IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 13, 2019, accepted September 29, 2019, date of publication October 2, 2019,
date of current version October 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2944981

Improvement and Application of Adaptive Hybrid Cuckoo Search Algorithm

ZHIWEN CHENG, JIQUAN WANG, MINGXIN ZHANG, HAOHAO SONG, TIEZHU CHANG, YUSHENG BI, AND KEXIN SUN
College of Engineering, Northeast Agricultural University, Harbin 150030, China

Corresponding author: Jiquan Wang (wang-jiquan@163.com)

This work was supported in part by the project of National Key Research and Development Plan under Grant 2018YFD0300105, and in part by the Natural Science Foundation under Grant 71671034.

ABSTRACT Aiming at the problem of ease of falling into local optimum and low solution quality when solving optimization problems, this paper proposes an adaptive hybrid cuckoo search (AHCS) algorithm. AHCS improves the Lévy flight method and population evolution strategy of the cuckoo search (CS) algorithm, and introduces a mutation operation operator. Inspired by the idea of position update of particle swarm optimization (PSO) algorithm, this paper introduces the inertia weight w in the Lévy flight method of CS algorithm, and gives the new dynamic adjustment methods of parameters α and β respectively. In order to enhance the local search ability and optimization speed of the algorithm, this paper introduces the mutation operation operator, and presents a new evolution strategy of the hybrid cuckoo search algorithm. In addition, in order to verify the performance of AHCS, 30 benchmark functions and CEC 2017 optimization problems were selected. The calculation results of the 30 benchmark functions and CEC 2017 optimization problems show that compared with other algorithms, the number of winning cases of t-test values and the Friedman average ranking for AHCS are significantly better than other algorithms. Finally, AHCS and various intelligent optimization methods in the literature are used to optimize the structural parameters of the reducer and the cantilever beam. The optimization results show that the quality of AHCS solution is significantly better than other algorithms.

INDEX TERMS Hybrid cuckoo search algorithm, adaptive parameter adjustment, mutation operator, evolutionary strategy.

I. INTRODUCTION

In 2009, Yang and Deb [1] proposed a random optimization algorithm for swarm search, namely cuckoo search (CS) algorithm, based on the interesting breeding behavior such as brood parasitism of certain species of cuckoos. Compared with other intelligent optimization algorithms, CS has fewer parameter settings, simple operation, and clear process. Therefore, it is easier to implement. At present, CS has been widely used in travel sales problem [2]–[4], job scheduling [5]–[8], location problem [9], [10], fault diagnosis and prediction [11], [12] and image processing and classification [13]–[16].

In recent years, CS has gradually attracted people's attention with its unique and excellent performance. Many scholars have conducted in-depth research on CS and obtained some research results. According to the mechanism and technology used, the progress and attempts made by CS can be divided into the following categories: (1) research on the improvement of position update mechanism of CS [12], [17]–[23]; (2) improvement of the parameter adjustment method of CS [17], [18], [20]–[28]; (3) application research of CS [3], [8], [25], [29]–[40]. The improvement and research of these algorithms not only improve the optimization performance of CS, but also promote the development of CS theory and application.

The individual position update method of the CS is a major factor affecting the performance of the CS. Therefore, many scholars have focused their research on the position update of CSs. In 2012, Rani *et al.* [17] introduced a linearly decreasing inertia weight with increasing number of iterations in the Lévy flight of CS. In 2013, Kaveh and Bakhshpoori [18] proposed a Lévy flight

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Shen.

method with contemporary optimal individual guidance in CS. In 2018, Dhabal and Venkateswaran [19] proposed an optimal individual guidance position update method in the CS, which solved the problem of slow convergence of the algorithm by replacing the contemporary individuals with the random generation method. In 2019, Chen and Wang [20] improved the Lévy flight method in the CS, avoiding the situation where the individual in the population is the optimal individual and the position update fails. In 2019, Zhang *et al.* [12] improved the Lévy flight method by fully combining the information of individual fitness values in the CS. In 2019, Gmili *et al.* [21] proposed a hybrid particle swarm optimization-cuckoo search (PSO-CS) algorithm, and applied the improved hybrid algorithm to quadrotor control and trajectory tracking. In 2019, Thirugnanasambandam *et al.* [22] proposed an improved CS. In the improved algorithm, the original discarding strategy is modified, and a parameter memory matrix is introduced to guide the individuals of the population. In 2019, Zhang *et al.* [23] proposed an improved CS. The improved algorithm proposes a new boundary processing method. At the same time, a new Lévy flight method and a random search method are also given.

In the research of the parameter adjustment methods of CS, in 2015, Maribel Guerrero et al. described the enhancement of the CS Algorithm via Lévy flights using a fuzzy system to dynamically adapt its parameters. In 2017, Jaballah and Meddeb [25] proposed a CS with adaptive parameter adjustment. The algorithm adopts an adjustment method that linearly decreases with the increase of the number of iterations for the parameter p_a, and adopts an adjustment method that nonlinearly increase with the increase of the number of iterations for the parameter α. In 2017, Chi *et al.* [26] combined CS with PSO to propose a hybrid CS-PSO. At the same time, the hybrid algorithm uses an adjustment method for the parameter α in the Lévy flight with a linear decrease as the number of iterations increases. In 2018, Ma *et al.* [27] introduced a parameter adaptive parameter adjustment method in CS. In order to avoid falling into local optimum, this method draws on the idea of grey wolf optimization (GWO) and gives certain weights to the best individuals in the three optimal subpopulations to generate new individuals. At the same time, each of individual in the population is compared with the newly generated individual to complete the adaptive adjustment of the parameter α. In 2019, Zhang *et al.* [12] proposed a CS algorithm with dynamic adjustment parameters. The algorithm adopts different adjustment methods for the parameter α and the parameter p_a, which are nonlinearly decreasing as the number of iterations increases. In 2019, Chen and Wang [20] proposed a hybrid CS that uses a method of nonlinear decrement as the number of iterations increases to adjust the parameter α. In 2019, Ong and Zainuddin [28] proposed a CS with adaptive parameter adjustment. In the adjustment of the parameter α, the algorithm fully considers the information of the average fitness value of the contemporary population. In 2019, Thirugnanasambandam *et al.* [22] proposed a CS with adaptive parameter adjustment and applied it to multi-peak numerical optimization problems. In the improved algorithm, the information of the optimal individual fitness value in the contemporary population is considered, and the parameter α is adaptively adjusted.

In the application research of CS, in 2013, Ouaarab *et al.* [3] proposed an improved discrete CS and applied it to the traveling salesman problem. In 2016, Gonzalez *et al.* [16] presented the optimization of a fuzzy edge detector based on the traditional Sobel technique combined with interval type-2 fuzzy logic, and applied the CS and genetic algorithm to optimize the fuzzy inference systems. In 2017, Mohammadrezapour *et al.* [29] used the CS for the optimization of water allocation and crop planning under different weather conditions. In 2017, Zhang *et al.* [30] proposed an improved CS and applied it to solve linear equation problems. In 2017, Jaballah and Meddeb [25] proposed an improved CS with a new adaptive parameter adjustment and applied it to solve complex RFID network planning problems. In 2018, Raha *et al.* [31] applied the CS to the unpowered scheduling based on superconducting magnetic energy storage systems. In 2018, Zhu *et al.* [8] proposed a hybrid cuckoo-differential evolution (CS-DE) algorithm, and applied the hybrid algorithm to the no-wait flow shop scheduling. In 2018, Agasthian *et al.* [32] optimized the parameters of the support vector machine by the CS, and applied the support vector machine optimized by the CS to the fault classification and detection of the wind turbine. In 2018, Hosseinalizadeh *et al.* [33] proposed a hybrid CS and applied it to the improvement of steam turbine speed regulation and excitation system identification procedures. In 2018, Biswal *et al.* [34] proposed an adaptive CS algorithm and applied the improved algorithm to time-frequency analysis and classification of power signals. In 2018, Cheng *et al.* [35] proposed a CS with memory and used it in the fault diagnosis of hydroelectric generating sets. In 2018, Chen *et al.* [36] proposed an improved CS and applied it to solve the inverse geometric heat conduction problem. In 2018, Prasath and Kumanan [37] proposed a distance-oriented CS, and applied the improved algorithm to the problem of water quality analysis based on water quality images. In 2019, Wu *et al.* [38] proposed a hybrid model based on an improved multi-objective CS for short-term load forecasting of power systems. In 2019, Chen and Zhou [39] proposed a hybrid CS using quasi-Newton method for boundary condition identification of non-Fourier heat conduction problems. In 2019, Meng *et al.* [40] proposed an improved CS for multi-target hydropower station operation.

In summary, many scholars have contributed on the improvement of CS. Although scholars have made some achievements in the improvement of the CS, there are still some problems. The improved CS proposed in the literature [12], [20], [23] and [25]–[28] only improves the adjustment method of the parameter α or β, and improves the convergence speed of the algorithm. However, the algorithm still has the problem of premature convergence and the inability to find the global optimal solution. A good CS should have the following properties: (1) the position update formula should make full use of the information of the fitness value of the contemporary population, the average fitness value of the population, and the number of iterations of the algorithm; (2) the loop statement should be reduced in programming to increase the speed of computation; (3) it can converge faster, while avoiding premature convergence; (4) the maximum number of iterations should be avoided as an iterative termination condition to ensure fairness due to differences in the complexity of different algorithms. In response to the above problems, this paper proposes AHCS.

The main contributions of this study are as follows:

- We propose an adaptive hybrid cuckoo search algorithm. There are three main aspects for the improvement of AHCS:
  1) The Lévy flight method of the AHCS is improved. In the Lévy flight method of CS algorithm, this paper introduces the dynamic inertia weight w, and gives the new adaptive adjustment methods of parameters α and β respectively.
  2) The mutation operator is introduced in the AHCS.
  3) An improved population evolution strategy of the AHCS is proposed.
- The proposed AHCS and other improved algorithms in literature have been used to optimize 30 benchmark functions and CEC 2017 optimization problems, and the AHCS is superior to other algorithms in the number of winning cases of t-test values and the Friedman average ranking.
- We compare the proposed AHCS with other improved algorithms in literature in the parameter optimization of the reducer and the cantilever beam, and the proposed AHCS outperforms them in quality of the solutions.

The paper is organized as follows. Section II introduces the processing method of the constraint optimization problems-penalty function method. Section III is a brief introduction of the basic CS algorithm. The overall structure and improvement of the AHCS are presented in section IV. The experimental results of AHCS and other improved algorithms are compared in section V and section VI. Finally, we conclude and make a summary in section VII.

II. PENALTY FUNCTION METHOD FOR CONSTRAINED OPTIMIZATION PROBLEMS

The mathematical model of a constrained optimization problem can be generally expressed as follows:

$$\begin{cases} \min f(X), \ X = [X_1, X_2, \cdots, X_k, \cdots, X_n] \in R \\ s.t. \begin{cases} h_i(X) = 0, & i = 1, 2, \cdots, p \\ g_j(X) \geq 0, & j = 1, 2, \cdots, q \end{cases} \end{cases} \tag{1}$$

where $s.t.$ is short for "subject to", n is the population size, $h_i(X) = 0$ is the i-th equation constraint, p is the number of equation constraints,$g_j(X) \geq 0$ is the j-th inequality constraint, q is the number of inequality constraints, and X_k is a m-dimensional vector $X_k = (x_{k1}, x_{k2}, \ldots, x_{km})$.

Eq. (1) can be expressed as

$$\begin{cases} \min f(X), X = [X_1, X_2, \cdots, X_k, \cdots X_n] \in R \\ s.t. \ R = \{X | h_i(X) = 0, i = 1, 2, \cdots, p; \ g_j(X) \geq 0, \\ \quad j = 1, 2, \cdots, q\} \end{cases} \tag{2}$$

Letting $X*$ be the optimal solution to the constrained optimization problem means $\forall X \in R$: $f(X*) \leq f(X)$. In addition, if $g_j(X*) = 0$, the constraint is referred to as active constraint. Under this concept, all the equation constraints $h_i(X) = 0$ $(i = 1, 2, \cdots, p)$ are active at $X*$.

The penalty function method can be used to convert a constrained optimization problem to an unconstrained optimization problem. For this purpose, the penalty function is constructed by [41]

$$P(X, M) = f(X) + M_1 \sum_{i=1}^{p} [h_i(X)]^2 + M_2 \sum_{j=1}^{q} \left[\min(0, g_j(X))\right]^2 \tag{3}$$

where M_1 and M_2 are the penalty factors, generally chosen as large enough positive constants; the second and third terms on the right are the penalty terms, and $P(X, M)$ is the penalty function.

In Eq. (3), when $X \in R$, there should be no penalty to the feasible points, thus $P(X, M) = f(X)$; when $X \notin R$, for the non-feasible points, M_1 and M_2 should be big, therefore, the value of the second and third terms in Eq. (3) are large, which is equivalent to the "penalty" for the infeasible point. Moreover, when X gets farther away from the feasible region, the penalty should be larger. When M_1 and M_2 become sufficiently large, the minimal point $X(M)$ of the unconstrained optimization problem of Eq. (3) is close enough to the minimum point of the original constrained optimization problem. When $X(M) \in R$, it becomes the minimal point of the original constraint problem.

The minimum value of Eq. (3) is given by

$$\min P(X, M) \tag{4}$$

which is equivalent to the minimum value of Eq. (1).

III. BASIC CUCKOO SEARCH ALGORITHM

CS is a novel algorithm based on the breeding behavior of cuckoo species. For the sake of simplicity, the breeding behavior of cuckoo species can be idealized as the following rules [1]:

1) Each of cuckoo lays one egg at a time, and a random nest is selected to dump the egg.
2) The best nests with high quality of eggs will carry over to the next generations.
3) The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability $p_a \in [0,1]$. In this case, the host bird can either discard the egg or the nest so as to build a completely new nest in a new position.

In the basic CS, Yang and Deb also introduced the principle of Lévy flight. The Lévy flight process, named by the French mathematician Paul Lévy, is essentially a model of random walks that is characterized by random step lengths drawn from a power law distribution [3].

Let N be the population size and D be the dimension of the variable. The i-th individual in the population is $X_i = [x_{i1}, x_{i2}, \ldots, x_{iD}]$, The lower and upper boundaries of $x_{i1}, x_{i2}, \ldots, x_{iD}$ are $a_1, a_2, \ldots, a_D$ and $b_1, b_2, \ldots, b_D$, respectively. At the same time, let $a = [a_1, a_2, \ldots, a_D]^T$, $b = [b_1, b_2, \ldots, b_D]^T$.

During the generation of new solution $x_i(t+1)$, a Lévy flight [42]is carried out as in Eq. (5).

$$x_i(t+1) = x_i(t) + \alpha \oplus \text{Lévy}(\beta) \tag{5}$$

where $x_i(t)$ represents the position of the i-th nest at the t-th iteration, $\oplus$ represents dot multiplication, α is step parameter, usually taking $\alpha = 0.01$, Lévy is the Lévy flight utilized for a random walk, and Lévy can be expressed as [33]

$$\text{Lévy}(\beta) = \frac{\mu}{|\nu|^{1/\beta}} \tag{6}$$

where both μ and ν follow the normal distribution.

$$\mu \sim N\left(0, \sigma_\mu^2\right), \quad \nu \sim N\left(0, \sigma_\nu^2\right), \quad \sigma_\mu = 1$$
$$\sigma_\nu = \left(\frac{\Gamma(1+\beta) * \sin(\pi * \beta/2)}{\Gamma((1+\beta)/2) * \beta * 2^{(\beta-1)/2}}\right)^{1/\beta} \tag{7}$$

where $\beta = 1.5$, and Γ is a gamma distribution function.

Abandon the worst nests with a probability (p_a), and build the new ones at new locations according to Eq. (8)

$$x_i(t+1) = \begin{cases} x_i(t) + \nu\left(x_j(t) - x_k(t)\right), & r < p_a \\ x_i(t), & r \geq p_a \end{cases} \tag{8}$$

where t is the number of algorithm iterations, r and ν are uniform random number in [0,1], p_a is the probability that the nest will be discarded, $x_j(t)$ and $x_k(t)$ are the two randomly selected nest positions in the t-th iteration.

The pseudo code of the basic CS algorithm is shown in Algorithm 1.In Algorithm 1, N is the population size, t is the number of iterations, and *MaxGen* is the maximum number of iterations, and p_a is the probability of dropping.

Algorithm 1 Basic Cuckoo Search Algorithm (BCS)

Begin
Randomly generate N initial nests
Evaluate the fitness value of all initial nests
$t = 0$
While t <*MaxGen*
  $t = t+1$;
  For $i = 1$ to N
    Generate new solution $x_i(t+1)$ using Eqs. (5)-(7)
    Evaluate the fitness value of $x_i(t+1)$
    If $f(x_i(t+1)) < f(x_i(t))$
      Replace the i-th solution and accept the new solution as $x_i(t+1)$
    End if
  End for
  Abandon the worst nests with a probability (p_a), and build the new ones at new locations according to Eq. (8)
  Keep the best nests
  Rank the best solutions and find the current best
End while
Output the best solution and optimal value
End

IV. IMPROVED ADAPTIVE HYBRID CUCKOO SEARCH ALGORITHM

A. IMPROVEMENT OF STEP SEARCH METHOD

Lévy flight is a random walk method based on the distribution of heavy-tailed probability [1], which has strong global search ability [40], [44]. In the early stage of the CS algorithm optimization, the Lévy flight method can enhance the global search ability of the algorithm. However, in the later stage of algorithm optimization, Lévy flight can generate large random walk steps, which is not conducive to the local search of the algorithm, resulting in slow convergence of the algorithm. In order to solve this problem, the idea of PSO with inertia weight is used [1], [45]. This paper introduces the inertia weight w in the Lévy flight of CS to accelerate the convergence of the algorithm. The improvement of the Lévy flight method of CS is as follows:

$$x_i(t+1) = wx_i(t) + \alpha \oplus \text{Lévy}(\beta) \oplus (x_i(t) - g_{best}(t)) \tag{9}$$

where w is inertia weight, $x_i(t)$ represents the position of the i-th nest at the t-th iteration,$\oplus$ represents dot multiplication, α is step parameter, and β is a parameter, $g_{best}(t)$ is the best nest.

Technically, the larger w has the greater global search ability whereas the smaller w has greater local search ability. Based on (9), in order to make the CS have a better performance, w was nonlinearly decreased from a relatively large value to a small value. The update method of w is as follows:

$$w = 1 - e^{-1/t} \tag{10}$$

where t is the number of iterations.

Similar to the PSO velocity update method, the Lévy flight method in CS is also sensitive to parameter settings. Therefore, some new CSs have been proposed in succession by introducing some new parameter adjustment methods. The literature [46] summarized the types of two dynamic parameter adjustments of the Lévy flight method in CS, and proposed the adjustment methods of the parameters α and β according to the two types. The literature [47] proposed a method for adjusting the parameter β with nonlinear increment as the number of iterations increases. The literature [48] proposed a method to adaptively adjust the parameter α by fully considering the information of the optimal solution and the optimal value in the population. The literature [49] proposed a method to adjust the parameter β by referring to the swarm information and the individual's own experience. The literature [50] proposed a method for adjusting the parameter α with nonlinear decrement as the number of iterations increases. As can be seen from the simulation results in references [46]–[50] that the improved parameter adjustment methods effectively improve the convergence speed of CS. At the same time, through the analysis of the parameter adjustment methods, it is found that the adjustment of these literatures parameters is related to the maximum number of iterations setting by the algorithm. The maximum number of iterations is set differently, and the convergence effect of the algorithm is also different. In order to reduce the influence of the maximum iteration number setting on CS optimization performance and improve the convergence speed of CS, the new adjustment methods of parameter α and parameter β are given respectively. The adjustment methods of the parameters α and β are as follows:

$$\alpha_i(t) = 0.5 + 1.5(\frac{1}{\sqrt{t}})^{\left|\frac{f_{best}(t)-f_i(t)}{f_{best}(t)-f_{worst}(t)+\varepsilon}\right|} \tag{11}$$

$$\beta_i(t) = 0.5 + 0.1\left|\frac{f_{best}(t)-f_i(t)}{f_{best}-f_{worst}+\varepsilon}\right|^{t^{0.1}} \tag{12}$$

where t is the number of algorithm iterations, $f_i(t)$ is the fitness value of the i-th individual in the t-th iteration of the population, $f_{bset}(t)$ is the optimal fitness value of the t-th iteration of the population,$f_{worst}(t)$ is the worst fitness value of the t-th iteration of the population, ε used to avoid zero-division-error, is the smallest constant in the computer.

It can be seen from equations (9) and (10) that the algorithm can converge quickly due to the introduction of inertia weights that decrease nonlinearly with increasing number of iterations. At the same time, the algorithm can converge to the global optimal solution faster because of the introduction of the optimal individual in the population. In the new adjustment method of the parameter α, individuals in the population can adaptively adjust according to their fitness values and contemporary optimal fitness values. In the adjustment method of the parameter β, the individuals in the population adaptively adjust according to the fitness values of the population and the number of iterations. By analyzing the updating methods of parameters α and β, it is known that individuals with better fitness values have larger parameter values, and individuals with poor fitness values have smaller parameter values. In the update formulas of parameters α and β, the update of the parameters is not affected by the maximum number of iterations.

B. ADDITION OF MUTATION OPERATOR

According to the iterative update mechanism of CS, the algorithm adopts the position update method of Lévy flight to generate a large search step. In the early stage of the algorithm search, CS can perform global search with a large search step. In the later stage of the algorithm search, the larger step size leads to slower convergence of the algorithm and weaker local search ability. In response to this problem, this paper introduces a mutation operator. The operation method of the mutation operator is as follows:

$$x_i(t+1) = \begin{cases} x_i(t) + (x_j(t) - x_i(t))(1 - r_1^{e^{-\lambda/t}}), \\ \qquad r_2 < p_c \& r_3 < 1/2 \\ x_i(t) - (x_j(t) - x_i(t))(1 - r_1^{e^{-\lambda/t}}), \\ \qquad r_2 < p_c \& r_3 \geq 1/2 \\ x_i(t) + r_4(x_j(t) - x_k(t)), \quad r_2 \geq p_c \end{cases} \tag{13}$$

where t is the number of algorithm iterations, the parameter $\lambda = 1.2$, p_c is the probability of mutation, r_1, r_2, r_3 and r_4 are uniform random numbers in [0,1], $x_i(t)$ represents the position of the i-th nest at the t-th iteration, $x_j(t)$ represents the position of the j-th nest at the t-th iteration, $x_k(t)$ represents the position of the k-th nest at the t-th iteration, and i, j and k are not equal to each other.

Analysis of the mutation operator given in this paper shows that in the late iteration of the algorithm, individuals in the population can perform local search in a small step size. The mutation operator given in this paper can effectively enhance the local search ability of the algorithm.

C. EVOLUTIONARY STRATEGY OF ADAPTIVE HYBRID CUCKOO SEARCH ALGORITHM

When the improved CS performs a local search, the algorithm may still have a locally optimal situation. In order to reduce the probability that CS falls into local optimum, this paper proposes a new evolutionary strategy. The evolutionary strategy of the adaptive hybrid cuckoo search is as follows:

(1) Timing begins, initialize the relevant parameters of AHCS, such as the population size N, disturbed iteration threshold T_0, the maximum runtime of the algorithm *maxruntime* and so on.

(2) The initial population is generated as follows:

$$x = a + rand(1, D). * (b - a) \tag{14}$$

where x is an individual in the randomly generated initial population, b is the upper bound vector of the variable, a is the lower bound vector of the variable,D is the dimension of the variable, *rand* (1, D) is a random uniform vector in [0,1], and ".∗" shows the dot product of two vectors.

(3) Calculate the fitness values of all nest positions in the population, and sort the fitness from small to large, and record the best nest position and its fitness.

(4) For each nest, according to Eqs. (6), (7) and (9)-(12), the positional update of the Lévy flight is performed on the AHCS. After that, the fitness value of the nest position after the position update is calculated, and compared with the fitness of the bird's nest position before the update, the better bird's nest position and its fitness value are retained.

(5) For each nest, the nest position is discarded according to Eqs. (15)-(18). After that, the fitness value of the nest position after the discarding operation is calculated, and compared the fitness value of the nest position before the discarding, the

better nest position and its fitness are retained.

$$x_i(t+1) = \begin{cases} x_i(t) + H_1\left(x_p(t) - x_i(t)\right) \\ \quad +H_2\left(x_g(t) - x_i(t)\right) \\ +rr_1\left(x_j(t) - x_k(t)\right), & r < p_a \\ x_i(t), & r \geq p_a \end{cases} \tag{15}$$

$$p_a = \frac{1}{1 + e^{(-(\frac{t-1}{t})^{50})}} - 0.5 \tag{16}$$

$$H_1 = \begin{cases} 0, & p_a < rr_2 \\ 0.5, & p_a = rr_2 \\ 1, & p_a > rr_2 \end{cases} \tag{17}$$

$$H_2 = \begin{cases} 0, & p_a < rr_3 \\ 0.5, & p_a = rr_3 \\ 1, & p_a > rr_3 \end{cases} \tag{18}$$

where t is the number of algorithm iterations, parameter p_a is the drop probability, $x_p(t)$ is the historical optimal solution for the i-th bird's nest position,$x_g(t)$ is the global optimal solution of the population,$x_i(t)$, $x_j(t)$, and $x_k(t)$ are different bird nest positions, r, rr_1, rr_2, and rr_3 are uniform random numbers in [0,1].

(6) According to Eq. (13), all nest positions are subjected to mutation operations. After that, for each nest, the fitness value of the nest position after the mutation operation is calculated, and compared with the adaptability of the nest position before the mutation, and the better nest position and its fitness are retained.

(7) Record the optimal nest position and its fitness value in each iteration of the algorithm. If the number of fitness stagnations exceeds T_0, the optimal position of the population is perturbed using Eq. (19). After that, the fitness value of the nest position after the disturbance is calculated, and compared with the fitness value of the best nest position before the disturbance, the better nest position and its fitness value are retained.

$$x_{best}(t+1) = x_{best}(t) + N(0, 1). * (x_i(t) - x_j(t)) \tag{19}$$

where x_{best} is the optimal solution in the population, and N(0,1) is a Gaussian distribution with mean 0 and variance 1, ".*" represents dot multiplication, and $x_i(t)$ and $x_j(t)$ are any two different nest positions in the population of the t-th iteration.

(8) Record the running time of the algorithm *runtime*, if the iteration stop condition is satisfied, the optimal solution and the optimal value are output; if not, then return to (4).

The pseudo code of the adaptive hybrid cuckoo search is shown in Algorithm 2.

In Algorithm 2, N is the population size, t is the iteration number, p_a is the probability of dropping, p_c is the probability of mutation, *runtime* is the running time of the algorithm, *maxruntime* is the maximum runtime of the algorithm, T_0 is the maximum number of stagnation of the optimal value in the population, T is the number of stagnations of the optimal value in the population.

Algorithm 2 Adaptive Hybrid Cuckoo Search (AHCS)

Begin
Let timing initial value is *runtime*=0, start the timer
Randomly generate N initial nests
Evaluate the fitness value of all initial nests
$t = 0$;
While *runtime*<*maxruntime*
  $t = t+1$;
  For $i = 1$ to N
    Generate a new solution $x_i(t+1)$ using Eqs. (6), (7) and (9)-(12)
    Evaluate the fitness value of $x_i(t+1)$
    If $f(x_i(t+1)) < f(x_i(t))$
      Replace the i-th solution and accept the new solution as $x_i(t+1)$
    End if
  End for
  For $i = 1$ to N
    Abandon the worst nests with a probability (p_a), and generate a new solution $x_i(t+1)$ according to Eqs. (15)-(18)
    Evaluate the fitness value of $x_i(t+1)$
    If $f(x_i(t+1)) < f(x_i(t))$
      Replace the i-th solution and accept the new solution as $x_i(t+1)$
    End if
  End for
  For $i = 1$ to N
    Mutation operation with a probability (p_c), and generate a new solution $x_i(t+1)$ according to Eq. (13)
    Evaluate the fitness value of $x_i(t+1)$
    If $f(x_i(t+1)) < f(x_i(t))$
      Replace the i-th solution and accept the new solution as $x_i(t+1)$
    End if
  End for
  If $T > T_0$
    generate a new solution $x_{best}(t+1)$ according to Eq. (19)
    Evaluate the fitness value of $x_{best}(t+1)$
    If $f(x_{best}(t+1)) < f(x_{best}(t))$
      Replace the i-th solution and accept the new solution as $x_{best}(t+1)$
    End if
  End if
  Record running time *runtime*
End while
Output the best solution and optimal value
End

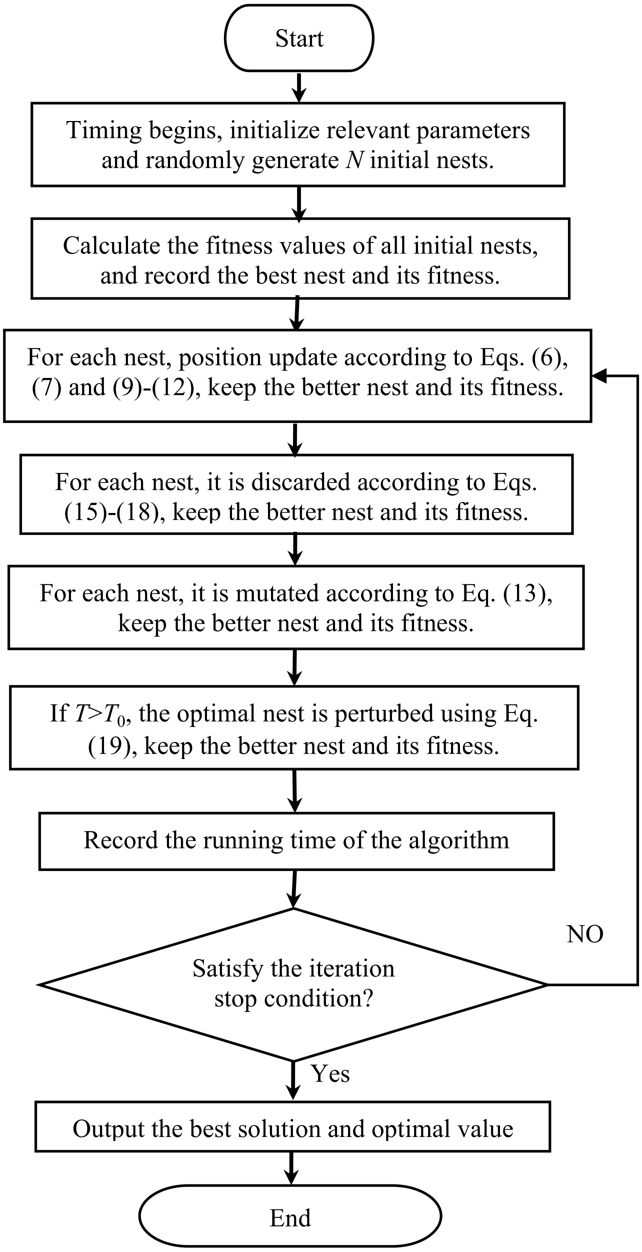

FIGURE 1. The flow chart of the AHCS algorithm.

The flow chart of the adaptive hybrid cuckoo search algorithm is shown in FIGURE 1.

V. NUMERICAL EXPERIMENTS

A. BENCHMARK FUNCTIONS

To evaluate the performance of the proposed AHCS and conduct a further comparative study, a set of well-known test functions [23], [51] are used as benchmark problems. The dimension of these test functions can be fixed or unfixed. Functions f_1-f_{14} and f_{21}-f_{23} are unimodal functions, and functions f_{15}-f_{20} and f_{24}-f_{30} are multimodal functions. The 30 benchmark functions are shown in TABLE 1.

In TABLE 1, Dim represents the dimension of these test functions. D is the dimension of the variable, and it is a variable value. This paper takes D=30.

B. ALGORITHM PERFORMANCE EVALUATION INDEX

In this section, in order to measure the performance of AHCS, the mean value, standard deviation value, t-test values, Friedman average ranking given in the literature [22], and VS are used as performance evaluation indicators.

The mean value is defined as the average function value achieved by the algorithm out of the total number of independent runs R times. The mean value is abbreviated as $MEAN$, and is represented in Eq. (20).

$$MEAN = \frac{\sum_{i=1}^{R} f_i}{R} \tag{20}$$

where f_i is defined as the best fitness value that the algorithm achieved in the i-th run.

VS is an evaluation index of the mean comparison between the other algorithm and the proposed algorithm. "+", "−" and "=" are three results of the mean comparison between the other algorithm and the proposed algorithm. "+" signifies the condition where mean of the other algorithm is better than the proposed algorithm, "−" signifies the condition where mean of the other algorithm is worse than the proposed algorithm, "=" signifies the condition where both the proposed and the algorithm in comparison have the same mean results.

The standard deviation is defined as the actual deviation that exists between the average function values achieved in R times runs. The standard deviation is abbreviated as SD, and is represented in Eq. (21).

$$SD = \sqrt{\frac{\sum_{i=1}^{R} (f_i - MEAN)^2}{R}} \tag{21}$$

The t-test values are calculated for every function using its mean and standard deviation in each existing algorithm. The t-test value is abbreviated as T_{val}, and is calculated using the Eq. (22).

$$T_{val} = \frac{MEAN_1 - MEAN_2}{\sqrt{\frac{SD_1^2+SD_2^2}{R}} + \varepsilon} \tag{22}$$

where $MEAN_1$, $MEAN_2$ and SD_1, SD_2 are respectively the mean and standard deviation values of other algorithms and AHCS algorithm, R is the number of independent runs of the algorithm, ε is used to avoid zero-division-error, and ε is the smallest constant in the computer. The t-test has been carried out with the significance level of $\alpha = 0.05$. The winning (w), tie (t) and lost (l) cases of t-test values for AHCS over existing algorithms are represented in the last row of each table with the attributes namely $w/t/l$. For each algorithm that compares with AHCS, w represents the number of t-test values greater than 0, t represents the number of t-test values equal to 0, and l represents the number of t-test values less than 0.

In order to check the significant difference of the proposed AHCS algorithm, the Friedman average ranking test has been used to compare the performance of the AHCS and other algorithms. With this test, the relative ranking of each algorithm has been calculated based on the mean

TABLE 1. 30 benchmark functions with dimension in experiments.

Name	Function	Dim	Range	Min
Sphere	$f_1(X)=\sum_{i=1}^{D}x_i^2$	D	[-100, 100]	0
Schwefel 2.22	$f_2(X)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	D	[-10, 10]	0
Eason	$f_3(X)=-\cos(x_1)\cos(x_2)\exp(-(x_1-\pi)^2-(x_2-\pi)^2)$	2	[-100,100]	-1
Step	$f_4(X)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	D	[-100,100]	0
Sum Square	$f_5(X)=\sum_{i=1}^{D}ix_i^2$	D	[-100,100]	0
Quartic	$f_6(X)=\sum_{i=1}^{D}ix_i^4+random(0,1)$	D	[-1.28,1.28]	0
Dixion-Price	$f_7(X)=(x_1-1)^2+\sum_{i=2}^{D}i(2x_i^2-x_{i-1})^2$	D	[-10,10]	0
Schwefel 2.21	$f_8(X)=\max\{\lvert x_i\rvert,\ 1\le i\le D\}$	D	[-100, 100]	0
Zakharov	$f_9(X)=\sum_{i=1}^{D}x_i^2+(\sum_{i=1}^{D}0.5ix_i)^2+(\sum_{i=1}^{D}0.5ix_i)^4$	D	[-5,10]	0
Matyas	$f_{10}(X)=0.26(x_1^2+x_2^2)-0.48x_1x_2$	2	[-10,10]	0
Trid6	$f_{11}(X)=\sum_{i=1}^{D}(x_i-1)^2-\sum_{i=2}^{D}x_ix_{i-1}$	6	[-36,36]	-50
Powell	$f_{12}(X)=\sum_{i=1}^{D/4}\left[(x_{4i-3}+10x_{4i-2})^2+5(x_{4i-1}-x_{4i})^2+(x_{4i-2}-x_{4i-1})^4+10(x_{4i-3}-x_{4i})^4\right]$	24	[-4,5]	0
Beale	$f_{13}(X)=(1.5-x_1+x_1x_2)^2+(2.25-x_1+x_1x_2^2)^2+(2.625-x_1+x_1x_2^3)^2$	2	[-4.5,4.5]	0
Trid10	$f_{14}(X)=\sum_{i=1}^{D}(x_i-1)^2-\sum_{i=2}^{D}x_ix_{i-1}$	10	[-100,100]	-210
Rastrigin	$f_{15}(X)=\sum_{i=1}^{D}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	D	[-5.12, 5.12]	0
Griewank	$f_{16}(X)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(x_i/\sqrt{i})+1$	D	[-600, 600]	0
Weierstrass	$f_{17}(X)=\sum_{i=1}^{D}(\sum_{k=0}^{k_{max}}[a^k\cos(2\pi b^k(x_i+0.5))])-D\sum_{k=0}^{k_{max}}[a^k\cos(2\pi b^k0.5)]$ $a=0.5,\ b=3,\ k_{max}=20$	D	[-0.5, 0.5]	0
Bohachevsky1	$f_{18}(X)=x_1^2+2x_2^2-0.3\cos(3\pi x_1)-0.4\cos(4\pi x_2)+0.7$	2	[-100,100]	0
Bohachevsky2	$f_{19}(X)=x_1^2+2x_2^2-0.3\cos(3\pi x_1)\cos(4\pi x_2)+0.3$	2	[-100,100]	0
Bohachevsky3	$f_{20}(X)=x_1^2+2x_2^2-0.3\cos(3\pi x_1+4\pi x_2)+0.3$	2	[-100,100]	0
Elliptic	$f_{21}(X)=\sum_{i=1}^{D}(10^6)^{\frac{i-1}{D-1}}x_i^2$	D	[-100,100]	0
SumPower	$f_{22}(X)=\sum_{i=1}^{D}\lvert x_i\rvert^{(i+1)}$	D	[-1,1]	0
Exponential	$f_{23}(X)=\exp(0.5\sum_{i=1}^{D}\lvert x_i\rvert)-1$	D	[-1.28,1.28]	0
Rosenbrock	$f_{24}(X)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(1-x_i)^2]$	D	[-5,10]	0
NCRastrigin	$f_{25}(X)=\sum_{i=1}^{D}\left[y_i^2-10\cos(2\pi y_i)+10\right]$ $y_i=\begin{cases}x_i, & \lvert x_i\rvert<0.5\\ 0.5round(2x_i), & \lvert x_i\rvert\ge 0.5\end{cases}$	D	[-5.12,5.12]	0
Ackley	$f_{26}(X)=-20\exp(-0.2\sqrt{\sum_{i=1}^{D}x_i^2/D})-\exp([\sum_{i=1}^{D}\cos(2\pi x_i)]/D)+20+e$	D	[-32,32]	0
Penalized1	$f_{27}(X)=\pi/D\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+$ $(y_D-1)^2\}+\sum_{i=1}^{D}u(x_i,10,100,4)$ $y_i=1+0.25(x_i+1),\ u_{x_i,a,k,m}=\begin{cases}k(x_i-a)^m, & x_i>a\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^m, & x_i<-a\end{cases}$	D	[-50,50]	0
Penalized2	$f_{28}(X)=0.1\{\sin^2(\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+$ $(x_D-1)^2[1+\sin^2(2\pi x_D)]\}+\sum_{i=1}^{D}u(x_i,5,100,4)$	D	[-50,50]	0
Alpine	$f_{29}(X)=\sum_{i=1}^{D}\lvert x_i\sin(x_i)+0.1x_i\rvert$	D	[-10,10]	0
Levy	$f_{30}(X)=\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+\sin^2(3\pi x_1)+\lvert x_D-1\rvert[1+\sin^2(3\pi x_D)]$	D	[-10,10]	0

TABLE 2. Parameter Settings for seven improved CSs.

Algorithm	Year	Parameters
BCS	2009	α=0.01, p_a=0.25, β=1.5
ICS	2018	m=0.8, k=0.2, p=5, S_{min}=0.01, p_{amin}=0.1
CS-EO	2018	a_0=0.01, b_0=1, p_b=0.25, $p_a\in[0,1]$, $\sigma\in[0,1]$, $m\in[0,1]$, $\lambda\in(1,3]$
DCS	2019	α_{max}=0.5, α_{min}=0.01, p_{amax}=0.5, p_{amin}=0.01, $r'\in[0,1]$, $\lambda\in(1,3]$
RCS	2019	C=1, w=2, φ=2, $\alpha(0)$=0.1, $\beta\in(1,3]$
MCS	2019	α_{max}=1.5, α_{min}=0.1, $\lambda\in(1,3]$, $p_a\in[0,1]$
CV 1.0	2018	α=0.01, p_a=0.5, $F\in[0,1]$
AHCS	2019	w_{max}=0.7, w_{min}=0.4, s=0.8, T_0=10, λ=1.2, M=10^8

values for every function and finally the average ranking and final ranking of all the algorithms are given. The final ranking of all algorithms is based on their average ranking. The average ranking is computed using the following process.

Step1: Gather statistical results for k algorithm.

Step2: For each function i, rank mean values from 1 (best result) to k(worst result), the average ranks are assigned in case of ties. Denote these ranks as $r_i^j(1\leq j\leq k)$.

Step3: For each algorithm j, average the ranks obtained in all functions to obtain the final average ranking R_j, the R_j is computed by Eq. (23).

$$R_j=\frac{1}{n}\sum\nolimits_{i=1}^{n} r_i^j \quad (j=1,2,\cdots,k) \tag{23}$$

where n is the number of functions, k is the number of algorithms for participation in comparison.

C. COMPARSION OF ADAPTIVE HYBRID CUCKOO SEARCH ALGORITHM AND OTHER IMPROVED CUCKOO SEARCH ALGORITHMS

AHCS was compared to seven improved CSs in the literature. The seven improved CSs that participated in the comparison are as follows: (1) the basic CS in reference [1] is abbreviated as BCS; (2) the improved CS in reference [52] is abbreviated as ICS; (3) the proposed CS with explosion operator in reference [20] is abbreviated as CS-EO; (4) the proposed dynamic CS in reference [12] is abbreviated as DCS; (5) the proposed reinforced cuckoo search in reference [22] is abbreviated as RCS; (6) the modified CS in reference [28] is abbreviated as MCS; (7) the proposed cuckoo search version 1.0 in reference [53] is abbreviated as CV 1.0.

The size of all populations participating in the comparison algorithm is $N = 40$, and the maximum runtime is *maxruntime*=20 seconds. See TABLE 2 for other parameter settings for various algorithms. For each algorithm, each test function is run independently 30 times. In order to ensure fairness of comparison, and to quickly obtain statistical results, it is guaranteed that for the same test functions, all algorithms participating in the comparison are tested on the same operating system of the same computer. At the same time, different test functions can be tested in the same operating system in different computers.

The statistical results of the mean and standard deviation are shown in TABLE 3. The comparisons of the statistical results are shown in TABLE 4.

As seen in TABLE 3, the algorithm proposed in this paper can converge global optimal solution for most of the test functions, which are f_1-f_3, f_5-f_{23}, f_{25}, and f_{29}; while other intelligent methods have some cases where the convergence accuracy of the test function is low, for example, BCS for functions $f_1, f_{16}, f_{17}, f_{21}, f_{24}$-$f_{28}$ and f_{30}, ICS for functions f_2, $f_9, f_{15}, f_{17}, f_{21}$, and f_{24}-f_{27}, DCS for functions f_{15} and so on.

As seen in TABLE 4, compared to the other seven improved CSs, the number of winning cases of t-test values for AHCS is greater than the number of lost cases of t-test values, and the Friedman average ranking of AHCS is better than the other seven improved CSs. Therefore, the performance of AHCS is significantly better than the other seven improved algorithms.

D. COMPARSION OF ADAPTIVE HYBRID CUCKOO SEARCH ALGORITHM AND OTHER IMPROVED ALGORITHMS

AHCS was compared to seven improved algorithms in the literature. The seven improved algorithms that participated in the comparison are as follows: (1) the improved chicken swarm optimization in reference [54] is abbreviated as ICSO; (2) the improved chicken swarm optimization in reference [55] is abbreviated as MDCSO; (3) the improved firefly algorithm in reference [56] is abbreviated as NAFA; (4) the improved firefly algorithm in reference [57] is abbreviated as DUFA; (5) the improved PSO in reference [58] is abbreviated as OBLPSOGD; (6) the improved grey wolf optimizer in reference [59] is abbreviated as AGWO; (7) the improved real code genetic algorithm in reference [60] is abbreviated as IRCGA.

The size of all populations participating in the comparison algorithm is $N = 40$, and the maximum runtime is *maxruntime*=20 seconds. See TABLE 5 for other parameter settings for various algorithms. For each algorithm, each test function is run independently 30 times. In order to ensure fairness of comparison, and to quickly obtain statistical results, it is guaranteed that for the same test functions, all algorithms participating in the comparison are tested in the same operating system of the same computer. At the same time, different

TABLE 3. The statistical results of AHCS and seven improved CSs on 30 benchmark functions.

Function	Evaluation indicator	Algorithm							
		AHCS	BCS	ICS	CS-EO	DCS	RCS	MCS	CV 1.0
f_1	*MEAN*	0.00E+00	3.32E-03	2.31E-01	1.06E-19	3.13E-17	1.10E+03	3.42E-12	0.00E+00
	SD	0.00E+00	1.20E-03	5.42E-02	1.77E-19	8.73E-18	6.19E+02	7.65E-12	0.00E+00
f_2	*MEAN*	0.00E+00	5.12E-01	5.07E+02	2.08E-12	3.06E-09	1.81E+11	2.12E-09	0.00E+00
	SD	0.00E+00	5.25E-02	7.21E+02	4.65E-12	6.19E-09	2.52E+11	4.73E-09	0.00E+00
f_3	*MEAN*	-1.00E+00	-1.00E+00	-1.00E+00	-1.00E+00	-1.00E+00	-4.00E-01	-1.00E+00	-1.00E+00
	SD	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.48E-01	0.00E+00	0.00E+00
f_4	*MEAN*	6.16E-34	3.01E-03	3.03E-03	0.00E+00	2.79E-17	9.47E+02	8.46E-06	3.24E+00
	SD	1.38E-33	3.10E-04	6.63E-04	0.00E+00	1.02E-17	5.38E+02	1.89E-05	5.80E-01
f_5	*MEAN*	0.00E+00	6.84E-03	3.17E-02	3.96E-28	3.17E-17	5.08E+00	3.71E-17	0.00E+00
	SD	0.00E+00	2.31E-03	1.01E-02	8.85E-28	1.03E-17	2.81E+00	8.29E-17	0.00E+00
f_6	*MEAN*	0.00E+00	5.86E-06	1.63E-04	7.84E-31	7.29E-18	7.89E-03	2.57E-10	0.00E+00
	SD	0.00E+00	2.24E-06	9.17E-05	1.74E-30	3.89E-19	9.52E-03	5.74E-10	0.00E+00
f_7	*MEAN*	0.00E+00	5.06E-04	1.33E-05	4.40E-29	3.33E-17	3.02E-01	1.92E-13	1.11E+01
	SD	0.00E+00	1.50E-04	1.67E-06	9.83E-29	3.40E-18	1.80E-01	4.29E-13	4.32E+00
f_8	*MEAN*	0.00E+00	4.10E+01	2.32E-01	1.14E-04	1.12E-03	1.94E+01	2.60E-03	0.00E+00
	SD	0.00E+00	2.23E+00	3.67E-02	1.56E-04	1.04E-03	1.77E+00	5.82E-03	0.00E+00
f_9	*MEAN*	0.00E+00	4.16E-02	1.36E+02	2.43E+01	4.08E-02	7.39E+08	2.30E+00	0.00E+00
	SD	0.00E+00	5.58E-03	6.17E+01	2.38E+01	1.30E-02	7.93E+08	3.23E+00	0.00E+00
f_{10}	*MEAN*	0.00E+00	1.01E-58	2.44E-71	1.24E-45	1.33E-20	5.76E-02	4.00E+00	0.00E+00
	SD	0.00E+00	2.07E-58	3.07E-71	2.77E-45	2.97E-20	5.13E-02	0.00E+00	0.00E+00
f_{11}	*MEAN*	-5.00E+01	-5.00E+01	-5.00E+01	-5.00E+01	-5.00E+01	3.05E+01	-5.00E+01	-5.00E+01
	SD	0.00E+00	2.56E-14	0.00E+00	0.00E+00	3.12E-14	2.05E+01	0.00E+00	0.00E+00
f_{12}	*MEAN*	0.00E+00	9.67E-02	3.23E-01	4.68E-02	6.96E-03	5.63E+00	1.13E-03	0.00E+00
	SD	0.00E+00	1.27E-02	8.32E-02	6.55E-02	3.13E-03	4.13E+00	6.28E-04	0.00E+00
f_{13}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.51E-30	1.84E+00	2.17E-21	0.00E+00
	SD	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.61E-30	8.05E-01	4.85E-21	0.00E+00
f_{14}	*MEAN*	-2.10E+02	-2.10E+02	-2.10E+00	-2.10E+02	-2.10E+02	7.82E+02	-2.10E+02	-2.10E+02
	SD	4.07E-13	4.07E-13	0.00E+00	4.98E-13	4.07E-13	4.58E+02	1.11E-12	4.07E-13
f_{15}	*MEAN*	0.00E+00	6.24E+01	1.57E+02	0.00E+00	7.79E+01	1.32E+02	9.98E+01	0.00E+00
	SD	0.00E+00	7.59E+00	6.73E+01	0.00E+00	1.16E+01	1.89E+01	3.78E+01	0.00E+00
f_{16}	*MEAN*	0.00E+00	1.04E+01	1.61E+00	1.89E-15	1.78E-05	1.73E+02	0.00E+00	0.00E+00
	SD	0.00E+00	2.77E+00	3.66E-01	4.22E-15	1.76E-05	2.94E+01	0.00E+00	0.00E+00
f_{17}	*MEAN*	0.00E+00	1.84E+01	3.71E+01	1.13E+00	3.61E+00	3.65E+01	6.41E+00	0.00E+00
	SD	0.00E+00	8.15E-01	1.17E+00	9.87E-01	2.88E-01	1.62E+00	7.10E+00	0.00E+00
f_{18}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.20E+02	0.00E+00	0.00E+00
	SD	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.45E+02	0.00E+00	0.00E+00
f_{19}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.47E+02	0.00E+00	0.00E+00
	SD	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.77E+02	0.00E+00	0.00E+00
f_{20}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.19E+02	0.00E+00	0.00E+00
	SD	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.85E+01	0.00E+00	0.00E+00
f_{21}	*MEAN*	0.00E+00	9.56E+04	3.27E+07	1.71E-10	8.21E-24	3.22E+08	2.29E+05	0.00E+00
	SD	0.00E+00	1.02E+05	1.25E+07	2.34E-10	1.84E-23	2.19E+08	5.13E+05	0.00E+00
f_{22}	*MEAN*	0.00E+00	7.39E-08	1.62E-05	2.27E-52	8.41E-19	1.49E-03	8.48E-14	0.00E+00
	SD	0.00E+00	3.84E-08	9.66E-06	5.08E-52	1.14E-18	1.04E-03	1.90E-13	0.00E+00
f_{23}	*MEAN*	0.00E+00	1.34E-01	4.75E-01	3.56E-16	1.79E-11	5.90E-01	4.10E-13	0.00E+00
	SD	0.00E+00	8.11E-03	5.72E-02	2.98E-16	1.52E-11	3.43E-01	9.16E-13	0.00E+00
f_{24}	*MEAN*	3.44E-09	1.50E+01	1.33E+02	1.88E+01	1.92E+01	1.21E+03	1.03E+01	2.87E+01
	SD	2.52E-09	5.67E+00	1.36E+02	4.19E+00	4.23E+00	3.84E+02	1.37E+01	1.65E-02
f_{25}	*MEAN*	0.00E+00	6.71E+01	1.36E+02	1.22E-01	6.27E+01	9.73E+01	1.21E+02	0.00E+00
	SD	0.00E+00	6.06E+00	1.45E+01	1.44E-01	4.35E+00	5.23E+01	1.62E+01	0.00E+00
f_{26}	*MEAN*	8.88E-16	1.87E+01	1.81E+01	2.82E-10	7.70E-07	1.96E+01	2.02E-03	8.89E-16
	SD	0.00E+00	3.33E-01	1.71E-01	3.33E-10	1.57E-06	1.48E-01	4.52E-03	0.00E+00
f_{27}	*MEAN*	2.07E-33	6.10E+07	8.96E+07	1.85E-07	9.05E+00	4.68E+08	2.76E-17	7.20E+00
	SD	2.83E-33	3.34E+04	6.79E+07	4.14E-07	3.78E-01	3.08E+08	6.17E-17	1.34E+00
f_{28}	*MEAN*	1.75E-33	1.53E+08	1.04E+08	8.50E-11	9.58E-11	1.03E+09	1.29E-20	1.96E+01
	SD	5.51E-34	1.03E+08	5.42E+07	1.90E-10	9.79E-11	9.87E+08	2.89E-20	1.01E+00
f_{29}	*MEAN*	0.00E+00	1.87E+00	8.05E+00	1.25E-02	2.25E+00	625E+00	4.43E-03	0.00E+00
	SD	0.00E+00	1.19E-01	4.47E+00	2.70E-02	6.32E-01	2.52E+00	6.17E-03	0.00E+00
f_{30}	*MEAN*	4.19E-03	1.15E+02	9.68E+00	2.85E-02	5.58E-02	1.60E+02	2.35E-03	3.49E+00
	SD	2.71E-03	3.69E+01	2.59E+00	2.84E-02	3.89E-02	3.16E+01	2.46E-03	2.89E+00

TABLE 4. The comparison results of AHCS and seven improved CSs on 30 benchmark functions.

Function	Evaluation indicator	Algorithm							
		AHCS	BCS	ICS	CS-EO	DCS	RCS	MCS	CV 1.0
f_1	*Rank*	1.5	6	7	3	4	8	5	1.5
	T_{val}	/	1.52E+01	2.33E+01	4.77E-04	1.40E-01	9.73E+00	2.45E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_2	*Rank*	1.5	6	7	3	5	8	4	1.5
	T_{val}	/	5.34E+01	3.85E+00	2.45E+00	2.71E+00	3.93E+00	2.45E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_3	*Rank*	4	4	4	4	4	8	4	4
	T_{val}	/	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.00E+00	0.00E+00	0.00E+00
	VS	/	=	=	=	=	-	=	=
f_4	*Rank*	2	5	6	1	3	8	4	7
	T_{val}	/	5.32E+01	2.50E+01	-2.77E-18	1.25E-01	9.64E+00	2.45E+00	3.06E+01
	VS	/	-	-	+	-	-	-	-
f_5	*Rank*	1.5	6	7	3	4	8	5	1.5
	T_{val}	/	1.62E+01	1.72E+01	1.78E-12	1.42E-01	9.90E+00	1.56E-01	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_6	*Rank*	1.5	6	7	3	4	8	5	1.5
	T_{val}	/	1.43E+01	9.74E+00	3.53E-15	3.28E-02	4.54E+00	2.45E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_7	*Rank*	1	6	5	2	3	7	4	8
	T_{val}	/	1.85E+01	4.36E+01	1.98E-13	1.50E-01	9.19E+00	2.44E+00	1.41E+01
	VS	/	-	-	-	-	-	-	-
f_8	*Rank*	1.5	8	6	3	4	7	5	1.5
	T_{val}	/	1.01E+02	3.46E+01	4.00E+00	5.90E+00	6.00E+01	2.45E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_9	*Rank*	1.5	4	7	6	3	8	5	1.5
	T_{val}	/	4.08E+01	1.21E+01	5.59E+00	1.72E+01	5.10E+00	3.90E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{10}	*Rank*	1.5	4	3	5	6	7	8	1.5
	T_{val}	/	4.55E-43	1.10E-55	5.58E-30	5.99E-05	6.15E+00	1.80E+16	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{11}	*Rank*	4	4	4	4	4	8	4	4
	T_{val}	/	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.15E+01	0.00E+00	0.00E+00
	VS	/	=	=	=	=	-	=	=
f_{12}	*Rank*	1.5	6	7	5	4	8	3	1.5
	T_{val}	/	4.17E+01	2.13E+01	3.91E+00	1.22E+01	7.47E+00	9.86E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{13}	*Rank*	3	3	3	3	6	8	7	3
	T_{val}	/	0.00E+00	0.00E+00	0.00E+00	1.13E-14	1.25E+01	9.77E-06	0.00E+00
	VS	/	=	=	=	-	-	-	=
f_{14}	*Rank*	3.5	3.5	7	3.5	3.5	8	3.5	3.5
	T_{val}	/	0.00E+00	2.79E+15	0.00E+00	0.00E+00	1.19E+01	0.00E+00	0.00E+00
	VS	/	=	-	=	=	-	=	=
f_{15}	*Rank*	2	4	8	2	5	7	6	2
	T_{val}	/	4.50E+01	1.28E+01	0.00E+00	3.68E+01	3.83E+01	1.45E+01	0.00E+00
	VS	/	-	-	=	-	-	-	=
f_{16}	*Rank*	2	7	6	4	5	8	2	2
	T_{val}	/	2.06E+01	2.41E+01	1.90E+00	5.54E+00	3.22E+01	0.00E+00	0.00E+00
	VS	/	-	-	-	-	-	=	=
f_{17}	*Rank*	1	6	8	3	4	7	5	1.5
	T_{val}	/	1.24E+02	1.74E+02	6.27E+00	6.87E+01	1.23E+02	4.94E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{18}	*Rank*	4	4	4	4	4	8	4	4
	T_{val}	/	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.31E+00	0.00E+00	0.00E+00
	VS	/	=	=	=	=	-	=	=
f_{19}	*Rank*	4	4	4	4	4	8	4	4
	T_{val}	/	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.64E+00	0.00E+00	0.00E+00
	VS	/	=	=	=	=	-	=	=

TABLE 4. *(Continued.)* **The comparison results of AHCS and seven improved CSs on 30 benchmark functions.**

f_{20}	*Rank*	4	4	4	4	4	8	4	4
	T_{val}	/	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.99E+01	0.00E+00	0.00E+00
	VS	/	=	=	=	=	-	=	=
f_{21}	*Rank*	1.5	5	7	4	3	8	6	1.5
	T_{val}	/	5.13E+00	1.43E+01	4.00E+00	3.70E-08	8.05E+00	2.44E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{22}	*Rank*	1.5	6	7	3	4	8	5	1.5
	T_{val}	/	1.05E+01	9.19E+00	1.02E-36	3.78E-03	7.85E+00	2.43E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{23}	*Rank*	1.5	6	7	3	5	8	4	1.5
	T_{val}	/	9.05E+01	4.55E+01	1.29E+00	6.45E+00	9.42E+00	2.45E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{24}	*Rank*	1	3	7	4	5	8	2	6
	T_{val}	/	1.45E+01	5.36E+00	2.46E+01	2.49E+01	1.73E+01	4.12E+00	9.53E+03
	VS	/	-	-	-	-	-	-	-
f_{25}	*Rank*	1.5	5	8	3	4	6	7	1.5
	T_{val}	/	6.06E+01	5.14E+01	4.64E+00	7.89E+01	1.02E+01	4.09E+01	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{26}	*Rank*	1	7	6	3	4	8	5	2
	T_{val}	/	3.08E+02	5.80E+02	4.64E+00	2.69E+00	7.25E+02	2.45E+00	4.50E-03
	VS	/	-	-	-	-	-	-	-
f_{27}	*Rank*	1	6	7	3	5	8	2	4
	T_{val}	/	1.00E+04	7.23E+00	2.45E+00	1.31E+02	8.32E+00	1.18E-01	2.94E+01
	VS	/	-	-	-	-	-	-	-
f_{28}	*Rank*	1	7	6	3	4	8	2	5
	T_{val}	/	8.14E+00	1.05E+01	2.45E+00	5.36E+00	5.72E+00	5.81E-05	1.06E+02
	VS	/	-	-	-	-	-	-	-
f_{29}	*Rank*	1.5	5	7	4	6	8	3	1.5
	T_{val}	/	8.61E+01	9.86E+00	2.54E+00	1.95E+01	1.36E+03	3.93E+00	0.00E+00
	VS	/	-	-	-	-	-	-	=
f_{30}	*Rank*	2	7	6	3	4	8	1	5
	T_{val}	/	1.71E+01	2.05E+01	4.67E+00	7.25E+00	2.77E+01	-2.75E+00	6.61E+00
	VS	/	-	-	-	-	-	+	-
Average Ranking		2.02	5.25	6.07	3.42	4.25	7.77	4.28	2.95
Final Ranking		1	6	7	3	4	8	5	2
w/t/l		/	23/7/0	24/6/0	21/8/1	24/6/0	30/0/0	22/7/1	7/23/0

Note: "/" indicates that the same items do not compare.

test functions can be tested in the same operating system on different computers.

The statistical results of the mean and standard deviation are shown in TABLE 6. The comparisons of the statistical results are shown in TABLE 7.

As seen in TABLE 6, the algorithm proposed in this paper can converge global optimal solution for most of the test functions, which are f_1-f_3, f_5-f_{23}, f_{25}, and f_{29}; while other intelligent methods have some cases where the convergence accuracy of the test function is low, for example, ICSO for functions f_4, f_{24}, f_{28} and f_{30}, MDCSO for functions f_4, f_7, and f_{24}, and so on. For functions f_4 and f_{30}, all algorithms cannot converge global optimal solution.

As seen in TABLES 7, compared to the other seven improved algorithms, the number of winning cases of t-test values for AHCS is greater than the number of lost cases of t-test values, and the Friedman average ranking of AHCS is better than the other seven improved algorithms. Therefore, the performance of AHCS is significantly better than the other seven improved algorithms.

E. TEST ON CEC 2017 OPTIMIZATION PROBLEMS

We further tested the performance of the proposed approach on a set of 28 CEC 2017 optimization problems. TABLE 8 show these test problems, the detail of which can be found in reference [61].

The size of all populations participating in the comparison algorithm is $N = 40$, and the maximum runtime is *maxruntime*=20 seconds. The number of the decision variables of the optimization problems is $D = 30$, and the penalty factor is $M = 10^8$. The other parameters are set in TABLES 2 and 5. Each algorithm is run independently 30 times for each CEC 2017 optimization problem. In order to ensure fairness of comparison, and to quickly obtain statistical results, it is guaranteed that for the same test functions, all algorithms participating in the comparison are tested in the same operating system of the same computer. At the same time, different test functions can be tested in the same operating system on different computers.

TABLES 9 and 11 show the statistical results of AHCS and other improved algorithms. The comparison results of AHCS

TABLE 5. Parameter Settings for seven improved algorithms.

Algorithm	Year	Parameters
ICSO	2016	RN=0.2N, HN=0.6N, CN=0.2N, MN=0.1N, w_{min}=0.4, w_{max}=0.9, C=0.4, G=10
MDCSO	2018	RN=0.2N, HN=0.6N, CN=0.2N, MN=0.1N, $FL\in$[0.4,1], G=10
NAFA	2017	α=0.5, γ=1, $\beta_{\min}$=0.2, β_0=1, k=3
DUFA	2018	α=0.3, γ=0.9, d_D=0.9, P=3
OBLPSOGD	2018	P_0=0.3, α=3.2, k=15, σ=0.3, w_{min}=0.4, w_{max}=0.9
AGWO	2018	$r_1\in$[0,1], $r_2\in$[0,1], $rand\in$[0,1]
IRCGA	2019	K=5, k=5, S=5, p_c=1, p_m=1, M=10^{10}
AHCS	2019	w_{max}=0.7, w_{min}=0.4, s=0.8, T_0=10, λ=1.2, M=10^8

and other seven improved CSs are shown in TABLE 10, and the comparison results of AHCS and seven other improved algorithms are shown in TABLE 12.

In TABLE 8, D is the number of decision variables, I is the number of inequality constraints, and E is the number of equality constraints.

As seen in TABLES 9 and 11, the algorithm proposed in this paper can converge with high precision for most of the test functions; while other intelligent methods have some cases where the convergence accuracy of the test function is low.

As seen in TABLE 10, compared to the other seven improved CSs on CEC 2017 30-dimensional optimization problems, the number of winning cases of t-test values for AHCS is greater than the number of lost cases of t-test values, and the Friedman average ranking of AHCS on CEC 2017 30-dimensional optimization problems is better than the other seven improved CSs. Therefore, the performance of AHCS is significantly better than the other seven improved CSs.

As seen in TABLE 12, compared to the other seven improved algorithms on CEC 2017 30-dimensional optimization problems, the number of winning cases of t-test values for AHCS is greater than the number of lost cases of t-test values, and the Friedman average ranking of AHCS on CEC 2017 30-dimensional optimization problems is better than the other seven improved algorithms. Therefore, the performance of AHCS is significantly better than the other seven improved algorithms.

VI. APPLICATION STUDIES ON TWO ENGINEERING CASES

In order to verify the effectiveness of AHCS in solving constrained practical engineering optimization problems, the parameter optimization problem of the reducer in literature [62] and the optimization of the cantilever beam parameters in literature [63] are selected. AHCS and other improved algorithms in the literature are used to optimize the structural parameters of the two engineering cases.

A. REDUCER DESIGN PROBLEM

The design problem of the reducer is a classic constrained optimization problem proposed by the famous scholar Mezura-Montes. The structure of the reducer is shown in FIGURE 2.

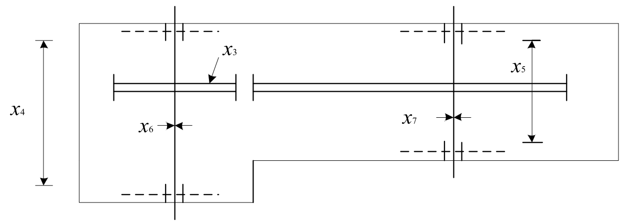

FIGURE 2. The reducer design structure.

There are 7 variables and 11 constraints in the design optimization problem of the reducer. The mathematical model of the design optimization problem is as follows:

$$\begin{aligned}\min f(x) = {} & 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 \\ & - 43.0934) - 1.508x_1(x_6^2 + x_7^2) \\ & + 7.477(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \end{aligned} \tag{24}$$

$$s.t.\begin{cases} g_1(x) = 27/x_1x_2^2x_3 - 1 \le 0 \\ g_2(x) = 397.5/x_1x_2^2x_3^2 - 1 \le 0 \\ g_3(x) = 1.93x_4^3/x_2x_3x_6^4 - 1 \le 0 \\ g_4(x) = 1.93x_5^3/x_2x_3x_7^4 - 1 \le 0 \\ g_5(x) = [(745x_4/x_2x_3)^2 + 16.9*10^6]^{1/2}/110x_6^3 - 1 \le 0 \\ g_6(x) = [(745x_5/x_2x_3)^2 + 157.5*10^6]^{1/2}/85x_7^3 - 1 \le 0 \\ g_7(x) = x_2x_3/40 - 1 \le 0 \\ g_8(x) = 5x_2/x_1 - 1 \le 0 \\ g_9(x) = x_1/12x_2 - 1 \le 0 \\ g_{10}(x) = (1.5x_6 + 1.9)/x_4 - 1 \le 0 \\ g_{11}(x) = (1.1x_7 + 1.9)/x_5 - 1 \le 0 \end{cases} \tag{25}$$

In Eqs. (24) and (25), the range of variables x_1-x_7 are $2.6\le x_1 \le3.6$, $0.7\le x_2 \le0.8$, $17\le x_3 \le28$, $7.3\le x_4 \le8.3$, $7.3\le x_5 \le8.3$, $2.9\le x_6 \le3.9$, and $5.0\le x_7 \le5.5$.

In order to verify the feasibility of AHCS in solving the design problem of constrained reducer, AHCS is compared with seven improved CSs (BCS [1], ICS [48], CS-EO [20], DCS [12], RCS [22], MCS [28], CV 1.0 [53]). In addition, HCS is compared to other seven algorithms (EOCSO [64], PSO-DE [65], MBA [66], HEAA [67],

TABLE 6. The statistical results of AHCS and seven other algorithms on 30 benchmark functions.

Function	Evaluation indicator	Algorithm							
		AHCS	ICSO	MDCSO	NAFA	DUFA	OBLPSOGD	AGWO	IRCGA
f_1	*MEAN*	0.00E+00	0.00E+00	0.00E+00	2.90E+03	5.45E-02	0.00E+00	0.00E+00	4.65E+01
	SD	0.00E+00	0.00E+00	0.00E+00	8.36E+02	4.97E-03	0.00E+00	0.00E+00	7.78E+00
f_2	*MEAN*	0.00E+00	0.00E+00	0.00E+00	3.19E+01	6.31E+03	0.00E+00	0.00E+00	2.92E+00
	SD	0.00E+00	0.00E+00	0.00E+00	3.72E+00	1.34E+04	0.00E+00	0.00E+00	2.51E-01
f_3	*MEAN*	-1.00E+00	-1.00E+00	-1.00E+00	-7.72E-01	-4.00E-01	-1.0E+00	-8.00E-01	-1.00E+00
	SD	0.00E+00	0.00E+00	0.00E+00	4.34E-01	5.48E-01	0.00E+00	4.47E-01	2.42E-09
f_4	*MEAN*	6.16E-33	2.01E+00	1.26E-05	3.02E+03	5.11E-02	8.73E-02	7.05E+00	5.19E+01
	SD	1.38E-33	3.79E-01	2.16E-05	1.36E+02	6.28E-03	9.48E-02	1.2E-01	7.63E+00
f_5	*MEAN*	0.00E+00	0.00E+00	0.00E+00	2.85E+02	7.18E-01	0.00E+00	0.00E+00	7.67E+00
	SD	0.00E+00	0.00E+00	0.00E+00	9.77E+01	8.16E-02	0.00E+00	0.00E+00	2.39E+00
f_6	*MEAN*	0.00E+00	0.00E+00	0.00E+00	2.31E-01	4.71E-03	0.00E+00	0.00E+00	2.03E-04
	SD	0.00E+00	0.00E+00	0.00E+00	1.41E-01	1.57E-03	0.00E+00	0.00E+00	6.98E-05
f_7	*MEAN*	0.00E+00	3.65E+00	9.08E-06	4.96E+01	5.30E-02	0.00E+00	2.63E+01	5.18E-01
	SD	0.00E+00	2.20E+00	2.00E-05	1.25E+01	5.05E-03	0.00E+00	2.19E+00	1.00E-01
f_8	*MEAN*	0.00E+00	0.00E+00	0.00E+00	1.85E+01	9.52E-02	2.47E-03	0.00E+00	6.39E+00
	SD	0.00E+00	0.00E+00	0.00E+00	2.01E+00	7.57E-03	1.03E-03	0.00E+00	8.69E-01
f_9	*MEAN*	0.00E+00	0.00E+00	0.00E+00	1.14E+09	9.15E-02	3.99E-01	0.00E+00	1.80E+01
	SD	0.00E+00	0.00E+00	0.00E+00	1.00E+09	8.08E-03	1.08E-01	0.00E+00	2.45E+00
f_{10}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	3.56E-05	7.71E-10	0.00E+00	0.00E+00	1.79E-12
	SD	0.00E+00	0.00E+00	0.00E+00	5.22E-05	5.46E-10	0.00E+00	0.00E+00	9.83E-13
f_{11}	*MEAN*	-5.00E+01	-5.00E+01	-5.00E+01	-3.18E-01	-5.00E+01	-5.00E+01	-1.33E+01	-5.00E+01
	SD	0.00E+00	1.67E-03	8.06E-04	4.68E+01	3.34E-05	0.00E+00	1.89E+01	4.17E-04
f_{12}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	1.23E+02	3.37E-01	1.54E-01	1.88E-293	2.25E+00
	SD	0.00E+00	0.00E+00	0.00E+00	5.48E+01	6.13E-02	1.28E-01	0.00E+00	9.11E-01
f_{13}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	5.61E-01	1.90E-01	0.00E+00	1.30E-05	1.57E-11
	SD	0.00E+00	0.00E+00	0.00E+00	6.90E-01	2.60E-01	0.00E+00	2.07E-05	2.46E-11
f_{14}	*MEAN*	-2.10E+02	-2.10E+02	-2.10E+02	1.21E+02	-2.00E+00	-2.10E+02	5.08E+00	-2.09E+02
	SD	4.07E-13	3.06E-01	5.61E-01	1.38E+02	1.74E-09	2.05E-03	4.93E+00	4.64E-01
f_{15}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	2.21E+02	1.58E+02	0.00E+00	0.00E+00	1.43E+02
	SD	0.00E+00	0.00E+00	0.00E+00	1.72E+01	3.29E+01	0.00E+00	0.00E+00	6.80E+01
f_{16}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	2.64E+01	1.61E+01	0.00E+00	0.00E+00	1.53E+00
	SD	0.00E+00	0.00E+00	0.00E+00	6.37E+00	6.50E+00	0.00E+00	0.00E+00	1.44E-01
f_{17}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	2.16E+01	2.81E+01	2.29E-02	0.00E+00	1.84E+01
	SD	0.00E+00	0.00E+00	0.00E+00	1.33E+00	2.04E+00	2.02E-02	0.00E+00	3.09E+00
f_{18}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	2.15E-01	9.99E-08	0.00E+00	0.00E+00	4.94E-08
	SD	0.00E+00	0.00E+00	0.00E+00	2.86E-01	5.36E-08	0.00E+00	0.00E+00	6.65E-08
f_{19}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	1.09E-01	7.47E-08	0.00E+00	0.00E+00	3.21E-08
	SD	0.00E+00	0.00E+00	0.00E+00	1.23E-01	6.13E-08	0.00E+00	0.00E+00	2.52E-08
f_{20}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	6.12E-02	2.30E-08	0.00E+00	0.00E+00	8.60E+00
	SD	0.00E+00	0.00E+00	0.00E+00	8.14E-02	2.06E-08	0.00E+00	0.00E+00	3.90E-09
f_{21}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	7.71E+07	1.35E+05	0.00E+00	0.00E+00	2.28E+06
	SD	0.00E+00	0.00E+00	0.00E+00	3.76E+07	3.98E+04	0.00E+00	0.00E+00	1.10E+06
f_{22}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	8.21E-05	2.79E-06	0.00E+00	0.00E+00	3.48E-08
	SD	0.00E+00	0.00E+00	0.00E+00	1.42E-04	1.81E-06	0.00E+00	0.00E+00	2.60E-08
f_{23}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	3.25E+00	6.06E-01	0.00E+00	0.00E+00	2.02E-01
	SD	0.00E+00	0.00E+00	0.00E+00	6.66E-01	4.90E-02	0.00E+00	0.00E+00	3.76E-02
f_{24}	*MEAN*	3.44E-09	2.80E+01	2.41E-04	7.49E+00	4.36E+01	6.10E+00	2.90E+01	1.58E+02
	SD	2.52E-09	4.33E-01	4.29E-04	2.72E+04	2.84E+01	5.10E+00	1.38E-02	7.05E+01
f_{25}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	1.89E+02	1.96E+02	0.00E+00	0.00E+00	1.47E+02
	SD	0.00E+00	0.00E+00	0.00E+00	1.37E+01	2.98E+01	0.00E+00	0.00E+00	3.32E+01
f_{26}	*MEAN*	8.88E-16	8.88E-16	8.88E-16	9.98E+00	1.84E+01	8.88E-16	8.88E-16	3.44E+00
	SD	0.00E+00	0.00E+00	0.00E+00	4.37E-01	3.11E-01	0.00E+00	0.00E+00	2.81E-01
f_{27}	*MEAN*	2.07E-33	4.09E+00	6.76E-02	2.79E+06	4.19E+01	7.56E+00	2.33E+01	1.63E+01
	SD	2.83E-33	7.29E-01	2.12E-02	2.63E+06	1.17E+01	1.33E+00	2.93E+00	7.74E+00
f_{28}	*MEAN*	1.75E-33	2.30E+00	2.83E-04	2.93E+07	8.27E+01	1.52E+00	2.98E+00	3.24E+02
	SD	5.51E-34	9.21E-01	2.99E-04	2.50E+07	8.64E+00	3.13E-01	4.28E-02	4.36E+02
f_{29}	*MEAN*	0.00E+00	0.00E+00	0.00E+00	1.46E+01	4.62E+00	0.00E+00	0.00E+00	1.24E+01
	SD	0.00E+00	0.00E+00	0.00E+00	7.31E-01	1.09E+00	0.00E+00	0.00E+00	5.56E+01
f_{30}	*MEAN*	4.19E-03	4.11E+00	1.66E-04	3.37E+01	1.38E+02	6.40E-03	2.94E+01	7.31E+01
	SD	2.71E-03	5.02E+00	2.33E-04	5.95E+00	2.11E+01	2.49E-03	9.23E-01	6.07E+01

TABLE 7. The comparison results of AHCS and seven improved algorithms on 30 benchmark functions.

Function	Evaluation indicator	Algorithm							
		AHCS	ICSO	MDCSO	NAFA	DUFA	OBLPSOGD	AGWO	IRCGA
f_1	*Rank*	3	3	3	8	6	3	3	7
	T_{val}	/	0.00E+00	0.00E+00	1.90E+01	6.01E+01	0.00E+00	0.00E+00	3.27E+01
	VS	/	=	=	-	-	=	=	-
f_2	*Rank*	3	3	3	7	8	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	4.70E+01	2.58E+00	0.00E+00	0.00E+00	6.37E+01
	VS	/	=	=	-	-	=	=	-
f_3	*Rank*	3	3	3	7	8	3	6	3
	T_{val}	/	0.00E+00	0.00E+00	2.88E+00	6.00E+00	0.00E+00	2.45E+00	0.00E+00
	VS	/	=	=	-	-	=	-	=
f_4	*Rank*	1	5	2	8	3	4	6	7
	T_{val}	/	2.90E+01	3.20E+00	1.22E+02	4.46E+01	5.04E+00	3.22E+02	3.73E+01
	VS	/	-	-	-	-	-	-	-
f_5	*Rank*	3	3	3	8	6	3	3	7
	T_{val}	/	0.00E+00	0.00E+00	1.60E+01	4.82E+01	0.00E+00	0.00E+00	1.76E+01
	VS	/	=	=	-	-	=	=	-
f_6	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	8.97E+00	1.64E+01	0.00E+00	0.00E+00	1.59E+01
	VS	/	=	=	-	-	=	=	-
f_7	*Rank*	1.5	6	3	8	4	1.5	7	5
	T_{val}	/	9.09E+00	2.49E+00	2.17E+01	5.75E+01	0.00E+00	6.58E+01	2.84E+01
	VS	/	-	-	-	-	=	-	-
f_8	*Rank*	2.5	2.5	2.5	8	6	5	2.5	7
	T_{val}	/	0.00E+00	0.00E+00	5.04E+01	6.89E+01	1.31E+01	0.00E+00	4.03E+01
	VS	/	=	=	-	-	-	=	-
f_9	*Rank*	2.5	2.5	2.5	8	5	6	2.5	7
	T_{val}	/	0.00E+00	0.00E+00	6.24E+00	6.20E+01	2.02E+01	0.00E+00	4.02E+01
	VS	/	=	=	-	-	-	=	-
f_{10}	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	3.74E+00	7.73E+00	0.00E+00	0.00E+00	9.96E+00
	VS	/	=	=	-	-	=	=	-
f_{11}	*Rank*	3.5	3.5	3.5	8	3.5	3.5	7	3.5
	T_{val}	/	0.00E+00	0.00E+00	5.81E+00	0.00E+00	0.00E+00	1.06E+01	0.00E+00
	VS	/	=	=	-	=	=	-	=
f_{12}	*Rank*	2	2	2	8	6	5	4	7
	T_{val}	/	0.00E+00	0.00E+00	1.23E+01	3.01E+01	6.59E+00	8.47E-278	1.35E+01
	VS	/	=	=	-	-	-	-	-
f_{13}	*Rank*	2.5	2.5	2.5	8	7	2.5	6	5
	T_{val}	/	0.00E+00	0.00E+00	4.45E+00	4.00E+00	0.00E+00	3.44E+00	3.50E+00
	VS	/	=	=	-	-	=	-	-
f_{14}	*Rank*	2.5	2.5	2.5	8	6	2.5	7	5
	T_{val}	/	0.00E+00	0.00E+00	1.31E+01	6.55E+11	0.00E+00	2.39E+02	1.18E+01
	VS	/	=	=	-	-	=	-	-
f_{15}	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	7.04E+01	2.63E+01	0.00E+00	0.00E+00	1.15E+01
	VS	/	=	=	-	-	=	=	-
f_{16}	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	2.27E+01	1.36E+01	0.00E+00	0.00E+00	5.82E+01
	VS	/	=	=	-	-	=	=	-
f_{17}	*Rank*	2.5	2.5	2.5	7	8	5	2.5	6
	T_{val}	/	0.00E+00	0.00E+00	8.90E+01	7.54E+01	6.21E+00	0.00E+00	3.26E+01
	VS	/	=	=	-	-	-	=	-
f_{18}	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	4.12E+00	1.02E+01	0.00E+00	0.00E+00	4.07E+00
	VS	/	=	=	-	-	=	=	-
f_{19}	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	4.85E+00	6.67E+00	0.00E+00	0.00E+00	6.98E+00
	VS	/	=	=	-	-	=	=	-
f_{20}	*Rank*	3	3	3	7	6	3	3	8
	T_{val}	/	0.00E+00	0.00E+00	4.12E+00	6.12E+00	0.00E+00	0.00E+00	1.21E+10
	VS	/	=	=	-	-	=	=	-

TABLE 7. *(Continued.)* **The comparison results of AHCS and seven improved algorithms on 30 benchmark functions.**

f_{21}	*Rank*	3	3	3	8	6	3	3	7
	T_{val}	/	0.00E+00	0.00E+00	1.12E+01	1.86E+01	0.00E+00	0.00E+00	1.14E+01
	VS	/	=	=	-	-	=	=	-
f_{22}	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	3.17E+00	8.44E+00	0.00E+00	0.00E+00	7.33E+00
	VS	/	=	=	-	-	=	=	-
f_{23}	*Rank*	3	3	3	8	7	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	2.67E+01	6.77E+01	0.00E+00	0.00E+00	2.94E+01
	VS	/	=	=	-	-	=	=	-
f_{24}	*Rank*	1	5	2	4	7	3	6	8
	T_{val}	/	3.54E+02	3.08E+00	1.51E-03	8.41E+00	6.55E+00	1.15E+04	1.23E+01
	VS	/	-	-	-	-	-	-	-
f_{25}	*Rank*	3	3	3	7	8	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	7.56E+01	3.60E+01	0.00E+00	0.00E+00	2.43E+01
	VS	/	=	=	-	-	=	=	-
f_{26}	*Rank*	3	3	3	7	8	3	3	6
	T_{val}	/	0.00E+00	0.00E+00	1.25E+02	3.24E+02	0.00E+00	0.00E+00	6.71E+01
	VS	/	=	=	-	-	=	=	-
f_{27}	*Rank*	1	3	2	8	7	4	6	5
	T_{val}	/	3.07E+01	1.75E+01	5.81E+00	1.96E+01	3.11E+01	4.36E+01	1.15E+01
	VS	/	-	-	-	-	-	-	-
f_{28}	*Rank*	1	4	2	8	6	3	5	7
	T_{val}	/	1.37E+01	5.18E+00	6.42E+00	5.24E+01	2.66E+01	3.81E+02	4.07E+00
	VS	/	-	-	-	-	-	-	-
f_{29}	*Rank*	3	3	3	8	6	3	3	7
	T_{val}	/	0.00E+00	0.00E+00	1.09E+02	2.32E+01	0.00E+00	0.00E+00	1.22E+00
	VS	/	=	=	-	-	=	=	-
f_{30}	*Rank*	2	4	1	6	8	3	5	7
	T_{val}	/	4.48E+00	-8.10E+00	3.10E+01	3.58E+01	3.29E+00	1.74E+02	6.60E+00
	VS	/	-	+	-	-	-	-	-
Average Ranking		2.55	3.20	2.70	7.60	6.48	3.30	4.02	6.15
Final Ranking		1	3	2	8	7	4	5	6
w/t/l		/	6/24/0	5/24/1	30/0/0	29/1/0	9/21/0	11/19/0	28/2/0

Note: "/" indicates that the same items do not compare.

HGA [68], SSBA [69], and GOKA [70]). The optimization results of AHCS and the seven improved CSs in the literature are shown in TABLE 13. The optimization results of AHCS and other seven algorithms in the literature are shown in TABLE 14.

In order to verify that the solution quality of the AHCS is better than other algorithms, each algorithm is run 30 times, and the maximum runtime for each run is *maxruntime*=20 seconds, the population size is $N = 40$, and the penalty factor is $M = 10^8$. The optimal value and the worst value of the various algorithms after 30 results are counted, and the average optimal value and standard deviation of the optimal operating objective function for 30 times are calculated. The statistical results are shown in TABLE 15.

In TABLE 15, Best represents the optimal value among the 30 results, Worst represents the worst value of the 30 run results, Mean represents the average value of the 30 results, and Std represents standard deviation the 30 results.

As can be seen from TABLES 13 and 14 that the optimized value of AHCS solution is not worse than the other seven improved CSs and seven other algorithms involved in the comparison. In addition, as can be seen from TABLE 15, the optimal value, mean value, and worst value of AHCS are better than the other 14 algorithms that participate in the comparison. Therefore, the quality of AHCS solution is significantly better than other algorithms.

B. CANTILEVER BEAM DESIGN PROBLEM

The goal of the cantilever beam design optimization problem is to determine the optimal combination of five different cross-sectional areas to minimize the volume of the cantilever beam. The design problem has 10 variables, namely five width variables $b_i(i = 1, 2, \ldots, 5)$ and five height variables $h_i(i = 1, 2, \ldots, 5)$. In addition, there are 11 constraints. The free end of the cantilever beam exerts an external force $p = 50000$N, the maximum allowable stress at the left end of each section is $\sigma_{\max} = 14000\text{N}/cm^2$, the material elastic modulus E is 200 GPa, and the length $l_i(i = 1, 2, \ldots, 5)$ is 100 cm. The maximum allowable disturbance $y_{max} = 2.715cm$, and the aspect ratio of each cross section is limited to 20 cm or less. The cantilever beam design structure with discrete rectangular sections is shown in FIGURE 3.

The design variables for the cantilever beam design optimization problem are as follows:

$$\begin{aligned} X &= [b_1, h_1, b_2, h_2, b_3, h_3, b_4, h_4, b_5, h_5]^T \\ &= [x_1, x_2, \ldots, x_{10}]^T \end{aligned} \tag{26}$$

TABLE 8. Details of 28 CEC 2017 optimization problems.

Problem	Name/Search Range	Type of Objective	Number of Constraints	
			E	*I*
F1	C01 $[-100, 100]^D$	Non Separable	0	1 Separable
F2	C02 $[-100, 100]^D$	Non Separable Rotated	0	1 Non Separable, Rotated
F3	C03 $[-100, 100]^D$	Non Separable	1 Separable	1 Separable
F4	C04 $[-10,10]^D$	Separable	0	2 Separable
F5	C05 $[-10,10]^D$	Non Separable	0	2 Non Separable, Rotated
F6	C06 $[-20,20]^D$	Separable	6	0 Separable
F7	C07 $[-20,20]^D$	Separable	2 Separable	0
F8	C08 $[-100, 100]^D$	Separable	2 Non Separable	0
F9	C09 $[-10, 10]^D$	Separable	2 Non Separable	0
F10	C10 $[-100, 100]^D$	Separable	2 Non Separable	0
F11	C11 $[-100, 100]^D$	Separable	1 Non Separable	1 Non Separable
F12	C12 $[-100, 100]^D$	Separable	0	2 Separable
F13	C13 $[-100, 100]^D$	Non Separable	0	3 Separable
F14	C14 $[-100, 100]^D$	Non Separable	1 Separable	1 Separable
F15	C15 $[-100, 100]^D$	Separable	1	1
F16	C16 $[-100, 100]^D$	Separable	1 Non Separable	1 Separable
F17	C17 $[-100, 100]^D$	Non Separable	1 Non Separable	1 Separable
F18	C18 $[-100, 100]^D$	Separable	1	2 Non Separable
F19	C19 $[-50, 50]^D$	Separable	0	2 Non Separable
F20	C20 $[-100, 100]^D$	Non Separable	0	2
F21	C21 $[-100, 100]^D$	Rotated	0	2 Rotated
F22	C22 $[-100, 100]^D$	Rotated	0	3 Rotated
F23	C23 $[-100, 100]^D$	Rotated	1 Rotated	1 Rotated
F24	C24 $[-100, 100]^D$	Rotated	1 Rotated	1 Rotated
F25	C25 $[-100, 100]^D$	Rotated	1 Rotated	1 Rotated
F26	C26 $[-100, 100]^D$	Rotated	1 Rotated	1 Rotated
F27	C27 $[-100, 100]^D$	Rotated	1 Rotated	2 Rotated
F28	C28 $[-50, 50]^D$	Rotated	0	2 Rotated

TABLE 9. The statistical results among CSs on CEC 2017 30-dimensional optimization problems.

Problem	Evaluation indicator	Algorithm							
		AHCS	BCS	ICS	CS-EO	DCS	RCS	MCS	CV 1.0
F1	*MEAN*	6.44E-21	5.15E+03	5.62E+04	2.43E+03	5.84E+01	1.48E+06	7.45E-02	2.84E+04
	SD	1.08E-20	2.84E+03	1.16E+04	1.69E+03	4.22E+01	1.52E+06	1.16E-01	1.31E+04
F2	*MEAN*	5.16E-21	3.36E+03	4.57E+04	2.63E+03	8.54E+01	2.80E+05	8.33E+00	2.57E+04
	SD	1.05E-20	1.37E+03	8.26E+03	3.84E+03	6.44E+01	2.57E+05	1.76E+01	1.15E+04
F3	*MEAN*	2.10E+07	2.29E+07	5.00E+08	1.03E+08	5.42E+05	6.06E+14	8.08E+07	4.11E+07
	SD	4.50E+07	4.46E+07	1.11E+09	1.66E+06	2.12E+05	1.35E+15	4.50E+07	5.48E+07
F4	*MEAN*	1.47E+02	2.59E+02	7.88E+02	3.34E+00	2.30E+02	1.50E+03	2.82E+02	5.32E+02
	SD	1.72E+01	2.50E+01	1.17E+02	5.84E+00	1.70E+01	1.55E+02	6.81E+00	9.94E+01
F5	*MEAN*	7.97E-01	2.16E+01	1.87E+04	1.78E+01	2.16E+01	1.00E+07	3.54E+00	6.16E+05
	SD	1.78E+00	1.86E+00	9.02E+03	1.36E+00	2.48E+00	3.02E+06	5.60E+00	3.44E+05
F6	*MEAN*	1.55E+09	7.64E+08	1.33E+09	2.88E+10	1.79E+09	8.92E+09	4.41E+10	3.39E+08
	SD	2.66E+09	5.13E+07	1.20E+09	9.06E+09	4.16E+08	3.56E+09	1.37E+10	1.49E+08
F7	*MEAN*	5.42E+11	1.91E+12	2.07E+13	1.36E+11	1.28E+10	2.74E+12	9.05E+12	3.86E+12
	SD	3.10E+11	2.93E+12	2.02E+12	1.83E+10	3.54E+10	1.66E+12	1.50E+13	3.12E+12
F8	*MEAN*	-2.82E-04	4.97E+15	8.02E+17	1.21E+14	2.27E+11	3.18E+18	1.77E+14	4.11E+16
	SD	9.12E-07	4.11E+15	3.02E+17	2.60E+14	3.05E+11	4.22E+18	3.97E+14	1.71E+16
F9	*MEAN*	5.70E-01	1.41E+02	6.16E+09	1.17E+03	9.82E+00	9.06E+23	1.05E-01	4.11E+09
	SD	5.24E-01	4.32E+01	8.06E+09	2.62E+03	1.01E+01	5.03E+23	2.41E-01	4.33E+09
F10	*MEAN*	-1.02E-04	1.34E+17	1.85E+17	3.59E+05	1.70E-04	1.67E+19	-1.01E-04	8.35E+17
	SD	3.54E-07	5.98E+16	1.36E+17	7.95E+05	6.86E-05	1.03E+19	4.41E-06	5.18E+17
F11	*MEAN*	3.52E+07	4.20E+15	2.40E+17	2.02E+12	1.01E+13	4.39E+18	4.76E+12	1.87E+17
	SD	4.57E+07	3.84E+15	4.85E+16	3.55E+12	1.19E+13	2.13E+18	8.92E+12	5.20E+16
F12	*MEAN*	1.06E+02	1.78E+02	6.54E+10	2.19E+02	2.27E+02	2.40E+17	2.09E+02	9.54E+16
	SD	3.81E+01	1.31E+01	5.00E+10	2.46E+01	1.84E+01	8.29E+16	1.46E+01	4.23E+16
F13	*MEAN*	9.43E+09	1.11E+14	1.08E+13	6.75E-08	4.50E+10	3.24E+17	1.47E+12	8.30E+16
	SD	7.49E+09	4.34E+13	2.58E+12	1.51E-07	1.17E+10	1.10E+17	2.86E+12	3.69E+16
F14	*MEAN*	2.00E+00	2.21E+00	1.70E+11	2.21E+00	2.74E+00	6.48E+17	2.26E+00	2.64E+17
	SD	1.49E-01	7.21E-02	1.43E+11	7.25E-03	1.09E+00	1.12E+17	8.20E-02	1.07E+17
F15	*MEAN*	1.93E+01	1.74E+01	1.62E+01	1.68E+01	1.70E+01	2.99E+17	1.60E+01	1.08E+17
	SD	3.58E+00	1.40E+00	1.72E+00	4.76E+00	1.46E+00	2.76E+16	1.49E+00	7.79E+16
F16	*MEAN*	1.69E+02	1.97E+02	1.94E+02	1.68E+02	1.71E+02	2.31E+17	1.51E+02	1.31E+17
	SD	2.36E+01	1.33E+01	9.19E+00	3.13E+01	1.88E+01	7.67E+16	1.30E+01	4.71E+16
F17	*MEAN*	9.61E+10	9.61E+10	9.97E+10	9.61E+10	9.61E+10	3.69E+17	9.61E+10	1.86E+17
	SD	9.03E-03	8.55E-01	8.02E+09	1.46E+01	1.29E+03	9.44E+16	4.51E+00	1.03E+17
F18	*MEAN*	4.39E+01	1.02E+21	2.89E+24	8.91E+10	1.01E+09	4.77E+29	3.69E+01	9.37E+27
	SD	8.34E+00	2.23E+21	2.43E+24	1.99E+11	1.81E+09	8.56E+28	7.05E-01	9.29E+27
F19	*MEAN*	1.83E+17	1.84E+17	1.85E+17	1.83E+17	1.85E+17	1.85E+17	1.84E+17	1.84E+17
	SD	5.77E+01	1.45E+14	2.55E+13	1.81E+06	6.89E+13	9.14E+13	1.03E+15	1.09E+14
F20	*MEAN*	2.57E+00	2.53E+00	7.79E+00	6.32E+00	4.05E+00	1.31E+01	9.34E+00	5.16E+00
	SD	2.06E-01	2.05E-01	8.10E-01	5.88E-01	1.43E-01	9.81E-01	3.22E-01	7.54E-01
F21	*MEAN*	1.28E+02	2.02E+02	1.31E+10	2.20E+02	2.30E+02	2.42E+17	3.22E+15	8.34E+16
	SD	2.68E+01	1.22E+01	1.13E+10	1.92E+01	2.79E+01	2.53E+16	7.20E+15	5.68E+16
F22	*MEAN*	2.23E+10	3.95E+14	1.17E+13	7.97E+11	1.62E+11	2.85E+17	6.90E+11	4.59E+16
	SD	2.07E+10	1.03E+14	1.58E+12	1.12E+11	5.58E+10	5.82E+16	8.43E+10	3.03E+16
F23	*MEAN*	1.79E+00	2.25E+00	3.71E+10	2.16E+00	2.27E+00	3.30E+17	2.02E+00	8.48E+16
	SD	8.39E-02	5.78E-02	2.59E+10	7.32E-02	5.68E-02	1.29E+17	1.13E-01	9.13E+16
F24	*MEAN*	1.62E+01	1.68E+00	1.68E+01	1.87E+01	1.66E+01	2.02E+17	1.74E+01	4.47E+16
	SD	2.81E+00	1.72E+00	1.72E+00	1.40E+00	1.59E+00	5.42E+16	1.40E+00	1.28E+16
F25	*MEAN*	1.65E+02	2.10E+02	1.99E+02	1.75E+02	1.83E+02	1.76E+17	1.61E+02	5.49E+16
	SD	8.78E+00	9.25E+00	1.63E+01	1.26E+01	1.26E+01	4.76E+16	1.00E+01	3.89E+16
F26	*MEAN*	9.61E+10	9.61E+10	9.61E+10	9.61E+10	9.61E+10	2.33E+17	9.61E+10	6.27E+16
	SD	7.20E-03	8.85E-01	4.33E-03	1.67E+01	8.30E+01	8.58E+16	3.87E+01	2.05E+16
F27	*MEAN*	5.82E+01	1.90E+21	3.17E+25	2.09E+13	1.27E+13	4.08E+29	4.68E+13	4.69E+27
	SD	4.81E+01	2.73E+21	2.32E+25	3.91E+13	1.96E+13	7.74E+28	1.04E+14	4.94E+27
F28	*MEAN*	1.84E+17	1.85E+17	1.85E+17	1.85E+17	1.85E+17	1.85E+17	1.85E+17	1.84E+17
	SD	2.23E+14	4.34E+13	5.29E+13	5.73E+13	2.09E+13	1.59E+14	8.53E+13	6.34E+13

TABLE 10. The comparison results among CSs on CEC 2017 30-dimensional optimization problems.

Function	Evaluation indicator	Algorithm							
		AHCS	BCS	ICS	CS-EO	DCS	RCS	MCS	CV 1.0
F1	*Rank*	1	5	7	4	3	8	2	6
	T_{val}	/	9.93E+00	2.65E+01	7.88E+00	7.58E+00	5.33E+00	3.52E+00	1.19E+01
	VS	/	-	-	-	-	-	-	-
F2	*Rank*	1	5	7	4	3	8	2	6
	T_{val}	/	1.34E+01	3.03E+01	3.75E+00	7.26E+00	5.97E+00	2.59E+00	1.22E+01
	VS	/	-	-	-	-	-	-	-
F3	*Rank*	2	3	7	6	1	8	5	4
	T_{val}	/	1.64E-01	2.36E+00	9.97E+00	-2.49E+00	2.46E+00	5.15E+00	1.55E+00
	VS	/	-	-	-	+	-	-	-
F4	*Rank*	2	4	7	1	3	8	5	6
	T_{val}	/	2.02E+01	2.97E+01	-4.33E+01	1.88E+01	4.75E+01	4.00E+01	2.09E+01
	VS	/	-	-	+	-	-	-	-
F5	*Rank*	1	4.5	6	3	4.5	8	2	7
	T_{val}	/	4.43E+01	1.14E+01	4.16E+01	3.73E+01	1.81E+01	2.56E+00	9.81E+00
	VS	/	-	-	-	-	-	-	-
F6	*Rank*	4	2	3	7	5	6	8	1
	T_{val}	/	-1.62E+00	-4.13E-01	1.58E+01	4.88E-01	9.08E+00	1.67E+01	-2.49E+00
	VS	/	+	+	-	-	-	-	+
F7	*Rank*	3	4	8	2	1	5	7	6
	T_{val}	/	2.54E+00	5.40E+01	-7.16E+00	-9.29E+00	7.13E+00	3.11E+00	5.80E+00
	VS	/	-	-	+	+	-	-	-
F8	*Rank*	1	5	7	3	2	8	4	6
	T_{val}	/	6.62E+00	1.45E+01	2.55E+00	4.08E+00	4.13E+00	2.44E+00	1.32E+01
	VS	/	-	-	-	-	-	-	-
F9	*Rank*	2	4	7	5	3	8	1	6
	T_{val}	/	1.78E+01	4.19E+00	2.44E+00	5.01E+00	9.87E+00	-4.42E+00	5.20E+00
	VS	/	-	-	-	-	-	+	-
F10	*Rank*	1	5	6	4	3	8	2	7
	T_{val}	/	1.23E+01	7.45E+00	2.47E+00	2.17E+01	8.88E+00	1.24E+00	8.83E+00
	VS	/	-	-	-	-	-	-	-
F11	*Rank*	1	5	7	2	4	8	3	6
	T_{val}	/	5.99E+00	2.71E+01	3.12E+00	4.65E+00	1.13E+01	2.92E+00	1.97E+01
	VS	/	-	-	-	-	-	-	-
F12	*Rank*	1	2	6	4	5	8	3	7
	T_{val}	/	9.79E+00	7.16E+00	1.36E+01	1.57E+01	1.59E+01	1.38E+01	1.24E+01
	VS	/	-	-	-	-	-	-	-
F13	*Rank*	2	6	5	1	3	8	4	7
	T_{val}	/	1.40E+01	2.29E+01	-6.90E+00	1.40E+01	1.61E+01	2.80E+00	1.23E+01
	VS	/	-	-	+	-	-	-	-
F14	*Rank*	1	2.5	6	2.5	5	8	4	7
	T_{val}	/	6.95E+00	6.51E+00	7.71E+00	3.68E+00	3.17E+01	8.37E+00	1.35E+01
	VS	/	-	-	-	-	-	-	-
F15	*Rank*	6	5	2	3	4	8	1	7
	T_{val}	/	-2.71E+00	-4.28E+00	-2.30E+00	-3.26E+00	5.93E+01	-4.66E+00	7.59E+00
	VS	/	+	+	+	+	-	+	-
F16	*Rank*	3	6	5	2	4	8	1	7
	T_{val}	/	5.66E+00	5.41E+00	-1.40E-01	3.63E-01	1.65E+01	-3.66E+00	1.52E+01
	VS	/	-	-	+	-	-	+	-
F17	*Rank*	3	3	6	3	3	8	3	7
	T_{val}	/	0.00E+00	2.46E+00	0.00E+00	0.00E+00	2.14E+01	0.00E+00	9.89E+00
	VS	/	=	-	=	=	-	=	-
F18	*Rank*	2	5	6	4	3	8	1	7
	T_{val}	/	2.51E+00	6.51E+00	2.45E+00	3.06E+00	3.05E+01	-4.58E+00	5.52E+00
	VS	/	-	-	-	-	-	+	-
F19	*Rank*	1.5	4	7	1.5	7	7	4	4
	T_{val}	/	3.78E+01	4.30E+02	0.00E+00	1.59E+02	1.20E+02	5.32E+00	5.02E+01
	VS	/	-	-	=	-	-	-	-

TABLE 10. ***(Continued.)*The comparison results among CSs on CEC 2017 30-dimensional optimization problems.**

	Rank	2	1	6	5	3	8	7	4
F20	T_{val}	/	-7.54E-01	3.42E+01	3.30E+01	3.23E+01	5.75E+01	9.70E+01	1.81E+01
	VS	/	+	-	-	-	-	-	-
	Rank	1	2	5	3	4	8	6	7
F21	T_{val}	/	1.38E+01	6.35E+00	1.53E+01	1.44E+01	5.24E+01	2.45E+00	8.04E+00
	VS	/	-	-	-	-	-	-	-
	Rank	1	6	5	4	2	8	3	7
F22	T_{val}	/	2.10E+01	4.05E+01	3.73E+01	1.29E+01	2.68E+01	4.21E+01	8.30E+00
	VS	/	-	-	-	-	-	-	-
	Rank	1	4	6	3	5	8	2	7
F23	T_{val}	/	2.47E+01	7.85E+00	1.82E+01	2.59E+01	1.40E+01	8.95E+00	5.09E+00
	VS	/	-	-	-	-	-	-	-
	Rank	2	1	4	6	3	8	5	7
F24	T_{val}	/	-2.41E+01	9.97E-01	4.36E+00	6.79E-01	2.04E+01	2.09E+00	1.91E+01
	VS	/	+	-	-	-	-	-	-
	Rank	2	6	5	3	4	8	1	7
F25	T_{val}	/	1.93E+01	1.01E+01	3.57E+00	6.42E+00	2.03E+01	-1.65E+00	7.73E+00
	VS	/	-	-	-	-	-	+	-
	Rank	3.5	3.5	3.5	3.5	3.5	8	3.5	7
F26	T_{val}	/	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.49E+01	0.00E+00	1.68E+01
	VS	/	=	=	=	=	-	=	-
	Rank	1	5	6	3	2	8	4	7
F27	T_{val}	/	3.81E+00	7.48E+00	2.93E+00	3.55E+00	2.89E+01	2.46E+00	5.20E+00
	VS	/	-	-	-	-	-	-	-
	Rank	1.5	5.5	5.5	5.5	5.5	5.5	5.5	1.5
F28	T_{val}	/	2.41E+01	2.39E+01	2.38E+01	2.45E+01	2.00E+01	2.29E+01	0.00E+00
	VS	/	-	-	-	-	-	-	=
Average Ranking		1.91	4.07	5.75	3.50	3.52	7.70	3.54	6.02
Final Ranking		1	5	6	2	3	8	4	7
w/t/l		/	22/2/4	25/1/2	20/3/5	23/2/3	28/0/0	21/2/5	26/1/1

Note: "/" indicates that the same items do not compare.

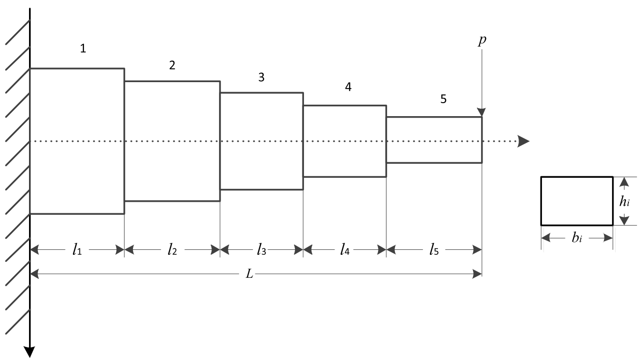

FIGURE 3. **The cantilever beam design structure.**

The mathematical model of cantilever beam structure optimization design can be expressed as follows:

$$\min f(X) = 100 * (x_1x_2+x_3x_4+x_5x_6+x_7x_8+x_9x_{10}) \quad (27)$$

constraint:

(1) Pressure constraint

$$\begin{aligned}
g_1(X) &= 10.7143 - x_1x_2^2/10^3 \leq 0\\
g_2(X) &= 8.5714 - x_3x_4^2/10^3 \leq 0\\
g_3(X) &= 6.4286 - x_5x_6^2/10^3 \leq 0\\
g_4(X) &= 4.2957 - x_7x_8^2/10^3 \leq 0\\
g_5(X) &= 2.1428 - x_9x_{10}^2/10^3 \leq 0\\
g_6(X) &= 10^4 * (244/x_1x_2^3 + 148/x_3x_4^3 + 76/x_5x_6^3\\
&\quad + 28/x_7x_8^3 + 4/x_9x_{10}^3) - 10.8611 \leq 0 \quad (28)
\end{aligned}$$

(2) Geometric constraint

$$\begin{aligned}
g_7(X) &= x_2 - 20x_1 \leq 0\\
g_8(X) &= x_4 - 20x_3 \leq 0\\
g_9(X) &= x_6 - 20x_5 \leq 0\\
g_{10}(X) &= x_8 - 20x_7 \leq 0\\
g_{11}(X) &= x_{10} - 20x_9 \leq 0 \quad (29)
\end{aligned}$$

In Eqs. (26)-(29), the range of design variables for the cantilever beam is $1\leq x_i \leq 5 (i = 1, 3, 5, 7, 9)$, $30\leq x_i \leq 65$ $(i = 2, 4, 6, 8, 10)$.

In order to verify the feasibility of AHCS in solving the design problem of constrained cantilever beam, AHCS is compared with seven improved CSs (BCS [1], ICS [48], CS-EO [20], DCS [12], RCS [22], MCS [28], CV 1.0 [53]). In addition, AHCS is compared to seven other algorithms (SDGAMINLS [63], MPNN [71], SUMT [72], RNES [73], CAD [74], GA [75], GA-APM [76]). The optimization results of AHCS and the seven improved CSs in the

TABLE 11. The statistical results among AHCS and other seven algorithms on CEC 2017 30-dimensional optimization problems.

Problem	Evaluation indicator	Algorithm							
		AHCS	ICSO	MDCSO	NAFA	DUFA	OBLPSOGD	AGWO	IRCGA
F1	*MEAN*	6.44E-21	5.27E+01	2.26E+02	4.47E+05	3.11E-01	1.24E+06	4.57E+05	1.65E+03
	SD	1.08E-20	1.49E+01	1.92E+02	4.58E+05	2.40E-02	5.05E+04	2.94E+05	6.44E+02
F2	*MEAN*	5.16E-21	5.18E+02	1.14E+02	1.17E+05	3.13E-01	2.50E+05	8.57E+04	9.54E+02
	SD	1.05E-20	9.05E+02	4.06E+01	7.82E+04	3.36E-02	2.07E+04	3.59E+04	1.51E+02
F3	*MEAN*	2.10E+07	1.03E+08	2.81E+06	2.98E+07	4.40E+06	4.35E+06	1.07E+08	1.18E+08
	SD	4.50E+07	8.57E+05	8.78E+05	5.75E+07	3.51E+06	6.06E+05	3.06E+06	1.44E+07
F4	*MEAN*	1.47E+02	4.36E+02	4.22E+02	7.81E+02	1.03E+03	4.98E+02	8.76E+02	3.10E+02
	SD	1.72E+01	8.08E+01	5.08E+01	3.68E+01	8.64E+01	6.12E+01	4.21E+01	4.65E+01
F5	*MEAN*	7.97E-01	1.49E+02	1.58E+05	1.31E+06	4.31E+01	1.68E+03	1.30E+06	2.48E+02
	SD	1.78E+00	4.74E+01	1.38E+05	2.18E+05	1.82E+00	1.28E+03	2.97E+05	1.70E+02
F6	*MEAN*	1.55E+09	1.15E+09	8.39E+08	5.61E+10	1.08E+09	6.55E+09	5.04E+10	1.38E+10
	SD	2.66E+09	6.79E+08	5.76E+08	3.06E+10	1.54E+08	2.10E+09	6.96E+09	5.15E+09
F7	*MEAN*	5.42E+11	8.51E+13	8.52E+13	4.10E+14	-2.44E+02	8.11E+00	5.26E+14	2.03E+13
	SD	3.10E+11	5.76E+13	1.90E+14	1.45E+14	1.26E+02	1.27E+02	7.30E+13	6.66E+12
F8	*MEAN*	-2.82E-04	6.19E+16	7.55E+16	4.31E+17	3.25E+06	4.24E+11	2.13E+18	2.05E+13
	SD	9.12E-07	1.92E+16	1.28E+16	3.88E+17	6.94E+05	3.71E+11	3.42E+18	1.29E+13
F9	*MEAN*	5.70E-01	6.19E+16	1.49E+15	2.88E+15	5.55E+04	1.45E+08	6.41E+15	9.34E+08
	SD	5.24E-01	1.92E+16	1.52E+15	2.70E+15	2.86E+04	2.04E+08	3.30E+15	1.74E+09
F10	*MEAN*	-1.02E-04	5.81E+12	1.84E+18	1.10E+21	2.79E+07	4.09E+11	1.09E+19	1.94E+13
	SD	3.54E-07	3.68E+12	7.13E+17	8.36E+19	7.32E+06	3.82E+11	5.43E+18	1.08E+13
F11	*MEAN*	3.52E+07	2.75E+16	2.10E+17	1.94E+18	1.52E+10	2.66E+16	1.01E+18	9.92E+14
	SD	4.57E+07	1.88E+16	1.74E+17	3.85E+16	9.04E+09	1.59E+16	3.69E+17	5.43E+14
F12	*MEAN*	1.06E+02	3.63E+16	8.78E+16	6.17E+17	1.01E+02	8.23E+09	4.40E+17	5.59E+10
	SD	3.81E+01	1.76E+16	2.91E+16	2.98E+16	6.50E+02	8.56E+09	6.26E+16	3.18E+10
F13	*MEAN*	9.43E+09	3.69E+16	1.02E+17	6.21E+17	8.05E+14	5.83E+12	3.85E+17	4.12E+12
	SD	7.49E+09	1.82E+16	3.63E+16	1.73E+16	2.03E+14	4.48E+12	6.42E+16	1.04E+12
F14	*MEAN*	2.00E+00	7.36E+16	1.65E+17	1.21E+18	2.20E+00	2.87E+10	1.02E+18	8.22E+10
	SD	1.49E-01	4.25E+16	1.00E+17	6.38E+16	2.90E-02	2.31E+10	1.20E+17	3.01E+10
F15	*MEAN*	1.93E+01	1.88E+16	5.43E+16	5.69E+17	2.56E+01	1.30E+01	3.85E+17	2.60E+01
	SD	3.58E+00	9.77E+15	1.82E+16	1.54E+16	5.25E+00	1.72E+00	1.74E+16	3.19E+00
F16	*MEAN*	1.69E+02	1.94E+16	6.66E+16	5.83E+17	2.29E+02	1.34E+02	4.07E+17	2.30E+02
	SD	2.36E+01	1.17E+16	4.00E+16	1.91E+16	1.14E+01	1.10E+01	9.94E+16	1.49E+01
F17	*MEAN*	9.61E+10	4.48E+16	8.00E+16	6.14E+17	9.61E+10	9.61E+10	4.40E+17	9.61E+10
	SD	9.03E-03	2.22E+16	3.20E+16	8.72E+15	7.30E-03	4.15E+00	8.80E+16	1.07E+05
F18	*MEAN*	4.39E+01	6.99E+26	1.23E+28	9.17E+28	2.99E+12	2.44E+17	6.02E+28	5.77E+16
	SD	8.34E+00	5.49E+26	5.75E+27	5.75E+27	2.68E+12	3.06E+17	1.73E+28	5.87E+16
F19	*MEAN*	1.83E+17	1.85E+17	1.85E+17	1.85E+17	1.85E+17	1.84E+17	1.85E+17	1.84E+17
	SD	5.77E+01	6.48E+13	1.33E+14	4.78E+13	1.90E+14	1.25E+14	6.85E+13	3.84E+14
F20	*MEAN*	2.57E+00	7.58E+00	5.65E+00	1.02E+01	3.66E+00	8.34E+00	9.76E+00	8.70E+00
	SD	2.06E-01	7.77E-01	1.12E+00	4.34E01	4.76E-01	5.51E-01	1.20E+00	1.08E+00
F21	*MEAN*	1.28E+02	1.89E+16	3.85E+16	2.02E+17	1.07E+02	1.02E+12	1.90E+17	3.08E+10
	SD	2.68E+01	1.20E+16	2.99E+16	2.55E+16	1.01E+01	1.12E+12	5.07E+16	6.69E+09
F22	*MEAN*	2.23E+10	1.80E+16	3.56E+16	1.92E+17	5.34E+14	3.40E+13	1.71E+17	8.13E+12
	SD	2.07E+10	9.87E+15	2.07E+16	4.90E+16	7.77E+13	1.29E+13	6.65E+16	1.66E+12
F23	*MEAN*	1.79E+00	3.17E+16	7.90E+16	3.30E+17	2.13E+00	6.45E+12	3.23E+17	1.04E+11
	SD	8.39E-02	2.42E+16	3.45E+16	4.45E+16	7.41E-02	9.79E+12	8.67E+16	4.88E+10
F24	*MEAN*	1.62E+01	5.58E+15	4.01E+16	1.18E+17	2.31E+01	1.67E+01	1.47E+17	9.00E+01
	SD	2.81E+00	2.75E+15	1.97E+16	3.65E+16	3.58E+00	1.66E+00	6.55E+16	1.24E+02
F25	*MEAN*	1.65E+02	8.21E+15	2.81E+16	1.64E+17	2.38E+02	1.42E+02	1.54E+17	2.31E+02
	SD	8.78E+00	3.63E+15	2.21E+16	5.34E+16	1.60E+016	7.64E+00	4.51E+16	1.21E+01
F26	*MEAN*	9.61E+10	1.66E+16	5.08E+16	1.76E+17	9.61E+10	1.70E+12	1.94E+17	9.61E+10
	SD	7.20E-03	9.44E+15	2.13E+16	2.81E+16	5.81E-03	6.91E+11	3.94E+16	2.18E+06
F27	*MEAN*	5.82E+01	2.29E+26	1.52E+27	1.30E+28	1.03E+13	1.45E+18	2.07E+28	1.82E+18
	SD	4.81E+01	1.75E+26	1.15E+27	7.34E+27	2.32E+13	1.75E+18	1.12E+28	4.00E+18
F28	*MEAN*	1.84E+17	1.85E+17	1.85E+17	1.85E+17	1.85E+17	1.84E+17	1.85E+17	1.85E+17
	SD	2.23E+14	7.67E+13	1.33E+14	3.78E+13	1.03E+14	1.62E+14	6.29E+13	1.27E+14

Note: "/" indicates that the same items do not compare.

TABLE 12. The comparison results among AHCS and other seven algorithms on CEC 2017 30-dimensional optimization problems.

Function	Evaluation indicator	Algorithm							
		AHCS	ICSO	MDCSO	NAFA	DUFA	OBLPSOGD	AGWO	IRCGA
F1	*Rank*	1	3	4	6	2	8	7	5
	T_{val}	/	1.94E+01	6.45E+00	5.35E+00	7.10E+01	1.34E+02	8.51E+00	1.40E+01
	VS	/	-	-	-	-	-	-	-
F2	*Rank*	1	4	3	7	2	8	6	5
	T_{val}	/	3.14E+00	1.54E+01	8.19E+00	5.10E+01	6.62E+01	1.31E+01	3.46E+01
	VS	/	-	-	-	-	-	-	-
F3	*Rank*	4	6	1	5	3	2	7	8
	T_{val}	/	9.98E+00	-2.21E+00	6.60E-01	-2.01E+00	-2.03E+00	1.04E+01	1.12E+01
	VS	/	-	+	-	+	+	-	-
F4	*Rank*	1	4	3	6	8	5	7	2
	T_{val}	/	1.92E+01	2.81E+01	8.55E+01	5.49E+01	3.02E+01	8.78E+01	1.80E+01
	VS	/	-	-	-	-	-	-	-
F5	*Rank*	1	3	6	8	2	5	7	4
	T_{val}	/	1.71E+01	6.27E+00	3.29E+01	9.10E+01	7.19E+00	2.40E+01	7.96E+00
	VS	/	-	-	-	-	-	-	-
F6	*Rank*	4	3	1	8	2	5	7	6
	T_{val}	/	-7.98E-01	-1.43E+00	9.73E+00	-9.66E-01	8.08E+00	3.59E+01	1.16E+01
	VS	/	+	+	-	+	-	-	-
F7	*Rank*	3	5	6	7	1	2	8	4
	T_{val}	/	8.04E+00	2.44E+00	1.55E+01	-9.58E+00	-9.58E+00	3.94E+01	1.62E+01
	VS	/	-	-	-	+	+	-	-
F8	*Rank*	1	5	6	7	2	3	8	4
	T_{val}	/	1.77E+01	3.23E+01	6.08E+00	2.56E+01	6.26E+00	3.41E+00	8.70E+00
	VS	/	-	-	-	-	-	-	-
F9	*Rank*	1	8	5	6	2	3	7	4
	T_{val}	/	1.77E+01	5.37E+00	5.84E+00	1.06E+01	3.89E+00	1.06E+01	2.94E+00
	VS	/	-	-	-	-	-	-	-
F10	*Rank*	1	4	6	8	2	3	7	5
	T_{val}	/	8.65E+00	1.41E+01	7.21E+01	2.09E+01	5.86E+00	1.10E+01	9.84E+00
	VS	/	-	-	-	-	-	-	-
F11	*Rank*	1	5	6	8	2	4	7	3
	T_{val}	/	8.01E+00	6.61E+00	2.76E+02	9.19E+00	9.16E+00	1.50E+01	1.00E+01
	VS	/	-	-	-	-	-	-	-
F12	*Rank*	2	5	6	8	1	3	7	4
	T_{val}	/	1.13E+01	1.65E+01	1.13E+02	-4.21E-02	5.27E+00	3.85E+01	9.63E+00
	VS	/	-	-	-	+	-	-	-
F13	*Rank*	1	5	6	8	4	3	7	2
	T_{val}	/	1.11E+01	1.54E+01	1.97E+02	2.17E+01	7.12E+00	3.28E+01	2.16E+01
	VS	/	-	-	-	-	-	-	-
F14	*Rank*	1	5	6	8	2	3	7	4
	T_{val}	/	9.49E+00	9.04E+00	1.04E+02	7.22E+00	6.81E+00	4.66E+01	1.50E+01
	VS	/	-	-	-	-	-	-	-
F15	*Rank*	2	5	6	8	3	1	7	4
	T_{val}	/	1.05E+01	1.63E+01	2.02E+02	5.43E+00	-8.69E+00	1.21E+02	7.65E+00
	VS	/	-	-	-	-	+	-	-
F16	*Rank*	2	5	6	8	3	1	7	4
	T_{val}	/	9.08E+00	9.12E+00	1.67E+02	1.25E+01	-7.36E+00	2.24E+01	1.20E+01
	VS	/	-	-	-	-	+	-	-
F17	*Rank*	2.5	5	6	8	2.5	2.5	7	2.5
	T_{val}	/	1.11E+01	1.37E+01	3.86E+02	0.00E+00	0.00E+00	2.74E+01	0.00E+00
	VS	/	-	-	-	=	=	-	=
F18	*Rank*	1	5	6	8	2	4	7	3
	T_{val}	/	6.97E+00	1.17E+01	8.73E+01	6.11E+00	4.37E+00	1.91E+01	5.38E+00
	VS	/	-	-	-	-	-	-	-
F19	*Rank*	1	6	6	6	6	2.5	6	2.5
	T_{val}	/	1.69E+02	8.24E+01	2.29E+02	5.77E+01	4.38E+01	1.60E+02	1.43E+01
	VS	/	-	-	-	-	-	-	-

TABLE 12. *(Continued.)* **The comparison results among AHCS and other seven algorithms on CEC 2017 30-dimensional optimization problems.**

F20	*Rank*	1	4	3	8	2	5	7	6
	T_{val}	/	3.41E+01	1.48E+01	9.63E-01	1.15E+01	5.37E+01	3.23E+01	3.05E+01
	VS	/	-	-	-	-	-	-	-
F21	*Rank*	2	5	6	8	1	4	7	3
	T_{val}	/	8.63E+00	7.05E+00	4.34E+01	-4.02E+00	4.99E+00	2.05E+01	2.52E+01
	VS	/	-	-	-	+	-	-	-
F22	*Rank*	1	5	6	8	4	3	7	2
	T_{val}	/	9.99E+00	9.42E+00	2.15E+01	3.76E+01	1.44E+01	1.41E+01	2.67E+01
	VS	/	-	-	-	-	-	-	-
F23	*Rank*	1	5	6	8	2	4	7	3
	T_{val}	/	7.17E+00	1.25E+01	4.06E+01	1.66E+01	3.61E+00	2.04E+01	1.17E+01
	VS	/	-	-	-	-	-	-	-
F24	*Rank*	1	5	6	7	3	2	8	4
	T_{val}	/	1.11E+01	1.11E+01	1.77E+01	8.30E+00	8.39E-01	1.23E+01	3.26E+00
	VS	/	-	-	-	-	-	-	-
F25	*Rank*	2	5	6	8	4	1	7	3
	T_{val}	/	1.24E+01	6.96E+00	1.68E+01	2.50E-14	-1.08E+01	1.87E+01	2.42E+01
	VS	/	-	-	-	-	+	-	-
F26	*Rank*	2	5	6	7	2	4	8	2
	T_{val}	/	9.63E+00	1.31E+01	3.43E+01	0.00E+00	1.27E+01	2.70E+01	0.00E+00
	VS	/	-	-	-	=	-	-	=
F27	*Rank*	1	5	6	7	2	3	8	4
	T_{val}	/	7.17E+00	7.24E+00	9.70E+00	2.43E+00	4.54E+00	1.01E+01	2.49E+00
	VS	/	-	-	-	-	-	-	-
F28	*Rank*	1.5	5.5	5.5	5.5	5.5	1.5	5.5	5.5
	T_{val}	/	2.32E+01	2.11E+01	2.42E+01	2.23E+01	0.00E+00	2.36E+01	2.13E+01
	VS	/	-	-	-	-	=	-	-
Average Ranking		1.57	4.84	5.20	7.30	2.75	3.41	7.05	3.88
Final Ranking		1	5	6	8	2	3	7	4
w/t/l		/	27/0/1	26/0/2	28/0/0	21/2/5	21/2/5	28/0/0	26/2/0

Note: "/" indicates that the same items do not compare.

TABLE 13. **Optimization results for various improved CSs in the reducer design optimization problem.**

Algorithm	x_1	x_2	x_3	x_4	x_5	x_6	x_7	$f(x)$
AHCS	3.499814	0.699968	17.000000	7.300000	7.715302	3.350214	5.286644	2994.23
BCS	3.499814	0.699968	16.999999	7.300000	7.715302	3.350214	5.286644	2994.23
ICS	3.499814	0.6999678	16.999999	7.300000	7.715302	3.350214	5.286644	2994.23
CS-EO	3.499814	0.699968	16.999999	7.300000	7.715302	3.350214	5.286644	2994.23
DCS	3.499814	0.699968	16.999999	7.300000	7.715302	3.350214	5.286644	2994.23
RCS	3.499650	0.699946	22.679463	7.489845	7.945432	3.726208	5.286095	4233.77
MCS	3.499814	0.699968	16.999999	7.300000	7.715302	3.350214	5.286644	2994.23
CV 1.0	3.499715	0.699947	17.000025	7.300165	7.715693	3.350235	5.286658	2994.30

TABLE 14. **Optimization results for various improved algorithms in the reducer design optimization problem.**

Algorithm	x_1	x_2	x_3	x_4	x_5	x_6	x_7	$f(x)$
AHCS	3.499814	0.699968	17.000000	7.300000	7.715302	3.350214	5.286644	2994.23
EOCSO	3.500000	0.700000	17.000000	7.300000	7.715320	3.350214	5.286654	2994.34
PSO-DE	3.500000	0.700000	17.000000	7.300000	7.800000	3.350214	5.286683	2996.35
MBA	3.500000	0.700000	17.000000	7.300033	7.715772	3.350218	5.286654	2994.48
HEAA	3.500023	0.700000	17.000013	7.300427	7.715377	3.350231	5.286664	2994.50
HGA	3.500000	0.700000	17.000000	7.300000	7.715332	3.350215	5.286664	2994.47
SSBA	3.500007	0.700000	17.000000	7.327602	7.715322	3.350267	5.286655	2994.74
GOKA	3.500000	0.700000	17.000000	7.300000	7.710000	3.350000	5.290000	2996.31

literature are shown in TABLE 16. The optimization results of AHCS and other seven algorithms in the literature are shown in TABLE 17.

In order to verify that the solution quality of the AHCS is better than other algorithms, each algorithm is run 30 times, the maximum runtime for each run is

TABLE 15. The quality of optimization values for various algorithms in the reducer design optimization problem.

Algorithm	Best	Mean	Worst	Std
AHCS	2994.228064	2994.228064	2994.228064	2.27E-13
BCS	2994.228064	2994.228064	2994.228064	1.82E-09
ICS	2994.228064	2994.228064	2994.228064	3.94E-13
CS-EO	2994.228064	2994.228064	2994.228064	0.00E+00
DCS	2994.228064	2994.228064	2994.228064	0.00E+00
RCS	4233.770468	5376.204254	7983.441769	1.53E+03
MCS	2994.228064	2994.228064	2994.228064	3.94E-13
CV 1.0	2994.296475	2995.561612	2999.422099	2.18E+00
EOCSO	2994.341316	2994.341317	2994.341318	1.20E-06
PSO-DE	2996.348167	2996.348174	2996.348204	6.40E-06
MBA	2994.482453	2996.769019	2999.652444	1.56E+00
HEAA	2994.499107	2994.613368	2994.752311	7.00E-02
HGA	2994.47	NA	NA	NA
SSBA	2994.74	NA	NA	NA
GOKA	2996.31	NA	NA	NA

Note: NA means not available.

TABLE 16. Optimization results for various improved CSs in Cantilever beam design optimization problem.

Variables	AHCS	BCS	ICS	CS-EO	DCS	RCS	MCS	CV 1.0
x_1	3.045403	3.027535	3.045314	3.055495	3.050828	3.605859	3.045298	3.048868
x_2	60.907994	60.374814	60.880110	61.109916	61.003579	62.573491	60.904092	60.976745
x_3	2.811145	2.806519	2.808552	2.797220	2.803124	4.152304	2.810453	2.807071
x_4	56.222875	56.102387	55.934739	55.942491	56.059324	58.094172	56.208141	56.141185
x_5	2.523614	2.556202	2.538435	2.528112	2.524290	4.329599	2.524627	2.523950
x_6	50.471989	51.061015	50.729881	50.535512	50.467691	63.993629	50.491906	50.478989
x_7	2.206277	2.220513	2.213725	2.206679	2.210856	4.952004	2.206496	2.206300
x_8	44.125373	44.322935	44.122555	44.123960	44.211776	63.279849	44.129841	44.126003
x_9	1.749741	1.755242	1.750250	1.749793	1.751118	5.282324	1.750645	1.749887
x_{10}	34.994815	35.039979	34.999148	34.995551	34.988346	61.360506	34.985752	34.993386
$f(x)$	62949.07	63068.47	63020.16	62956.64	62966.23	8108817.88	62953.47	62949.85

TABLE 17. Optimization results for various algorithms in Cantilever beam design optimization problem.

Variables	AHCS	$SDGAM_{INLS}$	MPNN	SUMT	RNES	CAD	GA-1	GA-APM
x_1	3.045403	3.0459	3.0606	2.17	3	3	3	3
x_2	60.907994	60.8969	61.2115	42.74	60	60	60	60
x_3	2.811145	2.8018	2.8161	2.27	3.1	3.1	3.1	3.1
x_4	56.222875	56.0168	56.3214	44.99	55	55	55	55
x_5	2.523614	2.5251	2.5216	2.82	2.6	2.6	2.6	2.6
x_6	50.471989	50.4643	50.4290	50.47	50	50	50	50
x_7	2.206277	2.2252	2.2136	2.79	2.311	2.279	2.3	2.289
x_8	44.125373	44.4745	44.2759	55.42	43.108	45.553	45.5	45.626
x_9	1.749741	1.7678	1.7503	3.00	1.822	1.75	1.8	1.793
x_{10}	34.994815	34.8462	35.0141	59.77	34.307	35.004	35	34.593
$f(x)$	62949.07	63044.17	63240.67	65678.00	64269.59	64403	64558	64698.56

maxruntime=20 seconds, the population size is $N = 40$, and the penalty factor is $M = 10^8$. The optimal value and the worst value of the various algorithms after 30 results are counted, and the average optimal value and standard deviation of the optimal operating objective function for 30 times are calculated. The statistical results are shown in Table 18.

As can be seen from TABLES 16 and 17 that the optimized value of AHCS solution is better than the other seven improved CSs and seven other algorithms involved in the comparison. In addition, as can be seen from TABLE 18, the optimal value, mean value, worst value, and standard deviation of AHCS are better than the other 14 algorithms that participate in the comparison. Therefore, the quality of AHCS solution is significantly better than other algorithms.

TABLE 18. The quality of optimization values for various algorithms in Cantilever beam design optimization problem.

Algorithm	Best	Mean	Worst	Std
AHCS	62949.070362	62949.307760	62949.705517	3.17E-01
BCS	63068.477345	63161.790272	63221.810517	6.64E+01
ICS	63020.161651	63120.002991	63191.079758	6.38E+01
CS-EO	62956.647263	62958.839898	62962.729714	2.55E+00
DCS	62966.235074	62973.304528	62982.591467	6.50E+00
RCS	8108817.8859	103883406.58	325349487.77	1.30E+08
MCS	62953.473989	62994.262591	63142.755801	8.31E+01
CV 1.0	62949.852413	62951.460513	62952.807681	1.11E+00
$SDGAM_{INLS}$	63044.17	NA	NA	NA
MPNN	63240.67	NA	NA	NA
SUMT	65678.00	NA	NA	NA
RNES	64269.59	NA	NA	NA
CAD	64403	NA	NA	NA
GA-1	64558	NA	NA	NA
GA-APM	64698.56	NA	NA	NA

Note: NA means not available.

VII. CONCLUSION

The improved CS proposed in the literature has a low solution quality and is easy to fall into a local optimum when solving optimization problems. This paper proposes AHCS. There are three main aspects for the improvement of the AHCA: (1) the Lévy flight method of the HCS is improved, and the parameters α and β are dynamically adjusted respectively; (2) the mutation operator is introduced in the AHCS; (3) an improved population evolution strategy of the AHCS is given.

In the improvement study of Lévy flight method in HCS, in order to accelerate the convergence speed of the algorithm, the inertia weight is introduced in the Lévy flight method of the CS algorithm. At the same time, in order to minimize the influence of the maximum iteration number on the algorithm position update, and consider the individual fitness value information in the population, the parameters α and β are dynamically adjusted, and new adaptive adjustment method are given respectively.

In the improved research of the hybrid cuckoo search algorithm, in order to make the algorithm have strong local search ability in the late iteration, the mutation operator is introduced into the cuckoo search algorithm.

In the study of population evolution strategy, a new hybrid cuckoo search evolution strategy is proposed. The evolutionary strategy draws on the idea of the speed update of the PSO algorithm, supplements the individuals the population, and restores the initial size of the population. In addition, in order to avoid the algorithm falling into local optimum, the optimal individual in the population is perturbed.

In order to verify the performance of AHCS, 30 benchmark functions and CEC 2017 optimization problems were selected, and run with various improved CSs and other improved algorithms in the literature. The calculation results of 30 benchmark functions and CEC 2017 optimization problems show compared with other algorithms, the number of winning cases of t-test values and the Friedman average ranking for AHCS are significantly better than other algorithms.

In addition, AHCS and various improved algorithms in the literature are used to optimize the structural parameters optimization of the reducer and the cantilever beam. The optimization results show that the quality of the solution of HCS is significantly better than other algorithms.

Although the AHCS has achieved good results on the structural parameters optimization of the reducer and the cantilever beam, further research is needed to extend the application area of the algorithm. In the future, our proposed AHCS will be applied to more optimization examples and some new engineering optimization applications.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable and constructive comments that greatly helped improve the quality and completeness of the article.

REFERENCES

[1] X.-S. Yang and S. Deb, “Cuckoo search via Lévy flights,” in *Proc. World Congr. Nature Biologically Inspired Comput. (NaBIC)*, Coimbatore, India, Dec. 2009, pp. 210–214.

[2] X. Ouyang, Y. Zhou, Q. Luo, and H. Chen, “A novel discrete cuckoo search algorithm for spherical traveling salesman problem,” *Appl. Math. Inf. Sci.*, vol. 7, no. 2, pp. 777–784, Mar. 2013.

[3] A. Ouaarab, B. Ahiod, and X.-S. Yang, “Discrete cuckoo search algorithm for the travelling salesman problem,” *Neural Comput. Appl.*, vol. 24, nos. 7–8, pp. 1659–1669, 2014.

[4] A. Ouaarab, B. Ahiod, and X.-S. Yang, “Random-key cuckoo search for the travelling salesman problem,” *Soft Comput.*, vol. 19, no. 4, pp. 1099–1106, 2015.

[5] S. Burnwal and S. Deb, “Scheduling optimization of flexible manufacturing system using cuckoo search-based approach,” *Int. J. Adv. Manuf. Technol.*, vol. 64, nos. 5–8, pp. 951–959, Feb. 2013.

[6] M. Prakash, R. Saranya, K. R. Jothi, and A. Vigneshwaran, “An optimal job scheduling in grid using cuckoo algorithm,” *Int. J. Comput. Sci. Telecommun.*, vol. 3, no. 2, pp. 65–69, Feb. 2012.

[7] M. Rabiee and H. Sajedi, “Job scheduling in grid computing with cuckoo optimization algorithm,” *Int. J. Comput. Appl.*, vol. 62, no. 16, pp. 38–44, Jan. 2013.

[8] H. Zhu, X. Qi, F. Chen, L. Chen, and Z. Zhang, “Quantum-inspired cuckoo co-search algorithm for no-wait flow shop scheduling,” *Appl. Intell.*, vol. 49, no. 2, pp. 791–803, Feb. 2019.

[9] A. R. Yildiz, “Cuckoo search algorithm for the selection of optimal machining parameters in milling operations,” *Int. J. Adv. Manuf. Technol.*, vol. 64, nos. 1–4, pp. 55–61, Jan. 2013.

[10] R. Chi, Y. Su, Z. Qu, and X. Chi, “A hybridization of cuckoo search and differential evolution for the logistics distribution center location problem,” *Math. Problems Eng.*, vol. 2019, Feb. 2019, Art. no. 7051248.

[11] K. Chi, J. Kang, X. Zhang, and Z. Yang, "Bearing fault diagnosis based on stochastic resonance with cuckoo search," *Int. J. Performability Eng.*, vol. 14, no. 3, pp. 413–424, Mar. 2018. doi: 10.23940/ijpe.18.03.p2.413424.
[12] W. Zhang, G. Han, J. Wang, and Y. Liu, "A BP neural network prediction model based on dynamic cuckoo search optimization algorithm for industrial equipment fault prediction," *IEEE Access*, vol. 7, pp. 11736–11746, 2019.
[13] A. K. Bhandari, V. K. Singh, A. Kumar, and G. K. Singh, "Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy," *Expert Syst. Appl.*, vol. 41, no. 7, pp. 3538–3560, Jun. 2014.
[14] A. S. Ashour, S. Samanta, N. Dey, N. Kausar, W. B. Abdessalemkaraa, and A. E. Hassanien, "Computed tomography image enhancement using cuckoo search: A log transform based approach," *J. Signal Inf. Process.*, vol. 6, no. 3, pp. 244–257, Jul. 2015.
[15] M. A. Sanchez, O. Castillo, and J. R. Castro, "Information granule formation via the concept of uncertainty-based information with interval type-2 fuzzy sets representation and Takagi–Sugeno–Kang consequents optimized with Cuckoo search," *Appl. Soft Comput.*, vol. 27, pp. 602–609, Feb. 2015.
[16] C. I. Gonzalez, P. Melin, J. R. Castro, O. Castillo, and O. Mendoza, "Optimization of interval type-2 fuzzy systems for image edge detection," *Appl. Soft Comput.*, vol. 47, pp. 631–643, Oct. 2016.
[17] K. N. A. Rani, M. F. AbdulMalek, and S. Neoh, "Nature-inspired cuckoo search algorithm for side lobe suppression in a symmetric linear antenna array," *Radio Eng. J.*, vol. 21, no. 3, pp. 865–874, Sep. 2012.
[18] A. Kaveh and T. Bakhshpoori, "Optimum design of space trusses using Cuckoo search algorithm with levy flights," *Iranian J. Sci. Technol., Trans. Civil Eng.*, vol. 37, no. C1, pp. 1–15, 2013.
[19] S. Dhabal and P. Venkateswaran, "An improved global-best-guided cuckoo search algorithm for multiplierless design of two-dimensional IIR filters," *Circuits, Syst., Signal Process.*, vol. 38, no. 2, pp. 805–826, Feb. 2019.
[20] Y. Chen and N. Wang, "Cuckoo search algorithm with explosion operator for modeling proton exchange membrane fuel cells," *Int. J. Hydrogen Energy*, vol. 44, no. 5, pp. 3075–3087, Jan. 2019.
[21] N. El Gmili, M. Mjahed, A. El Kari, and H. Ayad, "Particle swarm optimization and cuckoo search-based approaches for quadrotor control and trajectory tracking," *Appl. Sci.*, vol. 9, no. 8, p. 1719, Apr. 2019.
[22] K. Thirugnanasambandam, S. Prakash, V. Subramanian, S. Pothula, and V. Thirumal, "Reinforced cuckoo search algorithm-based multimodal optimization," *Appl. Intell.*, vol. 49, no. 6, pp. 2059–2083, Jun. 2019.
[23] X. Zhang, X.-T. Li, and M.-H. Yin, "Hybrid Cuckoo search algorithm with covariance matrix adaption evolution strategy for global optimisation problem," *Int. J. Bio-Inspired Comput.*, vol. 13, no. 2, pp. 102–110, Mar. 2019.
[24] M. Guerrero, O. Castillo, and M. García, "Fuzzy dynamic parameters adaptation in the Cuckoo search algorithm using fuzzy logic," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2015, pp. 441–448.
[25] A. Jaballah and A. Meddeb, "A new variant of cuckoo search algorithm with self adaptive parameters to solve complex RFID network planning problem," *Wireless Netw.*, vol. 25, no. 4, pp. 1585–1604, May 2019.
[26] R. Chi, Y. X. Su, D. H. Zhang, X. X. Chi, and H. J. Zhang, "A hybridization of cuckoo search and particle swarm optimization for solving optimization problems," *Neural Comput. Appl.*, vol. 31, pp. 653–670, Jan. 2019.
[27] H.-S. Ma, S.-X. Li, S.-F. Li, Z.-N. Lv, and J.-S. Wang, "An improved dynamic self-adaption Cuckoo search algorithm based on collaboration between subpopulations," *Neural Comput. Appl.*, vol. 31, no. 5, pp. 1375–1389, May 2019.
[28] P. Ong and Z. Zainuddin, "Optimizing wavelet neural networks using modified Cuckoo search for multi-step ahead chaotic time series prediction," *Appl. Soft Comput.*, vol. 80, pp. 374–386, Jul. 2019.
[29] O. Mohammadrezapour, I. Yoosefdoost, and M. Ebrahimi, "Cuckoo optimization algorithm in optimal water allocation and crop planning under various weather conditions (case study: Qazvin plain, Iran)," *Neural Comput. Appl.*, vol. 31, no. 6, pp. 1879–1892, Jun. 2019.
[30] X. Zhang, Q. Wan, and Y. Fan, "Applying modified Cuckoo search algorithm for solving systems of nonlinear equations," *Neural Comput. Appl.*, vol. 31, no. 2, pp. 553–576, Feb. 2019.
[31] S. B. Raha, K. K. Mandal, and N. Chakraborty, "Hybrid SMES based reactive power dispatch by Cuckoo search algorithm," *IEEE Trans. Ind. Appl.*, vol. 55, no. 1, pp. 907–917, Jan./Feb. 2019.
[32] A. Agasthian, R. Pamula, and L. A. Kumaraswamidhas, "Fault classification and detection in wind turbine using Cuckoo-optimized support vector machine," *Neural Comput. Appl.*, vol. 31, no. 5, pp. 1503–1511, May 2019.
[33] T. Hosseinalizadeh, S. M. Salamati, S. A. Salamati, and G. B. Gharehpetian, "Improvement of identification procedure using hybrid Cuckoo search algorithm for turbine-governor and excitation system," *IEEE Trans. Energy Convers.*, vol. 34, no. 2, pp. 585–593, Jun. 2019.
[34] B. Biswal, P. K. Karn, M. V. S. Sairam, and B. R. Surekhabolli, "Time-frequency analysis and classification of power signals using adaptive cuckoo search algorithm," *Int. J. Numer. Model., Electron. Netw., Devices Fields*, vol. 32, no. 1, p. e2477, Jan./Feb. 2019.
[35] J. T. Cheng, L. Wang, and Y. Xiong, "Cuckoo search algorithm with memory and the vibrant fault diagnosis for hydroelectric generating unit," *Eng. Comput.*, vol. 35, no. 2, pp. 687–702, Apr. 2019.
[36] H.-L. Chen, B. Yu, H.-L. Zhou, and Z. Meng, "Improved Cuckoo search algorithm for solving inverse geometry heat conduction problems," *Heat Transf. Eng.*, vol. 40, nos. 3–4, pp. 362–374, 2019.
[37] R. Prasath and T. Kumanan, "Distance-oriented cuckoo search enabled optimal histogram for underwater image enhancement: A novel quality metric analysis," *Imag. Sci. J.*, vol. 67, no. 2, pp. 76–89, 2019.
[38] Z. Wu, X. Zhao, Y. Ma, and X. Zhao, "A hybrid model based on modified multi-objective cuckoo search algorithm for short-term load forecasting," *Appl. Energy*, vol. 237, pp. 896–909, Mar. 2019.
[39] H. Chen and H. Zhou, "Identification of boundary conditions for non-Fourier heat conduction problems by differential transformation DRBEM and improved Cuckoo search algorithm," *Numer. Heat Transf., B, Fundam.*, vol. 74, no. 6, pp. 818–839, Apr. 2019.
[40] X. Meng, J. Chang, X. Wang, and Y. Wang, "Multi-objective hydropower station operation using an improved Cuckoo search algorithm," *Energy*, vol. 168, pp. 425–439, Feb. 2019.
[41] G. V. Shenoy, *Operations Research for Management*. Beijing, China: Tsinghua Univ. Press, 2012, pp. 176–178.
[42] M. Guerrero, O. Castillo, and M. García, "Cuckoo search via Lévy flights and a comparison with genetic algorithms," in *Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics*. Cham, Switzerland: Springer, 2015, pp. 91–103.
[43] I. Pavlyukevich, "Lévy flights, non-local search and simulated annealing," *J. Comput. Phys.*, vol. 226, no. 2, pp. 1830–1844, Oct. 2007.
[44] M. F. Shlesinger, "Mathematical physics: Search research," *Nature*, vol. 443, no. 7109, pp. 281–282, Sep. 2006.
[45] X. S. Yang and S. Deb, "Engineering optimization by cuckoo search," *Int. J. Math. Model. Numer. Optim.*, vol. 1, no. 4, pp. 330–343, Dec. 2010.
[46] R. Hinterding, Z. Michalewicz, and A. E. Eiben, "Adaptation in evolutionary computation: A survey," in *Proc. IEEE Int. Conf. Evol. Comput.* Piscataway, NJ, USA: IEEE Press, Apr. 1997, pp. 65–69.
[47] W. L. Xue, "A parameter turning based on fitness variance of Cuckoo search algorithm," *Value Eng.*, no. 34, pp. 154–157, Dec. 2015. doi: 10.14018/j.cnki.cn13-1085/n.2015.34.059.
[48] Z. Gong, C. Y. Jiang, Z. D. Yu, and L. D. Shi, "Parameter estimation of Muskegon model based on CS algorithm," *Low Temp. Archit. Technol.*, no. 12, pp. 78–79, Dec. 2016. doi: 10.13905/j.cnki.dwjz.2016.12.028.
[49] Y. Y. Li, X. Q. Y. Fang, and Z. J. Li, "Reactive power voltage control on power distribution network with photovoltaic power generation based on improved Cuckoo search algorithm," *Guangdong Electr. Power*, vol. 29, no. 8, pp. 79–84, 2016. doi: 10.3969/j.issn.1007-290X.2016.08.015.
[50] R. Y. Li and R. W. Dai, "Adaptive step-size Cuckoo search algorithm," *Comput. Sci.*, vol. 44, no. 5, pp. 235–240, May 2017. doi: 10.11896/j.issn.1002-137X.2017.05.042.
[51] W. Gao, Z. Wei, Y. Luo, and J. Cao, "Artificial bee colony algorithm based on Parzen window method," *Appl. Soft Comput.*, vol. 74, pp. 679–692, Jan. 2019.
[52] X. Du, J. Wang, V. Jegatheesan, and G. Shi, "Parameter estimation of activated sludge process based on an improved Cuckoo search algorithm," *Bioresource Technol.*, vol. 249, pp. 447–456, Feb. 2018.
[53] R. Salgotra, U. Singh, and S. Saha, "New cuckoo search algorithms with enhanced exploration and exploitation properties," *Expert Syst. Appl.*, vol. 95, pp. 384–420, Apr. 2018.
[54] D. Wu, S. Xu, and F. Kong, "Convergence analysis and improvement of the chicken swarm optimization algorithm," *IEEE Access*, vol. 4, pp. 9400–9412, 2016.
[55] M. Han, "Hybrid chicken swarm algorithm with dissipative structure and differential mutation," *J. Zhejiang Univ., Sci. Ed.*, vol. 45, pp. 272–283, May 2018.
[56] H. Wang, W. Wang, X. Zhou, H. Sun, J. Zhao, X. Yu, and Z. Cui, "Firefly algorithm with neighborhood attraction," *Inf. Sci.*, vols. 382–383, pp. 374–387, Mar. 2017.
[57] M. Yang, A. Wang, S. Geng, and Z. Ying, "Deploying charging nodes in wireless rechargeable sensor networks based on improved firefly algorithm," *Comput. Elect. Eng.*, vol. 72, pp. 719–731, Nov. 2018.

[58] M. Zhou, Z. Zhao, C. Xiong, and Q. Kang, "An opposition-based particle swarm optimization algorithm for noisy environments," in *Proc. IEEE 15th Int. Conf. Netw., Sens. Control*, Mar. 2018, pp. 1–6.

[59] M. H. Qais, H. M. Hasanien, and S. Alghuwainem, "Augmented grey wolf optimizer for grid-connected PMSG-based wind energy conversion systems," *Appl. Soft Comput.*, vol. 69, pp. 504–515, Aug. 2018.

[60] Y. Song, F. Wang, and X. Chen, "An improved genetic algorithm for numerical function optimization," *Appl. Intell.*, vol. 49, no. 5, pp. 1880–1902, May 2019.

[61] G. H. Wu, R. Mallipeddi, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization," Nat. Univ. Defense Technol., Changsha, China, Kyungpook Nat. Univ., Daegu, South Korea, Nanyang Technol. Univ., Singapore, Tech. Rep., Sep. 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

[62] R. Reynolds and M. Ali, "Embedding a social fabric component into cultural algorithms toolkit for an enhanced knowledge-driven engineering optimization," *Int. J. Intell. Comput. Cybern.*, vol. 1, no. 4, pp. 563–597, Oct. 2008.

[63] H.-C. Kuo and C.-H. Lin, "A directed genetic algorithm for global optimization," *Appl. Math. Comput.*, vol. 219, no. 14, pp. 7348–7364, Mar. 2013.

[64] C. Qu, S. Zhao, Y. Fu, and W. He, "Chicken swarm optimization based on elite opposition-based learning," *Math. Problems Eng.*, vol. 2017, Mar. 2017. Art. no. 2734362.

[65] H. Liu, Z. Cai, and Y. Wang, "Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization," *Appl. Soft Comput.*, vol. 10, no. 2, pp. 629–640, Mar. 2010.

[66] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, "Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems," *Appl. Soft Comput.*, vol. 13, no. 5, pp. 2592–2612, May 2013.

[67] Y. Wang, Z. Cai, Y. Zhou, and Z. Fan, "Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique," *Struct. Multidisciplinary Optim.*, vol. 37, no. 4, pp. 395–413, 2009.

[68] K. Deep and K. N. Das, "A novel hybrid genetic algorithm for constrained optimization," *Int. J. Syst. Assurance Eng. Manage.*, vol. 4, no. 1, pp. 86–93, Mar. 2013.

[69] T. Ray and K. M. Liew, "Society and civilization: An optimization algorithm based on the simulation of social behavior," *IEEE Trans. Evol. Comput.*, vol. 7, no. 4, pp. 386–396, Aug. 2003.

[70] H. L. Li and P. Papalambros, "A production system for use of global optimization knowledge," *J. Mech., Transmiss., Autom.*, vol. 107, no. 2, pp. 277–284, 1985.

[71] K. C. Gupta and J. Li, "Robust design optimization with mathematical programming neural networks," *Comput. Struct.*, vol. 76, pp. 507–516, Jul. 2000.

[72] A. V. Fiacco and G. P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. New York, NY, USA: Wiley, 1968.

[73] T. Y. Chen and H. C. Chen, "Mixed–discrete structural optimization using a rank-niche evolution strategy," *Eng. Optim.*, vol. 41, no. 1, pp. 39–58, Jan. 2009.

[74] P. B. Thanedar and G. N. Vanderplaats, "Survey of discrete variable optimization for structural design," *J. Struct. Eng.*, vol. 121, no. 2, pp. 301–306, Feb. 1995.

[75] F. Erbatur, O. Hasançebi, I. Tütüncü, and H. Kiliç, "Optimal design of planar and space structures with genetic algorithms," *Comput. Struct.*, vol. 75, no. 2, pp. 209–224, Mar. 2000.

[76] A. C. C. Lemonge and H. J. C. Barbosa, "An adaptive penalty scheme for genetic algorithms in structural optimization," *Int. J. Numer. Methods Eng.*, vol. 59, no. 5, pp. 703–736, Feb. 2004.

ZHIWEN CHENG received the B.E. degree in logistics management from the Hefei University of Technology, China, in 2017. He is currently pursuing the master's degree with the Engineering College, Northeast Agricultural University.

His current research interest includes genetic algorithm theory and its application.

JIQUAN WANG received the B.E. and M.E. degrees in agricultural electrification and automation from Northeast Agricultural University, China, in 1996 and 2004, respectively, and the Ph.D. degree in agricultural equipment engineering technology from Shenyang Agricultural University, China, in 2011.

He is currently an Associate Professor with the Engineering College, Northeast Agricultural University. He published over 40 articles in domestic and international academic journals and conference proceedings. His current research interests include genetic algorithm theory and its application, neural network theory and its application, parallel computing, and image recognition.

MINGXIN ZHANG received the B.E. degree in information management and information system from the Hebei University of Engineering, China, in 2018. He is currently pursuing the master's degree with the Engineering College, Northeast Agricultural University.

His current research interest includes genetic algorithm theory and its application.

HAOHAO SONG received the B.E. degree in logistics management from the North China University of Water Resources and Electric Power, China, in 2019. She is currently pursuing the master's degree with the Engineering College, Northeast Agricultural University.

Her current research interest includes genetic algorithm theory and its application.

TIEZHU CHANG received the B.E. degree in industry engineering from Northeast Agricultural University, China, in 2019, where he is currently pursuing the master's degree with the Engineering College.

His current research interest includes genetic algorithm theory and its application.

YUSHENG BI received the B.E. degree in industry engineering from Northeast Agricultural University, China, in 2018, where he is currently pursuing the master's degree with the Engineering College.

His current research interest includes genetic algorithm theory and its application.

KEXIN SUN received the B.E. degree in industry engineering from Northeast Agricultural University, China, in 2018, where he is currently pursuing the master's degree with the Engineering College.

His current research interest includes genetic algorithm theory and its application.