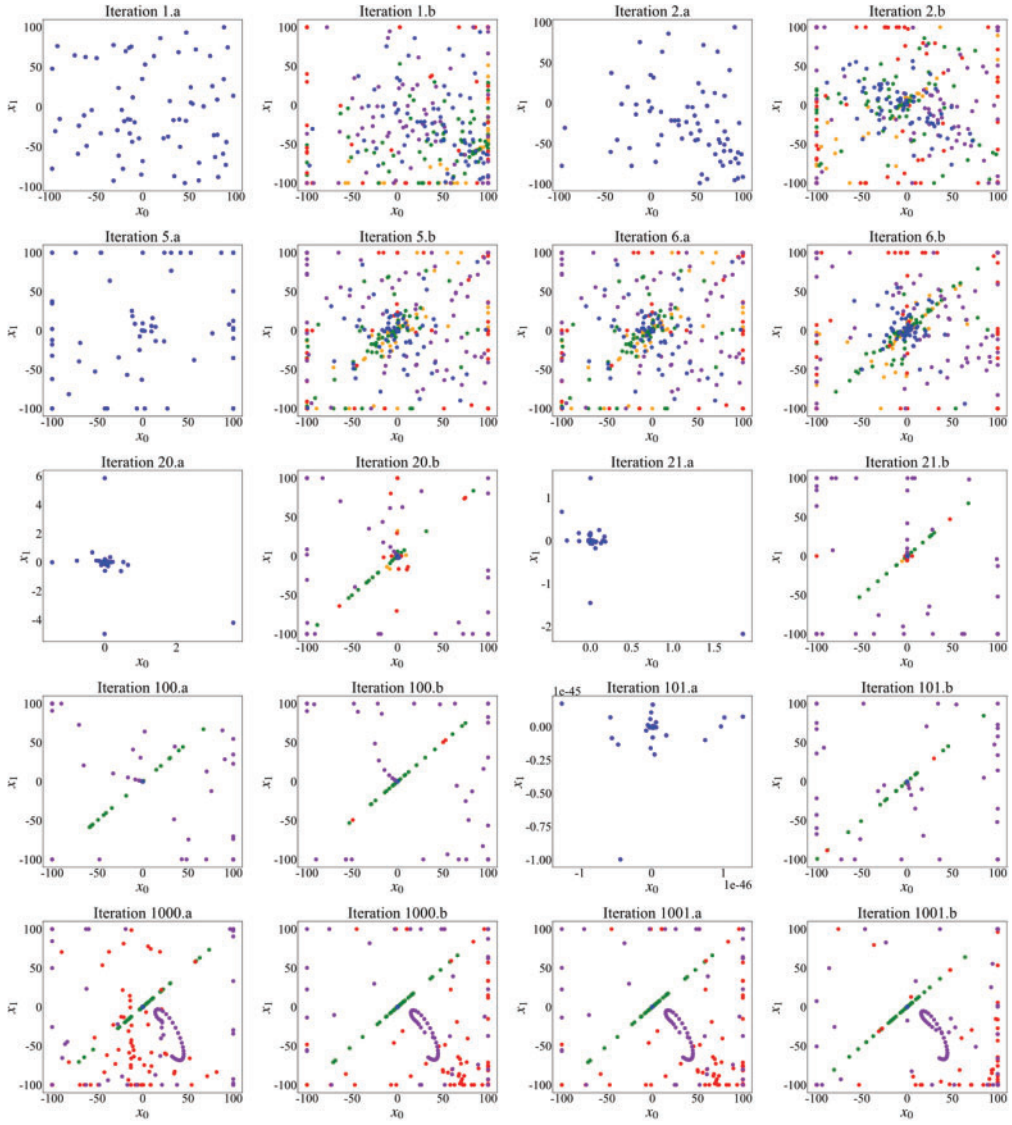


**Figure 6:** (Continued)

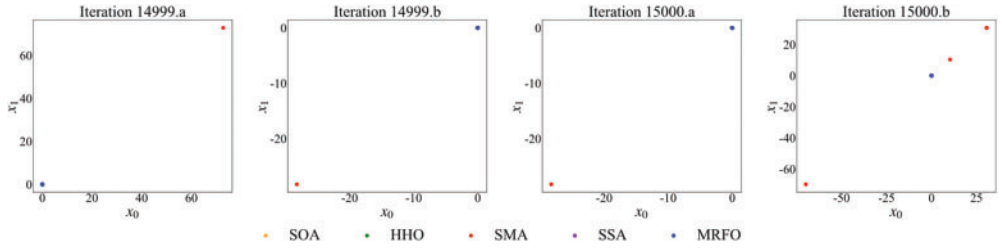


**Figure 6:** The virtual population distribution diagram of the EPM5 algorithm under some iterations in the process of solving the function F3

**Table 9:** Number of partial iterations on F3 function and historical optimal solution

| t | H | t | H | t | H | t | H | t | H | t | H |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 102389.5 | 4 | 0.178303 | 19 | 5.78E−16 | 99 | 2.72E−97 | 999 | 0 | 14998 | 0 |
| 1 | 374.6011 | 5 | 0.103464 | 20 | 1.78E−17 | 100 | 2.09E−99 | 1000 | 0 | 14999 | 0 |
| 2 | 314.3702 | 6 | 0.103464 | 21 | 1.48E−17 | 101 | 1.32E−99 | 1001 | 0 | 15000 | 0 |

In Iteration 1.a, it is observed that the positions of the five virtual populations come from the same real population before the start of the first iteration. After the high-level generation sub-stage, the five virtual populations update their locations independently, and the positions of the populations are different, as shown in Iteration 1.b. Because the fitness value of the best elite is smaller than the historical optimum, it materializes the population where the optimal elite resides, and the population distribution after materialization is shown by Iteration 2.a. It transforms the five virtual populations into the real population. In Iteration 5.b and Iteration 6.a, the best elite of the virtual populations after the fifth iteration is not less than an order of magnitude smaller than the historically best fitness value. Thus, the virtual population continues to exist. In Iteration 20.a, the entity population has converged at this time. However, after the high-level generation sub-stage, the virtual population can still be distributed in every corner of the map, as shown in Iteration 20.b. After the high-level selection sub-stage, it obtains the population distributed in iteration 21.a, and the virtual populations are transformed into the real population again. It can be seen from this figure that the premature convergence of single or multiple populations will not lead to the convergence of all populations, and other populations still have good exploratory properties. In Iteration 100.b and Iteration 101.b, it is observed that the red population has converged. But in Iteration 1000.a, the red population is dispersed again, indicating that the converged population has a chance to reexplore. In Iteration 14999.a and Iteration 15000.b, all populations have converged. The red population has several individual sporadic distributions related to the corresponding algorithm characteristics of the population.

### *4.6 Engineering Application Problems*

The pressure vessel design optimization problem optimizes the cost of welding, material, and forming of the pressure vessel by adjusting the shell thickness, head thickness, inner diameter, and length of the vessel under four constraints [59]. The three-bar truss design optimization problem is based on the stress constraint of each bar to optimize the total weight of the bar structures [60]. Table 10 gives the mathematical description of these problems. It abbreviates the first engineering problem as E1 and the second engineering problem as E2.

**Table 10:** Engineering optimization problems

| No. | Problem name | D | Initialization range | Constraints | Known optimum |
|---|---|---|---|---|---|
| E1 | Pressure Vessel Design<br>$f(x) = 0.6224z_1x_3x_4 + 1.7781z_2x_3{}^2 + 3.1661z_1{}^2x_4 + 19.84z_1{}^2x_3$ | 4 | $1 \le x_1 \le 99$(int),<br>$1 \le x_2 \le 99$(int),<br>$10 \le x_3 \le 200$,<br>$10 \le x_4 \le 200$ | $0.00954x_3 \le z_2$,<br>$0.0193x_3 \le z_1$,<br>$x_4 \le 240$,<br>$-\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 \le -1296000$,<br>$z_1 = 0.0625x_1$<br>$z_2 = 0.0625x_2$ | 5885.3327736 |
| E2 | Three-bar Truss Design Problem<br>$f(x) = l\left(2\sqrt{2}x_1 + x_2\right)$ | 2 | $0 \le x_1 \le 1$,<br>$0 \le x_2 \le 1$ | $\frac{x_2}{2x_2x_1 + \sqrt{2}x_1^2}P - \sigma$<br>$\frac{x_2 + \sqrt{2}x_1}{2x_2x_1 + \sqrt{2}x_1^2}P - \sigma$<br>$\frac{1}{x_1 + \sqrt{2}x_2}P - \sigma$<br>$l = 100, P = 2, \sigma = 2$ | 263.89584338 |

The average and standard deviation of the best solutions obtained by all test algorithms in 25 independent runs and 500 iterations in each dimension were recorded and compared (see Table 11). For the optimization results of the pressure vessel design problem, EPM3, EPM4, and EPM5 are better than their integrated original algorithms. EPM2 is better than SOA but worse than HHO. For the optimization results of the three-bar truss design problem, both EPM4 and EPM5 are better than their integrated original algorithm. EPM3 is better than HHO and SOA but worse than SMA. EPM2 is better than SOA but worse than HHO. Two practical engineering problems obtain better optimization results from the average optimization results by adopting the ensemble framework.

**Table 11:** The results of engineering optimization problems by different optimization algorithms

| | | EPM5 | EPM4 | EPM3 | EPM2 | HHO | SOA | SMA | SSA | MRFO |
|---|---|---|---|---|---|---|---|---|---|---|
| E1 | AVE | 6066.6913 | 6085.2628 | 6098.4798 | 8074.5080 | 7964.7286 | 13856.8187 | 6216.5912 | 6353.9354 | 6271.0216 |
| | STD | 12.5129 | 20.6386 | 33.2818 | 1752.8158 | 1414.2115 | 6088.7058 | 219.9751 | 246.7167 | 228.2141 |
| E2 | AVE | 263.895861 | 263.896028 | 263.896757 | 263.917593 | 263.914440 | 266.142034 | 263.896307 | 263.896183 | 263.895862 |
| | STD | 1.7451E−05 | 2.6238E−04 | 1.1884E−03 | 3.8590E−02 | 2.5215E−02 | 2.0515E+00 | 8.2330E−04 | 5.1781E−04 | 1.8544E−05 |

## 5 Conclusion

We propose a new population-based metaheuristic ensemble framework in this paper. The main innovation is to use the common structural characteristics of the population-based metaheuristic algorithms to design a framework, which is a bridge for cooperative optimization of multiple algorithms through the transformation of virtual population and entity population and elite strategy. The framework leverages the differentiation and unification of real and virtual populations to coordinate exploration and exploitation at the population level. Meanwhile, an elitist strategy is adopted to communicate among virtual populations to guarantee diversity and reduce the internal influence on each metaheuristic algorithm. In the experiment, five algorithms, SOA, HHO, SMA, SSA, and MRFO, are integrated using the ensemble framework to obtain EPM algorithms. The EPM algorithms perform superior on 23 standard benchmark functions and the CEC2017 benchmark functions. According to Friedman's average ranking, the results of algorithms integrated with the

EPM framework, in most cases, outperformed the results of the same algorithms integrated with the PE framework, demonstrating the superiority of the ensemble framework. The ensemble framework also demonstrated excellent results in solving two practical engineering problems and obtained better solutions.

With the increase in the number of optimization algorithms integrated into the ensemble framework, the information exchange between various algorithms will increase, and these algorithms will be more likely to jump out of the local optimization. However, we cannot distinguish the role of each algorithm in optimization and the influence of algorithm parameters on the optimization results. Meanwhile, each additional algorithm in the ensemble framework will increase the calculation of the virtual population corresponding to the algorithm, the storage cost of the calculation results, and the cost of information exchange, reducing the optimization speed. In future research, we will study two aspects: improving the optimization ability and speed. We will analyze the contribution of a single algorithm in the integration framework and adjust the algorithm parameters to improve the optimization ability. We will study the factors that affect the optimization speed and reduce unnecessary computing and storage costs to improve the optimization speed according to the role of different algorithms in the different stages.

**Acknowledgement:** The authors would like to acknowledge the editor-in-chief, associate editors, and reviewers for their contributions to the improvement of this paper.

**Funding Statement:** This work was supported by National Natural Science Foundation of China under Grant 62073330. The auther J. T. received the grant.

**Author Contributions:** Study conception and design: H. Li, J. Tang, S. Lao; data collection: Q. Pan, J. Zhan; analysis and interpretation of results: H. Li, Q. Pan; draft manuscript preparation: H. Li, J. Zhan. All authors reviewed the results and approved the final version of the manuscript.

**Availability of Data and Materials:** Data available on request from the authors. The data that support the findings of this study are available from the corresponding author, (J. T.), upon reasonable request.

**Conflicts of Interest:** The authors declare that they have no conflicts of interest to report regarding the present study.

## References

[1] E. G. Talbi, "Optimization methods," in *Metaheuristics: From Design to Implementation*, vol. 1. Hoboken, NJ, USA: John Wiley & Sons, no. 3, pp. 18–33, 2009.

[2] T. G. Crainic and M. Toulouse, "Meta-heuristics and parallelism," in *Handbook of Metaheuristics*, vol. 17. New York, NY, USA: Springer, no. 2, pp. 498–505, 2010.

[3] I. Boussaïd, J. Lepagnot and P. Siarry, "A survey on optimization metaheuristics," *Information Sciences*, vol. 237, pp. 82–117, 2013.

[4] D. P. Bertsekas, "Lagrangian methods–local convergence," in *Constrained Optimization and Lagrange Multiplier Methods*, vol. 4. New York, NY, USA: Academic Press, no. 5, pp. 231–256, 1982.

[5] K. Deb, "Two approaches to multi-objective optimization," in *Search Methodologies*, vol. 15. New York, NY, USA: Springer, no. 2, pp. 407–410, 2014.

[6] J. H. Vandermeer, "Niche theory," *Annual Review of Ecology and Systematics*, vol. 3, no. 1, pp. 107–132, 1972.

[7] J. Tang, G. Liu and Q. Pan, "A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 10, pp. 1627–1643, 2021.
[8] E. Pacini, C. Mateos and C. G. Garino, "Distributed job scheduling based on swarm intelligence: A survey," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 252–269, 2014.
[9] B. H. Nguyen, B. Xue and M. Zhang, "A survey on swarm intelligence approaches to feature selection in data mining," *Swarm and Evolutionary Computation*, vol. 54, pp. 100663, 2020.
[10] J. Tang, H. Duan and S. Lao, "Swarm intelligence algorithms for multiple unmanned aerial vehicles collaboration: A comprehensive review," *Artificial Intelligence Review*, vol. 56, no. 5, pp. 4295–4327, 2022.
[11] K. V. Price, "Differential evolution," in *Handbook of Optimization*, vol. 2. New York, NY, USA: Springer, no. 1, pp. 187–214, 2013.
[12] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, no. 2, pp. 65–85, 1994.
[13] J. R. Koza, "Overview of genetic programming," in *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, vol. 5. Cambridge, MA, USA: MIT Press, no. 1, pp. 73–78, 1992.
[14] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *1997 IEEE Int. Conf. on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, Orlando, FL, USA, vol. 5, pp. 4104–4108, 1997.
[15] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.
[16] X. S. Yang and S. Deb, "Cuckoo search via lévy flights," in *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, Coimbatore, India, pp. 210–214, 2009.
[17] A. Seyyedabbasi and F. Kiani, "Sand cat swarm optimization: A nature-inspired algorithm to solve global optimization problems," *Engineering with Computers*, pp. 1–25, 2022.
[18] F. Glover, "Tabu search—Part I," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190–206, 1989.
[19] Z. Feng, W. Niu and S. Liu, "Cooperation search algorithm: A novel metaheuristic evolutionary intelligence algorithm for numerical optimization and engineering optimization problems," *Applied Soft Computing*, vol. 98, pp. 106734, 2021.
[20] Y. Shi, "Brain storm optimization algorithm," in *Int. Conf. in Swarm Intelligence*, Chongqing, China, pp. 303–309, 2011.
[21] Q. Askari, M. Saeed and I. Younas, "Heap-based optimizer inspired by corporate rank hierarchy for global optimization," *Expert Systems with Applications*, vol. 161, pp. 113702, 2020.
[22] T. S. Ayyarao, N. S. Ramakrishna, R. M. Elavarasan, N. Polumahanthi, M. Rambabu *et al.,* "War strategy optimization algorithm: A new effective metaheuristic algorithm for global optimization," *IEEE Access*, vol. 10, pp. 25073–25105, 2022.
[23] E. Rashedi, H. Nezamabadi-Pour and S. Saryazdi, "GSA: A gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.
[24] O. K. Erol and I. Eksin, "A new optimization method: Big bang–big crunch," *Advances in Engineering Software*, vol. 37, no. 2, pp. 106–111, 2006.
[25] Ş. İ. Birbil and S. C. Fang, "An electromagnetism-like mechanism for global optimization," *Journal of Global Optimization*, vol. 25, no. 3, pp. 263–282, 2003.
[26] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.
[27] J. A. Koupaei, S. M. M. Hosseini and F. M. Ghaini, "A new optimization algorithm based on chaotic maps and golden section search method," *Engineering Applications of Artificial Intelligence*, vol. 50, pp. 201–214, 2016.
[28] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 376, pp. 113609, 2021.
[29] N. Chopra and M. M. Ansari, "Golden jackal optimization: A novel nature-inspired optimizer for engineering applications," *Expert Systems with Applications*, vol. 198, pp. 116924, 2022.

[30] Y. Zhang, Z. Jin and S. Mirjalili, "Generalized normal distribution optimization and its applications in parameter extraction of photovoltaic models," *Energy Conversion and Management*, vol. 224, pp. 113301, 2020.
[31] I. A. Ibrahim, M. Hossain and B. C. Duck, "A hybrid wind driven-based fruit fly optimization algorithm for identifying the parameters of a double-diode photovoltaic cell model considering degradation effects," *Sustainable Energy Technologies and Assessments*, vol. 50, pp. 101685, 2022.
[32] J. Tang, X. Chen, X. Zhu and F. Zhu, "Dynamic reallocation model of multiple unmanned aerial vehicle tasks in emergent adjustment scenarios," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 2, pp. 1139–1155, 2023.
[33] Z. Tian, "Backtracking search optimization algorithm-based least square support vector machine and its applications," *Engineering Applications of Artificial Intelligence*, vol. 94, pp. 103801, 2020.
[34] A. R. Jordehi, "Binary particle swarm optimisation with quadratic transfer function: A new binary optimization algorithm for optimal scheduling of appliances in smart homes," *Applied Soft Computing*, vol. 78, pp. 465–480, 2019.
[35] M. H. Nadimi-Shahraki, H. Zamani and S. Mirjalili, "Enhanced whale optimization algorithm for medical feature selection: A COVID-19 case study," *Computers in Biology and Medicine*, vol. 148, pp. 105858, 2022.
[36] S. Salcedo-Sanz, "Modern meta-heuristics based on nonlinear physics processes: A review of models and design procedures," *Physics Reports*, vol. 655, pp. 1–70, 2016.
[37] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.
[38] A. K. Qin, V. L. Huang and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2008.
[39] Q. Pan, J. Tang, H. Wang, H. Li, X. Chen *et al.,* "SFSADE: An improved self-adaptive differential evolution algorithm with a shuffled frog-leaping strategy," *Artificial Intelligence Review*, vol. 55, no. 5, pp. 3937–3978, 2021.
[40] W. Gong, A. Zhou and Z. Cai, "A Multi-operator search strategy based on cheap surrogate models for evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 746–758, 2015.
[41] M. Z. Ali, N. H. Awad and P. N. Suganthan, "Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization," *Applied Soft Computing*, vol. 33, pp. 304–327, 2015.
[42] H. Rakhshani and A. Rahati, "Intelligent multiple search strategy cuckoo algorithm for numerical and engineering optimization problems," *Arabian Journal for Science and Engineering*, vol. 42, no. 2, pp. 567–593, 2017.
[43] C. Li, S. Yang and T. T. Nguyen, "A self-learning particle swarm optimizer for global optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 3, pp. 627–646, 2011.
[44] N. Lynn and P. N. Suganthan, "Ensemble particle swarm optimizer," *Applied Soft Computing*, vol. 55, pp. 533–548, 2017.
[45] S. X. Zhang, S. Y. Zheng and L. M. Zheng, "An efficient multiple variants coordination framework for differential evolution," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2780–2793, 2017.
[46] S. Thangavelu and C. S. Velayutham, "An investigation on mixing heterogeneous differential evolution variants in a distributed framework," *International Journal of Bio-Inspired Computation*, vol. 7, no. 5, pp. 307–320, 2015.
[47] G. Wu, X. Shen, H. Li, H. Chen, A. Lin *et al.,* "Ensemble of differential evolution variants," *Information Sciences*, vol. 423, pp. 172–186, 2018.
[48] S. M. Elsayed, R. A. Sarker and E. Mezura-Montes, "Self-adaptive mix of particle swarm methodologies for constrained optimization," *Information Sciences*, vol. 277, pp. 216–233, 2014.

[49] J. A. Vrugt, B. A. Robinson and J. M. Hyman, "Self-adaptive multimethod search for global optimization in real-parameter spaces," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 243–259, 2008.
[50] Y. Xue, S. Zhong, Y. Zhuang and B. Xu, "An ensemble algorithm with self-adaptive learning techniques for high-dimensional numerical optimization," *Applied Mathematics and Computation*, vol. 231, pp. 329–346, 2014.
[51] S. M. Elsayed, R. A. Sarker and D. L. Essam, "Adaptive configuration of evolutionary algorithms for constrained optimization," *Applied Mathematics and Computation*, vol. 222, pp. 680–711, 2013.
[52] X. Yao, Y. Liu and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.
[53] G. Wu, R. Mallipeddi and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization," *Technical Report*, 2017. [Online]. Available: https://www.researchgate.net/publication/317228117_Problem_Definitions_and_Evaluation_Criteria_for_the_CEC_2017_Competition_and_Special_Session_on_Constrained_Single_Objective_Real-Parameter_Optimization
[54] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja *et al.,* "Harris hawks optimization: Algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.
[55] G. Dhiman and V. Kumar, "Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems," *Knowledge-Based Systems*, vol. 165, pp. 169–196, 2019.
[56] S. Li, H. Chen, M. Wang, A. A. Heidari and S. Mirjalili, "Slime mold algorithm: A new method for stochastic optimization," *Future Generation Computer Systems*, vol. 111, pp. 300–323, 2020.
[57] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris *et al.,* "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017.
[58] W. Zhao, Z. Zhang and L. Wang, "Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications," *Engineering Applications of Artificial Intelligence*, vol. 87, pp. 103300, 2020.
[59] E. Sandgren, "Nonlinear integer and discrete programming in mechanical design," in *Int. Design Engineering Technical Conf. and Computers and Information in Engineering Conf.*, Kissimmee, FL, USA, vol. 26584, pp. 95–105, 1988.
[60] H. Nowacki, "Optimization in pre-contract ship design," in *Int. Conf. on Computer Applications in the Automation of Shipyard Operation and Ship Design*, Tokyo, Japan, pp. 1–12, 1973.