Applied Soft Computing 39 (2016) 199–222

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Cognitive behavior optimization algorithm for solving optimization problems

Mudong Li*, Hui Zhao, Xingwei Weng, Tong Han

Institute of Aeronautics and Astronautics Engineering, Air Force Engineering University, Xi'an, Shaanxi 710038, China

ARTICLE INFO

Article history:
Received 18 May 2015
Received in revised form 9 October 2015
Accepted 6 November 2015
Available online 29 November 2015

Keywords:
Nature-based algorithms
Global optimization
Cognitive behavior model
Exploration
Exploitation

ABSTRACT

Nature-based algorithms have become popular in recent fifteen years and have been widely applied in various fields of science and engineering, such as robot control, cluster analysis, controller design, dynamic optimization and image processing. In this paper, a new swarm intelligence algorithm named cognitive behavior optimization algorithm (COA) is introduced, which is used to solve the real-valued numerical optimization problems. COA has a detailed cognitive behavior model. In the model of COA, the common phenomenon of foraging food source for population is summarized as the process of exploration–communication–adjustment. Matching with the process, three main behaviors and two groups in COA are introduced. Firstly, cognitive population uses Gaussian and Levy flight random walk methods to explore the search space in the rough search behavior. Secondly, the improved crossover and mutation operator are used in the information exchange and share behavior between the two groups: cognitive population and memory population. Finally, the intelligent adjustment behavior is used to enhance the exploitation of the population for cognitive population. To verify the performance of our approach, both the classic and modern complex benchmark functions considered as the unconstrained functions are employed. Meanwhile, some well-known engineering design optimization problems are used as the constrained functions in the literature. The experimental results, considering both convergence and accuracy simultaneously, demonstrate the effectiveness of COA for global numerical and engineering optimization problems.

1. Introduction

Global optimization which can be defined as the process of searching for the global optimum in an optimization problem is a hotspot in applied mathematics [1,2]. However, the classical methods are difficult to find the global optimum for the problems as the dimension size of search space increased [3]. How to solve such complex problems becomes a key issue for global optimization.

The behaviors observed in creatures such as reproduction, foraging food, defending oneself, communication and information exchange are very promising and complex. It is because of the behaviors that the problem-solving skills of super organisms are highly developed. As a result, the nature-based algorithms (NAs) come next, enabling non-linear and non-differentiable optimization problems to be solved effectively [4–6]. In addition, such optimization algorithms have been applied to various fields such as dynamic optimization [7], inverse geophysical problems [8], twin-screw configuration problem [9], IIR filter design [10], image processing [11], distribution transformer design problem [12], mechanical design optimization problems [13], sensor deployment problems [14], task scheduling [15], data mining applications [16], chemical processes [17] and many other engineering problems.

Generally speaking, NAs can be divided into three main classes: swarm intelligence algorithms (SIAs), evolutionary algorithms (EAs), and physical phenomenon algorithms (PPAs) [18]. EAs are inspired by the genetic or evolution behaviors of creatures. Genetic algorithm (GA) [19] and differential evolution (DE) [20] algorithm can be described as representative algorithms in EAs. Firstly, the EAs evolve an initial random population for optimization. Then the combination and mutation strategies are used to generate the new population. Finally, greedy selection method is usually used to select a better solution between the new population and original population. In contrast, DE has a relative simpler structure and is more efficient for optimization. Many researchers have proposed a number of advanced variants of DE to improve its optimization performance, such as self-adapting control parameters in differential evolution (jDE) [21], the strategy adaption self-adaptive differential evolution algorithm (SADE) [22], the parameter adaptive

* Corresponding author. Tel.: +86 18392193176.
E-mail address: modern_lee@163.com (M. Li).

http://dx.doi.org/10.1016/j.asoc.2015.11.015
1568-4946/

differential evolution algorithm (JADE) [5], the DE based on covariance matrix learning and bimodal distribution parameter setting (CoBiDE) [23], and differential search algorithm (DSA) [24].

The second main class of NAs is swarm intelligence algorithms. The SIAs as well as EAs are usually inspired by the behaviors of intelligent nature creatures. But most of SIAs use genetic rules only, which are different from the EAs. On the other hand, SIAs always take fully advantages of each solution in search space to provide better solutions to the problem. Some of the popular SIAs are particle swarm optimization (PSO) inspired by the social behavior of bird flocking [25], symbiosis organisms search (SOS) that simulates the interactive behavior seen among organisms in nature [26], artificial bee colony (ABC) that mimics the honey-bees food searching behavior [27], cuckoo search (CS) inspired by parasitic bio-interactions in living creatures [28], animal migration optimization (AMO) algorithm that gets the idea from the behavior of the animal migration [29], gray wolf optimizer (GWO) inspired by hunting mechanism of gray wolves [30].

Physical phenomenon algorithms are the third class of nature-based algorithms. Different from the other two nature-inspired algorithms EAs and SIAs, the PPAs are mostly inspired by physical rules in the nature. There are some popular PPAs such as central force optimization (CFO) [31], big-bang big-crunch (BBBC) [32], charged system search (CSS) [33] and gravitational search algorithm (GSA) algorithm [34]. These algorithms in the three classes of the nature-based algorithms have been used to solve complex computational optimization problems. However, fast convergence along with accuracy is still a challenge need to be solved for the NAs.

As a result, we develop a new nature-inspired algorithm named cognitive behavior optimization (COA) in this paper. Firstly, according to analyzing the classic behavior model of ABC and DE, the common phenomenon of finding food source is summarized as the process of exploration–communication–adjustment. Secondly, combining with social behaviors of human, a relative detailed cognitive behavioral model of swarm intelligence is proposed. Finally, based on the developed model, which contains three main behaviors: rough search, information exchange and share, and intelligent adjustment, COA is proposed which satisfy both fast convergence and accuracy for the selected global optimization problems in this paper.

Preliminary studies show that the NAs such as ABC, DE, GSA, CoBiDE, DSA, GWO, AMO, CMA-ES [35] and other well-known optimization algorithms are very promising and could be used as the compared algorithms for evaluating COA's performance in solving global optimization problems.

The rest of this paper is organized as follows. Section 2 reviews the two typical behavior models of NAs. Then the COA is presented in Section 3. The experiment sets and experimental analysis for different tests are presented in Section 4. Finally, Section 5 concludes the work and suggests some directions for the future.

2. Typical behavior model in NAs

Many researchers are looking forward to a perfect way to model the intelligence of nature creatures' behaviors so as to solve real world complex problems. Herein DE is a kind of evolutionary algorithm which has the ability of information exchange. Besides, ABC is typical bionic algorithm based on the self-organized model and swarm intelligence of bee colony in nature.

Based on optimization technique that models the foraging behavior of the honey bees in the nature, Karaboga proposed the ABC algorithm [4]. The model of artificial bees in ABC contains three groups namely employed bees, onlookers and scouts. It can be concluded from the behavior model of ABC algorithm that based on the division of the swarm. The model is composed of three kinds of behavior for three kinds of bees: preliminary search, accurate search and abandoning a food source. ABC algorithm finds the global optimal value through various individuals of local optimization behavior. Compared with other nature-based algorithms, the detailed behaviors of foraging food and cooperation in ABC are emphasized so as to make it have faster convergence rate. However, it is simplified by using selected individuals' information as the whole population's information to exchange. It may result in loss of useful information. Meanwhile, the primary search stage for employed bees and the accuracy search stage for onlooker bees have the same search equation. It unfits the demand of different search behavior in ABC model.

On the other hand, DE, proposed by Storn and Price [20], is a simple, yet powerful, evolutionary algorithm with the generate-and-test feature for global optimization. It can be summarized that the behavior model of DE is mainly composed of the mutation and crossover operation, which is a process of information exchange. In DE, each individual in the population is called a target vector. DE produces a mutant vector by making use of the mutation operator, which perturbs a target vector using the difference vector of other individuals in the population. Then, the crossover operator is applied to the target vector and the mutant vector to generate a trial vector. Finally, the trial vector competes with its target vector for survival according to their objective function values. In a word, the function of information exchange in DE, which is explained by the mutant and crossover operator, is emphasized and it plays a very important role in the DE model.

3. Cognitive behavior optimization algorithm (COA)

In this section, we propose a novel swarm intelligence algorithm named cognitive behavior optimization algorithm (COA) inspired by ABC and DE. Based on the behavior model of ABC and DE, cognitive behavior model is proposed and it contains three main behaviors: rough search, information exchange and share and intelligent adjustment.

3.1. Cognitive behavior model of COA

As we have known, the exploration refers to the ability to investigate the various unknown regions in the solution space to discover the global optimum, while the exploitation refers to the ability to apply the knowledge of the previous good solutions to find better solutions [36]. From the perspective of balancing the performance of exploration and exploitation, based on the biological behavior model of ABC and DE, the common process of finding food source for social insects is summarized as exploration–communication–adjustment. Meanwhile, combining with social behaviors of human, the cognitive behavior model is generalized as Fig. 1 shown. The model contains three main behaviors: rough search, information exchange and share, and intelligent adjustment, which compose the main parts of COA. As ABC designed, the population of COA is also divided into two important groups namely cognitive population (*Cpop*) and memory population (*Mpop*) in this model, which play different roles in the COA algorithm. The abstract behavior model of foraging food source is introduced as follows:

1) Rough search. For cognitive population *Cpop*, the individuals use the Levy flight and Gaussian random walk method to explore the search space randomly (RS and RS_1 in Fig. 1). Gaussian random walk method used in the optimization algorithm has a promising performance in finding global minima, while Levy flight converges faster than that of Gaussian random walk. As a result, the two random walk methods are used in this search behavior.

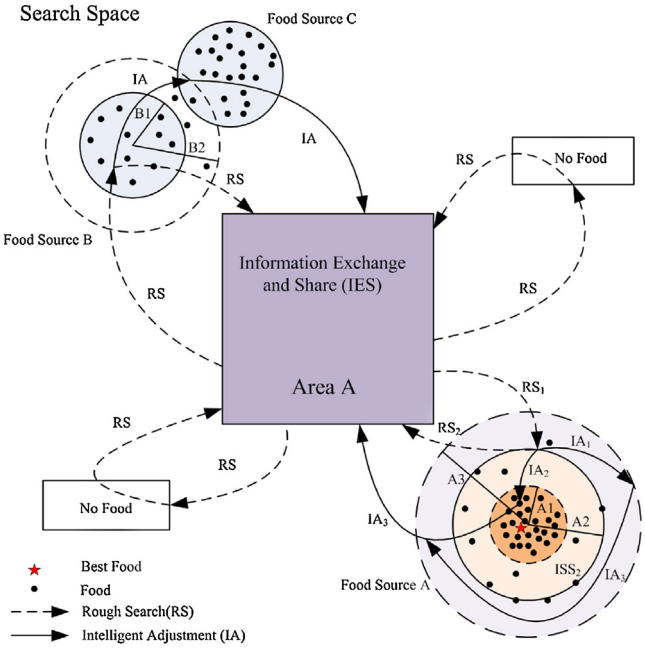

Fig. 1. Cognitive behavior model for foraging food source.

2) Information exchange and share. Based on the results of first stage, the memory population *Mpop* is introduced to store the food position of previous population for this behavior. In addition, the new crossover and mutation operator inspired by the DE algorithm is proposed to realize the behavior of information exchange and share (IES in Fig. 1).
3) Intelligent adjustment. The exploitation and exploration are balanced by the selection and adjustment in this stage (IA, IA_1 and IA_2 in Fig. 1). In this part, with the ranks of the individuals among the population, the chance of passing better food position to the next generation will increase. On the other hand, the chance of adjusting the position of individuals which have not obtained a good food position will increase.

In the early generations, the population prefers to explore a greater scope (RS in Fig. 1) because of lacking of useful information or avoiding loss information (Food Source C in Fig. 1) of the problems (B1 expands to B2 and A2 expands to A3 in Fig. 1). While, with the generation increased and useful information exchanged, more suitable solutions are remembered and the population trends to exploit some specific locations around the best food position (A2 narrows to A1 in Fig. 1). Rough search can random explore the search space. The information of individuals in *Cpop* and *Mpop* are exchanged by the information exchange and share operation. Meanwhile, the third behavior enables the individuals to fine-tune the position so as to retain the fairly performance to the next generation. In a word, these three behaviors and two groups in this model are collaboratively used to realize the foraging food phenomena and the detailed illustration of the three behaviors can be shown as follows.

3.2. Rough search

Before the rough search, according to the cognitive behavior model, the two important groups in COA are initialized as ABC designed by the Eqs. (1) and (2).

$$Cpop_i = rand \cdot (up - low) + low \tag{1}$$

$$Mpop_i = rand \cdot (up - low) + low \tag{2}$$

where $Cpop_i$ is the ith cognitive population and $Mpop_i$ is the ith memory population ($i = 1,2,3,\ldots,N/2$). And the size of $Cpop = Mpop = N/2$. N is the population size, *low* and *up* are the lower and upper bounds of the search space and *rand* is a random value selected from the range [0,1].

In this stage, there are two main steps. Firstly, the methods of Gaussian random walk and the Levy flight are used to generate the new individual around the current individual. As we known, Levy flight converges faster than Gaussian walk in some generations, but Gaussian random walk is more promising in finding the global best solutions. As a result, the two different random walk methods are used in this behavior to balance the exploration and exploitation as Eqs. (3) and (4) described.

$$Cpop'_i = Gaussian(G_{best}, \sigma) + (r_1 \cdot G_{best} - r_2 \cdot Cpop_i), \quad rand \le 0.5 \tag{3}$$

$$Cpop'_i = Cpop_i + \alpha \otimes Levy(s) \otimes (Cpop_i - G_{best}) | s = \frac{\mu}{|\upsilon|^{1/\beta}}, \quad rand > 0.5 \tag{4}$$

where r_1 *and* r_2 are random numbers produced in the range [0,1]. G_{bset} is the current best solution. The step size σ is calculated by the search direction $(Cpop_i - G_{best})$. Meanwhile, in order to balance the exploration and exploitation, the term $\log(g)/g$ is used to decrease the step size with the generation increased, where g is the current generation. As a result, the step size σ in Eq. (3) is calculated by Eq. (5)

$$\sigma = \left(\frac{\log(g)}{g}\right) \cdot (Cpop_i - G_{best}) \tag{5}$$

In Eq. (4), $\alpha > 0$ is the step size scaling factor. μ and υ are selected from the normal distribution $N(0, \sigma_\mu^2)$ and $N(0, \sigma_\upsilon^2)$, where

$$\sigma_\mu = \left(\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma\left[(1+\beta)/2\right]\beta 2^{(\beta-1)/2}}\right)^{1/\beta}, \quad \sigma_\upsilon = 1 \tag{6}$$

We use $\alpha = 0.01$ and $\beta = 3/2$ as CS set in [27]. If the new individual is better than the old solution, it is replaced by the new one; otherwise it keeps unchanged.

3.3. Information exchange and share

In this behavior, the improved crossover and mutation operation realized by the select probability *Pc* is proposed. Meanwhile the memory population *Mpop* which is used to store the individuals of previous population is introduced. The main process of the information exchange and share is introduced as follows.

Step 1: Calculate the crossover probability *Pc* by Eq. (7).

$$Pc_i = \frac{rank(fitCpop_i)}{N/2} \tag{7}$$

where $fitCpop_i$ is the fitness value of $Cpop_i$ and $rank(fitCpop_i)$ is considered as the rank of the individual $Cpop_i$ among the other individuals in the population.

Step 2: Confirm the *Mpop* by the 'if-then' rule and change the order of it:

$$\text{if } r_1 < r_2, \text{ then } Mpop = Cpop \tag{8}$$

$$Mpop = permuting\ (Mpop) \tag{9}$$

Step 3: Generate the new individual.

$$\begin{cases} Cpop''_{i,j} = Cpop_{k,j} + rand \cdot (G_{best,j} - Cpop_{i,j} + Mpop_{i,j} - Cpop_{h,j}), \\ \qquad \times rand \le Pc_i \\ Cpop''_{i,j} = Cpop_{i,j} + rand \cdot (Mpop_{i,j} - Cpop_{k,j}), \\ \qquad \times rand > Pc_i \end{cases} \tag{10}$$

where $i, k, h \in \{1, 2, 3, \ldots, N/2\}$, $i \neq k \neq h, j \in \{1, 2, 3, \ldots, D\}$ *and* D is the dimension of the problem. G_{best} is the current global best solution.

In this process, instead of the control parameter crossover rate *CR* in DE, Eq. (7) wants to state that the better the individual, the higher the probability. This equation is used to increase the chance of exchange the elements of individuals which have not obtained a good solution. Meanwhile, the chance of passing better solutions in the next behavior will increase. As a result, according to Eq. (10) without the scaling factor F in the standard DE, if $rand \le Pc_i$ is held for a new individual $Cpop_i$, the element of $Cpop_i$ is exchanged guided by the adjusted 'DE/best/2' so as to improve the exploitation performance, otherwise it is exchanged by the modified 'DE/rand/1' so as to improve the exploration performance. In addition, Eq. (8) ensures that *Mpop* belonging to a randomly selected previous generation as the memory population and remembers this historical population until it is changed. Eq. (9) changes the order of *Mpop* to enhance the capability of memory.

3.4. Intelligent adjustment process

The final operation is processed in the intelligent adjustment behavior. It is used to improve the quality of exploitation and the diversification property. In this process, the crossover probability *Pc* is also used. Different from the information exchange and share, if the condition $Pc_i > rand$ is held for a new individual, the current position of $Cpop_i$ is adjusted according to Eqs. (11) and (12), otherwise no adjustment occurs.

$$Cpop'''_i = Cpop_i + \varphi \cdot (Cpop_i - G_{best}) \quad |rand < 0.5 \tag{11}$$

$$Cpop'''_i = Cpop_i + \varphi \cdot (Cpop_i - Cpop_j) \quad |rand > 0.5 \tag{12}$$

where i is random selected index from $[1, 2, 3, \ldots, N/2]$ and $i \neq j$. φ is uniformly distributed random number restricted to $[-1,1]$. The adjusted individual will replace $Cpop_i$ if its fitness value is better than that of $Cpop_i$.

Some individuals of the population may overflow the allowed search space after the generation of new population. The individuals beyond the search-space limits are regenerated using the '**boundary control**' strategy. Its procedure is introduced as follows.

Step. 1 if $Cpop_{i,j} < low$ *or* $Cpop_{i,j} > up$
Step. 2 then $Cpop_{i,j} = rand \times (up - low) + low$

By combining the above three behaviors, the pseudo code of COA based on the fitness evaluations is presented as shown in Algorithm 1. From Algorithm 1, it is necessary to illustrate that the number of fitness evaluations for first two processes is N per one generation ($N/2 + N/2$). But for the last process (Intelligent Adjustment), the function is evaluated if $rand > Pc_i$. As a result, the total number of fitness evaluations per one generation cannot be statically determined.

Algorithm 1. Pseudo Code of COA

Settings: *low*, *up*, *N*, *D*, *ObjFun* and *MaxFES* (maximum function evolution numbers)
Begin
  Initialize the colony *Cpop* and *Mpop* according to Eq.(1) and Eq. (2);
  Evaluate the *Cpop* by $fitCpop_i = ObjFun(Cpop_i)$ and $FES = N/2$;
  $g = 1$;
  Repeat
    //Rough Search
    for i from 1 to $N/2$ **do**
      if $rand \le 0.5$ **then**
        $Cpop'_i = Gaussian(G_{best}, \sigma) + (r_1 \cdot G_{best} - r_2 \cdot Cpop_i)$ // r_1 *and* r_2 *are random values*;
      else
        $Cpop'_i = Cpop_i + \alpha \otimes Levy(s) \otimes (\text{Cpopi-}\ G_{best}\ ) ls = \mu / |v|^{1/\beta}$;
      end-if
      for j=1 to D **do** // *Boundary control strategy*
        if $Cpop_{i,j} < low$ or $Cpop_{i,j} > up$ **then**
          $Cpop_{i,j} = rand \times (up - low) + low$;
        end-if
      end-for
    end-for
    for i from 1 to $N/2$ **do** // *Function evaluation*
      $fitCpop_i' = ObjFun(Cpop_i')$;
      $FES = FES + 1$;
      if $fitCpop_i' < fitCpop_i$ **then**
        $Cpop_i = Cpop'_i$;
        $fitCpop_i = fitCpop_i'$;
      end-if
      if $FES == MaxFES$ **then**
        Memorize the best solution achieved so far and exit main repeat;
      end
    end-for
    //Information Exchange and Share
    Calculate the *Pc* through $Pc_i = rank(fitCpop_i)/(N/2)$
    if $r_1 < r_2$, **then** $Mpop = Cpop$, **end-if**
    $Mpop = permuting(Mpop)$
    for i from 1 to $N/2$ **do**
      for j from 1 to D **do**
        if $rand \le Pc_i$ **then**
          $Cpop''_{i,j} = Cpop_{k,j} + rand \cdot (G_{best,j} - Cpop_{i,j} + Mpop_{i,j} - Cpop_{h,j})$;
        else
          $Cpop''_{i,j} = Cpop_{i,j} + \text{rand} \cdot (Mpop_{i,j} - Cpop_{k,j})$;
        end-if
      end-for
      Check the boundary of $Cpop''_i$ through 'boundary control' strategy;
    end-for
    for i from 1 to $N/2$ **do** // *Function evaluation*
      $fitCpop_i'' = ObjFun(Cpop_i'')$;
      $FES = FES + 1$;
      if $fitCpop_i'' < fitCpop_i$ **then**
        $Cpop_i = Cpop''_i$;
        $fitCpop_i = fitCpop''_i$;
      end-if
      if $FES == MaxFES$ **then**
        Memorize the best solution achieved so far and exit main repeat;
      end
    end-for
    //Intelligent Adjustment
    Calculate the *Pc* through $Pc_i = rank(fitCpop_i)/(N/2)$;
    for i from 1 to $N/2$ **do**
      if $rand > Pc_i$ **then**
        if $rand < 0.5$ **then**
          $Cpop''' = Cpop_i + \varphi \cdot (Cpop_i - G_{best})$;
        else
          $Cpop''' = Cpop_i + \varphi \cdot (Cpop_i - Cpop_j)$;
        end-if
      end-if
      Check the boundary of $Cpop'''_i$ through 'boundary control' strategy;
      if $ObjFun(Cpop'''_i) < fitCpop_i$ **then** // *Function evaluation*
        $Cpop_i = Cpop'''_i$
        $FES = FES + 1$;
      end-if
      if $FES == MaxFES$ **then**
        Memorize the best solution achieved so far and exit main repeat;
      end
    end-for
    Memorize the best solution achieved so far and exit main repeat;
    $g = g + 1$;
  until $FES = MaxFES$
End

4. Experimental study

To study our COA algorithm, experiments are carried out on both unconstrained and constrained global optimization problems. For unconstrained problems, two types of benchmark functions including classic and modern benchmark functions (CEC 2014) are employed. For constrained benchmark functions, three engineering design optimization problems commonly usedin literature are

used. In addition, all the experiments in this section are performed on computer with 3.20 GHz Intel(R) Core(TM) i5-3470 processor and 4GB of RAM using MATLAB 2013a.

4.1. Test I – classic benchmark functions

To illustrate the relative success of COA on classic benchmark functions, a series of 20 widely used benchmarks selected from literature [29,37] are introduced in this test. Table 1 presents the detailed information about the different types of benchmarks, such as unimodal, multimodal, separable, non-separable. Our COA algorithm has been compared with six well-known nature-based algorithms: (1) EAs: DE, CoBiDE and DSA; (2) SIAs: GWO and ABC; (3) PPAs: GSA. Appendix A shows the download link of all algorithms used in this test.

In this test, the population size N is set to 50. The maximum function evolution numbers ($MaxFEs$) is set to 400,000 for f_1–f_{12} and 10,000 for f_{13}–f_{20}. For each test function, 30 independent runs were carried out with random seeds. To have a fair comparison, ABC is also modified according to [38]. For COA, the size of $Cpop = Mpop = N/2$ inspired by ABC algorithm. The parameter settings for the other compared algorithms in this test are detailed as follows:

1) DE: $F = 0.5$, $CR = 0.9$ as in [20] and the DE/rand/2/exp strategy is used;
2) CoBiDE: $pb = 0.4$, $ps = 0.5$ and mutation strategies and crossover strategies as in [23];
3) DSA: $p_1 = 0.3 \times rand$, $p_2 = 0.3 \times rand$ and objective direction method is used as in [24];
4) GWO: $a = 2\text{-}1 \times (2/MaxCycle)$ as in [30];
5) GSA: $G_o = 100$ and $a = 20$ [34];
6) ABC: $limit = (N{\cdot}D)/2$; size of employed-bee = onlooker-bee = (colony size)/2 as in [38].

Table 2 summarizes the statistical results including Mean, Worst, Best and SD of objective function values obtained by 30 independent runs for high-dimensional functions in Table 1. From Table 2, it can be found that COA nearly finds all the optimal values of the high-dimensional functions except function f_{05}, f_{07}, f_{11} and f_{12}. Compared with other 6 algorithms, COA exhibits the best performance on the unimodal nonseparable function f_{02} and

Table 1
Benchmark functions used in Test 1 (M: multimodal, U: unimodal, S: separable, N: nonseparable, D: dimension, range: limits of search space, optimum: global optimal value).

Test function	Name	Type	D	Range	Optimum
$f_{01}(x) = \sum_{i=1}^{D} x_i^2$	Sphere	US	30	[−100,100]	0
$f_{02}(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i-1}^{D} \lvert x_i \rvert$	Schwefel 2.22	UN	30	[−10,10]	0
$f_{03}(x) = \sum_{i=1}^{D} \left(\sum_{j-1}^{D} x_i\right)^2$	Schwefel 1.2	UN	30	[−100,100]	0
$f_{04}(x) = \max_i \left\{ \lvert x_i \rvert, 1 \le i \le D \right\}$	Schwefel 2.21	US	30	[−100,100]	0
$f_{05}(x) = \sum_{i=1}^{D} 100(x_{i+1}^2 - x_i^2)^2 + (x_i - 1)^2$	Rosenbrock	UN	30	[−30,30]	0
$f_{06}(x) = \sum_{i=1}^{D} (\lvert x_i + 0.5 \rvert)^2$	Step	US	30	[−100,100]	0
$f_{07}(x) = \sum_{i=1}^{D} i x_i^4 + random[0, 1)$	Quartic	US	30	[−1.28, 1.28]	0
$f_{08}(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	Rastrigin	MS	30	[−5.12, 5.12]	0
$f_9(x) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$	Ackley	MS	30	[−32,32]	8.8818e−16
$f_{10}(x) = \frac{1}{4000}\sum_{i=1}^{D}(x_i^2) - \left(\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)\right) + 1$	Griewank	MN	30	[−600,600]	0
$f_{11}(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_i) + \sum_{i-1}^{D-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_D - 1)\right\}$ $+ \sum_{i-1}^{D} u(x_i, 10, 100, 4) y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < a \end{cases}$	Penalized	MN	30	[−50,50]	0
$f_{12}(x) = 0.1\{\sin^2(3\pi x_i) + \sum_{i=1}^{D}(x_i - 1)^2[1 + \sin^2(3\pi x_i)] + (x_D - 1)^2[1 + \sin^2(2\pi x_D)]\}$ $+ \sum_{i-1}^{D} u(x_i, 5, 100, 4)$	Penalized2	MN	30	[−50,50]	0
$f_{13}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	Foxholes	MS	2	[−65.53, 65.53]	0.998004
$f_{14}(x) = \sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^{-1}$	Kowalik	MS	4	[−5,5]	0.0003075
$f_{15}(x) = 4x_1^2 - 2.1x_1^4 + 1/3x_1^6 + x_1x_2 - 4x_2^2 + x_2^4$	Six Hump Camel Back	MN	2	[−5,5]	−1.03163
$f_{16}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	Branin	MS	2	[−5,10] × [0,15]	0.398
$f_{17}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 +$ $(2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	Goldstein Price	MN	2	[−5,5]	3
$f_{18}(x) = -\sum_{i=1}^{4}(c_i\exp(-\sum_{j-1}^{3}a_{ij}(x_j - p_{ij})^2)$	Hartman 3	MN	3	[0,1]	−3.8628
$f_{19}(x) = -\sum_{i=1}^{4}(c_i\exp(-\sum_{j-1}^{6}a_{ij}(x_j - p_{ij})^2)$	Hartman 6	MN	6	[0,1]	−3.3220
$f_{20}(x) = -\sum_{i=1}^{5}((X - a_i)(X - a_i)^T + c_i)^{-1}$	Langermann	MN	4	[0,10]	−10.1532

Table 2
Best, worst, mean, and SD obtained by GSA, DSA, ABC, DE, CoBiDE, GWO and COA through 30 independent runs on 12 high-dimensional functions from Test 1 (Best: the best solution, Worst: the worst solution, Mean: mean solution, SD: standard deviation).

No.	Statistics	GSA	DSA	ABC	DE	CoBiDE	GWO	COA
f_{01}	Best	3.8715e−18	6.3612e−57	3.0297e−16	1.9365e−146	1.0886e−74	**0**	**0**
	Worst	1.1553e−17	3.5277e−52	9.5074e−16	1.6620e−142	4.8820e−71	**0**	**0**
	Mean	8.0433e−18	3.5854e−53	5.3075e−16	1.8052e−143	2.7231e−72	**0**	**0**
	SD	1.8760e−18	7.9075e−53	1.3318e−16	5.2089e−143	8.8889e−72	**0**	**0**
f_{02}	Best	1.1240e−08	2.9905e−32	9.9370e−16	1.1036e−78	8.0648e−36	**0**	**0**
	Worst	1.7516e−08	3.5783e−29	1.4428e−15	7.4164e−76	1.9765e−34	**0**	**0**
	Mean	1.4194e−08	4.0011e−30	1.2970e−15	1.6266e−76	4.7503e−35	**0**	**0**
	SD	1.7586e−09	8.5072e−30	1.2680e−16	2.2740e−76	3.9202e−35	**0**	**0**
f_{03}	Best	2.2394e−17	4.1069e+01	1.7374e+03	2.0626e−24	3.5188e−23	0	**0**
	Worst	8.3318e−17	1.6976e+02	4.9009e+03	1.9898e−19	4.2270e−20	5.8598e−201	**0**
	Mean	4.3696e−17	1.0224e+02	3.0600e+03	3.0041e−20	4.4107e−21	1.9533e−202	**0**
	SD	1.5156e−17	3.6830e+01	6.6910e+02	6.0708e−20	8.7461e−21	0	**0**
f_{04}	Best	1.0492e−09	1.5621e−05	4.9072e−02	2.4692e+00	6.5995e−12	6.1607e−147	**0**
	Worst	2.1041e−09	2.4706e−04	3.1778e−01	1.4579e+01	2.2837e−10	6.1674e−147	**0**
	Mean	1.6937e−09	6.4289e−05	1.3144e−01	9.0613e+00	4.1809e−11	6.1629e−147	**0**
	SD	2.4867e−10	4.9680e−05	4.9466e−02	3.6604e+00	4.2785e−11	1.8736e−150	**0**
f_{05}	Best	1.5801e+01	1.6455e−05	7.9367e−04	1.2734e+01	**1.5127e−24**	2.5156e+01	4.1238e−04
	Worst	1.6945e+01	8.0846e+01	2.5368e+00	2.1684e+01	3.9866e+00	2.7076e+01	**4.0544e−01**
	Mean	1.6612e+01	3.6460e+01	2.9110e−01	1.8254e+01	1.3289e−01	2.5985e+01	**8.7507e−02**
	SD	2.5785e−01	3.0616e+01	5.3709e−01	2.3371e+00	7.2785e−01	9.6215e−01	**1.3409e−01**
f_{06}	Best	5.0995e−18	**0**	3.2591e−16	0	**0**	5.0089e−09	**0**
	Worst	1.5623e−17	**0**	7.6612e−16	4.4373e−31	**0**	1.3087e−07	3.6978e−32
	Mean	8.6656e−18	**0**	5.4436e−16	4.9304e−32	**0**	1.7831e−08	1.7873e−32
	SD	2.3277e−18	**0**	8.8705e−17	1.3945e−31	**0**	2.9531e−08	1.1693e−32
f_{07}	Best	6.8563e−03	1.4503e−03	2.8355e−02	1.4043e−01	1.4712e−03	**3.4510e−06**	3.4479e−05
	Worst	1.7396e−02	6.9932e−03	8.3129e−02	3.4206e−01	4.8719e−03	**5.6173e−05**	7.3040e−04
	Mean	1.0530e−02	4.4783e−03	5.6886e−02	2.2256e−01	3.0542e−03	**1.3332e−05**	2.5809e−04
	SD	2.7981e−03	1.3682e−03	1.1645e−02	7.7071e−02	8.8752e−04	**1.5691e−05**	2.4550e−04
f_{08}	Best	3.9798e+00	**0**	**0**	7.9597e+00	**0**	**0**	**0**
	Worst	2.1889e+01	**0**	**0**	1.7909e+01	**0**	**0**	**0**
	Mean	1.3498e+01	**0**	**0**	1.4133e+01	**0**	**0**	**0**
	SD	4.2909e+00	**0**	**0**	2.9939e+00	**0**	**0**	**0**
f_{09}	Best	1.6126e−09	4.4409e−15	2.9310e−14	4.4409e−15	4.4409e−15	7.9936e−15	**8.8818e−16**
	Worst	2.7182e−09	7.9936e−15	4.3521e−14	7.9936e−15	4.4409e−15	7.9936e−15	**8.8818e−16**
	Mean	2.2263e−09	6.2172e−15	3.6771e−14	7.2831e−15	4.4409e−15	7.9936e−15	**8.8818e−16**
	SD	2.6882e−10	1.8067e−15	4.5084e−15	1.4980e−15	0	0	**0**
f_{10}	Best	0	**0**	0	0	**0**	**0**	**0**
	Worst	7.3960e−03	**0**	7.3961e−03	1.2316e−02	**0**	**0**	**0**
	Mean	4.9307e−04	**0**	2.4654e−04	1.2316e−03	**0**	**0**	**0**
	SD	1.8764e−03	**0**	1.3503e−03	3.8947e−03	**0**	**0**	**0**
f_{11}	Best	3.8128e−20	**1.5705e−32**	2.8351e−16	1.5705e−32	**1.5705e−32**	1.9671e−02	**1.5705e−32**
	Worst	1.0367e−01	**1.5705e−32**	7.4216e−16	1.6996e−32	**1.5705e−32**	1.9706e−02	1.6109e−32
	Mean	3.4556e−03	**1.5705e−32**	5.1711e−16	1.5964e−32	**1.5705e−32**	1.9688e−02	1.5789e−32
	SD	1.8927e−02	**5.5674e−48**	8.6262e−17	3.9668e−34	**5.5674e−48**	1.0855e−05	1.3645e−34
f_{12}	Best	5.4431e−19	**1.3498e−32**	2.9696e−16	1.3498e−32	**1.3498e−32**	5.4300e−09	**1.3498e−32**
	Worst	8.5542e−03	**1.3498e−32**	7.5235e−16	1.0987e−02	**1.3498e−32**	1.9408e−07	1.0987e−02
	Mean	2.8514e−04	**1.3498e−32**	5.2851e−16	3.2962e−03	**1.3498e−32**	3.1903e−08	1.0987e−03
	SD	1.5618e−03	**5.5674e−48**	9.2982e−17	5.3074e−03	**5.5674e−48**	4.3859e−08	4.1648e−03

f_{03}. For f_{05}, COA has the best mean value. For unimodal separable functions f_{01}, f_{04} and f_{06}, COA performs better than the compared algorithm in the terms of the accuracy and robustness. GWO has a promising performance for optimizing f_{07} and COA exhibits the second best performance on this function. With regard to the multimodal nonseparable functions (f_{10}–f_{12}), COA performs better than GSA, ABC and DE in the terms of the accuracy and robustness. And COA performs very similar with DSA, CoBiDE and GWO for f_{10}. However, DSA and CoBiDE perform best on f_{11} and f_{12} compared with other algorithms. According to the test results of multimodal separable functions, COA outperforms the other algorithms on f_{09}. Meanwhile, COA performs comparable to DSA, CoBiDE and GWO on f_{08}.

For the functions f_{13}–f_{20} with few local minima due to low dimensions, Table 3 reports the final results obtained by 30 independent runs. Different from the high-dimensional multimodal functions, these functions deal with low and fixed dimensions. As shown in Table 3, almost all algorithms could reach optimal or near optimal solutions for all test functions. However, COA performs best on f_{14}, f_{15}, f_{17}, f_{18} and f_{20} according to Mean and SD values. DE is similar to CoBiDE for optimizing f_{13} and it performs best on f_{16}. For f_{16} and f_{19}, GSA exhibits the promising performance.

From a graphical point of view, Figs. 2 and 3 present the convergence curves abstained by the seven algorithms with 30 independent runs. When examined Figs. 2 and 3, COA converges rapidly for most test functions. Especially for the unimodal

Table 3
Best, Worst, Mean and SD obtained by GSA, DSA, ABC, DE, CoBiDE, GWO and COA through 30 independent runs on 8 well known low-dimensional functions from Test 1 (Best: the best solution, Worst: the worst solution, Mean: mean solution, SD: standard deviation).

No.	Statistics	GSA	DSA	ABC	DE	CoBiDE	GWO	COA
f_{13}	Best	1.0197e+00	9.9800e−01	9.9800e−01	9.9800e−01	**9.9800e−01**	9.9800e−01	**9.9800e−01**
	Worst	1.6901e+01	9.9800e−01	9.9800e−01	9.9800e−01	**9.9800e−01**	1.9920e+00	**9.9800e−01**
	Mean	6.7683e+00	9.9800e−01	9.9800e−01	9.9800e−01	**9.9800e−01**	1.0311e+00	**9.9800e−01**
	SD	4.1213e+00	1.3518e−12	1.6956e−15	7.4015e−17	**4.1233e−17**	1.8148e−01	4.4197e−14
f_{14}	Best	2.3753e−03	6.5102e−04	4.9301e−04	3.0749e−04	6.3449e−04	4.5698e−04	**3.0749e−04**
	Worst	1.8408e−02	2.5556e−03	3.0656e−03	3.1467e−04	1.1548e−03	7.2313e−03	**3.1179e−04**
	Mean	7.2147e−03	1.4667e−03	1.0553e−03	3.0876e−04	1.0158e−03	6.9117e−04	**3.0860e−04**
	SD	4.0195e−03	4.6672e−04	4.7219e−04	2.2730e−06	7.6532e−05	1.2353e−03	**1.4357e−06**
f_{15}	Best	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	**−1.0316e+00**
	Worst	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	**−1.0316e+00**
	Mean	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	−1.0316e+00	**−1.0316e+00**
	SD	4.7908e−16	1.0461e−10	4.5356e−16	7.4015e−16	5.5319e−16	4.1208e−09	**1.8130e−16**
f_{16}	Best	**3.9789e−01**	3.9789e−01	3.9789e−01	**3.9789e−01**	3.9789e−01	3.9789e−01	**3.9789e−01**
	Worst	**3.9789e−01**	3.9789e−01	3.9789e−01	**3.9789e−01**	3.9789e−01	3.9789e−01	**3.9789e−01**
	Mean	**3.9789e−01**	3.9789e−01	3.9789e−01	**3.9789e−01**	3.9789e−01	3.9789e−01	**3.9789e−01**
	SD	**0**	1.3844e−06	2.2537e−09	**0**	6.5364e−09	4.7870e−06	2.7007e−14
f_{17}	Best	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00	**3.0000e+00**
	Worst	3.0000e+00	3.0082e+00	3.0001e+00	3.0000e+00	3.0000e+00	3.0000e+00	**3.0000e+00**
	Mean	3.0000e+00	3.0005e+00	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00	**3.0000e+00**
	SD	5.0688e−15	1.4823e−03	2.4947e−05	6.2804e−16	1.9792e−15	1.7935e−05	**2.3622e−16**
f_{18}	Best	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	**−3.8628e+00**
	Worst	−3.8513e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8619e+00	**−3.8628e+00**
	Mean	−3.8594e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8627e+00	**−3.8628e+00**
	SD	3.1051e−03	8.7879e−09	6.4192e−13	9.0043e−16	2.5391e−15	2.1408e−04	**7.8330e−16**
f_{19}	Best	**−3.3220e+00**	−3.3220e+00	−3.3220e+00	−3.3220e+00	−3.3220e+00	−3.2016e+00	−3.3220e+00
	Worst	**−3.3220e+00**	−3.3219e+00	−3.3220e+00	−3.2031e+00	−3.2031e+00	−3.2012e+00	−3.2031e+00
	Mean	**−3.3220e+00**	−3.3220e+00	−3.3220e+00	−3.2982e+00	−3.2625e+00	−3.2016e+00	−3.3101e+00
	SD	**4.1869e−16**	3.8015e−05	2.3236e−08	5.0130e−02	6.2661e−02	1.2288e−04	3.7597e−02
f_{20}	Best	−1.0153e+01	−1.0153e+01	−1.0153e+01	−1.0153e+01	−1.0153e+01	−1.0153e+01	**−1.0153e+01**
	Worst	−2.6305e+00	−8.5866e+00	−1.0099e+01	−3.7243e+00	−2.6829e+00	−2.6301e+00	**−1.0153e+01**
	Mean	−6.1673e+00	−1.0005e+01	−1.0151e+01	−1.0180e+01	−9.9042e+00	−9.4285e+00	**−1.0153e+01**
	SD	3.7922e+00	3.1299e−01	9.9525e−03	1.2193e+00	1.3639e+00	2.2146e+00	**8.2419e−09**

functions which are used to test the performance of the convergence for the NAs, COA has faster convergence rate for f_{02}, f_{03}, f_{04} and f_{07} compared with other algorithms. GWO performs better for f_{01} on convergence rate. For the most multimodal functions, COA also presents the promising performance in convergence rate as shown in Figs. 2 and 3.

For detailed comparison, the Wilcoxon Signed Ranks Test [39], in which Iman–Davenport's procedure is used as a post hoc procedure, is firstly used to make a pairwise comparison so as to illustrate the superiority of COA on the 20 test functions. The statistical results are summarized in Table 4. In Table 4, the 'R+' represents the sum of ranks for the problems which COA algorithm outperformed the compared one, and the 'R−' refers to the sum of ranks for the opposite. '+' indicates COA having superior performance than the compared algorithm. '−' indicates the opposite. The sign '≈' means there is no statistical difference between COA and the compared algorithm. According to the values of '+', '≈' and '−' in Table 4, we can conclude that COA algorithm outperforms the compared six algorithms in solving the problems used in the test. According to the last line of Table 4, overall COA is the statistically best compared with other six algorithms at a 0.05 significance level.

To further detect the significant differences between COA and the six competitors, the Friedman's test is carried out, in which Iman–Davenport's procedure is used as a post hoc procedure. It is necessary to emphasize that the Friedman's test is accomplished in this paper by using the KEEL software [40]. Table 5 summarizes the average rankings of Mean and SD values among seven algorithms obtained by the Friedman's test. The average rankings can be used as indicators to illustrate how successful the algorithm is. In other words, the lower the rank, the more successful the algorithm is [41]. From Table 5, it can be seen that the COA ranks first on Mean and SD value. For the mean value, Friedman statistic considering reduction performance distributed according to chi-square with 6 degrees of freedom is 9.208929. And the Iman and Davenport statistic considering reduction performance distributed according to *F*-distribution with 6 and 114 degrees of freedom is 1.579276 at $\alpha = 0.05$, and the critical value is 2.17 which can be found in the statistical tables [42]. According to [41], Critical difference *CD* is calculated by $CD = q_\alpha \sqrt{k(k+1)/6N}$ and $CD = 2.17\sqrt{7 \cdot 8/6 \cdot 20} = 1.48$, where q_α is the critical value, *N* is the number of problems, *k* the number of algorithms. The comparisons of all algorithms against each other using the critical difference from the Friedman and Iman and Davenport tests for Mean and SD are shown in Fig. 4. From Fig. 4(a), it can be found that there is significantly difference between COA and GSA on mean values. However, the group of the algorithms including COA, CoBiDE, ABC, DSA, GWO and DE is not significantly different each other. From Fig. 4(b), there is significantly difference between COA and GSA, DSA, GWO on SD values. However, the group of the algorithms including COA, CoBiDE, ABC, and DE is not significantly different each other.

With regard to the time consumption for optimization of these 20 test functions, Table 6 presents the results of mean time obtained by the COA and other compared algorithms with 30 independent runs. We also rank the algorithms from smallest mean time of the test functions to the highest mean time. As it can be concluded from Table 6, DSA performs the best among the algorithms.

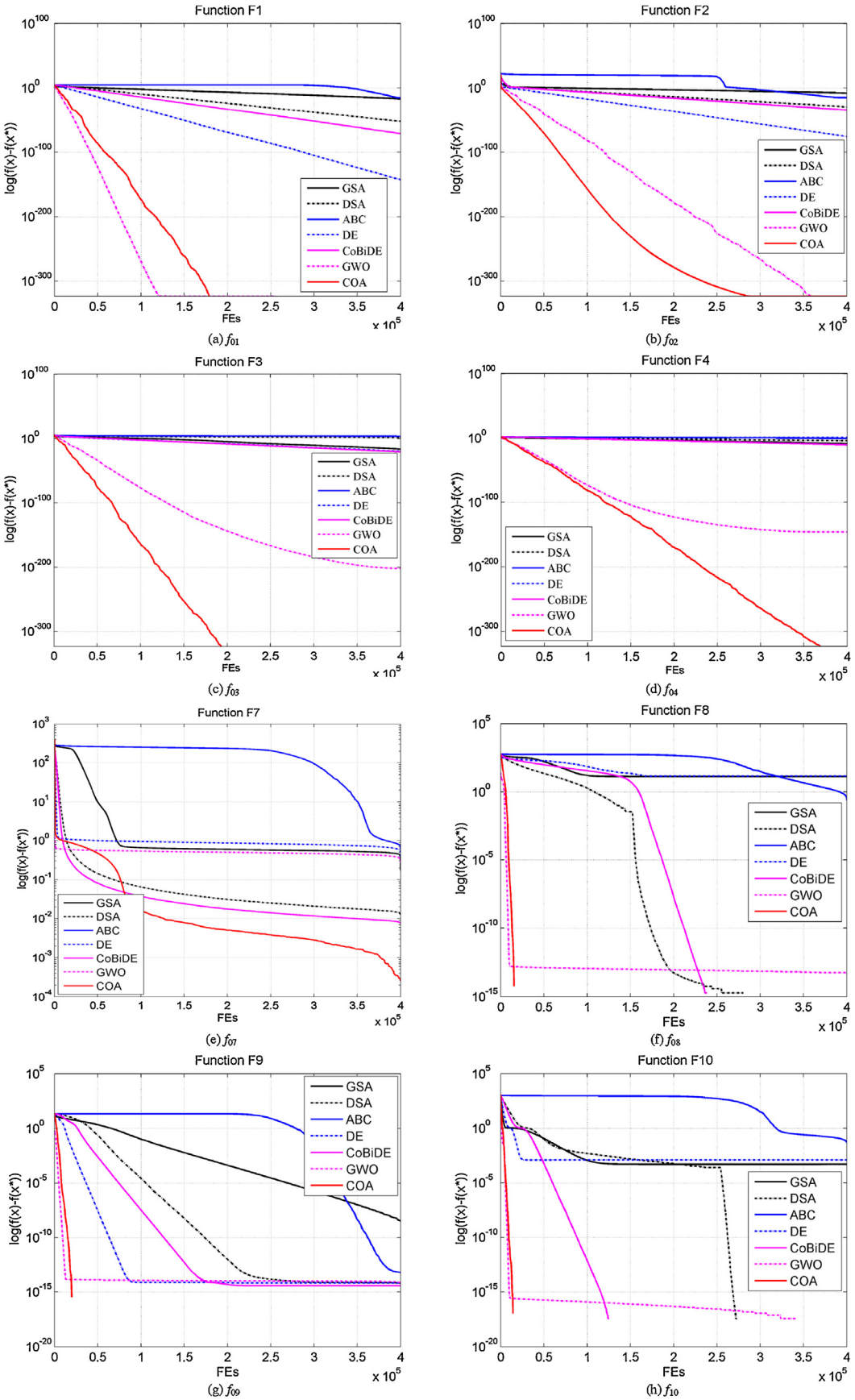

Fig. 2. Evolution of the mean function error values derived from GSA, DSA, ABC, DE, CoBiDE, GWO and COA vs. the number of FEs on 8 classic benchmarks.

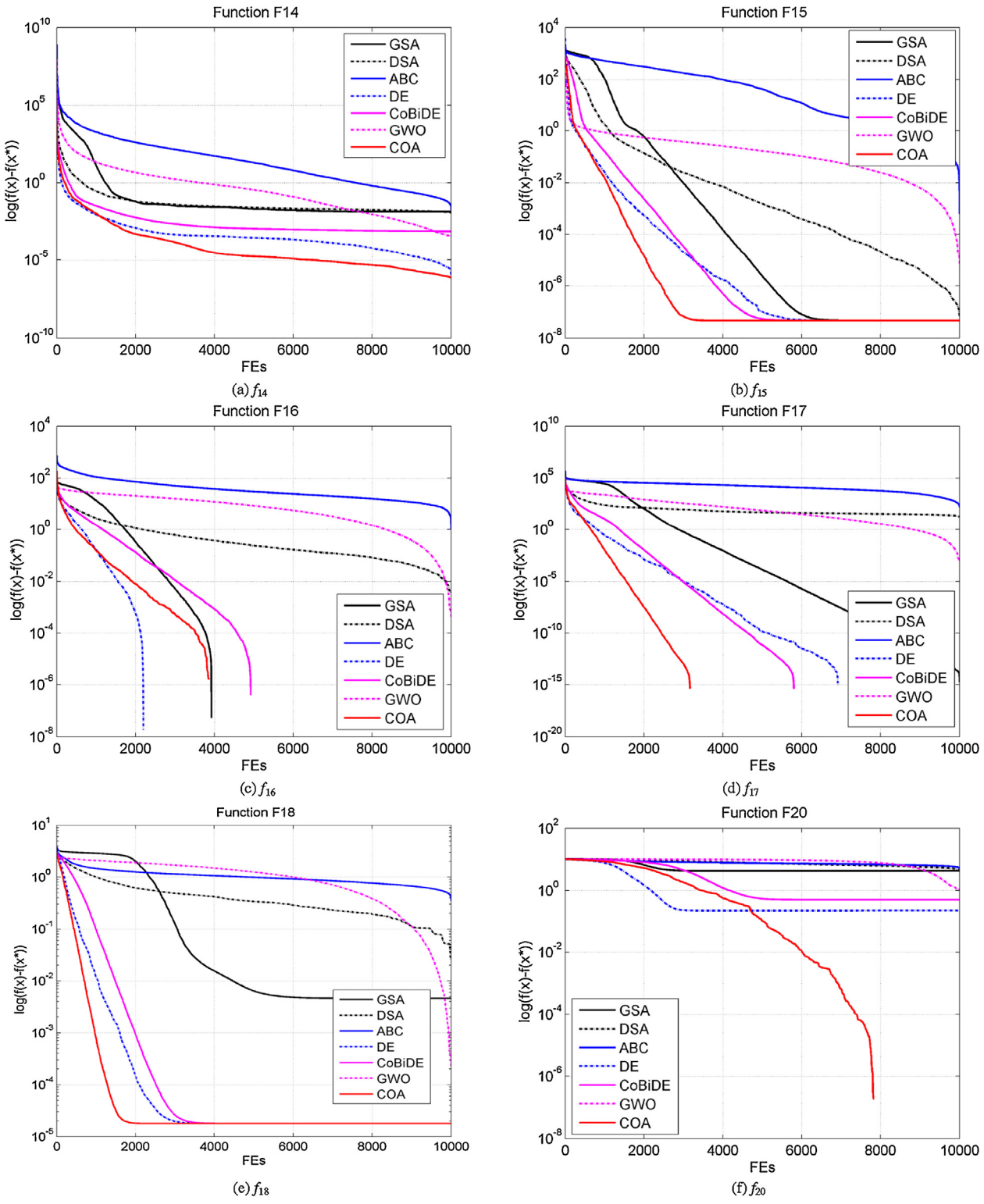

Fig. 3. Evolution of the mean function error values derived from GSA, DSA, ABC, DE, CoBiDE, GWO and COA vs. the number of FEs on 6 classic benchmarks.

Compared with ABC and DE, COA ranks fifth and it performs better than DE. As COA's information exchange and share behavior realize the information exchange among all the individuals and the intelligent adjust process is performed for the selected individuals per generation, while the selected or special individuals in ABC perform the information exchange operation for onlooker bees and scout bees. As a result, COA consumes more time than ABC.

4.1.1. COA search behaviors and parameter analysis

In this section, the effect of COA parameter, as well as three behaviors of COA is experimentally studied. To evaluate the effect

Table 4
The results of Wilcoxon Signed Ranks Test based on the best solution for each benchmarks with 30 independently runs ($\alpha = 0.05$).

No.	GSA vs. COA				DSA vs. COA				ABC vs. COA				DE vs. COA				GWO vs. COA				CoBiDE vs. COA			
	p-Value	R+	R−	Win	*p*-Value	R+	R−	Win	*p*-Value	R+	R−	Win	*p*-Value	R+	R−	Win	*p*-Value	R+	R−	Win	*p*-Value	R+	R−	Win
f_{01}	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	**1**	**0**	**0**	≈	1.7344e−06	465	0	+
f_{02}	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.6912e−06	465	0	+	**1**	**0**	**0**	≈	1.7344e−06	465	0	+
f_{03}	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.6912e−06	465	0	+	**1**	**1**	**0**	≈	1.7344e−06	465	0	+
f_{04}	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7333e−06	465	0	+	1.7344e−06	465	0	+
f_{05}	1.7344e−06	465	0	+	4.7292e−06	455	10	+	**1.1093e−01**	**310**	**155**	≈	1.7344e−06	465	0	+	1.7344e−06	465	0	+	**3.1108e−05**	**30**	**435**	−
f_{06}	1.7344e−06	465	0	+	**5.2251e−06**	**0**	**378**	−	1.7344e−06	465	0	+	9.6938e−02	120	258	+	7.0538e−07	465	0	+	**5.2251e−06**	**0**	**378**	−
f_{07}	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	**1.7289e−06**	**0**	**465**	−	1.7344e−06	465	0	+
f_{08}	1.7062e−06	465	0	+	**1**	**0**	**0**	≈	**1**	**0**	**0**	≈	1.6343e−06	465	0	+	**1**	**0**	**0**	≈	**1**	**0**	**0**	≈
f_{09}	1.7344e−06	465	0	+	8.2072e−07	465	0	+	1.1320e−06	465	0	+	3.2578e−07	465	0	+	4.3205e−08	465	0	+	4.3205e−08	465	0	+
f_{10}	**5.0000e−01**	**3**	**0**	≈	**1**	**0**	**0**	≈	5.0410e−06	276	0	+	3.9063e−03	145	0	+	**1**	**0**	**0**	≈	**1**	**0**	**0**	≈
f_{11}	1.7344e−06	465	0	+	**2.4414e−04**	**0**	**91**	−	1.7344e−06	465	0	+	**1.2425e−01**	**188**	**88**	≈	1.7344e−06	465	0	+	**2.4414e−04**	**0**	**91**	−
f_{12}	2.7624e−03	378	87	+	**4.8828e−04**	**0**	**78**	−	2.7646e−03	378	87	+	**1.0401e−01**	**110**	**43**	≈	2.7495e−03	378	87	+	4.8828e−04	78	0	+
f_{13}	1.7344e−06	465	0	+	5.9269e−03	264	61	+	**2.0377e−02**	**24**	**113**	−	3.9909e−05	0	231	−	1.4327e−06	465	0	+	**3.9909e−05**	**0**	**231**	−
f_{14}	1.7344e−06	465	0	+	1.7344e−06	465	0	+	2.3534e−06	462	3	+	2.7367e−03	87	378	−	1.7344e−06	465	0	+	1.7344e−06	465	0	+
f_{15}	**1**	**0**	**0**	≈	2.5614e−06	435	0	+	**1**	**0**	**0**	≈	2.5000e−01	0	6	≈	1.0144e−07	465	0	+	**1**	**0**	**0**	≈
f_{16}	**4.8828e−04**	**0**	**78**	−	1.7344e−06	465	0	+	**2.3498e−05**	**342**	**9**	−	4.8828e−04	0	78	−	1.6721e−06	465	0	+	**8.9038e−01**	**55**	**50**	≈
f_{17}	6.7632e−05	278	22	+	1.7344e−06	465	0	+	1.7333e−06	465	0	+	**5.0000e−01**	**0**	**3**	≈	1.5973e−06	465	0	+	4.3205e−03	65	0	+
f_{18}	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.5625e−02	28	0	+	1	0	0	+	1.5922e−06	465	0	+	**1**	**0**	**0**	≈
f_{19}	1.6626e−06	465	0	+	2.7367e−03	378	87	+	**3.7043e−01**	**276**	**189**	≈	**1.6428e−01**	**300**	**165**	≈	1.7333e−06	465	0	+	1.6912e−06	465	0	+
f_{20}	1.7116e−06	465	0	+	1.7344e−06	465	0	+	2.3038e−02	343	122	+	1.7344e−06	465	0	+	1.7246e−06	465	0	+	1.7333e−06	465	0	+
+/≈/−		17/2/1				15/2/3				16/4/2				12/5/3				14/5/1				11/5/4		

Table 5
Average rankings of GSA, DSA, ABC, DE, CoBiDE, GWO and COA according to the Friedman test at $\alpha = 0.05$ significance.

Algorithms	GSA	DSA	ABC	DE	CoBiDE	GWO	COA
Mean rankings[a]	4.975	3.9	3.875	4.4	3.5	4.2	**3.15**
SD rankings[b]	4.85	4.3	3.9	4.225	3.4	4.55	**2.775**

a: 1) Friedman statistic considering reduction performance distributed according to chi-square with 6 degrees of freedom: 9.208929.
2) *p*-Value computed by Friedman Test: 0.16216453212480603.
3) Iman and Davenport statistic considering reduction performance distributed according to *F*-distribution with 6 and 114 degrees of freedom: 1.579276, and the critical value: 2.17.
4) *p*-Value computed by Iman and Daveport Test: 0.15946975240197525.
b: 1) Friedman statistic considering reduction performance distributed according to chi-square with 6 degrees of freedom: 13.0125.
2) *p*-Value computed by Friedman Test: 0.04283787485230284.
3) Iman and Davenport statistic considering reduction performance distributed according to *F*-distribution with 6 and 114 degrees of freedom: 2.510901, and the critical value: 2.17.
4) *p*-Value computed by Iman and Daveport Test: 0.03835557770818043.

of each behavior in COA cognitive behavior model, three configurations are designed. The first one configuration (Configuration 1) is designed without rough search behavior. In the second configuration (Configuration 2), the information exchange and share behavior is removed from the COA. The last configuration (Configuration 3) is developed without the intelligent adjustment behavior. These three configurations along with COA are tested on the above 10 typical classic benchmark functions. The *MaxFEs* of each function are set as its previous condition. Table 7 represents the average results of 30 runs for each configuration. From Table 7, it can be observed that COA performs the best performance for the accuracy and robustness. Configuration 1 presents the worst optimization performance for the selected unimodal and multimodal benchmark functions. It means that the rough search behavior is rather important for COA because of the successful use of the random walk method.

Fig. 5 shows the convergence curves of the three configurations and COA algorithm. From Fig. 5, COA converges faster for the multimodal functions than the other configurations, while Configuration 3 performs better for the unimodal function Sphere and Schwefel 2.22. The obtained results from Table 7 and Fig. 5 prove that each behavior in COA can significantly affect the quality of final results, and it also shows that COA with three behaviors together make a coherent system to solve optimization problems more successfully.

As mentioned before, different from DE algorithm, COA does not have any special control parameters such as *CR* or *F* in DE. However, COA is sensitive to the population size *N* designed from the idea of ABC. In order to analyze the effect of *N* for the optimizing performance of COA, it is tested on the above 10 typical functions and the *MaxFEs* of each function are set as its previous condition. Table 8 lists the average results of 30 independent runs based on the different *N* values. From Table 8, it can be found that the optimization performance of COA is considerably improved as *N* is decreased under the same *MaxFEs*, while the time consumption is increased as a result. Figs. 6 and 7 show the effect of *N* on the convergence rate for several test functions and mean time consumption of the previous ten test functions. From Fig. 6, it can be found that COA with $N = 30$ converges faster than the other conditions. However, its time consumption is larger than that of other conditions. From the above results, *N* is better to set as 50–100 based on a trade-off between convergence rate and time consumption.

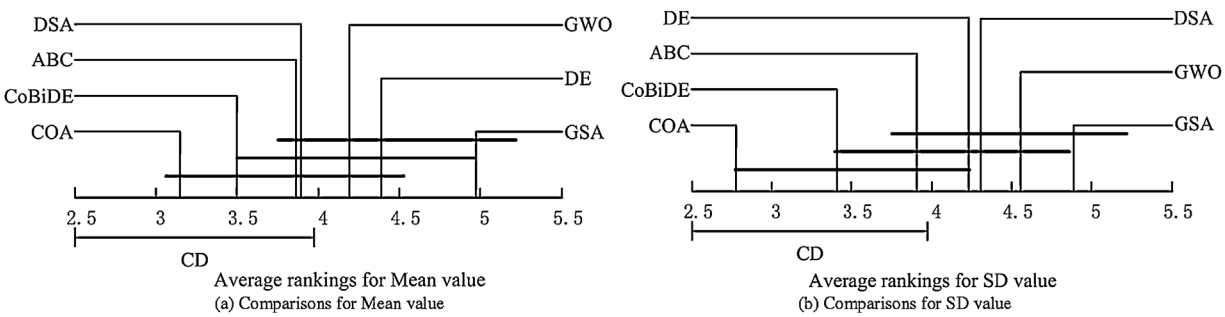

Fig. 4. Comparisons between 7 algorithms against each other. The groups of algorithms that are not significantly different are connected. The critical difference *CD* is visible below the axis.

Table 6
Mean time consumption in seconds about GSA, DSA, ABC, DE, CoBiDE, GWO and COA through 30 independent runs on 20 well known functions from Test 1 (Mean: mean time of 20 test functions; Rank: ranking based on mean time).

No.	GSA	DSA	ABC	DE	CoBiDE	GWO	COA
f_{01}	4.1512e+01	3.5856e+00	8.7326e+00	2.4966e+01	8.4022e+00	7.8675e+00	2.3771e+01
f_{02}	4.1871e+01	4.1198e+00	9.2993e+00	2.7016e+01	9.0310e+00	9.0972e+00	2.4014e+01
f_{03}	6.0363e+01	2.1996e+01	2.6146e+01	8.0326e+01	2.6709e+01	2.8232e+01	4.8702e+01
f_{04}	4.3282e+01	4.9601e+00	9.1208e+00	2.8654e+01	9.9415e+00	6.8059e+00	2.4208e+01
f_{05}	4.3417e+01	5.2573e+00	8.3999e+00	3.0727e+01	1.0130e+01	7.1720e+00	2.3791e+01
f_{06}	4.2144e+01	3.6031e+00	9.8282e+00	2.5311e+01	8.1865e+00	5.5479e+00	2.1152e+01
f_{07}	4.4565e+01	5.8115e+00	8.7063e+00	3.1835e+01	1.0883e+01	7.6986e+00	2.3470e+01
f_{08}	4.2338e+01	4.2564e+00	8.6548e+00	2.8224e+01	9.2636e+00	6.2516e+00	2.2060e+01
f_{09}	4.4107e+01	5.9646e+00	1.0102e+01	3.1637e+01	1.0942e+01	7.7482e+00	2.3358e+01
f_{10}	4.3318e+01	4.9698e+00	9.4470e+00	2.9683e+01	9.9853e+00	6.8100e+00	2.3877e+01
f_{11}	5.0954e+01	1.2754e+01	1.9990e+01	5.3164e+01	1.9301e+01	1.4338e+01	3.5618e+01
f_{12}	5.0873e+01	1.2539e+01	1.6945e+01	4.9647e+01	1.7142e+01	1.4222e+01	3.8947e+01
f_{13}	1.3457e+00	8.8713e−01	8.8918e−01	2.7224e+00	1.1837e+00	8.8289e−01	2.2653e+00
f_{14}	6.1769e−01	1.0825e−01	1.8004e−01	4.4766e−01	1.8865e−01	1.1496e−01	2.3866e−01
f_{15}	5.3628e−01	1.1217e−01	1.2707e−01	2.9749e−01	1.5546e−01	1.4510e−01	2.5296e−01
f_{16}	5.4599e−01	1.7182e−01	1.9454e−01	2.8632e−01	1.4800e−01	9.1680e−02	2.3196e−01
f_{17}	5.3044e−01	1.4965e−01	1.5425e−01	2.8033e−01	1.4707e−01	1.5547e−01	2.2964e−01
f_{18}	6.3704e−01	1.4838e−01	2.6167e−01	5.5231e−01	2.3049e−01	1.5416e−01	3.9583e−01
f_{19}	7.2376e−01	2.2512e−01	3.3705e−01	8.7628e−01	3.0805e−01	2.3121e−01	4.2327e−01
f_{20}	7.1944e−01	2.6863e−01	1.6864e−01	5.8567e−01	2.3768e−01	1.6324e−01	4.2818e−01
Mean	2.7720e+01	**4.5944e+00**	7.3842e+00	2.2362e+01	7.6258e+00	6.1865e+00	1.6871e+01
Rank	7	**1**	3	6	4	2	5

4.2. Test II – modern benchmark functions in CEC 2014

We also used 30 single objective and complex real-parameter unconstrained numerical optimization problems in IEEE CEC2014 to evaluate the performance of COA for solving complex problems in this section. Table 9 summarizes several features of the benchmark problems in this test. Detailed information on these problems is provided in [43]. To evaluate CEC 2014 benchmark functions, the results of COA are compared with state-of-the-art algorithms such as: DE [20], CoBiDE [23], AMO [29], GWO [30], CMA-ES [35] and ABC [38]. AMO is s novel swarm intelligence algorithms and it is similar with COA on the term of the algorithm structure. CMA-ES is a famous evolutionary algorithm and it is often used as a compared algorithm for complex benchmark functions. In this test, the population size N is set to 100, function's dimension is 30, *MaxFEs* is set to 300,000 with 51 independent runs [43]. It is necessary to emphasize that the Matlab codes of CMA-ES and AMO are directly taken from Appendix A. The other parameter settings for DE, CoBiDE, ABC,

Table 7
Statistical results obtained by COA's different behaviors for optimization of 10 typical classic functions.

No.	Configuration 1		Configuration 2		Configuration 3		COA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Sphere	1.0503e−103	2.0081e−103	**0**	**0**	**0**	**0**	**0**	**0**
Schwefel 2.22	3.1326e−54	2.3247e−54	**4.8540e−244**	**0**	**0**	**0**	**0**	**0**
Schwefel 2.21	3.0267e−04	6.8874e−04	**9.1899e−261**	**0**	**0**	**0**	**0**	**0**
Rastrigin	9.1536e+00	3.4721e+00	**0**	**0**	**0**	**0**	**0**	**0**
Ackley	6.9278e−15	1.7161e−15	**8.8818e−16**	**0**	**8.8818e−16**	**0**	**8.8818e−16**	**0**
Griewank	1.2321e−03	3.8962e−03	**0**	**0**	**0**	**0**	**0**	**0**
Foxholes	9.9803e−01	7.3133e−05	1.0974e+00	3.1434e−01	9.9800e−01	0	**9.9800e−01**	**9.2740e−16**
Branin	3.9789e−01	3.3044e−07	3.9789e−01	3.0814e−06	3.9789e−01	0	**3.9789e−01**	**3.8668e−14**
Goldstein Price	3.0000e+00	8.2098e−14	3.0000e+00	7.1480e−11	3.0000e+00	4.1466e−15	**3.0000e+00**	**2.8358e−15**
Hartman 3	−3.8628e+00	1.3098e−13	−3.8628e+00	3.8823e−10	−3.8628e+00	9.0043e−16	**−3.8628e+00**	**7.4015e−16**

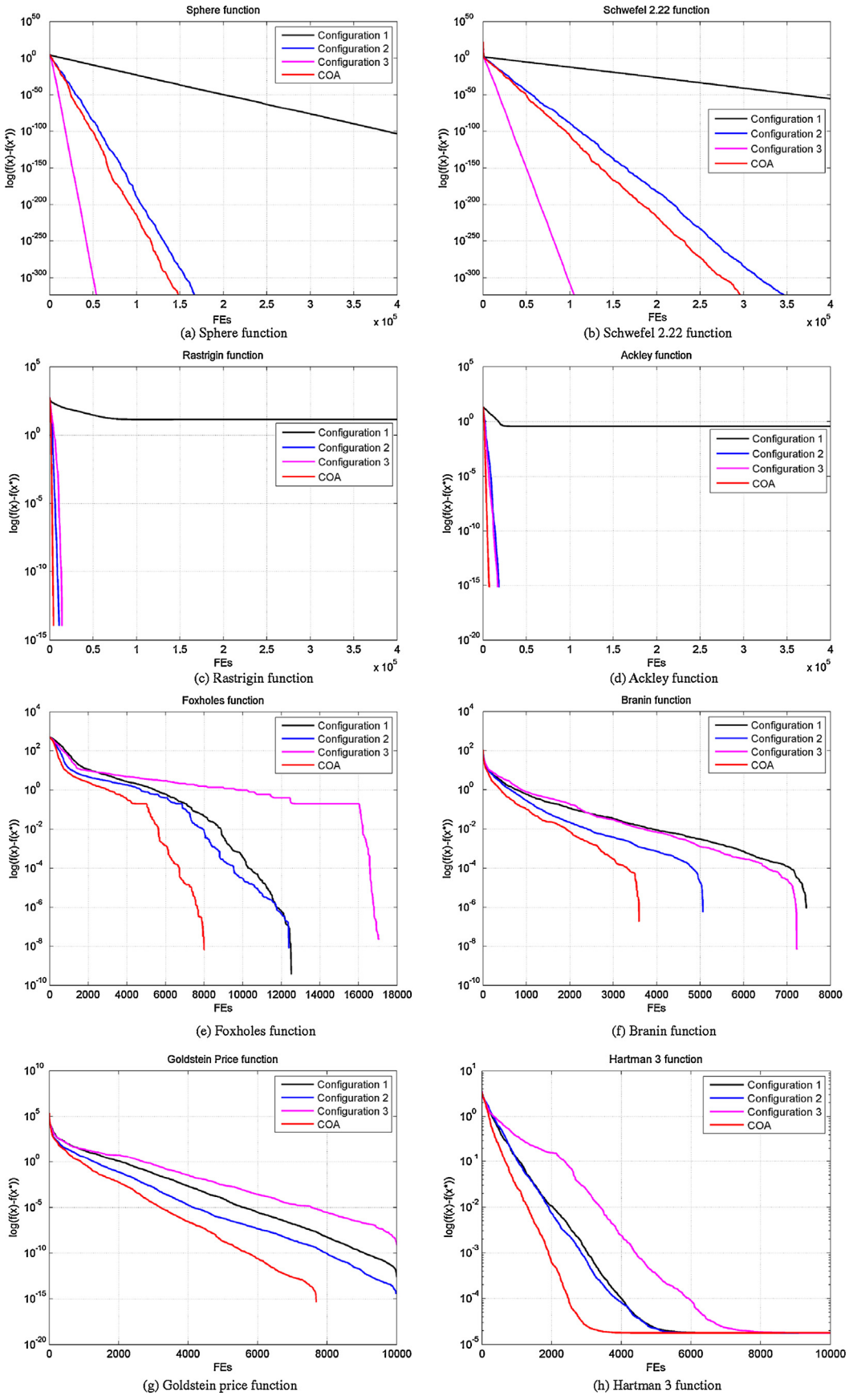

Fig. 5. Convergence curves of three configurations and COA on 8 classic benchmark functions.

Table 8
Statistical results obtained by COA with different values of N on 10 typical classic functions.

Function		$N=30$	$N=50$	$N=80$	$N=100$	$N=130$	$N=150$	$N=200$	$N=300$
Sphere	Mean	**0**	**0**	**0**	1.6134e−317	9.1323e−235	1.8958e−197	3.3882e−143	7.9160e−91
	SD	**0**	**0**	**0**	0	0	0	1.0714e−142	2.4213e−90
	Time	2.4305e+01	2.3721e+01	2.2995e+01	2.2972e+01	2.3059e+01	2.3280e+01	2.3137e+01	2.3309e+01
Schwefel 2.22	Mean	**0**	**0**	2.9331e−207	1.1535e−153	2.4654e−114	1.1998e−100	3.0777e−72	1.0935e−48
	SD	**0**	**0**	0	3.6294e−153	7.6830e−114	3.7756e−100	9.7324e−72	1.5187e−48
	Time	2.4157e+01	2.3463e+01	2.2995e+01	2.2972e+01	2.3059e+01	2.3280e+01	2.3137e+01	2.3309e+01
Schwefel 2.21	Mean	**0**	**0**	1.6120e−211	1.6835e−146	2.2300e−110	1.0879e−99	6.8765e−71	1.5473e−45
	SD	**0**	**0**	0	5.3236e−146	7.0519e−110	3.4371e−99	2.1742e−70	4.8915e−45
	Time	2.5321e+01	2.3607e+01	2.2276e+01	2.4443e+01	2.4693e+01	2.4555e+01	2.4516e+01	2.4856e+01
Rastrigin	Mean	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	SD	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Time	2.3929e+01	2.3631e+01	2.3297e+01	2.3253e+01	2.3624e+01	2.3509e+01	2.3643e+01	2.3817e+01
Ackley	Mean	**8.8818e−16**	**8.8818e−16**	**8.8818e−16**	**8.8818e−16**	**8.8818e−16**	**8.8818e−16**	**8.8818e−16**	**8.8818e−16**
	SD	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Time	2.6162e+01	2.5550e+01	2.5491e+01	2.5267e+01	2.5640e+01	2.5411e+01	2.5876e+01	2.5868e+01
Griewank	Mean	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	SD	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Time	2.6819e+01	2.6140e+01	2.6039e+01	2.5878e+01	2.6179e+01	2.5764e+01	2.5063e+01	2.5350e+01
Foxholes	Mean	**9.9800e−01**	9.9800e−01	9.9800e−01	9.9800e−01	9.9800e−01	9.9800e−01	9.9800e−01	9.9872e−01
	SD	**1.4803e−16**	2.3056e−14	2.6837e−09	3.8192e−08	8.6981e−07	8.9327e−07	1.5957e−06	1.4543e−03
	Time	1.2545e+00	1.2442e+00	1.2345e+00	1.2385e+00	1.2398e+00	1.2441e+00	1.2697e+00	1.2748e+00
Branin	Mean	**3.9789e−01**	3.9789e−01	3.9789e−01	3.9789e−01	3.9789e−01	3.9789e−01	3.9789e−01	3.9798e−01
	SD	**0**	1.9368e−13	1.4964e−07	2.3621e−06	9.6807e−06	1.9889e−05	2.5904e−05	1.1974e−04
	Time	2.5931e−01	2.3789e−01	2.4267e−01	2.4034e−01	2.4055e−01	2.3962e−01	2.4243e−01	2.4664e−01
Goldstein Price	Mean	**3.0000e+00**	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00	3.0000e+00
	SD	**2.0040e−15**	2.4143e−15	1.0867e−13	4.1738e−10	2.1665e−08	1.2399e−07	6.9477e−06	6.1676e−05
	Time	2.5995e−01	2.4999e−01	2.4486e−01	2.4676e−01	2.4109e−01	2.4134e−01	2.4189e−01	2.4333e−01
Hartman 3	Mean	−3.8628e+00	**−3.8628e+00**	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00	−3.8628e+00
	SD	9.0043e−16	**6.4525e−16**	1.5262e−12	1.7469e−09	2.1353e−08	5.2692e−08	4.3514e−07	5.5584e−06
	Time	3.8151e−01	3.6843e−01	3.6699e−01	3.6349e−01	3.6673e−01	3.6030e−01	3.6670e−01	3.6870e−01

GWO and COA are the same as Test I used. The parameter settings of other two algorithms are detailed as follows:

1) AMO: The number of animals in each group was set to 5 [29];
2) CMA-ES: $\sigma = 0.25$ and $\mu = \left\lfloor \frac{4+\lfloor 2\cdot\log(N)\rfloor}{2} \right\rfloor$ as in [35].

Tables 10–13 summarize the Mean and SD of the objective function values over 51 independent runs for each problem. We rank the algorithms from smallest mean solution to the highest solution. At last of the tables, the average ranks and the overall ranks obtained by algorithms are concluded.

(1) Unimodal functions $f_{01CEC} \sim f_{03CEC}$: Table 10 presents the results of the algorithms for optimizing the unimodal functions. From Table 10, it can be found that AMO can find the global best global optimum for f_{02CEC}. Although COA cannot reach the best global optimum for the three functions, it exhibits the best performance on the three unimodal functions. CMA-ES shows the worst performance on f_{02CEC}, GWO performs the worst for f_{01CE} and f_{03CEC}. When examined the last line of Table 10, COA ranks first for these three functions.
(2) Simple multimodal functions $f_{04CEC} \sim f_{16CEC}$: Table 11 summarizes the results obtained by the algorithms for simple multimodal functions. Clearly, CMA-ES exhibits the best performance on f_{06CEC}, f_{09CEC}, f_{11CEC}, f_{12CEC} and f_{15CEC} compared with other six algorithms. COA shows the best performance among the algorithms for f_{04CEC}. For f_{14CEC}, GWO outperforms than the other six algorithms. CoBiDE presents the best performance on f_{08CEC}, and AMO performs better than the other algorithms for f_{05CEC}, f_{07CEC} and f_{13CEC}. DE shows the worse optimization performance for the multimodal functions. However, when examined the last line of Table 11, COA ranks first for these multimodal functions as a whole because the three behaviors effectively balance the performance of exploration and exploitation.
(3) Hybrid functions $f_{17CEC} \sim f_{22CEC}$: The solution of these 6 functions is more difficult than that of previous 16 functions. The final results of hybrid functions are reported in Table 12. When examined Table 12, we can observe that the performance of CoBiDE outperforms the compared algorithms for f_{17CEC}, f_{18CEC} and f_{21CEC}. COA performs better on f_{22CEC} than the other compared algorithms. As can be seen from Table 12, CoBiDE and DE rank toward the top for these 6 hybrid functions, while COA performs overall better than GWO, ABC, AMO and CMA-ES.
(4) Composition functions $f_{23CEC} \sim f_{30CEC}$: These eight functions are the most difficult in this test. Table 13 summarizes the obtained results of the algorithms with 51 independent runs. As seen from Table 13, the results provided by the seven algorithms are far away from the global optima for these 8 test functions. However, according to overall rankings of Table 13, the COA still ranks first, while CoBiDE and AMO rank second and third, respectively. CMA-ES performs the worst performance among the algorithms.

We also perform the Wilcoxon Signed Ranks Test in this test, in which Iman–Davenport's procedure is used as a post hoc procedure [39]. The statistical results are summarized in Table 14. When examined the last line of the Table 14, we can observe that COA provides higher '+' values than '−' or '≈' values compared with other six algorithms. It means that, COA performs better

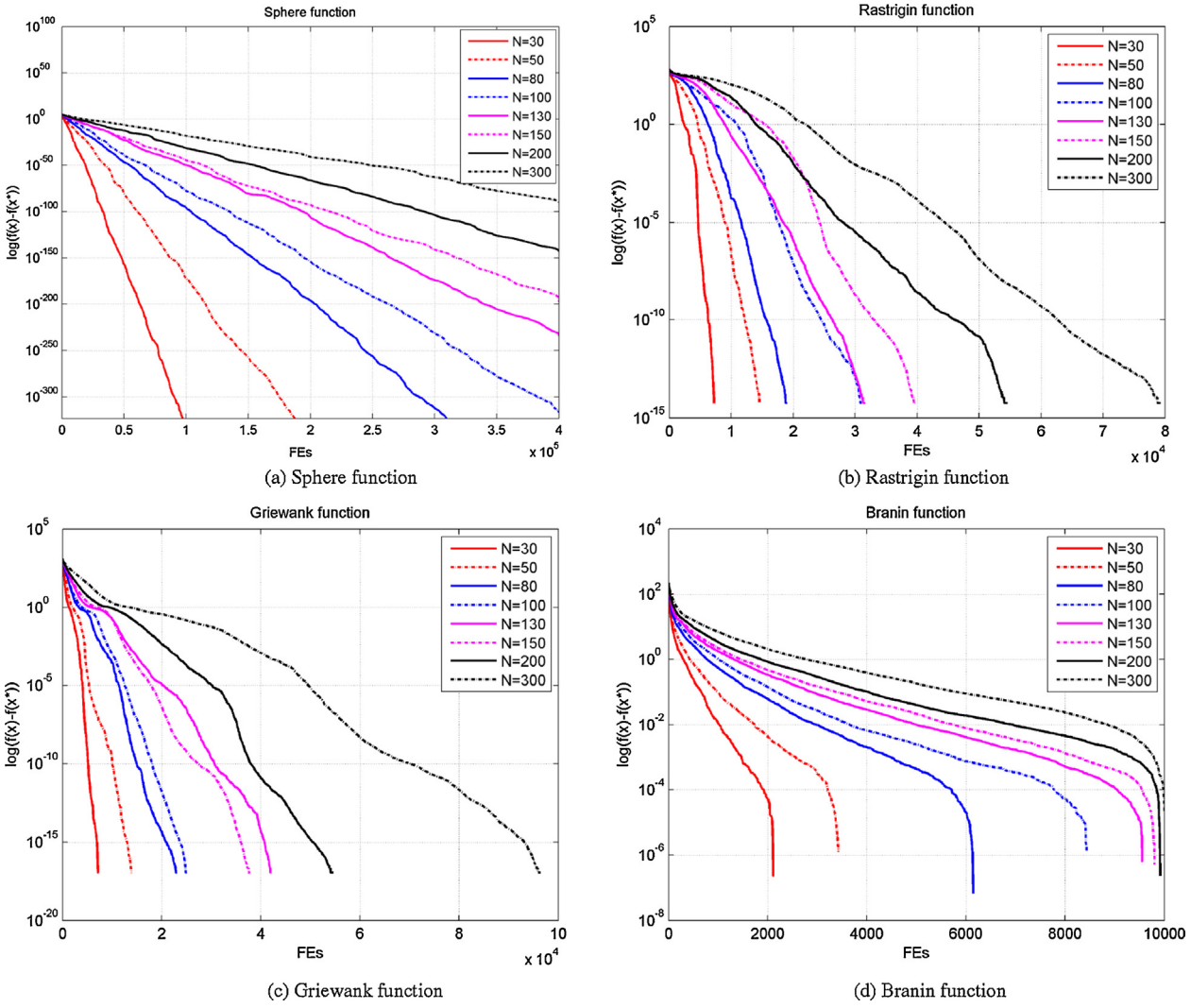

Fig. 6. Effect of N on convergence rate for four classical benchmark functions.

than the compared algorithms. As ABC's mechanism, COA and AMO also divide the population into different functional parts or have different operations for the population. However, COA outperforms the ABC and AMO according to the results of Table 14.

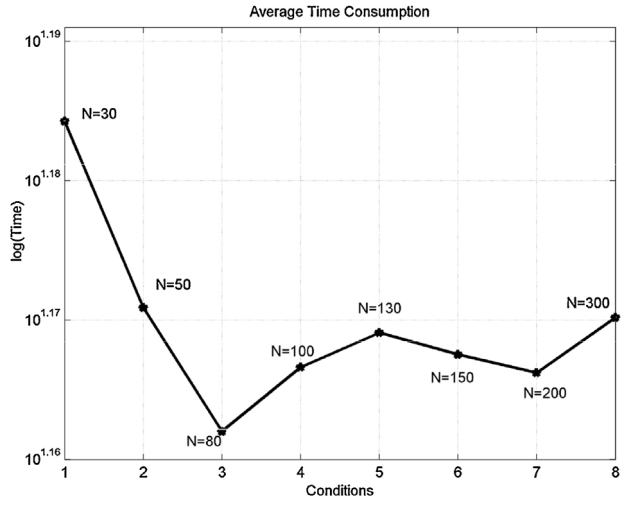

Fig. 7. Effect of N on average time consumption for 10 classic benchmark functions.

As a result, the behaviors of COA are superior to that of ABC and AMO.

In addition, Friedman test is used to detect the significance differences between COA and the compared algorithms on both 30 benchmarks of CEC 2014, in which Iman–Davenport's procedure is used as a post hoc procedure. Also, the Friedman's test was used by KEEL software [40]. The detailed test results of the seven algorithms on Mean and SD are shown in Table 15. From Table 15, it can be found that COA ranks first on Mean and SD among the seven algorithms. We can also observe that Friedman statistic considering reduction performance distributed according to chi-square with 6 degrees of freedom for mean value is 49.035714, and the Iman and Davenport statistic considering reduction performance distributed according to F-distribution with 6 and 174 degrees of freedom is 10.8582 at $\alpha = 0.05$, the critical value is 2.15 which can also be found in the statistical tables [42]. Again, we calculate critical difference CD for this test and $CD = 2.15\sqrt{7 \cdot 8/6 \cdot 20} = 1.199$. Fig. 8 shows the comparisons of all algorithms against each other using the critical difference from the Friedman and Iman and Davenport tests for Mean and SD rankings. From Fig. 8(a), it can be found that there are not significantly differences among the COA, CoBiDE and AMO for 30 CEC2014 benchmark functions. Compared with the rest four algorithms, COA is significantly different. As seen from Fig. 8(b), the group of the algorithms including COA, CoBiDE, GWO and AMO is not significantly different each other.

Table 9
Benchmark functions used in Test 2 (Optimum: Global optimal value).

No.	Types	Name	Optimum
f_{01CEC}	Unimodal functions	Rotated High Conditioned Elliptic Function	100
f_{02CEC}		Rotated Bent Cigar Function	200
f_{03CEC}		Rotated Discus Function	300
f_{04CEC}	Simple multimodal functions	Shifted and Rotated Rosenbrock's Function	400
f_{05CEC}		Shifted and Rotated Ackley's Function	500
f_{06CEC}		Shifted and Rotated Weierstrass Function	600
f_{07CEC}		Shifted and Rotated Griewank's Function	700
f_{08CEC}		Shifted Rastrigin's Function	800
f_{09CEC}		Six Hump Camel Back	900
f_{10CEC}		Shifted and Rotated Rastrigin's Function	1000
f_{11CEC}		Shifted and Rotated Schwefel's Function	1100
f_{12CEC}		Shifted and Rotated Katsuura Function	1200
f_{13CEC}		Shifted and Rotated HappyCat Function	1300
f_{14CEC}		Shifted and Rotated HGBat Function	1400
f_{15CEC}		Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500
f_{16CEC}		Shifted and Rotated Expanded Scaffer's F6 Function	1600
f_{17CEC}	Hybrid functions	Hybrid Function 1 ($n=3$)	1700
f_{18CEC}		Hybrid Function 2 ($n=3$)	1800
f_{19CEC}		Hybrid Function 3 ($n=4$)	1900
f_{20CEC}		Hybrid Function 4 ($n=4$)	2000
f_{21CEC}		Hybrid Function 5 ($n=5$)	2100
f_{22CEC}		Hybrid Function 6 ($n=5$)	2200
f_{23CEC}	Composition functions	Composition Function 1 ($n=5$)	2300
f_{24CEC}		Composition Function 2 ($n=3$)	2400
f_{25CEC}		Composition Function 3 ($n=3$)	2500
f_{26CEC}		Composition Function 4 ($n=5$)	2600
f_{27CEC}		Composition Function 5 ($n=5$)	2700
f_{28CEC}		Composition Function 6 ($n=5$)	2800
f_{29CEC}		Composition Function 7 ($n=3$)	2900
f_{30CEC}		Composition Function 8 ($n=3$)	3000
Search range: [−100,100]		Dimension: $D=30$	

From a graphical point of view, Figs. 9 and 10 present the convergence curves for 16 typical functions in CEC 2014. When examined Fig. 9, COA algorithm converges rapidly for most test functions compared with other 6 algorithms. For f_{02CEC}, AMO converges faster than COA. From Fig. 10, it is clear that COA performs better performance in convergence than the compared algorithms for $f_{14(CEC)}$, $f_{16(CEC)}$, $f_{22(CEC)}$, $f_{23(CEC)}$ and $f_{28(CEC)}$. For $f_{20(CEC)}$, DE converges little faster than COA. In a word, the superior experimental and analysis results of COA on these 30 CEC2014 test functions owe to its cooperative behaviors of the cognitive behavior modal.

4.3. Test III – constrained engineering design optimization

So far, the COA is invested on types of unconstrained problems. For the engineering design optimization problems, which are the most important and ubiquitous types of global constrained optimization problems, COA is also employed to evaluate its preference of solving such problems in this section.

Generally, the global constrained optimization problem is defined as follows:

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & g_k(x) \le 0,\ k = 1, 2, 3, \ldots, q \\ & h_j(x) = 0,\ j = 1, 2, 3, \ldots, m \\ & low_i \le x_i \le up_i,\ i = 1, 2, 3, \ldots, D \end{aligned} \tag{13}$$

where $f(x)$ is the objective function, $x=(x_1, x_2, \ldots, x_D)$ is an D dimensional vector, $g_k(x)$ is the inequality constraints and $h_j(x)$ is the equality constraints. low_i and up_i are the lower and upper bound of x_i, respectively.

Table 10
Mean and SD obtained by CMA-ES, GWO, DE, CoBiDE, ABC, AMO and COA through 51 independent runs on CEC 2014 Unimodal Functions in 30 dimension (Mean: mean solution, SD: Standard deviation, Rank: ranking based on mean values).

No.	Algorithms	CMA-ES	GWO	DE	CoBiDE	ABC	AMO	COA
f_{01CEC}	Mean	3.1574e+05	5.5795e+07	1.6422e+07	6.1660e+06	2.1518e+07	4.9073e+06	**2.2758e+05**
	SD	2.6581e+05	1.1446e+07	5.3320e+06	1.5119e+06	1.0816e+07	1.0231e+06	**1.8422e+05**
	Rank	2	7	5	4	6	3	**1**
f_{02CEC}	Mean	4.5854e+10	1.2156e+08	7.4607e+06	2.0233e+02	7.5252e+04	**2.0000e+02**	2.0030e+02
	SD	9.7194e+09	1.3617e+06	3.4854e+06	1.4248e+00	7.6169e+04	**7.3124e−04**	4.8879e−01
	Rank	7	6	5	3	4	**1**	2
f_{03CEC}	Mean	2.9597e+03	4.0459e+04	3.6508e+02	3.9199e+02	3.4880e+02	6.4058e+02	**3.3801e+02**
	SD	1.2905e+03	1.3581e+03	2.4769e+01	2.9956e+01	8.4478e+01	3.1021e+02	**7.1843e+01**
	Rank	6	7	3	4	2	5	**1**
Average rank		5	6.6667	4.3333	3.6667	4	3	**1.3333**
Overall rank		6	7	5	3		2	**1**

Table 11
Mean and SD obtained by CMA-ES, GWO, DE, CoBiDE, ABC, AMO and COA through 51 independent runs on CEC 2014 Simple Multimodal Functions in 30 dimension (Mean: mean solution, SD: Standard deviation, Rank: ranking based on mean values).

No.	Algorithms	CMA-ES	GWO	DE	CoBiDE	ABC	AMO	COA
f_{04CEC}	Mean	4.7081e+03	6.5478e+02	4.5935e+02	4.2096e+02	5.6098e+02	4.7398e+02	**4.1272e+02**
	SD	1.3341e+03	2.5761e+01	1.3186e+01	2.9201e+01	5.8162e+01	2.3954e+01	**2.4388e+01**
	Rank	7	6	3	2	5	4	**1**
f_{05CEC}	Mean	5.2799e+02	5.2968e+02	5.3061e+02	5.2787e+02	5.3005e+02	5.2856e+02	**5.2071e+02**
	SD	6.7573e+00	7.4676e+00	7.8175e+00	5.0706e+00	5.9864e+00	5.7099e+00	**8.4857e−02**
	Rank	3	5	7	2	6	4	**1**
f_{06CEC}	Mean	6.5751e+02	6.2378e+02	6.4830e+02	6.3065e+02	6.2916e+02	**6.1721e+02**	6.1834e+02
	SD	1.5673e+01	5.8168e+00	1.1624e+01	4.0069e+00	5.9521e+00	**4.1548e+00**	3.0201e+00
	Rank	7	3	6	5	4	**1**	2
f_{07CEC}	Mean	1.1254e+03	7.0259e+02	7.0153e+02	7.0128e+02	7.0834e+02	**7.0000e+02**	7.0002e+02
	SD	1.1583e+02	2.3591e−02	3.9787e−01	2.4255e−01	1.4033e+01	**3.5671e−10**	1.8563e−02
	Rank	7	5	4	3	6	**1**	2
f_{08CEC}	Mean	8.9812e+02	8.3621e+02	1.1408e+03	**8.1230e+02**	8.8994e+02	8.1437e+02	8.3495e+02
	SD	1.8366e+02	2.5678e−01	6.7999e+01	**3.5382e+00**	2.7558e+01	1.8637e+00	1.1030e+01
	Rank	6	4	7	**1**	5	2	3
f_{09CEC}	Mean	**9.1380e+02**	9.9739e+02	1.2429e+03	1.0463e+03	1.5060e+03	9.9605e+02	9.8789e+02
	SD	**4.2359e+00**	1.4589e+01	6.0140e+01	4.1435e+01	1.5898e+02	6.5317e+01	2.6295e+01
	Rank	**1**	4	6	5	7	3	2
f_{10CEC}	Mean	1.0882e+04	3.5937e+03	1.1439e+04	3.9604e+03	**1.6062e+03**	1.6118e+03	1.7211e+03
	SD	5.7839e+03	1.5765e+01	2.1679e+03	2.1057e+02	**3.0241e+02**	1.0981e+02	3.1248e+02
	Rank	6	4	7	5	**1**	2	3
f_{11CEC}	Mean	**2.1607e+03**	3.3378e+03	1.0302e+04	7.0533e+03	4.9782e+03	5.6058e+03	4.0219e+03
	SD	**1.1064e+03**	4.5227e+00	1.9959e+03	2.8484e+02	8.4914e+02	1.3007e+02	5.8756e+02
	Rank	**1**	2	7	6	4	5	3
f_{12CEC}	Mean	**1.2000e+03**	1.2001e+03	1.2033e+03	1.2012e+03	1.2007e+03	1.2010e+03	1.2009e+03
	SD	**8.1887e−04**	3.5887e−03	7.0939e−01	1.1580e−01	1.6392e−01	1.2036e−01	1.8841e−01
	Rank	**1**	2	7	6	3	5	4
f_{13CEC}	Mean	1.3070e+03	1.3005e+03	1.3010e+03	1.3005e+03	1.3022e+03	1.3004e+03	**1.3004e+03**
	SD	1.7035e+00	1.2776e−01	2.3826e−01	1.0061e−01	1.4143e+00	1.2240e−01	**9.7701e−02**
	Rank	7	4	5	3	6	2	**1**
f_{14CEC}	Mean	1.5664e+03	1.4004e+03	1.4005e+03	1.4003e+03	1.4262e+03	1.4003e+03	**1.4002e+03**
	SD	4.6612e+01	7.7044e−02	1.3717e−01	3.7606e−02	2.9081e+01	3.1037e−02	**3.3845e−02**
	Rank	7	4	5	2	6	3	**1**
f_{15CEC}	Mean	**1.5055e+03**	1.5316e+03	1.5314e+03	1.5141e+03	2.6923e+04	1.5116e+03	1.5096e+03
	SD	**1.4774e+00**	2.9462e+00	8.0041e+00	1.0174e+00	2.6503e+04	3.4395e+00	2.7030e+00
	Rank	**1**	6	5	4	7	3	2
f_{16CEC}	Mean	1.6228e+03	1.6133e+03	1.6216e+03	1.6128e+03	1.6152e+03	**1.6115e+03**	1.6118e+03
	SD	6.8508e+00	2.0415e+00	4.0132e+00	4.1497e−01	2.3262e+00	**2.0643e−01**	3.5652e−01
	Rank	7	4	6	3	5	**1**	2
Average rank		4.6923	4.0769	5.7692	3.6154	5.0000	2.7692	**2.0769**
Overall rank		5	4	7	3	6	2	**1**

In order to study the performance of solving the engineering design optimization problems, COA is applied to three well-known constrained engineering design optimization problems: tension/compression spring, welded beam and pressure vessel designs. The above algorithms including ABC, GSA, CoBiDE, DSA, GWO, DE and AMO are used as the compared algorithms. In addition, the same constraint handling mechanism used in [44] was used for the problems as Eq. (14) shown, and the results of 30 independent runs on COA and other algorithms are placed at Tables 16–21. The population size for these problems is set to 20 and the *MaxFEs* is set to 50,000 for each problem.

$$f_P(x) = f(x) + 10^4 \left[\sum_k \max\{g_k(x), 0\} + \sum_j |h_j(x)| \right] \tag{14}$$

4.3.1. Tension/compression spring design problem

Spring design optimization problem is a well-known engineering design problem. The main goal of this problem is to design a spring for a minimum weight by achieving optimum values of the variables as shown in Fig. 11. It contains four nonlinear inequality constrains and three continuous variables: the wire diameter w (x_1), the mean coil diameter d (x_2) and the length (or number of coils) L (x_3). This problem can be modeled as follows:

$$\min f(x_1, x_2, x_3) = (x_3 + 2) x_1^2 x_2 \tag{15}$$

Subject to

$$\begin{aligned} g_1(X) &= 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0 \\ g_2(X) &= \frac{x_2(4x_2 - x_1)}{12566 x_1^3 (x_2 - x_1)} + \frac{1}{5108 x_1^2} - 1 \le 0 \\ g_3(X) &= 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0 \\ g_4(X) &= \frac{2(x_1 + x_2)}{3} - 1 \le 0 \end{aligned} \tag{16}$$

Variable range : $0.05 \le x_1 \le 2,\ 0.25 \le x_2 \le 1.3,\ 2.0 \le x_3 \le 15.0$

Table 12
Mean and SD obtained by CMA-ES, GWO, DE, CoBiDE, ABC, AMO and COA through 51 independent runs on CEC 2014 Hybrid Functions in 30 dimension (Mean: mean solution, SD: Standard deviation, Rank: ranking based on mean values).

No.	Algorithms	CMA-ES	GWO	DE	CoBiDE	ABC	AMO	COA
f_{17CEC}	Mean	4.4197e+03	3.3241e+05	5.0759e+03	**3.8467e+03**	2.8529e+06	4.9128e+05	2.3516e+04
	SD	9.1896e+02	2.1537e+04	9.0313e+02	**1.9537e+02**	2.4267e+06	1.0814e+05	2.9349e+04
	Rank	2	5	3	**1**	7	6	4
f_{18CEC}	Mean	2.4990e+03	2.6885e+07	1.9604e+03	**1.8611e+03**	3.3186e+06	2.0526e+03	3.2729e+03
	SD	2.6558e+02	2.1376e+05	3.8459e+01	**8.2124e+00**	4.3661e+06	2.6891e+02	1.8139e+03
	Rank	4	6	2	**1**	7	3	5
f_{19CEC}	Mean	2.0470e+03	1.9254e+03	1.9127e+03	1.9150e+03	1.9203e+03	**1.9061e+03**	1.9155e+03
	SD	2.1917e+02	3.1373e−02	3.4775e+00	9.9131e−01	4.4620e+00	**5.3753e−01**	1.8121e+00
	Rank	7	6	2	3	5	1	4
f_{20CEC}	Mean	2.7635e+03	2.4042e+03	**2.1183e+03**	2.1119e+03	6.0443e+04	6.5271e+03	2.2222e+03
	SD	2.4535e+02	6.1374e+02	**2.5186e+01**	2.2758e+01	2.5379e+04	1.2013e+03	3.7268e+01
	Rank	5	4	**1**	2	6	7	3
f_{21CEC}	Mean	3.5261e+04	1.6544e+05	3.9721e+03	**3.0480e+03**	2.1525e+05	1.1845e+05	1.5306e+04
	SD	1.2234e+04	1.6345e+04	4.7218e+02	**1.4423e+02**	3.0304e+05	2.3601e+04	1.3940e+04
	Rank	4	6	2	**1**	7	5	3
f_{22CEC}	Mean	5.1729e+03	2.4772e+03	2.9342e+03	2.5957e+03	3.0855e+03	2.4829e+03	**2.4686e+03**
	SD	1.9392e+03	1.1576e+01	2.9707e+02	1.1816e+02	2.6873e+02	1.0674e+01	**1.1493e+01**
	Rank	7	2	5	4	6	3	**1**
Average rank		4.8333	4.8333	2.5000	**1.8333**	6.3333	4.1667	3.3333
Overall rank		5	5	2	**1**	7	4	3

For this problem, Table 16 lists the best solutions of this problem obtained by COA and other seven compared algorithms with 30 independent runs and the statistical results of them are presented in Table 17. When examined Tables 16 and 17, it can be found that the best result obtained by COA is 0.01266523278, which is equal to that of DE algorithm. However, the standard deviation of the results by COA is clearly smaller than that of compared six algorithms. In addition, the optimization results of inequality constraint g_1 and g_2 of COA reach to the 0 based on 50,000 NFEs.

4.3.2. Welded beam design problem

The welded beam design optimization problem (see Fig. 12) involves four design variables including the width $h\,(x_1)$ and length $l\,(x_2)$ of the welded area, the depth $t\,(x_3)$ and thickness $b\,(x_4)$ of the

Table 13
Mean and SD obtained by CMA-ES, GWO, DE, CoBiDE, ABC, AMO and COA through 51 independent runs on CEC 2014 Composition Functions in 30 dimension (Mean: mean solution, SD: Standard deviation, Rank: ranking based on mean values).

No.	Algorithms	CMA-ES	GWO	DE	CoBiDE	ABC	AMO	COA
f_{23CEC}	Mean	2.8145e+03	2.6218e+03	2.7758e+03	2.6140e+03	2.8452e+03	2.6152e+03	**2.5000e+03**
	SD	1.4345e+02	8.7657e−01	8.2158e+01	7.7709e−07	1.3752e+02	1.6217e−06	**1.0735e−05**
	Rank	6	4	5	2	7	3	**1**
f_{24CEC}	Mean	2.7641e+03	2.6000e+03	2.7857e+03	2.6264e+03	2.7367e+03	2.6243e+03	**2.6000e+03**
	SD	1.2166e+02	1.1453e−04	9.0800e+01	8.8470e+00	6.2711e+01	1.3547e−01	**1.1975e−06**
	Rank	6	2	7	4	5	3	**1**
f_{25CEC}	Mean	2.8244e+03	2.7138e+03	2.8061e+03	2.7003e+03	2.7890e+03	2.7090e+03	**2.7000e+03**
	SD	9.1702e+01	2.6571e−02	8.4767e+01	5.9221e−02	4.8424e+01	2.6317e+00	**4.1730e−13**
	Rank	7	4	6	2	5	3	**1**
f_{26CEC}	Mean	2.7459e+03	2.7385e+03	2.7663e+03	2.7586e+03	2.7581e+03	2.7482e+03	**2.7004e+03**
	SD	4.2824e+01	3.2789e+01	3.5608e+01	3.3092e+01	3.4547e+01	2.7358e+01	**8.2426e−02**
	Rank	3	2	7	6	5	4	**1**
f_{27CEC}	Mean	4.4958e+03	3.3376e+03	3.9477e+03	3.8271e+03	3.3827e+03	3.1073e+03	**2.9005e+03**
	SD	8.7577e+02	1.5734e+00	3.3227e+02	5.9870e+01	1.5227e+02	1.5634e+01	**2.3021e+00**
	Rank	7	3	6	5	4	2	**1**
f_{28CEC}	Mean	5.9755e+03	3.7277e+03	3.9560e+03	3.1693e+03	3.5094e+03	3.7375e+03	**3.0003e+03**
	SD	1.3575e+03	1.9371e+00	2.6180e+02	1.2684e+00	2.0077e+02	2.5761e+01	**1.2156e+00**
	Rank	7	4	6	2	3	5	**1**
f_{29CEC}	Mean	1.2699e+07	1.2315e+04	6.7102e+05	**3.2220e+03**	3.2464e+03	4.5466e+03	4.0555e+03
	SD	6.2770e+06	1.0657e+02	8.9206e+05	**6.9211e+01**	8.6647e+01	1.1007e+02	3.5224e+02
	Rank	7	5	6	**1**	2	4	3
f_{30CEC}	Mean	7.1763e+05	2.2756e+04	5.5504e+03	**3.8018e+03**	4.8538e+03	5.7585e+03	5.2474e+03
	SD	2.9604e+05	3.3461e+01	7.7932e+02	**1.4545e+02**	4.7455e+02	3.7651e+02	6.6901e+02
	Rank	7	6	4	**1**	2	5	3
Average rank		6.2500	3.7500	5.8750	2.8750	4.1250	3.6250	**1.5000**
Overall rank		7	4	6	2	5	3	**1**

Table 14
The results of Wilcoxon Signed Ranks Test based on the best solution for each benchmarks of CEC 2014 benchmarks with 51 independently runs (α=0.05).

F	CMAES vs. COA				GWO vs. COA				DE vs. COA				CoBiDE vs. COA				ABC vs. COA				AMO vs. COA			
	p-Value	R+	R−	Win	p-Value	R+	R−	Win	p-Value	R+	R−	Win	p-Value	R+	R−	Win	p-Value	R+	R−	Win	p-Value	R+	R−	Win
f_{01CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{02CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	1.2476e−09	1311	15	+	5.1453e−10	1326	0	+	**5.1363e−05**	**231**	**1095**	–
f_{03CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	6.0205e−04	1029	297	+	4.0330e−05	1101	225	+	**9.4768e−01**	**670**	**656**	≈	8.2719e−10	1318	8	+
f_{04CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	1.0467e−09	1314	12	+	8.6758e−03	943	383	+	5.1453e−10	1326	0	+	1.0467e−09	1314	12	+
f_{05CEC}	7.5527e−03	948	378	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	1.7684e−09	1305	21	+
f_{06CEC}	5.1274e−10	1326	0	–	**5.4232e−01**	**598**	**728**	≈	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	8.7743e−10	1317	9	+	**2.9657e−02**	**431**	**895**	–
f_{07CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	**7.6814e−07**	**136**	**1190**	–
f_{08CEC}	1.0467e−09	1314	12	+	1.0219e−02	937	389	+	5.1453e−10	1326	0	+	**5.1453e−10**	**0**	**1326**	–	7.3499e−10	1320	6	+	**5.4615e−10**	**1**	**1325**	–
f_{09CEC}	**3.9550e−08**	**77**	**1249**	–	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{10CEC}	1.2046e−08	1271	55	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	**2.0572e−04**	**267**	**1059**	–	**4.2901e−02**	**447**	**879**	–
f_{11CEC}	**7.9392e−05**	**242**	**1084**	–	**1.8583e−03**	**331**	**995**	–	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{12CEC}	**5.1453e−10**	**0**	**1326**	–	**8.9246e−05**	**245**	**1081**	–	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	**5.3761e−01**	**798**	**528**	≈	5.1453e−10	1326	0	+
f_{13CEC}	5.1453e−10	1326	0	+	4.6209e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	**5.1964e−01**	**768**	**558**	≈
f_{14CEC}	5.1453e−10	1326	0	+	**1.0436e−09**	**12**	**1314**	–	6.5284e−10	1322	4	**+**	2.4641e−05	1113	213	+	5.1453e−10	1326	0	+	**5.8583e−01**	**695**	**631**	**+**
f_{15CEC}	**9.3064e−10**	**10**	**1316**	–	5.1453e−10	1326	0	+	6.9272e−10	1321	5	+	1.3658e−05	1127	199	+	5.1453e−10	1326	0	+	1.0289e−01	1089	237	+
f_{16CEC}	5.1453e−10	1326	0	+	**5.1453e−10**	**0**	**1326**	–	5.1453e−10	1326	0	+	1.2657e−05	1039	287	+	1.8737e−09	1304	22	**+**	**2.4132e−01**	**538**	**788**	≈
f_{17CEC}	**5.1453e−10**	**0**	**1326**	–	5.1453e−10	1326	0	+	**5.1453e−10**	**0**	**1326**	–	**5.1453e−10**	**0**	**1326**	–	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{18CEC}	**1.0025e−04**	**248**	**1078**	–	5.1453e−10	1326	0	+	**1.2046e−08**	**55**	**1271**	–	**5.1453e−10**	**0**	**1326**	–	5.1453e−10	1326	0	+	**9.1483e−08**	**93**	**1233**	–
f_{19CEC}	1.6078e−07	1222	104	+	8.9151e−05	1081	245	+	**1.0354e−03**	**313**	**1013**	–	**8.9246e−05**	**245**	**1081**	–	8.9246e−05	1081	245	+	**5.7967e−10**	**2**	**1324**	–
f_{20CEC}	5.1453e−10	1326	0	+	6.5360e−09	1282	44	+	**5.1453e−10**	**0**	**1326**	–	**5.1453e−10**	**0**	**1326**	–	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{21CEC}	6.6970e−08	1239	87	+	5.1453e−10	1326	0	+	**5.1453e−10**	**0**	**1326**	–	**5.1453e−10**	**0**	**1326**	–	5.5737e−07	1197	129	+	5.1453e−10	1326	0	+
f_{22CEC}	1.2046e−08	1271	55	+	**2.1745e−05**	**210**	**1116**	–	5.4615e−10	1325	1	+	7.5527e−03	948	378	+	6.1519e−10	1323	3	+	4.8860e−04	1305	291	+
f_{23CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{24CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{25CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{26CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{27CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{28CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+
f_{29CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	**5.4232e−01**	**598**	**728**	≈	**2.3595e−09**	**26**	**1300**	–	5.1453e−10	1326	0	+
f_{30CEC}	5.1453e−10	1326	0	+	5.1453e−10	1326	0	+	3.6592e−02	886	440	+	**5.1453e−10**	**0**	**1326**	–	**6.5437e−09**	**87**	**1239**	–	5.6176e−04	1031	295	+
+/≈/−		24/0/6				24/1/5				25/0/5				22/1/7				25/2/3				21/2/7		

Table 15
Ranks of CMA-ES, GWO, DE, CoBiDE, ABC, AMO and COA for CEC 2014 benchmarks according to the Friedman test at $\alpha = 0.05$ significance.

Algorithms	CMA-ES	GWO	DE	CoBiDE	ABC	AMO	COA
Mean rank[a]	5.2833	4.3667	4.8167	3	4.9	3.35	**2.2833**
SD rank[b]	5.6333	3.3333	5	**2.6333**	5.3667	3.1667	2.8667

a: 1) Friedman statistic considering reduction performance distributed according to chi-square with 6 degrees of freedom: 49.035714.
2) *p*-Value computed by Friedman Test: 7.3582e−09.
3) Iman and Davenport statistic considering reduction performance (distributed according to *F*-distribution with 6 and 174 degrees of freedom: 10.8582, and the critical value: 2.15.
4) *p*-Value computed by Iman and Daveport Test: 4.8227e−11.
b: 1) Friedman statistic considering reduction performance distributed according to chi-square with 6 degrees of freedom: 63.1714.
2) *p*-Value computed by Friedman Test: 5.8988e−11.
3) Iman and Davenport statistic considering reduction performance (distributed according to *F*-distribution with 6 and 174 degrees of freedom: 15.6808, and the critical value: 2.15.
4) *p*-Value computed by Iman and Daveport Test: 2.3423e−14.

main beam. The main goal is to minimize the overall fabrication cost, under the bending stress σ_d (30,000 psi), appropriate constraints of shear stress τ_d (13,600 psi), maximum end deflection δ_d (0.25 in) and loading condition P (6000 lb). The problem can be written as:

$$\min f(x_1, x_2, x_3, x_4) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \quad (17)$$

Subject to:

$$\begin{aligned} &g_1(X) = x_1 - x_4 \le 0 \\ &g_2(X) = \delta - 0.25 \le 0 \\ &g_3(X) = \tau - 13,600 \le 0 \\ &g_4(X) = \sigma - 30,000 \le 0 \\ &g_5(X) = 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5.0 \le 0 \\ &g_6(X) = 0.125 - x_1 \le 0 \\ &g_7(X) = 6000 - F \le 0 \end{aligned} \quad (18)$$

where

$$\begin{aligned} &\sigma = 50400/x_3^2x_4 \quad Q = 6000\left(14 + x_2/2\right) \\ &D = \frac{1}{2}\sqrt{x_2^2 + (x_1 + x_3)^2}\ \beta = QD/J \\ &J = \sqrt{2}x_1x_2\left(x_2^2/6 + (x_1 + x_3)^2/2\right) \\ &\delta = 65856/3000x_4x_3^3 \quad \alpha = 6000/\left(\sqrt{2}x_1x_2\right) \\ &\tau = \sqrt{\alpha^2 + \alpha\beta x_2/D + \beta^2} \\ &F = 0.61423 \cdot 10^6 \frac{x_3x_4^3}{6}\left(1 - \frac{x_3\sqrt{30/48}}{28}\right) \end{aligned} \quad (19)$$

The variable ranges are $0.1 \le x_1, x_4 \le 2.0$, $0.1 \le x_2, x_3 \le 10$.

With 30 independent runs, the COA and the other compared algorithms are used to find the best solution of the design problem based on 50,000 NFEs per run. The comparison results are provided in Table 18 and the statistical results are listed in Table 19. The results from Tables 18 and 19 show that COA obtained the best cost value. Although, CoBiDE gets the 0 value for the inequality constraint g_3 and g_4, its cost value (1.72485442453) is larger than that of COA (1.72480865692). In addition, the best result obtained by DE is comparable to COA, but it performs worse on the SD value as seen from Table 19.

4.3.3. Pressure vessel problem

Pressure vessel design optimization problem is a mixed type of optimization and it has also been often used as a benchmark problem for evaluate different optimization algorithms. Fig. 13 shows a cylindrical pressure vessel capped at both ends by hemispherical heads. The object is to minimize the total cost, including the cost of forming, material and welding. There are four variables: the thickness *Ts* (x_1), thickness of the head *Th* (x_2), the inner radius *R* (x_3) and the length of the cylindrical section of the vessel *L* (x_4). The problem can be stated as follows:

$$\min f(x_1, x_2, x_3, x_4) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (20)$$

Subject to:

$$\begin{aligned} &g_1(X) = -x_1 + 0.0193x_3 \le 0 \\ &g_2(X) = -x_2 + 0.00954x_3 \le 0 \\ &g_3(X) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^2 + 1,296,000 \le 0 \\ &g_4(X) = x_4 - 240 \le 0 \\ &\textit{Variable ranges}: 1 \times 0.0625 \le x_1,\ x_2 \le 99 \times 0.0625,\ 10 \le x_3,\ x_4 \le 200 \end{aligned} \quad (21)$$

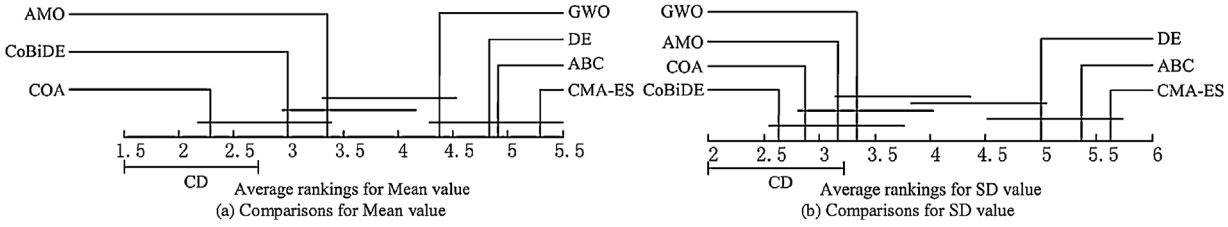

Fig. 8. Comparisons between 7 algorithms against each other based on CEC2014 functions. The groups of algorithms that are not significantly different are connected. The critical difference *CD* is visible below the axis.

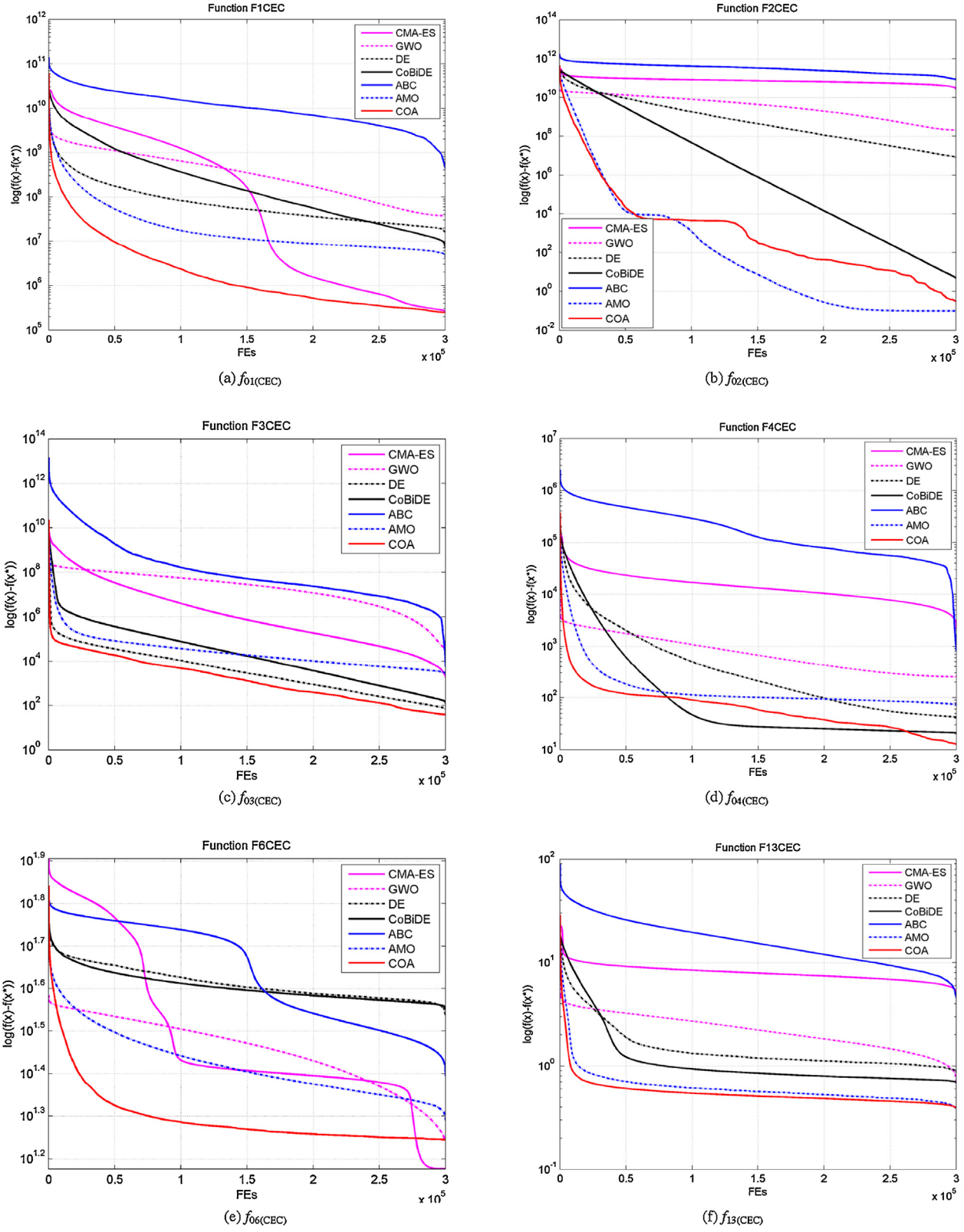

Fig. 9. Evolution of the mean function error values derived from COA and other competitors vs. the number of FEs on some of CEC 2014 benchmarks.

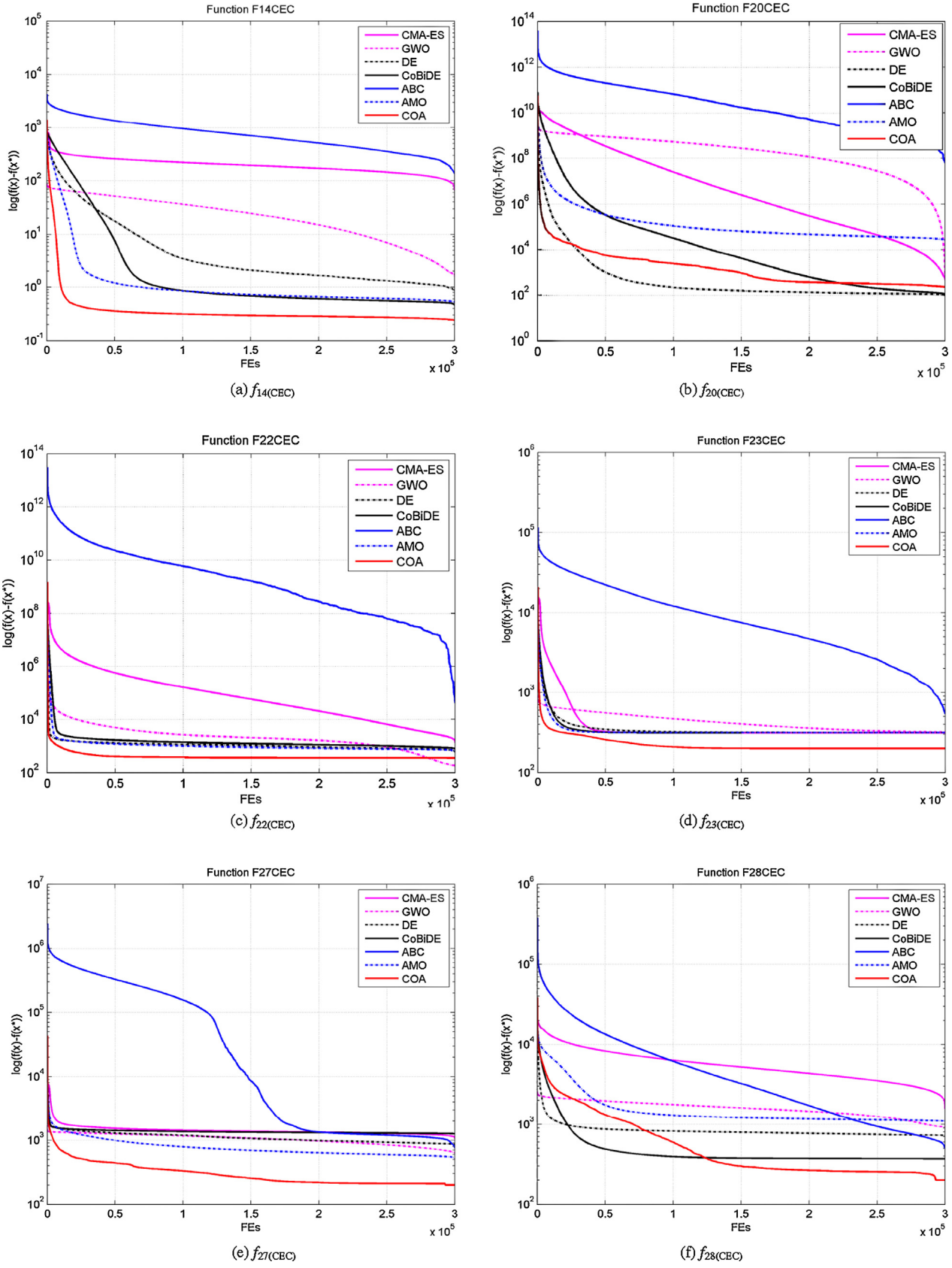

Fig. 10. Evolution of the mean function error values derived from COA and other competitors vs. the number of FEs on some of CEC 2014 benchmarks.

Table 16
Comparing of the best solution for Tension/compression spring design problem obtained by different algorithms.

	ABC	GSA	DE	CoBiDE	DSA	GWO	AMO	COA
$w(x_1)$	0.0500000000	0.05094493497	0.0516890612	0.05168852102	0.05144824965	0.05054535412	0.05013481404	0.0516890609
$d(x_2)$	0.3170446475	0.33152878490	0.35671774364	0.35670474754	0.35094608800	0.32982011146	0.31984505614	0.3567177361
$L(x_3)$	14.082469461	14.1678306531	11.28896552632	11.2897274875	11.6371153947	13.0609576953	13.8611740812	11.2889659655
g_1	−2.910268e−04	−0.0676510442	**0**	−9.5257135e−13	−1.1614252e−04	−1.0994527e−04	−6.3567549e−05	**0**
g_2	−9.441899e−04	−0.0177379013	**0**	−1.9926282e−12	−1.3602513e−05	−2.45781588e−06	−0.00152360360	**0**
g_3	−3.961032572	−3.5949090004	−4.0537856379	−4.0537599697	−4.0415745710	−3.9965897892	−3.96571568002	−4.0537856231
g_4	−0.755303568	−0.7450175201	−0.7277287967	−0.7277378209	−0.7317371082	−0.7464230229	−0.7533467532	−0.7277288019
cost	0.0127471521	0.01391153388	**0.01266523278**	0.01266523279	0.01266788549	0.01269089380	0.01275128024	**0.01266523278**

Table 17
Statistical results of different approaches for Tension/compression spring design problem.

	ABC	GSA	DE	CoBiDE	DSA	GWO	AMO	COA
Best	0.0127471521	0.0139115338	0.01266523278	0.0126652327	0.0126678854	0.0126908938	0.0127512802	**0.01266523278831**
Worst	0.0149549869	0.0306399954	0.01266523304	0.0126652688	0.0134555710	0.0127287695	0.0158526305	**0.01266523278831**
Mean	0.0131438327	0.0187552241	0.01266523279	0.0126652356	0.0128234434	0.0127143134	0.0132952021	**0.01266523278831**
SD	5.0741e−04	4.2064e−03	4.8229e−11	7.4113e−09	1.8406e−04	9.8039e−06	8.5909e−04	**2.5317e−18**

Table 18
Comparing of the best solution for Welded Beam design problem found by different algorithms.

	ABC	GSA	DE	CoBiDE	DSA	GWO	AMO	COA
$h(x_1)$	0.2126780338	0.1773206338	0.20579784052	0.2057302283	0.2054068156	0.2056185845	0.2052589009	0.205797840454702
$l(x_2)$	3.3973460533	4.8056207294	3.46902638916	3.4704809741	3.4825427990	3.4731518533	3.4816768782	3.469026390716161
$t(x_3)$	8.8841861816	8.8768123833	9.03661098457	9.0366109845	9.0504378392	9.0365960728	9.0366961078	9.036610984575198
$b(x_4)$	0.2142080944	0.2135250227	0.20573022832	0.2057302283	0.2056651747	0.2057412905	0.2061211320	0.205730228329383
g_1	−0.001530060	−0.036204388	6.761219e−05	6.9388939e−16	−2.583591e−04	−1.227061e−04	−8.6223111e−04	−6.76121253182e−05
g_2	−0.235385456	−0.235302137	−0.2355403019	−0.2355403019	−0.2356019205	−0.2355410077	−0.2355681322	−0.235540301901929
g_3	−43.31167677	−1.12961e+03	2.1173036e−09	**0**	−32.454896852	−0.8144985074	−3.2895240135	−2.16277840081e−09
g_4	−1.90179e+02	−45.11353999	8.1854523e−10	**0**	−82.135025999	−1.5140303087	−57.458382119	−7.71251507103e−10
g_5	−3.402424735	−3.281846944	−3.433109399	−3.4329822128	−3.4300199558	−3.4326666790	−3.4290154214	−3.433109399490641
g_6	−0.087678033	−0.052320633	−0.0807978405	−0.0807302283	−0.0804068156	−0.0806185845	−0.0802589009	−0.080797840454702
g_7	−6.96975e+02	−6.29449e+02	2.2218955e−09	−3.6379788e−12	−0.3353371982	−0.9614139173	−34.303823086	−2.27555574383e−09
cost	1.7625982372	1.8817837565	**1.72480865692**	1.72485442453	1.72788302612	1.7251234443	1.72861992627	**1.724808656925220**

Table 19
Statistical results of different approaches for Welded Beam design problem.

	ABC	GSA	DE	CoBiDE	DSA	GWO	AMO	COA
Best	1.76259823726	1.88178375657	**1.72480865692**	1.72485442453	1.72788302612	1.72512344431	1.72861992627	**1.72480865692**
Worst	2.23355070756	3.04230379237	1.72480865693	1.72485442453	2.25166950489	1.72650168786	2.00010791695	**1.72480865692**
Mean	1.93319893958	2.41271631755	1.72480865692	1.72485442453	1.93087452570	1.72531374891	1.78186897046	**1.72480865692**
SD	0.11890956612	0.27575315557	2.3342869e−12	4.5168102e−16	0.12810660660	2.3828242e−04	0.06334086502	**2.03181639e−16**

Table 20
Comparing of the best solution for Pressure vessel design problem found by different algorithms.

	ABC	GSA	DE	CoBiDE	DSA	GWO	AMO	COA
$Ts(x_1)$	0.83370106	1.1940896434	0.7781686414	0.77816864	0.78269568	0.78071454	0.78507757	0.77816864
$Th(x_2)$	0.41792373	0.5455388930	0.3846491626	0.38464916	0.38531758	0.38590940	0.38917280	0.38464916
$R(x_3)$	43.0948918	57.163907325	40.319618725	40.3196187	40.3859923	40.4513491	40.6371423	40.31961872
$L(x_4)$	164.781043	54.585987336	199.99999997	199.999998	199.881092	198.180293	195.973315	199.9999998
g_1	−0.00196965	−0.090826232	−5.3401727e−12	3.63664e−12	−0.0032460	−3.5034e−06	−7.8072e−04	3.622213e−12
g_2	−0.00852226	−0.002481773	−0.0016127847	−0.00161278	−0.0016506	−0.0016215	−0.00311995	−0.001612784
g_3	−6.5801e+02	−4.68158e+04	−1.7901416e−05	**0**	−4.1146e+03	−30.773829	−1.7989e+03	**0**
g_4	−75.21895678	−1.85414e+02	−40.000000021	−40.0000000	−40.1189082	−41.819706	−44.0266847	−40.000000000
cost	6021.770461	7352.3265407	**5885.33277386**	**5885.332773**	5928.486479	5889.85758	5913.450511	**5885.33277360**

Table 21
Statistical results of different approaches for Pressure vessel design problem.

	ABC	GSA	DE	CoBiDE	DSA	GWO	AMO	COA
Best	6021.770461	7352.3265407	5885.33277386	5885.3327736	5928.4864790	5889.8575771	5913.4505109	**5885.332773601229**
Worst	6921.359103	15,499.805944	5885.33281634	5903.8664666	7001.6489654	5908.4622851	6628.0066188	**5885.332773601229**
Mean	6469.974186	10,529.618172	5885.33278113	5885.9748903	6247.0215431	5893.7327023	6234.4498026	**5885.332773601229**
SD	219.9967162	1971.1024121	8.8785763e−06	3.3806558310	222.44196837	4.4144062643	202.78699568	**0**

Table 22
The results of Wilcoxon Signed Ranks Test based on the best solution for above three engineering problems with 30 independently runs and the average rankings based on the Friedman Test ($\alpha = 0.05$).

No.	Spring Design Problem				Welded Beam Design Problem				Pressure Vessel Problem				+/≈/−	Average rankings
	p-Value	R+	R−	Win	*p*-Value	R+	R−	Win	*p*-Value	R+	R−	Win		
ABC vs. COA	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	3/0/0	5.625
GSA vs. COA	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	3/0/0	7.125
DE vs. COA	4.2857e−06	456	9	+	2.5579e−06	435	0	+	4.5281e−07	269	196	≈	2/1/0	3.125
CoBiDE vs. COA	1.7344e−06	465	0	+	4.3205e−08	465	0	+	4.3778e−04	136	0	+	3/0/0	3.5
DSA vs. COA	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	3/0/0	5.125
GWO vs. COA	1.7344e−06	465	0	+	1.5362e−06	465	0	+	2.3493e−03	438	27	+	3/0/0	3.75
AMO vs. COA	1.7344e−06	465	0	+	1.7344e−06	465	0	+	1.7344e−06	465	0	+	3/0/0	4.625
COA	–	–	–	–	–	–	–	–	–	–	–	–	–	**3.125**

1) Friedman statistic considering reduction performance distributed according to chi−square with 7 degrees of freedom: 9.270833.
2) Iman and Davenport statistic considering reduction performance distributed according to *F*-distribution with 7 and 21 degrees of freedom: 1.484983, and the critical value: 2.4876.
3) *p*-Value computed by Iman and Daveport Test: 0.226243.

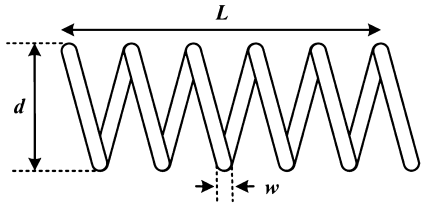

Fig. 11. Tension/compression spring design problem.

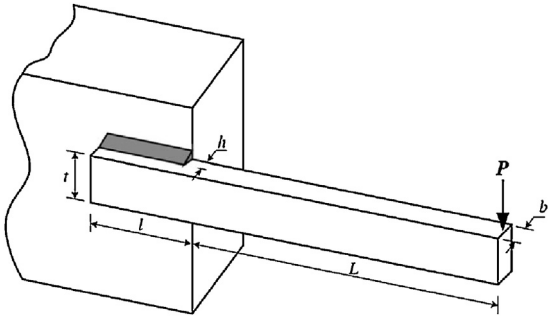

Fig. 12. Welded beam design problem.

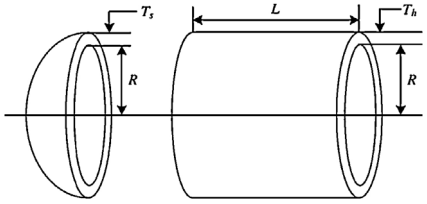

Fig. 13. Pressure vessel design problem.

For this problem, Table 20 summarizes the best results obtained by COA and other seven compared algorithms based on 30 independent runs and the statistical results of them are presented in Table 21. From Table 20, COA, CoBiDE and DE have the similar optimization performance according to the best results. However, the statistical results from Table 21 show that CoBiDE and DE have the worse standard deviation, while the SD of COA is 0.

For detailed comparison, the Wilcoxon Signed Ranks Test and Friedman test [39], in which Iman–Davenport's procedure is used as a post hoc procedure, are used to illustrate the superiority of COA on the three engineering optimization problems. The statistical results are summarized in Table 22. According to CD, which is equal to 4.3086 in this test, there are not significantly differences among the eight algorithms. However, when examined the Table 22, we can observe that COA provides higher '+' values than '−' or '≈' values compared with other algorithms. Meanwhile, COA and DE tied for first on the average rankings according to the Friedman test. As a result, COA still performs better optimization performance for these three engineering optimization problems.

5. Conclusion

During the past twenty years, nature-based algorithms which are inspired by the behavior or phenomena of nature have become a hotspot in the optimization computation. This paper proposes a new swarm intelligence algorithm cognitive behavior optimization algorithm based on the novel behavior model, which meticulously reflects the behaviors of foraging food. The model in COA contains three important behaviors namely rough search, information exchange and share, and intelligent adjustment. Meanwhile, two different groups: cognitive population *Cpop* and memory population *Mpop* are emphasized to search the best solutions through their cooperation. Rough search uses the Gaussian and Levy flight random walk methods to search the food region roughly for cognitive population. Between the cognitive population and memory population, information exchange and share utilizes the improved crossover and mutation strategy with the proposed select probability *Pc* to exchange the information among the individuals of the whole population. Intelligent adjustment uses the ranking of the individuals among the population to adjust the worse individuals to enhance the exploitation and improve the population diversity. To verify the performance of our algorithm, some benchmark functions including unconstrained benchmarks and constrained engineering design problems are used. Surprisingly, derived results by COA show that our new algorithm clearly outperforms some well-known nature-based algorithms. Meanwhile, the results also confirm the validity of the novel behavior model for constrained and unconstrained optimization problems.

As for the future work, the more cognitive behavior model which has a higher efficiency and lower time consumption for optimization computation is necessary to further study. Besides, the binary and multi-objective versions of COA will be developed.

Appendix A.

DSA: http://www.pinarcivicioglu.com/ds.html
CoBiDE: http://ist.csu.edu.cn/YongWang.htm
CMA-ES: http://dces.essex.ac.uk/staff/qzhang/
ABC: http://mf.erciyes.edu.tr/abc/software.htm
DE: http://academic.csuohio.edu/simond/bbo/

GWO: http://www.mathworks.cn/matlabcentral/fileexchange/44974-grey-wolf-optimizer–gwo-

GSA: http://www.mathworks.com/matlabcentral/fileexchange/27756-gravitational-search-algorithm-gsa

AMO: http://yinmh.nenu.edu.cn/

COA: It can be obtained from Cor. Author (Li M-D): modern_lee@163.com

References

[1] H. Ishibuchi, T. Yoshida, T. Murata, Balance between genetic search and local search in memetic algorithms for multi objective permutation flowshop scheduling, IEEE Trans. Evol. Comput. 7 (2003) 204–223.

[2] M.G. de Carvalho, A.H.F. Laender, M.A. Goncalves, A.S. da Solva, A genetic programming approach to record deduplication, IEEE Trans. Knowl. Data Eng. 24 (2012) 399–412.

[3] C. Kanzow, N. Yamashita, T. Fukushima, Levenberg–Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints, J. Comput. Appl. Math. 172 (2004) 375–397.

[4] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[5] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (2009) 945–958.

[6] F. Neri, V. Tirronen, Recent advances in differential evolution: a survey and experimental analysis, Artif. Intell. Rev. 33 (2010) 61–106.

[7] M. Helbig, A.P. Engelbrecht, Performance measures for dynamic multi-objective optimization algorithms, Inf. Sci. 250 (2013) 61–81.

[8] Ç. Balkaya, An implementation of differential evolution algorithm for inversion of geoelectrical data, J. Appl. Geophys. 98 (2013) 160–175.

[9] C. Teixeira, J. Covas, T. Stützle, A. Gaspar-Cunha, Hybrid algorithms for the twin-screw extrusion configuration problem, Appl. Soft Comput. 23 (2014) 298–307.

[10] N. Karaboga, A new design method based on artificial bee colony algorithm for digital IIR filters, J. Franklin I 346 (2009) 328–348.

[11] S. Ono, H. Maeda, K. Sakimoto, S. Nakayama, User-system cooperative evolutionary computation for both quantitative and qualitative objective optimization in image processing filter design, Appl. Soft Comput. 15 (2014) 203–218.

[12] S. Tamilselvi, S. Baskar, Modified parameter optimization of distribution transformer design using covariance matrix adaptation evolution strategy, Int. J. Electr. Power 61 (2014) 208–218.

[13] A. Baykasoğlu, F.B. Ozsoydan, Adaptive firefly algorithm with chaos for mechanical design optimization problems, Appl. Soft Comput. 36 (2015) 152–164.

[14] J.A. Koupaei, M. Abdechiri, Sensor deployment for fault diagnosis using a new discrete optimization algorithm, Appl. Soft Comput. 13 (2013) 2896–2905.

[15] J.T. Tsaia, J.C. Fanga, J.H. Chou, Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm, Comput. Oper. Res. 40 (2013) 3045–3055.

[16] T.Y. Chen, J.H. Huang, Application of data mining in a global optimization algorithm, Adv. Eng. Softw. 66 (2013) 24–33.

[17] F. Sun, W.M. Zhong, H. Cheng, F. Qian, Novel control vector parameterization method with differential evolution algorithm and its application in dynamic optimization of chemical processes, Chin. J. Chem. Eng. 21 (2013) 64–71.

[18] X.S. Yang, Nature-inspired Nature-based Algorithms, Luniver Press, 2008.

[19] J. Holland, Adaptation in Natural and Artificial Systems, MIT Press, Cambridge, MA, 1992.

[20] K.V. Price, R.M. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Berlin, Germany, Springer-Verlag, 2005.

[21] J. Brest, S. Greiner, B. Bošković, M. Mernik, V. Žumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (6) (2006) 646–657.

[22] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2009) 398–417.

[23] Y. Wang, H.X. Li, T.W. Huang, L. Li, Differential evolution based on covariance matrix learning and bimodal distribution parameter setting, Appl. Soft Comput. 18 (2014) 232–247.

[24] P. Civicioglu, Transforming geocentric Cartesian coordinates to geodetic coordinates by using differential search algorithm, Comput. Geosci.-UK 46 (2012) 229–247.

[25] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995, pp. 39–43.

[26] M.Y. Cheng, D. Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm, Comput. Struct. 139 (2014) 98–112.

[27] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (2007) 459–471.

[28] X.S. Yang, S. Deb, Cuckoo search via levy flights, in: Proceedings of World Congress on Nature and Biologically Inspired Computing, vol. 4, Nabic-2009, Coimbatore, India, 2009, pp. 210–214.

[29] X. Li, J. Zhang, M. Yin, Animal migration optimization: an optimization algorithm inspired by animal migration behavior, Neural Comput. Appl. 24 (7–8) (2014) 1867–1877.

[30] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61.

[31] R.A. Formato, Central force optimization: a new nature-based with applications in applied electromagnetics, Prog. Electromagn. Res. 77 (2007) 425–491.

[32] O.K. Erol, I. Eksin, A new optimization method: big bang–big crunch, Adv. Eng. Softw. 37 (2006) 106–111.

[33] A. Kaveh, S. Talatahari, A novel heuristic optimization method: charged system search, Acta Mech. 213 (2010) 267–289.

[34] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: a gravitational search algorithm, Inf. Sci. 179 (2009) 2232–2248.

[35] C. Igel, N. Hansen, S. Roth, Covariance matrix adaptation for multi-objective optimization, Evol. Comput. 15 (2007) 1–28.

[36] M. Črepinšek, S.H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: a survey, ACM Comput. Surv. 45 (2013) 1–33.

[37] M. Črepinšek, S.-H. Liu, M. Mernik, Replication and comparison of computational experiments in applied evolutionary computing: common pitfalls and guidelines to avoid them, Appl. Soft Comput. 19 (2014) 161–170.

[38] M. Mernik, S.H. Liu, M.D. Karaboga, M. Črepinšek, On clarifying misconceptions when comparing variants of the Artificial Bee Colony Algorithm by offering a new implementation, Inf. Sci. 291 (2015) 115–127.

[39] J. Derrac, S. Garcia, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (2011) 3–18.

[40] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms to data mining problems, Soft Comput. 13 (2009) 307–318 http://www.keel.es/.

[41] N. Veček, M. Mernik, M. Črepinšek, A chess rating system for evolutionary algorithms: a new method for the comparison and ranking of evolutionary algorithms, Inf. Sci. 277 (2014) 656–679.

[42] D.V. Lindley, New Cambridge Statistical Tables, Cambridge University Press, 1995.

[43] J.J. Liang, B.Y. Qu, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-parameter Numerical Optimization, Technical Report, 2013, pp. 1–32.

[44] A. Gandomi, X.S. Yang, A. Alavi, S. Talatahari, Bat algorithm for constrained optimization tasks, Neural Comput. Appl. 22 (6) (2013) 1239–1255.