Information Sciences 435 (2018) 240–262

Contents lists available at ScienceDirect

Information Sciences

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

Differential evolution with adaptive trial vector generation strategy and cluster-replacement-based feasibility rule for constrained optimization

Bin Xu[a,*], Xu Chen[b], Lili Tao[c]

[a] School of Mechanical Engineering, Shanghai University of Engineering Science, 201620, China
[b] School of Electrical and Information Engineering, Jiangsu University, Jiangsu 212013, China
[c] College of Engineering, Shanghai Second Polytechnic University, 201209, China

ARTICLE INFO

Article history:
Received 6 March 2017
Revised 3 January 2018
Accepted 7 January 2018
Available online 8 January 2018

Keywords:
Constrained optimization
Differential evolution
Adaptive strategy
Cluster
Feasibility rule

ABSTRACT

Constrained optimization problems (COPs) are common in many fields. To solve such problems effectively, in this paper, we propose a new constrained optimization evolutionary algorithm (COEA) named CACDE that combines an adaptive trial vector generation strategy-based differential evolution (DE) algorithm with a cluster-replacement-based feasibility rule. In CACDE, some potential mutation strategies, scale factors and crossover rates are stored in candidate pools, and each element in the pools is assigned a selection probability. During the trial vector generation stage, the mutation strategy, scale factor and crossover rate for each target vector are competitively determined based on these selection probabilities. Meanwhile, the selection probabilities are dynamically updated based on statistical information learned from previous searches in generating improved solutions. Moreover, to alleviate the greediness of the feasibility rule, the main population is divided into several clusters, and one vector in each cluster is conditionally replaced with an archived infeasible vector with a low objective value. The superior performance of CACDE is validated via comparisons with some state-of-the-art COEAs over 2 sets of artificial problems and 5 widely used mechanical design problems. The results show that CACDE is an effective approach for solving COPs, basically due to the use of adaptive DE and cluster-replacement-based feasibility rule.

1. Introduction

In real-world designs, many problems can be formulated as global numerical optimization problems with constraints. This type of problem is commonly known as a constrained optimization problem (COP), and it can be formulated as follows [28]:

$$\begin{array}{ll} \text{minimize} & f(x), x = [x_1, \dots, x_i, \dots, x_n]^T \in S \\ \text{subject to} & g_j(x) \le 0, \ (j = 1, 2, \dots, p) \\ & h_j(x) = 0, \ (j = p+1, \dots, q) \\ & a_i \le x_i \le b_i, \ (i = 1, 2, \dots, n) \end{array} \tag{1}$$

* Corresponding author.
E-mail address: xubin@mail.ecust.edu.cn (B. Xu).

https://doi.org/10.1016/j.ins.2018.01.014

where x is the n-dimensional decision variable in the search region S defined by the lower boundary $a = [a_1, \dots, a_n]^T$ and upper boundary $b = [b_1, \dots, b_n]^T$. The functions $f(x)$, $g(x)$ and $h(x)$ are, respectively, the objective function, inequality constraint and equality constraint. When a solution simultaneously satisfies all the p inequality and $q - p$ equality constraints, it is called a feasible solution; otherwise, it is an infeasible solution. Those inequality constraints that satisfy $g_j(x) = 0$ at a feasible solution are called active constraints. All equality constraints are therefore considered as active constraints at any feasible solutions.

Evolutionary algorithms (EA) are population-based metaheuristic algorithms that use mechanisms inspired by biological evolution such as reproduction, mutation, recombination, and selection. The primary advantage of EAs over traditional mathematical methods is that they require only that the objective function be calculable; other properties, such as differentiability and continuity, are unnecessary. The current popular EAs include particle swarm optimization (PSO), differential evolution (DE), artificial bee colony (ABC), artificial immune system (AIS), and teaching-learning-based optimization (TLBO). Among all these EAs, DE is a simple yet powerful algorithm for global optimization over continuous spaces [29,39]. Until now, DE has been widely and successfully applied in various domains [5,15,41,49]. However, it must be emphasized that DE was originally proposed for unconstrained problems [33], while the main goal of a constrained optimization algorithm is to find the feasible global optimum. Thus, to solve COPs effectively via DE, the following two issues should be considered [35,42]: (1) developing an effective constraint-handling technique (CHT); and (2) designing a powerful search engine.

In many constrained optimization evolutionary algorithms (COEAs), the CHTs serve as the main criterion for comparing multiple solutions during the selection process [28], and many different CHTs have been proposed. According to the ways in which the constraints are addressed, the existing CHTs can be grouped into three categories: (1) penalty function-based methods; (2) superiority of feasible solutions-based methods; (3) multiobjective optimization-based methods. Penalty function-based methods are simple to implement; however, they have been criticized because they require a fine-tuned penalty factor. When the penalty factor is too large, feasible solutions can be found quickly, but will have low quality. In contrast, when the penalty factor is too small, the quality of solutions may be high with respect to the objective function, but they may also be infeasible. To avoid this limitation, dynamic factor settings or adaptive factor settings [1] are applied. Superiority of feasible solutions-based methods prefer feasible solutions over infeasible ones, and they are both simple and parameter-free. However, they suffer from premature convergence because of the excessive greediness of the rule. To alleviate this greediness to some extent, some modifications have been proposed, such as the stochastic ranking method [38], the alpha constrained method [35,41], and the individual replacement technique [43]. Multiobjective optimization-based methods first combine the objective function with overall constraint violations to form a new bi-objective optimization problem; then, they optimize the bi-objective optimization problem [44]. However, some deficiencies still exist, and solving multiobjective optimization problems is still a challenging and often time-consuming task.

In its early stages, DE with slight modifications was directly combined with multifarious CHTs, such as penalty function methods [1], superiority of feasible solution-based methods [41], multiobjective-based methods [18], and multiple CHTs [22]. As stated above, designing a powerful search engine for COPs is also challenging. To improve the search capability of basic DE, some improved DE variants have been studied. Increasing the diversity is one significant direction. Tasgetiren and Suganthan [40] divided the main population into smaller sub-populations and then allowed them to search in parallel. Meanwhile, these sub-populations exchanged information periodically. Gao et al. [10] divided the main population into two sub-populations based on the feasibility of vectors and let the feasible sub-population focus on minimizing the objective, while the infeasible sub-population on the overall constraint violation. Mezura-Montes et al. [30] proposed a new COEA named Diversity-DE by incorporating a diversity mechanism into DE. Iacca et al. [14] proposed a multi-strategy approach that coevolved aging particles for global optimization. This method combined two components with complementary algorithm logic. Poikolainen et al. [34] proposed a cluster-based population initialization technique for DE that used the K-means clustering algorithm to group the solutions into two sets based on Euclidean distance.

In addition to introducing diversity, DE approaches that utilize multiple strategies are also popular. Wang and Cai [42] proposed a new DE framework called $(\mu + \lambda)$DE, in which three trial vectors are created for each target vector using three different mutation operators. Later, to overcome drawbacks in $(\mu + \lambda)$DE, Jia et al. [16] proposed an improved version called ICDE, which uses a new mutation strategy and an archive-based adaptive tradeoff model. In addition to using multiple mutation strategies, Elsayed et al. [8] included multiple mutation and crossover strategies into DE for COPs.

More recently, designing adaptive DE for COPs has attracted considerable research attention. Brest et al. [4] proposed the jDE-2 algorithm, in which the parameters F and CR are self-adaptively controlled based on previous search information. Ao and Chi [2] used a new mutation operator that did not require the F parameter. Moreover, they adaptively controlled the CR parameter to enhance their algorithm's adaptive capacity. Elsayed et al. [7] proposed an improved algorithm framework that uses a mixture of different mutation operators with a self-adaptive strategy. Zhang and Rangaiah [49] proposed a COEA named SaDETL, in which the mutation strategies and parameters are self-adjusted based on previous learning experiences. In addition, SaDETL applies a taboo list to avoid revisiting already searched areas. Qian et al. [35] proposed SADE-αCD, which solves complex constrained multiobjective problems by using multiple mutation-strategy-based DE operators with self-adaptively controlled F and CR. Elsayed et al. [8] included multiple mutation and crossover operators in DE by determining a mutation and a crossover operator for each vector in the population using an adaptive learning process. Kong et al. [17] proposed an adaptive grouping DE (AGDE) for COPs, in which the population is dynamically divided into three sub-populations, each with its own mutation strategies. During the evaluation process, AGDE adjusts F and CR adaptively based on the information of the entire population. Xia et al. [46] incorporated multiple mutation strategies and used credit assignment and probability matching methods to select the most appropriate strategy for each vector in a population. Moreover, they also adaptively determined F and CR to improve the success rate when solving COPs.

Generally speaking, there are two issues when using DE as a search engine [35]: (1) the trial vector generation strategy (i.e., mutation and crossover operators) must be determined; and (2) the control parameter values (i.e., scale factor F and crossover rate CR) must be set to appropriate values. These two issues are not well addressed by basic DE because all the vectors share the same trial vector generation strategy and control parameters. As Mallipeddi et al. indicated [24], the trial vector generation strategy and the associated F and CR values determine the performance of DE. As an alternative, DE with multiple trial vector generation strategies has been shown to be powerful and effective in improving the DE search capability [16,24,36]. On the other hand, the feasibility rule has been the most popular CHT over the last few years, but it has been criticized for its relatively excessive greediness. Considering these problems, in this paper, we propose a new constrained optimization algorithm by combining adaptive DE with a cluster-replacement-based feasibility rule (CACDE). The adaptive DE adaptively assigns mutation strategies, F and CR by learning from previous generations of creating potential vectors. Regarding the CHT, the feasibility rule is applied. Moreover, to alleviate the selection pressure, a clustering technique is used to replace some poor solutions with an archived infeasible vector with a low objective value. The main contributions of this study are as follows:

(1) A DE approach based on both multiple mutation strategies and control parameters is designed as a search engine useful for solving COPs;
(2) An adaptive selection mechanism is applied to determine the most appropriate mutation strategy and control parameters for each target vector;
(3) A cluster-replacement-based operator is integrated into the basic feasibility rule to alleviate the greediness;
(4) The performance of CACDE is comprehensively evaluated over 2 sets of artificial problems and 5 widely used constrained mechanical design problems.

The remainder of this paper is structured as follows. Section 2 briefly introduces the basic DE algorithm and some concepts regarding the feasibility rule. Section 3 presents our proposed CACDE algorithm in detail. Section 4 presents the experimental results and performance comparisons. Section 5 provides detailed discussions of our adaptive DE approach and the cluster-replacement-based CHT. Finally, Section 6 draws conclusions.

2. Basic concepts

2.1. Differential evolution

Like other EAs, DE maintains a main population with N n-dimensional vectors, i.e., $X^t = \{x_1^t, \ldots, x_N^t\}$, where t is the generation number and $x_i^t = [x_{i,1}^t, \ldots, x_{i,n}^t]^T$ is the i-th vector. DE employs mutation, crossover and selection operators sequentially and iteratively until a termination condition is met [33].

The widely used DE/rand/1 mutation operator functions as follows:

$$v_i^t = x_{r_1}^t + F \cdot (x_{r_2}^t - x_{r_3}^t) \tag{2}$$

where indices r_1, r_2 and r_3 are mutually exclusive integers randomly generated within [1,N] that are also different from index i, and the scale factor F is a real parameter that is usually within (0 1].

The most frequently used binomial crossover strategy operates as follows:

$$u_{i,j}^t = \begin{cases} v_{i,j}^t, & \text{if } (\text{rand}_{i,j}(0,1) \le CR) \text{ or } (j = sn), \\ x_{i,j}^t, & \text{otherwise.} \end{cases} \quad j = 1, 2, \ldots, n \tag{3}$$

where $sn = \text{rndint}(1, n)$ is an arbitrary number in $\{1, 2, \ldots, n\}$ and $u_{i,j}^t$ is the jth element of the ith new trial vector. Also, the crossover rate CR is a real parameter between 0 and 1.

After obtaining a trial vector, DE performs the greedy selection operation to form the main population of the next generation:

$$x_i^{t+1} = \begin{cases} u_i^t, & \text{if } f(u_i^t) \le f(x_i^t)), \\ x_i^t, & \text{otherwise.} \end{cases} \tag{4}$$

where x_i^{t+1} is the ith candidate solution in the main population.

2.2. Feasibility rule

At present, the feasibility rule is one of the most widely used CHTs because of its simplicity and efficiency [6]. Basically, the feasibility rule prefers feasible solutions over infeasible ones. To use this rule, the degree of the overall constraint violation must be calculated for every vector in the population [35]. To achieve this goal, the degree of the constraint violation on the jth constraint is calculated as follows:

$$\phi_j(x) = \begin{cases} \max\{g_j(x), 0\}, & j = 1, 2, \ldots, p, \\ \max\{|h_j(x)| - \delta, 0\}, & j = p+1, \ldots, q \end{cases} \tag{5}$$

Then, the overall constraint violation is calculated as the sum of all constraint violations:

$$G(x) = \sum_{j=1}^{q} \phi_j(x) \tag{6}$$

where δ is the positive tolerance value used when transforming the equality constraints into inequality ones.

After obtaining the objective function value and the overall constraint violation for each vector, the feasibility rule functions as follows: when comparing two solutions (i.e., x and y), solution x is considered better than solution y (denoted as $x \prec y$) when one of the following conditions is met:

(1) solution x is feasible but y is not, i.e., $G(x) = 0$ and $G(y) > 0$;
(2) both x and y are infeasible, but x has a smaller overall constraint violation, i.e., $0 < G(x) < G(y)$;
(3) both x and y are feasible, but x has a lower objective value, i.e., $G(x) = G(y) = 0$ and $f(x) < f(y)$.

Equivalently, solution y can be denoted as worse than solution x ($y \succ x$).

3. Our proposed algorithm: CACDE

3.1. Basic idea of CACDE

Our CACDE algorithm uses DE with an adaptive trial vector generation strategy as the main search engine, while the cluster-replacement-based feasibility rule serves as the main CHT. The main characteristic of the adaptive DE approach is the introduction of a mutation strategy candidate pool (Mpool), a scale factor candidate pool (Fpool) and a crossover rate candidate pool (CRpool), and each element in every candidate pool is assigned a selection probability. During the trial vector generation step, a mutation strategy, a scale factor value and a crossover rate are selected from the corresponding candidate pool according to the selection probabilities. The selection probabilities are dynamically updated at each generation based on the knowledge learned from previous searches that generated improved vectors. The cluster-replacement-based feasibility rule consists of two steps: First, each target vector is compared with the corresponding trial vector based on the feasibility rule. If a trial vector fails to survive to the main population of the next generation but has a lower objective value than the corresponding target vector, the trial vector will be selected and stored in an external archive. Then, the main population is grouped into several clusters, and one vector in each cluster is conditionally replaced with an archived infeasible vector. The pseudocode of the proposed CACDE is presented in Algorithm 1. Below, we elaborate on some of the main steps.

3.2. DE with multiple mutation strategies

Many mutation strategies exist in the DE literature, each with its own characteristics [33]. Since DE first appeared, DE algorithms that use multiple trial vector generation strategies have proven to be powerful and effective in improving the DE search capability [24,36]. To use multiple strategies, we must address two questions:

(1) How many and which mutation strategies should be chosen or designed?
(2) How do we use multiple strategies or how do we assign a strategy for each vector?

To address the first question, in CACDE, three strategies with diverse characteristics are selected to form a candidate pool (Mpool). The names and main characteristics of these strategies are as follows:

(1) DE/current-to-best/1: This strategy relies on the best solutions and usually performs well and converges quickly when solving unimodal problems. However, it is more likely to get stuck in a local optimum, which results in premature convergence when solving multimodal problems.
(2) DE/current-to-rand/1: This strategy applies the rotation-invariant arithmetic crossover rather than the binomial crossover to generate the trial vector. Thus, this strategy is rotation-invariant and suitable for problems with a shifted and rotated global optimum that have been rotated using a rotation matrix.
(3) DE/rand/2: This strategy usually converges slowly but exhibits a powerful exploration ability. Therefore, it is more appropriate than best-solution-based strategies when solving multimodal problems.

It is noteworthy that all three strategies in Mpool are two-difference vector-based strategies, which may result in better perturbation than one-difference vector-based strategies when solving complex problems.

To address the second question, an adaptive operator selection mechanism is applied [9]. Specifically, each candidate strategy in Mpool is assigned a selection probability, and one strategy is selected for every target vector during the mutation step. Moreover, an adaptive mechanism is proposed to update these selection probabilities at every generation based on knowledge learned from previous searches in generating improved vectors (this will be introduced in Section 3.4).

Algorithm 1 Pseudocode of CACDE algorithm

Set the parameter values: population size N, maximal number of function evaluations MaxFES, learning rate η, maximal cluster number $P_{\max}$, mutation strategy pool Mpool, scale factor pool Fpool and crossover rate pool CRpool;
Let the generation number $t = 0$ and generate the main population $X^t = \{x_1^t, \cdots, x_N^t\}$, with $x_i^t = [x_{i,1}^t, \cdots, x_{i,n}^t]^T$, $i = 1, \cdots, N$ uniformly distributed in the range $[a, b]$, where $a = [a_1, \cdots, a_n]^T$ and $b = [b_1, \cdots, b_n]^T$;
Randomly select a mutation strategy, a scale factor and crossover rate for each vector from the corresponding candidate pool;
Evaluate the objective value $f(x_i^t)$ and overall constraint violation $G(x_i^t)$, $i = 1, \cdots, N$;
Let FES = N and $Y^t = \emptyset$;
while FES < maxFES **do**
  for $i = 1 : N$ **do**
    Generate the trial vector $u_i^t = [u_{i,1}^t, \cdots, u_{i,n}^t]^T$ using the mutation strategy M_i^t, scale factor F_i^t and crossover rate CR_i^t associated with x_i^t;
    Evaluate the objective value $f(u_i^t)$ and overall constraint violation $G(u_i^t)$;
    FES = FES + 1;
  end for
  for $i = 1 : N$ **do**
    Compare x_i^t and u_i^t according to the feasibility rule
    if u_i^t is better than x_i^t **then**
      Replace x_i^t with u_i^t;
    else
      if $f(u_i^t) < f(x_i^t)$ **then**
        $Y^t = Y^t \cup u_i^t$
      end if
    end if
  end for
  Perform cluster and replacement operation (see Algorithm (2) for details);
  Update the selection probability for each element in every candidate pool;
  for all $i = 1 : N$ **do**
    Determine the mutation strategy, scale factor and crossover rate for the ith vector according to Equation (8);
  end for
  Let $Y^t = \emptyset$
  Let $t = t + 1$
end while

3.3. DE with multiple control parameters

In basic DE, all vectors use the same N, F and CR, and the values are kept unchanged from the beginning to the end of the evolution. Because the setting of N is highly dependent on the complexity of a given problem, we keep N fixed throughout the entire evolution process and choose some discrete parameters for potential F and CR values. As with the previous settings, each parameter value is also assigned a selection probability, and an adaptive mechanism is used to update the selection probabilities at every generation based on the knowledge learned from previous searches in generating improved vectors (this will be introduced in Section 3.4).

In general, a low F value can result in a small distribution around the base vector; thus, the DE algorithm usually converges quickly and achieves reliable results. In contrast, a large F value can result in a wide distribution over the search space and increase the trial vector diversity. For CR, a large value enhances trial vector diversity because it causes the trial vector to inherit more information from the mutant vectors. In contrast, a low value causes the trial vector to inherit little information from the mutant vectors. In this circumstance, it is very suitable for solving separable problems. Thus, to balance the search engine's exploration and exploitation abilities while solving particular problems with different characteristics, five values taken from the range [0.4, 1.2] with a step size of 0.2 constitute the Fpool, and five values taken from the range [0.2, 1.0] with a step size of 0.2 constitute the CRpool. Please note that we include a large F value (i.e., $F = 1.2$) in the Fpool to increase the probability of escaping from a local optimum. An $F > 1$ can solve some problems and is occasionally effective [39,47].

3.4. Adaptive mechanism for updating the selection probabilities

Being different from basic DE, each target vector in CACDE has its own mutation strategy, scale factor and crossover rate. Table 1 presents these relationships. To select the most suitable trial vector generation strategy from the corresponding pool, each element in the pool is assigned a selection probability, as illustrated in Table 2, and elements with larger

Table 1
Vector with independent mutation strategy, scale factor and crossover rate.

Vector	Mutation strategy	Scale factor	Crossover rate
x_1^t	M_1^t	F_1^t	CR_1^t
x_2^t	M_2^t	F_2^t	CR_2^t
$\cdots$	$\cdots$	$\cdots$	$\cdots$
x_N^t	M_N^t	F_N^t	CR_N^t

Table 2
Selection probability assignment.

Index	Mutation strategy		Scale factor		Crossover rate	
	Element	Probability	Element	Probability	Element	Probability
1	DE/current-to-best/1	$P_M(1)$	0.4	$P_F(1)$	0.2	$P_C(1)$
2	DE/current-to-rand/1	$P_M(2)$	0.6	$P_F(2)$	0.4	$P_C(2)$
3	DE/rand/2	$P_M(3)$	0.8	$P_F(3)$	0.6	$P_C(3)$
4			1.0	$P_F(4)$	0.8	$P_C(4)$
5			1.2	$P_F(5)$	1.0	$P_C(5)$
		$\sum_{i=1}^{3} P_M(i) = 1$		$\sum_{i=1}^{5} P_F(i) = 1$		$\sum_{i=1}^{5} P_C(i) = 1$

selection probabilities are more likely to be selected. Moreover, an adaptive mechanism is proposed to update the selection probabilities dynamically according to the previous search in generating improved solutions.

Suppose that the ith vector at generation t is x_i^t and that the mutation strategy, scale factor and crossover probability are M_i^t, F_i^t and CR_i^t, respectively. Initially, these values are determined randomly. After generating trial vectors u_i^t for each target vector at generation t, a pairwise comparison between x_i^t and u_i^t is performed according to the feasibility rule. Based on the comparison results, the selection probabilities for all elements are updated as follows [9]:

$$P(k) = (1 - \eta) \times P(k) + \eta \times \frac{ns(k)}{nt(k)} + \varepsilon \tag{7}$$

where η is the learning rate and is always a small positive value, $\varepsilon = 1 \times 10^{-30}$ is used to avoid possible null success rates, and $ns(k)$ and $nt(k)$ respectively denote the number of successful kth elements and the total number of kth elements in the candidate pool at generation t. Subsequently, all the selection probabilities are normalized for ease of use.

After obtaining the normalized selection probabilities, the new mutation strategy, scale factor and crossover rate for the ith vector at generation $t+1$ are updated as follows:

$$\Theta_i^{t+1} = \begin{cases} \Theta_i^t & \text{if} \quad u_i^t \prec x_i^t \\ RWS(\Theta\text{pool}) & \text{otherwise} \end{cases}, \Theta = M, F, CR \tag{8}$$

where function $RWS(\cdot)$ denotes the selection of a value from the corresponding candidate pool using the roulette wheel selection method. It can be observed from Eq. (8) that if u_i^t is better than x_i^t, then M_i^t, F_i^t and CR_i^t are kept unchanged; otherwise M_i^t, F_i^t and CR_i^t will be adjusted, and those elements with larger selection probabilities are more likely to be selected and, hence, propagate potential solutions.

3.5. Cluster-replacement-based feasibility rule

The feasibility rule is one of the most commonly used CHTs. However, it has also been criticized for its excessive greediness. To alleviate this greediness, some modifications have been proposed, such as stochastic ranking [38] and the replacement mechanism [43]. Following these guiding principles, we propose a new cluster-replacement-based feasibility rule as the CHT for CACDE,. This new CHT works as follows:

Suppose that the ith vector at generation t is x_i^t and the corresponding trial vector is u_i^t. First, the feasibility rule is applied directly to compare x_i^t with u_i^t, and the better of the two is selected to survive into the main population of the next generation. Meanwhile, when u_i^t is not selected to fill in the main population but u_i^t has a lower objective value than x_i^t (i.e., $u_i^t \succ x_i^t$ and $f(u_i^t) < f(x_i^t)$) then u_i^t is stored in the external archive, Y^t. It can be deduced that all vectors in Y^t are infeasible solutions. The reason is as follows: since $f(u_i^t) < f(x_i^t)$, if u_i^t is feasible, then u_i^t would be better than x_i^t. In fact, u_i^t is worse than x_i^t based on the feasibility rule. After storing all the infeasible u_i^t satisfying $f(u_i^t) < f(x_i^t)$ into Y^t, we group the main population into N_c disjoint clusters according to the locations of the vectors in the design space. We denote the vector with the maximal overall constraint violation in the kth cluster as z and the vector with the minimal overall constraint violation in Y^t as y. For each cluster, we compare z with y. When $f(y) < f(z)$, we replace z with y and remove y from Y^t.

Based on the above steps,the process of grouping population X with N vectors into N_c clusters occurs as detailed in Algorithm 2. Note that the parameter P_{max} is used to avoid excessive replacement. When P_{max} is too small, few replacements

Algorithm 2 Procedure of cluster and replacement loop

Input: main population: $X^t = \{x_1^t, \cdots, x_N^t\}$; external archive: $Y^t = \{y_1^t, \cdots, y_K^t\}$; maximal number of clusters: P_{max};
Output: X';
Let $X' = \varnothing$;
Calculate the number of clusters: $N_c = \min\{P_{max}, K\}$;
Equally assign the number of vectors in each cluster: $N_s(k), k = 1, 2, \cdots, N_c$;
Generate a reference vector R randomly from the search space;
$k = 1$
while $k \le N_c$ **do**
    Find the nearest vectors $\bar{x} \in X^t$ to R;
    Determine the kth cluster Z^k=$\{\bar{x}\} \bigcup \{N_s(k) - 1$ vectors in X^t that are nearest to $\bar{x}\}$;
    Eliminate Z^k from X^t: $X^t = X^t \backslash Z^k$
    Find the vector $z \in Z^k$ with the maximal overall constraint violation, i.e., $z = \arg\max(G(Z^k))$;
    Find the vector $y \in Y^t$ with the minimal overall constraint violation, i.e., $y = \arg\min(G(Y^t))$;
    if $(f(y) < f(z))$ **then**
        Replace z with y;
    end if
    Let $X' = X' \bigcup Z^k$ and $Y^t = Y^t \backslash y$;
    $k = k + 1$;
end while
return X' as the new target vectors at generation $t+1$;

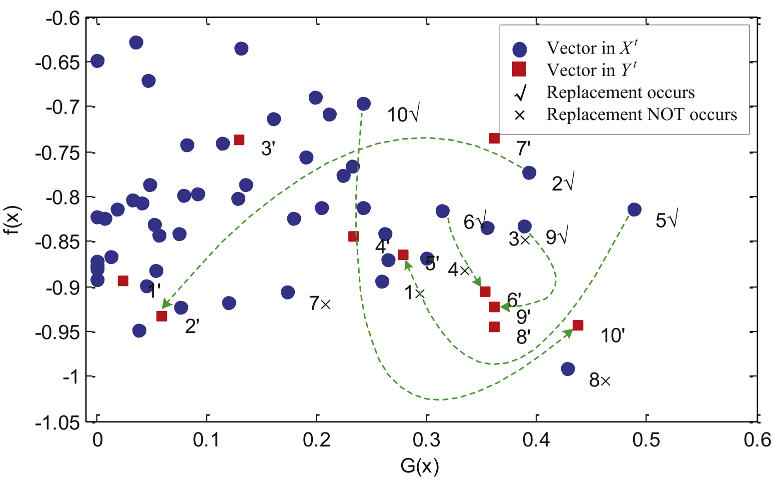

Fig. 1. Illustration of replacement-based feasibility rule.

will occur in a small region, and the goal of alleviating greediness will not be achieved. In contrast, when P_{max} is too large, replacements will occur frequently, which impairs the convergence.

Next, we employ an example (instance G12 in Section 4) to demonstrate the benefit of replacement. Suppose that the main population X^t contains 50 vectors and that the current external archive Y^t contains 10 vectors, as shown in Fig. 1. Then, the main population will be grouped into 10 clusters, and 10 pairwise comparisons will occur, as illustrated in Table 3. Clearly, 5 out of the 10 vectors (i.e., vectors 2, 5, 6, 9 and 10) satisfy $f(y) < f(z)$; therefore, replacement will occur. According to the values of $f(.)$ and $G(.)$, these 5 replacements can be classified into 2 types:

(1) Type I: Conditions $f(y) < f(z)$ and $G(y) < G(z)$ are satisfied;
(2) Type II: Conditions $f(y) < f(z)$ and $G(y) > G(z)$ are satisfied.

For Type I (see vectors 2, 5 and 9), it is beneficial to converge to a feasible global optimum because the vector z has both a lower objective value and a lower overall constraint violation. Regarding Type II (see vectors 6 and 10), vector z has a larger overall constraint violation than does y, but its objective value is lower. This type of replacement increases the diversity of the main population and is likely to guide the search towards a region with a lower objective value. Combining these two types of replacement makes the entire search more likely to converge to the global feasible optimum.

Table 3
Illustration of replacement based feasibility rule.

Index	y	z	$f(y) < f(z)$?	Repalce?	Type
1	x_1	$x_{1'}$	NO	×	–
2	x_2	$x_{2'}$	YES	√	I
3	x_3	$x_{3'}$	NO	×	–
4	x_4	$x_{4'}$	NO	×	–
5	x_5	$x_{5'}$	YES	√	I
6	x_6	$x_{6'}$	YES	√	II
7	x_7	$x_{7'}$	NO	×	–
8	x_8	$x_{8'}$	NO	×	–
9	x_9	$x_{9'}$	YES	√	I
10	x_{10}	$x_{10'}$	YES	√	II

4. Experimental studies

4.1. Test instances

To evaluate the performance of CACDE, we conducted comprehensive experiments on 42 well-known constrained real-parameter optimization problems from CEC2006 and CEC2010. Instances from CEC2006 involve five types of objective functions (linear, nonlinear, polynomial, quadratic, and cubic), with different numbers of design variables (between 2 and 24) and four kinds of constraints (linear inequalities, linear equalities, nonlinear inequalities, and nonlinear equalities) with different numbers of constraints (between 1 and 38). The instances from CEC2010 are more challenging than those from CEC2006. These test instances can be extended to 10 and 30 dimensions. In addition, 12 of 18 instances have one or more equality constraints, and 13 out of 18 have a feasibility ratio of less than 1.0×10^{-6}. The main characteristics of these instances can be found in [19] and [23].

4.2. Parameter settings

Based on the recommendation in [1,32,41], the population size N in CACDE is determined as follows:

$$N = \begin{cases} 50, & 0 < n \leq 5 \\ 80, & 5 < n \leq 10 \\ 100, & 10 < n \leq 30 \end{cases} \tag{9}$$

For the other parameters, we use $P_{max} = 0.1N$, $\eta = 0.01$. Note that for instances with equality constraints, dynamic setting of the tolerance value δ in Eq. (5) is adopted as follows to convert the equality constraints into inequality constraints [10,43]:

$$\delta(t+1) = \begin{cases} \delta(t)/\delta', & \text{if } \delta(t) > 0.0001, \\ 0.0001, & \text{otherwise.} \end{cases} \tag{10}$$

where t is the generation number and $\delta' = 1.015$ is the change rate. The initial value of $\delta(t)$, i.e., $\delta(0)$, is set to $n(\log_{10}(\max_j(b_j - a_j)) + 1)$.

4.3. General performance of CACDE on CEC2006 problems

As suggested by Liang et al. [19], 25 independent runs were executed for each instance. The optimal function error values $(f(x) - f(x^*))$ achieved after 5×10^3, 5×10^4 and 5×10^5 function evaluations (FEs) are summarized in terms of ‘Best’, ‘Median’, ‘Worst’, c, ‘$\overline{v}$’, ‘Mean’ and ‘Std’ (standard deviation) in Tables 4–7. Here, c is a sequence of three numbers denoting the number of violated constraints for the median solution that are greater than 1.0, between 0.01 to 1.0, and between 0.0001 to 0.1. The $\overline{v}$ value is the mean value of all constraint violations for the median solution. The numbers in the parenthesis after the error values of the best, median, and worst solutions are the numbers of constraints unable to satisfy the feasible condition for the best, median and worst solutions, respectively.

As shown in Tables 4–7, CACDE finds feasible solutions in all runs using 5×10^3 FEs for 14 test instances (i.e., G01, G02, G04, G06, G07, G08, G09, G10, G11, G12, G16, G18, G19 and G24). For 4 instances (i.e., G03, G05, G13 and G15), CACDE obtains feasible solutions using 5×10^4 FEs, and for another 4 test instances (i.e., G14, G17, G21 and G23), CACDE reaches feasible solutions using 5×10^5 FEs. Note that CACDE fails to obtain any feasible solutions for two instances, G20 and G22, in 5×10^5 FEs. These two instances are highly constrained and—to the best of our knowledge—no feasible solutions have been reported. The obtained median solution found for G20 by CACDE is slightly infeasible ($\overline{v} = 8.10 \times 10^{-3}$), and 9 out of 20 constraints are unsatisfied. In addition, the obtained best solution for G20 is far away from the feasible region, and 12 out of 20 constraints are unsatisfied in the median solution.

Table 4
Error values achieved when FES = 5×10^3, FES = 5×10^4, FES = 5×10^5 for instances 1–6.

FES		G01	G02	G03	G04	G05	G06
	Best	5.78E+00(0)	3.57E−01(0)	−1.18E−01(1)	7.11E+00(0)	−5.93E+00(3)	3.74E−01(0)
	Median	6.95E+00(0)	4.15E−01(0)	9.89E−01(1)	1.42E+01(0)	6.19E+00(3)	1.56E+00(0)
	Worst	8.31E+00(0)	4.74E−01(0)	1.00E+00(1)	3.22E+01(0)	9.33E+01(3)	5.49E+00(0)
5×10^3	c	0,0,0	0,0,0	0,1,0	0,0,0	0,3,0	0,0,0
	$\bar{v}$	0.00E+0	0.00E+0	1.72E−01	0.00E+0	2.20E−01	0.00E+0
	Mean	6.97E+0	4.15E−01	8.60E−01	1.55E+01	1.38E+01	1.82E+0
	Std	7.33E−01	3.35E−02	2.80E−01	5.90E+0	2.45E+01	1.20E+0
	Best	3.29E−08(0)	1.31E−03(0)	3.10E−04(0)	−3.64E−12(0)	7.25E−09(0)	−1.64E−11(0)
	Median	1.22E−07(0)	1.55E−02(0)	7.17E−04(0)	−3.64E−12(0)	1.05E−07(0)	−1.64E−11(0)
	Worst	4.82E−07(0)	1.03E−01(0)	1.02E−03(0)	−3.64E−12(0)	7.78E−07(0)	−1.64E−11(0)
5×10^4	c	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0
	$\bar{v}$	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0
	Mean	1.62E−07	2.77E−02	6.94E−04	−3.64E−12	1.95E−07	−1.64E−11
	Std	1.16E−07	2.53E−02	2.10E−04	0.00E+0	2.15E−07	0.00E+0
	Best	0.00E+00(0)	4.41E−10(0)	−2.44E−15(0)	−3.64E−12(0)	−9.09E−13(0)	−1.64E−11(0)
	Median	0.00E+00(0)	3.91E−09(0)	−2.00E−15(0)	−3.64E−12(0)	−9.09E−13(0)	−1.64E−11(0)
	Worst	0.00E+00(0)	5.69E−08(0)	−1.55E−15(0)	−3.64E−12(0)	0.00E+00(0)	−1.64E−11(0)
5×10^5	c	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0
	$\bar{v}$	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0
	Mean	0.00E+0	9.34E−09	−2.11E−15	−3.64E−12	−7.28E−13	−1.64E−11
	Std	0.00E+0	1.28E−08	2.04E−16	0.00E+0	3.71E−13	0.00E+0

Table 5
Error values achieved when FES = 5×10^3, FES = 5×10^4, FES = 5×10^5 for instances 7–12.

FES		G07	G08	G09	G10	G11	G12
	Best	8.53E+00(0)	1.15E−05(0)	1.96E+00(0)	3.03E+03(0)	−5.69E−04(1)	6.71E−11(0)
	Median	1.73E+01(0)	1.33E−04(0)	7.38E+00(0)	4.67E+03(0)	1.17E−01(0)	1.72E−09(0)
	Worst	2.56E+01(0)	9.60E−04(0)	1.41E+01(0)	9.63E+03(0)	2.43E−01(0)	1.30E−07(0)
5×10^3	c	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0
	$\bar{v}$	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0
	Mean	1.73E+01	2.60E−04	7.22E+0	5.37E+03	1.33E−01	8.86E−09
	Std	4.32E+0	2.64E−04	2.69E+0	1.81E+03	7.85E−02	2.56E−08
	Best	1.53E−06(0)	4.16E−17(0)	1.59E−10(0)	1.21E−01(0)	2.47E−10(0)	0.00E+00(0)
	Median	7.72E−06(0)	5.77E−06(0)	7.33E−10(0)	4.88E−01(0)	8.09E−05(0)	0.00E+00(0)
	Worst	2.22E−05(0)	2.88E−05(0)	7.60E−09(0)	1.77E+00(0)	1.51E−01(0)	0.00E+00(0)
5×10^4	c	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0
	$\bar{v}$	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0
	Mean	8.90E−06	7.87E−06	1.18E−09	5.05E−01	6.98E−03	0.00E+0
	Std	4.97E−06	7.43E−06	1.57E−09	3.60E−01	3.01E−02	0.00E+0
	Best	−2.42E−13(0)	2.78E−17(0)	−2.27E−13(0)	−8.19E−12(0)	1.10E−12(0)	0.00E+00(0)
	Median	−2.27E−13(0)	2.78E−17(0)	−1.14E−13(0)	−7.28E−12(0)	1.21E−11(0)	0.00E+00(0)
	Worst	−2.20E−13(0)	5.41E−07(0)	−1.14E−13(0)	−6.37E−12(0)	6.90E−11(0)	0.00E+00(0)
5×10^5	c	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0
	$\bar{v}$	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0
	Mean	−2.29E−13	2.80E−08	−1.27E−13	−7.31E−12	1.73E−11	0.00E+0
	Std	5.04E−15	1.10E−07	3.77E−14	3.19E−13	1.60E−11	0.00E+0

As suggested by Liang et al. [19], the 'Best', 'Median', 'Worst', 'Mean' and 'Std' values corresponding to the number of successful runs, feasible rate, success rate and success performance over 25 runs are reported in Table 8. Here, a feasible run is defined as a run during which at least one feasible solution is found in 5×10^4 FES; a successful run is defined as a run during which a feasible solution x satisfying $f(x)-f(x^*)\le 0.0001$ is found; the feasible rate is the ratio of the number of feasible runs to total runs; the success rate is the ratio of the number of successful runs to total runs; and the success performance is the mean FEs for successful runs multiplied by the total number of runs and divided by the number of successful runs.

Because CACDE did not find any feasible solutions in 5×10^5 FEs for G20 and G22, the corresponding results are not included in Table 8. For the other 22 instances, Table 8 shows that CACDE achieves a 100% feasible rate and success rate. Regarding the success performance, CACDE requires less than 5×10^3 FEs for 2 instances (i.e., G12 and G24), 5×10^4 FEs for 11 instances (i.e., G01, G04, G05, G06, G07, G08, G09, G13, G15, G16 and G18), 1×10^5 FEs for 7 instances (i.e., G02, G03, G11, G14, G17, G19 and G21), and 5×10^5 FEs for 2 instances (i.e., G10 and G23), respectively.

To provide more information regarding the performance of CACDE when solving all 22 instances, Figs. 2–5 show the convergence graphs of $\log_{10}(f(x)-f(x^*))$ versus the FEs of the median run over 25 independent runs. Please note that the solutions satisfying $f(x)-f(x^*)\le 0$ are not plotted in these figures.

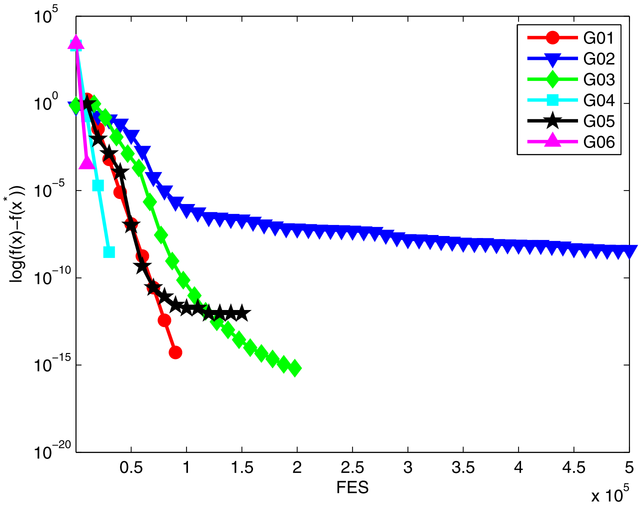

Fig. 2. Convergence graph for G01–G06.

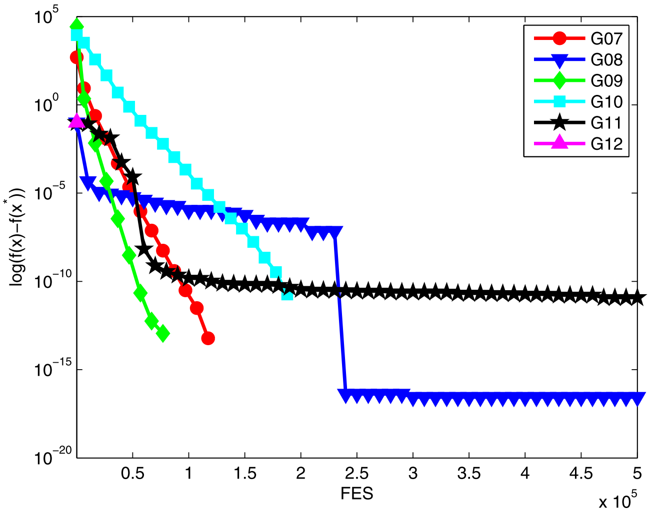

Fig. 3. Convergence graph for G07–G12.

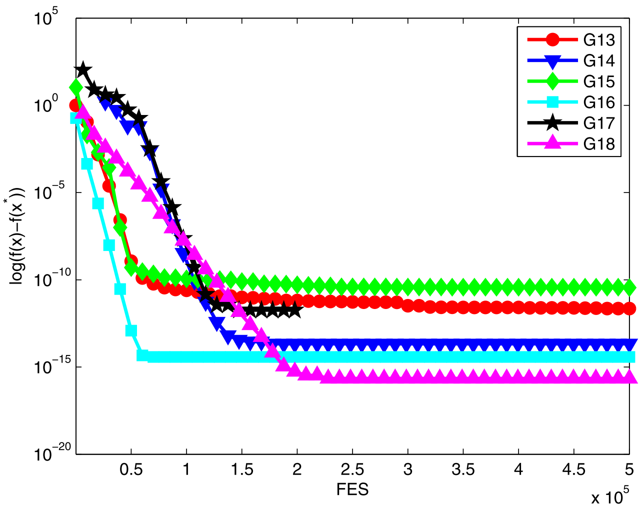

Fig. 4. Convergence graph for G13–G18.

Table 6
Error values achieved when FES = 5×10^3, FES = 5×10^4, FES = 5×10^5 for instances 13–18.

FES		G13	G14	G15	G16	G17	G18
5×10^3	Best	2.18E−02(3)	−1.55E+02(3)	−1.14E−01(2)	5.62E−03(0)	−9.78E+01(4)	4.00E−01(0)
	Median	6.51E−01(3)	−1.13E+02(3)	1.59E−01(2)	1.15E−02(0)	1.09E+02(4)	6.33E−01(0)
	Worst	4.33E+00(3)	−7.36E+01(3)	2.12E+00(2)	2.60E−02(0)	3.46E+02(4)	8.49E−01(2)
	c	0,2,1	3,0,0	0,2,0	0,0,0	1,3,0	0,0,0
	$\bar{v}$	1.95E−01	3.51E+0	3.46E−02	0.00E+0	8.98E−01	0.00E+0
	Mean	7.06E−01	−1.14E+02	2.34E−01	1.26E−02	1.04E+02	6.24E−01
	Std	8.36E−01	2.04E+01	4.31E−01	4.49E−03	1.07E+02	1.37E−01
5×10^4	Best	1.73E−10(0)	8.28E−03(2)	1.14E−10(0)	1.80E−14(0)	1.69E−02(4)	7.07E−06(0)
	Median	1.16E−09(0)	8.55E−02(3)	5.09E−10(0)	1.23E−13(0)	6.01E−01(4)	8.90E−05(0)
	Worst	3.19E−04(3)	1.07E+00(2)	2.27E−09(0)	1.66E−12(0)	1.00E+02(4)	3.09E−04(0)
	c	0,0,0	0,0,2	0,0,0	0,0,0	0,0,4	0,0,0
	$\bar{v}$	0.00E+0	2.40E−04	0.00E+0	0.00E+0	4.11E−04	0.00E+0
	Mean	1.76E−05	1.45E−01	6.78E−10	2.80E−13	3.64E+01	1.11E−04
	Std	6.73E−05	2.11E−01	5.52E−10	4.07E−13	4.88E+01	8.29E−05
5×10^5	Best	2.99E−13(0)	1.42E−14(0)	1.53E−11(0)	3.77E−15(0)	0.00E+00(0)	1.11E−16(0)
	Median	2.25E−12(0)	2.13E−14(0)	3.57E−11(0)	3.77E−15(0)	0.00E+00(0)	2.22E−16(0)
	Worst	7.08E−12(0)	2.13E−14(0)	7.62E−11(0)	3.77E−15(0)	0.00E+00(0)	2.22E−16(0)
	c	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0
	$\bar{v}$	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0
	Mean	2.26E−12	1.79E−14	3.68E−11	3.77E−15	0.00E+0	2.18E−16
	Std	1.37E−12	3.62E−15	1.35E−11	0.00E+0	0.00E+0	2.22E−17

Table 7
Error values achieved when FES = 5×10^3, FES = 5×10^4, FES = 5×10^5 for instances 19–24.

FES		G19	G20	G21	G22	G23	G24
5×10^3	Best	9.36E+01(0)	1.46E+00(20)	−2.73E+01(5)	4.95E+02(20)	−1.30E+03(5)	1.72E−05(0)
	Median	1.35E+02(0)	2.98E+00(20)	2.66E+02(5)	5.69E+03(19)	−4.85E+02(4)	6.99E−05(0)
	Worst	2.16E+02(0)	3.93E+00(20)	8.03E+02(5)	1.77E+04(19)	7.10E+02(5)	3.73E−04(0)
	c	0,0,0	2,18,0	0,5,0	18,1,0	2,2,0	0,0,0
	$\bar{v}$	0.00E+0	2.03E+0	3.66E−01	4.00E+06	7.95E−01	0.00E+0
	Mean	1.48E+02	2.95E+0	3.07E+02	5.92E+03	−4.69E+02	1.21E−04
	Std	3.62E+01	6.33E−01	2.37E+02	4.61E+03	4.37E+02	1.06E−04
5×10^4	Best	1.07E−02(0)	−1.37E−02(20)	4.37E−01(5)	−2.22E+02(20)	2.74E+00(4)	3.29E−14(0)
	Median	2.74E−02(0)	6.23E−03(19)	4.24E+00(5)	1.54E+02(20)	1.80E+01(3)	3.29E−14(0)
	Worst	1.04E−01(0)	2.69E−02(20)	1.19E+02(3)	2.52E+03(19)	7.22E+01(5)	3.29E−14(0)
	c	0,0,0	0,5,13	0,0,4	19,1,0	0,0,2	0,0,0
	$\bar{v}$	0.00E+0	2.15E−02	1.76E−04	2.79E+04	5.46E−04	0.00E+0
	Mean	3.47E−02	6.90E−03	1.72E+01	4.58E+02	2.31E+01	3.29E−14
	Std	2.18E−02	1.15E−02	2.84E+01	7.18E+02	1.94E+01	0.00E+0
5×10^5	Best	2.13E−14(0)	−2.70E−02(9)	−3.70E−10(0)	5.65E+02(12)	−6.82E−13(0)	3.29E−14(0)
	Median	2.84E−14(0)	−4.91E−03(8)	−3.36E−10(0)	1.47E+04(12)	−4.55E−13(0)	3.29E−14(0)
	Worst	2.84E−14(0)	6.40E−03(7)	−2.96E−10(0)	1.96E+04(10)	−1.14E−13(0)	3.29E−14(0)
	c	0,0,0	0,1,7	0,0,0	5,1,0	0,0,0	0,0,0
	$\bar{v}$	0.00E+0	8.10E−03	0.00E+0	2.73E+0	0.00E+0	0.00E+0
	Mean	2.79E−14	−6.02E−03	−3.35E−10	1.24E+04	−5.09E−13	3.29E−14
	Std	1.97E−15	6.87E−03	2.03E−11	6.08E+03	1.36E−13	0.00E+0

As Figs. 2–5 show, the error values decrease dramatically as the FEs increase for all instances. CACDE can obtain near-optimal solutions ($f(x) - f(x^*) \leq 0.0001$) for all instances in 1.5×10^5 FEs. More concretely, for 12 instances (i.e., G01, G03, G04, G05, G06, G07, G9, G10, G12, G13, G21 and G23), CACDE attains slightly better optimal solutions than the best known reported optimal results under 5×10^5 FES. For 4 instances (i.e., G14, G17, G21 and G23), no points are plotted during the early evaluation stage because no feasible solutions are found and the obtained best infeasible solutions have lower objective values than the best known optimal results. However, as the evaluation proceeds, some feasible solutions are found and those points are included in the figures. From the above analysis, it can be concluded that the proposed CACDE converges very quickly for most instances.

4.4. Comparison with COEAs on CEC2006 problems

In the previous subsection, the general performance of CACDE was verified using benchmark functions. In this subsection, CACDE is compared with 7 state-of-the-art COEAs from the literature on the 22 test instances (G20 and G22 are not included). Among these COEAs, the first 4 are DE-based COEAs (i.e., rank-iMDDE [12], CCiALF [11], NDE [31], and DE/HMO/1 [8]), while the remaining 3 are non-DE-based approaches (i.e., M-ABC [27], CVI-PSO [25], and TLBO [37]). To facilitate the

Table 8
Number of FES to achieve the fixed accuracy level $((f(x) - f(x*)) \leq 0.0001)$, success rate, feasible rate and success performance.

Prob.	Best	Median	Worst	Mean	Std	Feasible rate(%)	Success rate(%)	Success performance
G01	32,700	34,500	37,100	34,516	1259.85	100	100	34,516
G02	59,200	68,200	98,700	70,304	8391.24	100	100	70,304
G03	56,240	59,280	60,640	58,973	1100.71	100	100	58,973
G04	17,000	18,450	20,200	18,450	906.34	100	100	18,450
G05	39,500	40,150	42,850	40,442	766.58	100	100	40,442
G06	9100	10,750	11,700	10,628	685.88	100	100	10,628
G07	38,240	41,840	44,720	41,629	1631.67	100	100	41,629
G08	750	7350	15,950	7630	4064.07	100	100	7630
G09	22,640	24,960	27,840	25,056	1140.53	100	100	25,056
G10	84,160	99,840	126,080	102,019	9746.69	100	100	102,019
G11	17,800	49,100	274,500	76,988	67,831.86	100	100	76,988
G12	550	2450	3200	2382	571.89	100	100	2382
G13	28,100	34,550	54,000	35,844	5901.05	100	100	35,844
G14	67,840	73,360	122,400	81,370	15,956.04	100	100	81,370
G15	25,750	32,600	35,350	32,536	2018.84	100	100	32,536
G16	11,900	12,900	14,200	12,950	701.93	100	100	12,950
G17	72,400	75,520	277,600	91,686	55,900.28	100	100	91,686
G18	40,320	49,520	59,120	49,354	5571.59	100	100	49,354
G19	85,700	98,200	132,300	99,512	9209.52	100	100	99,512
G21	79,040	82,080	188,960	88,602	22,294.34	100	100	88,602
G23	94,800	105,440	317,120	114,694	42,883.90	100	100	114,694
G24	3650	4950	5850	4992	508.81	100	100	4992

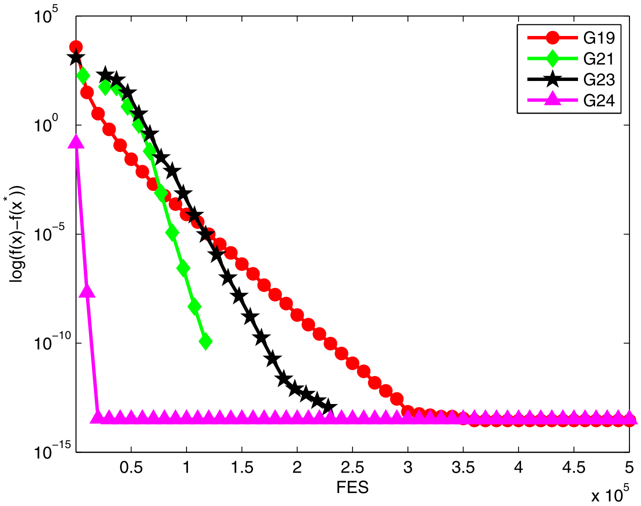

Fig. 5. Convergence graph for G19–G24.

comparison and analysis, the maximum number of function evaluations is set to 240,000 for CACDE and all competing approaches.

Table 9 summarizes the results in terms of ‘Best’, ‘Mean’ and ‘Std’. The statistical results of the 7 competing approaches are taken directly from the corresponding studies. As Table 9 shows, CACDE consistently obtains highly competitive results on all the functions compared with the 7 other COEAs. In terms of ‘Best’ and ‘Mean’, CACDE obtains the best or comparable results among all eight COEAs on all 22 functions.

Based on the ‘Mean’ values in Table 9, the average rankings of the 8 approaches according to the Friedman test are shown in Fig. 6, where CACDE is ranked slightly higher than NDE and lower than other COEAs, which means that CACDE performs a little worse than NDE and much better than other COEAs for CEC2006.

To determine the statistical differences systematically, multiple-paired Wilcoxon signed-rank tests were conducted for the above 8 approaches on the 22 test instances. Table 10 summarizes the results for all the pairwise comparisons involving CACDE. Here, R^+ represents the sum of ranks for all the problems in which CACDE outperforms the compared approach, and R^- represents the sum of ranks for the opposite case. In Table 10, CACDE exhibits significant improvements over M-ABC, CVI-PSO and TLBO at a significance level of $\alpha = 0.05$, while there are no significant differences between CACDE and rank-iMDDE, CCiALF, NDE and DE/HMO/1.

Table 9
Experimental comparison between CACDE with DE-based COEAs on CEC2016.

Algorithm	G01/−15.0000000000			G02/−0.8036191042		
	Best	Mean	Std	Best	Mean	Std
CACDE	−15	−15	0.00E+00	−0.8036191	−0.803619035	7.99E−08
rank-iMDDE	−15	−15	0.00E+00	−0.80361905	−0.802021189	4.57E−03
CCiALF	−15	−15	2.39E−08	−0.8036176	−0.7930875	8.30E−03
NDE	−15.000000	−15.000000	0.00E+00	−0.80348	−0.801809	5.10E−04
DE/HMO/1	−15.0000	−15.0000	0.00E+00	−0.8036191	−0.797650	5.21E−03
M-ABC	−15	−15	0.00E+00	−0.803598	−0.792412	6.84E−03
CVI-PSO	−15	−15	4.50E−16	−0.80009774	−0.790875577	1.09E−02
TLBO	−15	−15	0.00E+00	−0.803619	−0.803619	0.00E+00
Algorithm	G03/−1.0005001000			G04/−30665.5386717834		
	Best	Mean	Std	Best	Mean	Std
CACDE	−1.0005001	−1.0005001	9.39E−16	−30665.53867	−30665.53867	3.71E−12
rank-iMDDE	−1.0005001	−1.0005001	0.00E+00	−30665.5387	−30665.53867	0.00E+00
CCiALF	−1.000501	−1.000501	1.69E−08	−30665.539	−30665.539	9.80E−06
NDE	−1.0005001	−1.0005001	0.00E+00	−30665.539	−30665.539	0.00E+00
DE/HMO/1	−1.0005	−1.0005	5.21E−07	−30665.539	−30665.539	0.00E+00
M-ABC	−1.000	−1.000	4.68E−05	−30665.539	−30665.539	2.22E−11
CVI-PSO	−1.000000	−1.000000	3.70E−16	−30665.8217	−30665.82100	3.39E−03
TLBO	−1.0005	−1.0003	1.40E−04	−30665.539	−30665.539	0.00E+00
Algorithm	G05/5126.4967140071			G06/−6961.8138755802		
	Best	Mean	Std	Best	Mean	Std
CACDE	5126.496714	5126.496714	2.66E−12	−6961.813876	−6961.813876	0.00E+00
rank-iMDDE	5126.496714	5126.496714	0.00E+00	−6961.81388	−6961.813876	0.00E+00
CCiALF	5126.4967	5126.497	9.17E−08	−6961.814	−6961.8114	5.19E−11
NDE	5126.49671	5126.49671	0.00E+00	−6961.81388	−6961.81388	0.00E+00
DE/HMO/1	5126.497	5126.497	0.00E+00	−6961.814	−6961.814	0.00E+00
M-ABC	5126.736	5178.139	5.61E+01	−6961.814	−6961.814	0.00E+00
CVI-PSO	5127.277667	5127.277667	0.00E+00	−6961.81388	−6961.813876	0.00E+00
TLBO	5126.484	5168.7194	5.41E+01	−6961.814	−6961.814	0.00E+00
Algorithm	G07/24.3062090681			G08/−0.0958250415		
	Best	Mean	Std	Best	Mean	Std
CACDE	24.30620907	24.30620907	7.64E−15	−0.095825041	−0.095824835	3.59E−07
rank-iMDDE	24.3062091	24.3062091	0.00E+00	−0.09582504	−0.095825041	0.00E+00
CCiALF	24.3062	24.3062	6.82E−07	−0.09582505	−0.09582505	6.82E−07
NDE	24.306209	24.306209	1.35E−14	−0.095825	−0.095825	0.00E+00
DE/HMO/1	24.3062	24.3062	3.89E−05	−0.095825	−0.095825	0.00E+00
M-ABC	24.315	24.415	1.24E−01	−0.095825	−0.095825	4.23E−17
CVI-PSO	24.4738268	26.5612953	1.64E+00	−0.10545951	−0.105459505	0.00E+00
TLBO	24.3062	24.31	7.11E−03	−0.095825	−0.095825	0.00E+00
Algorithm	G09/680.6300573745			G10/7049.2480205286		
	Best	Mean	Std	Best	Mean	Std
CACDE	680.6300574	680.6300574	3.60E−13	7049.248021	7049.248021	2.72E−12
rank-iMDDE	680.6300574	680.6300574	0.00E+00	7049.248021	7049.248021	0.00E+00
CCiALF	680.6300	680.63	5.43E−08	7049.248	7049.248	6.04E−07
NDE	680.630057	680.630057	0.00E+00	7049.248020	7049.248020	3.41E−09
DE/HMO/1	680.630	680.630	0.00E+00	7049.24802	7049.39953	1.95E−01
M-ABC	680.632	680.647	1.55E−02	7051.706	7233.109	1.10E+02
CVI-PSO	680.6354008	680.7557052	7.92E−02	7049.276586	7053.214311	1.06E+01
TLBO	680.63	680.63	0.00E+00	7052.488	7143.45	1.13E+02
Algorithm	G11/0.7499000000			G12/−1.0000000000		
	Best	Mean	Std	Best	Mean	Std
CACDE	0.7499	0.749914912	7.46E−05	−1	−1	0.00E+00
rank-iMDDE	0.7499	0.7499	0.00E+00	−1	−1	0.00E+00
CCiALF	0.7498959	0.7498984	2.05E−16	−1	−1	0.00E+00
NDE	0.749999	0.749999	0.00E+00	−1	−1	0.00E+00
DE/HMO/1	0.7499	0.7499	0.00E+00	−1.0000	−1.0000	0.00E+00
M-ABC	0.75	0.75	2.30E−05	−1.0000	−1.0000	0.00E+00
CVI-PSO	0.7500000	0.7500000	0.00E+00	−1.0000	−1.0000	0.00E+00
TLBO	0.7499	0.74998	7.06E−05	−1	−1	0.00E+00

(*continued on next page*)

Table 9 (*continued*)

Algorithm	G13/0.0539415140			G14/−47.7648884595		
	Best	Mean	Std	Best	Mean	Std
CACDE	0.053941514	0.053941514	3.98E−12	−47.76488846	−47.76488846	2.24E−14
rank-iMDDE	0.053941514	0.053941514	0.00E+00	−47.7648885	−47.76488846	0.00E+00
CCiALF	0.05394151	0.05394261	4.03E−06	−47.7649	−47.7649	4.04E−08
NDE	0.0539415	0.0539415	0.00E+00	−47.7648885	−47.7648885	0.00E+00
DE/HMO/1	0.053942	0.053942	2.67E−12	−47.764888	−47.759325	4.49E−03
M-ABC	0.053985	0.158552	1.73E−01	−47.641	−47.271	2.46E−01
CVI-PSO	0.0555558210	0.065590744	1.02E−02	−47.45301143	−44.4246904	1.41E+00
TLBO	0.13314	0.83851	2.26E−01	−47.639	−43.805	2.32E+00
Algorithm	G15/961.7150222899			G16/−1.9051552586		
	Best	Mean	Std	Best	Mean	Std
CACDE	961.7150223	961.7150223	3.42E−11	−1.905155259	−1.905155259	4.53E−16
rank-iMDDE	961.7150223	961.7150223	0.00E+00	−1.90515526	−1.905155259	0.00E+00
CCiALF	961.7150	961.715	1.86E−08	−1.905155	−1.905155	9.77E−09
NDE	961.7150223	961.7150223	0.00E+00	−1.90515525	−1.90515525	0.00E+00
DE/HMO/1	961.71502	961.71502	0.00E+00	−1.905155	−1.905155	0.00E+00
M-ABC	961.715	961.719	1.42E−02	−1.905	−1.905	4.52E−16
CVI-PSO	961.71570715	961.7185955	6.87E−04	−1.9051550	−1.9051550	8.52E−15
TLBO	961.715	962.044	4.39E−01	−1.905155	−1.905155	0.00E+00
Algorithm	G17/8853.5338748065			G18/−0.8660254038		
	Best	Mean	Std	Best	Mean	Std
CACDE	8853.533875	8853.533965	3.14E−04	−0.866025404	−0.866025404	4.53E−17
rank-iMDDE	8853.539675	8853.539675	0.00E+00	−0.8660254	−0.866025404	0.00E+00
CCiALF	8857.447	8916.856	3.64E+01	−0.8660255	−0.8660255	3.58E−07
NDE	8853.533874	8853.533874	0.00E+00	−0.8660254	−0.8660254	0.00E+00
DE/HMO/1	8853.5397	8924.54464	1.51E+01	−0.866025	−0.866025	2.10E−13
M-ABC	8866.618	8987.459	9.57E+01	−0.866006	−0.7950187	9.39E−02
CVI-PSO	8853.539891	8853.539891	3.70E−12	−0.8646313	−0.809109259	6.27E−02
TLBO	8853.81	8895.7544	5.14E+01	−0.866025	−0.865755	5.09E−04
Algorithm	G19/32.6555929502			G21/193.7245100700		
	Best	Mean	Std	Best	Mean	Std
CACDE	32.65559295	32.65559295	5.79E−10	193.7245101	193.7245101	3.31E−11
rank-iMDDE	32.65559313	32.65561099	8.30E−05	193.7245101	193.7245101	0.00E+00
CCiALF	32.65561	32.66077	2.35E−04	193.7243	193.7352	1.20E−02
NDE	32.65559377	32.65562603	3.73E−05	193.7245101	193.7245101	6.26E−11
DE/HMO/1	32.655593	33.225925	3.01E−01	193.72451	205.23500	1.31E+01
M-ABC	33.285	34.267	6.31E−01	266.500	306.609	1.98E+01
CVI-PSO	32.82702709	35.06733711	2.286728	193.7869252	193.7869352	3.38E−05
TLBO	33.294	33.3699	7.87E−02	194.231	206.118	2.99E+01
Algorithm	G23/−400.0551000000			G24/−5.5080132716		
	Best	Mean	Std	Best	Mean	Std
CACDE	−400.0551	−399.992164	3.15E−01	−5.508013272	−5.508013272	9.06E−16
rank-iMDDE	−400.0551	−398.1808652	4.51E+00	−5.50801327	−5.508013272	0.00E+00
CCiALF	−400.0551	−400.0536	5.00E−03	−5.508013	−5.508013	1.30E−08
NDE	−400.0551	−400.0551	0.00E+00	−5.50801327	−5.50801327	0.00E+00
DE/HMO/1	−400.0551	−398.377245	2.82E+00	−5.508013	−5.508013	0.00E+00
M-ABC	−400.0551	−398.377245	2.82E+00	−5.508013	−5.508013	0.00E+00
CVI-PSO	−400.0000	−400.000000	0.00E+00	−5.50801327	−5.508013272	9.46E−15
TLBO	−387.716	−352.263	2.33E+01	−5.508013	−5.508013	0.00E+00

Table 10
Results obtained by the multiple-problem Wilcoxons test based on Mean values for CACDE on 22 instances.

VS	R^+	R^-	P-value
CACDE vs rank-iMDDE	145.5	85.5	≈
CACDE vs CCiALF	131.5	121.5	≈
CACDE vs NDE	76	155	≈
CACDE vs DE/HMO/1	177.5	75.5	≈
CACDE vs M-ABC	234.5	18.5	+
CACDE vs CVI-PSO	210	43	+
CACDE vs TLBO	219.5	33.5	+

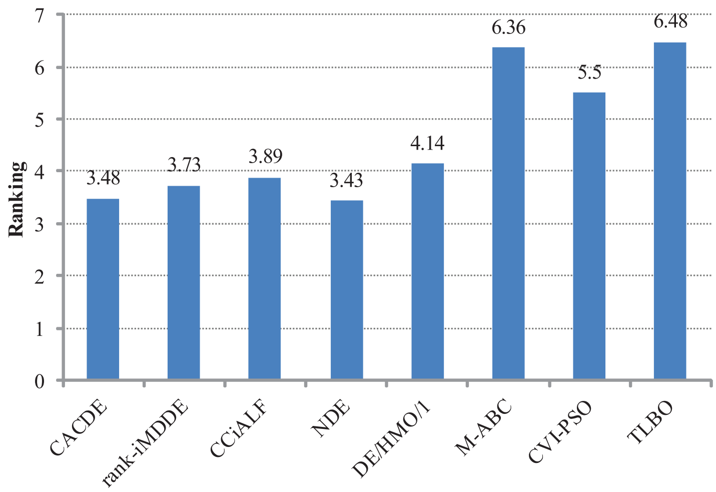

Fig. 6. Ranking of different COEAs by Friedman test for CEC 2006.

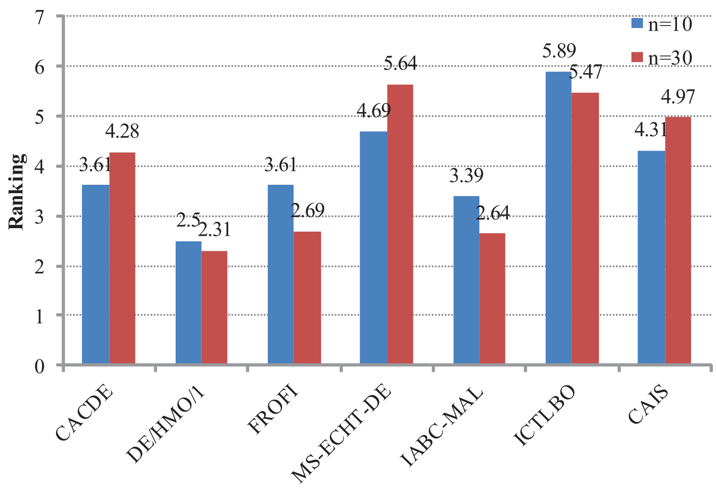

Fig. 7. Ranking of different COEAs by Friedman test for CEC 2010.

4.5. Comparisons with COEAs on CEC2010 problems

To further evaluate the performance of our proposed approach, we compare CACDE with 6 recent COEAs from the literature on 18 more challenging benchmark problems from CEC2010, which have dimensions of both 10 and 30. Among all these COEAs, the first 3 are DE-based COEAs (i.e., DE/HMO/1 [8], FROFI [43] and MS-ECHT-DE [44]), while the remaining 3 are non-DE-based approaches (i.e., IABC-MAL [21], ICTLBO [48] and CAIS [50]). Tables 11 and 12 summarize the comparison results in terms of 'Mean' and 'Std'. The statistical results of the 6 competing approaches are taken directly from the corresponding references.

In the case of $n = 10$, CACDE is ranked first for 8 problems (i.e., C01, C03, C04, C06, C07, C14, C17 and C18). In the case of $n = 30$, CACDE is ranked first for 3 problems (i.e., C01, C12 and C16) and second for 2 problems (i.e., C06 and C11).

Based on the 'Mean' values in Tables 11 and 12, the average rankings of all approaches according to the Friedman test are shown in Fig. 7. CACDE ranks third for the problems where $n = 10$ and fourth on those for which $n = 30$.

The results show that CACDE performs slightly worse than some of the compared approaches on the CEC2010 problems (e.g., DE/HMO/1 and FROFRI). According to the "No Free Lunch Theorems" [45], no algorithm performs well for all classes of problems. Based on this assertion, it can be deduced from the comparison in Section 4 that the CACDE approach is appropriate for solving problems with quadratic, nonlinear and polynomial objectives, such as those from CEC2006. However, the current version of CACDE seems inadequate for solving the more challenging benchmark problems, such as those with nonseparable/separable objective functions and constraints from CEC2010.

Table 11
Experimental comparison between CACDE with other COEAs from CEC2010 at $n = 10$.

Prob.	Criteria	CACDE	DE/HMO/1[8]	FROFI[43]	MS-ECHT-DE[44]	IABC-MAL[21]	ICTLBO[48]	CAIS[50]
C01	Mean	−7.47E−01	−7.47E−01	−7.47E−01	−7.45E−01	−7.47E−01	−7.44E−01	−7.47E−01
	Std	1.88E−03	1.35E−03	1.35E−03	6.20E−03	1.38E−03	5.45E−03	1.30E−03
C02	Mean	−2.26E+00	−2.28E+00	−2.02E+00	−2.27E+00	−2.09E+00	−1.72E+00	−2.27E+00
	Std	6.57E−02	1.15E−06	1.41E−01	1.54E−02	2.27E−01	8.14E−01	2.00E−03
C03	Mean	0.00E+00	0.00E+00	0.00E+00	1.07E+00	0.00E+00	1.41E+09	3.75E−09
	Std	0.00E+00	0.00E+00	0.00E+00	2.94E+00	0.00E+00	6.01E+09	4.81E−04
C04	Mean	−1.00E−05	−1.00E−05	−1.00E−05	−1.00E−05	−1.00E−05	−1.00E−05	−9.97E−06
	Std	0.00E+00	3.32E−16	0.00E+00	2.46E−16	0.00E+00	1.73E−21	4.28E−03
C05	Mean	−4.84E+02	−4.84E+02	−4.84E+02	−4.79E+02	−4.84E+02	−6.83E+01	−4.80E+02
	Std	3.48E−13	0.00E+00	8.09E−07	2.22E+01	3.06E+01	3.36E+01	6.30E+00
C06	Mean	−5.79E+02	−5.79E+02	−5.79E+02	−5.79E+02	−5.79E+02	−5.46E+02	−5.80E+02
	Std	1.68E−02	6.00E−06	5.04E−04	2.30E−03	3.35E−02	3.30E+01	7.30E−08
C07	Mean	0.00E+00	0.00E+00	0.00E+00	4.78E−01	0.00E+00	3.80E−24	1.17E−08
	Std	0.00E+00	0.00E+00	0.00E+00	1.32E+00	0.00E+00	1.50E−23	2.70E+00
C08	Mean	7.01E+00	0.00E+00	7.11E+00	5.89E+00	1.06E+01	1.72E+01	4.09E+00
	Std	5.01E+00	0.00E+00	4.79E+00	5.68E+00	4.66E+00	2.64E+01	1.46E+00
C09	Mean	2.10E+01	1.49E−20	2.50E+01	0.00E+00	4.56E+01	3.56E+05	2.70E+01
	Std	3.51E+01	6.83E−20	3.92E+01	0.00E+00	5.21E+01	1.00E+06	7.50E+01
C10	Mean	6.59E+01	1.53E−27	4.17E+01	7.70E−01	4.17E+01	1.32E+06	1.62E+03
	Std	4.40E+01	5.29E−27	8.69E−06	1.80E+00	2.10E−03	6.59E+06	5.00E+02
C11	Mean	−1.52E−03	−1.52E−03	−1.52E−03	−7.02E−02	−1.52E−03	−1.52E−03	−9.20E−04
	Std	1.30E−06	1.25E−10	5.63E−14	3.50E−02	3.81E−06	4.83E−14	8.23E−04
C12	Mean	−4.34E+02	−1.25E+02	−3.84E+02	−1.02E+02	−5.70E+02	−5.01E+01	−4.36E+02
	Std	2.49E+02	1.65E+02	2.17E+02	1.55E+02	3.13E+02	1.41E+02	6.02E+01
C13	Mean	−6.72E+01	−6.84E+01	−6.84E+01	−6.50E+01	−6.84E+01	−6.84E+01	−6.79E+01
	Std	1.04E+00	6.60E−07	2.52E−09	2.97E+00	1.68E−01	4.42E−14	3.11E−01
C14	Mean	0.00E+00	1.96E−28	0.00E+00	1.21E+03	0.00E+00	3.19E−01	1.22E−04
	Std	0.00E+00	9.78E−28	0.00E+00	6.01E+03	0.00E+00	1.10E+00	2.90E−08
C15	Mean	3.38E+00	1.29E−23	3.09E+00	8.86E+01	3.67E+00	3.73E+01	5.19E−09
	Std	1.02E+00	6.46E−23	1.37E+00	3.53E+02	2.35E+00	9.47E+01	1.10E−08
C16	Mean	4.52E-02	0.00E+00	1.19E−02	5.02E−02	0.00E+00	4.27E−12	9.96E−18
	Std	1.03E-01	0.00E+00	2.07E−02	4.79E−02	0.00E+00	2.13E−11	6.27E−15
C17	Mean	1.23E-33	6.90E−33	7.83E−02	1.34E−01	9.33E−22	7.68E+00	2.93E+00
	Std	2.52E-33	5.99E−33	2.25E−01	3.60E−01	4.10E−22	3.72E+01	2.29E+00
C18	Mean	0.00E+00	0.00E+00	5.23E−26	8.14E−33	1.73E−25	2.59E+00	1.66E+00
	Std	0.00E+00	0.00E+00	1.71E−25	1.91E−32	6.16E−25	1.30E+01	1.27E+00

4.6. Application to mechanical engineering design problems

The experimental results obtained using artificial problems verify that CACDE is very competitive. To further reveal the applicability of our CACDE to real-world cases, in this section, we tested its performance on five widely used constrained mechanical design optimization problems [11,20,31]: (1) welded beam design; (2) tension/compression spring; (3) speed reducer design; (4) three-bar truss design and (5) pressure vessel design.

We first applied CACDE 30 times independently to each problem and then compared the final statistical results with those of other competing approaches in terms of 'Best', 'Median', 'Worst', 'Mean' and 'Std'. The selected competing approaches are CDE [13], DELC [41], COMDE [32], rank-iMDDE [12], CCiALF [11], NDE [31], MVDE [26], PSO-DE [20], CVI-PSO [25], CB-ABC [3] and ICTLBO [48]. To ensure a fair comparison, we acquired the statistical results for the competing algorithms directly from the corresponding references.

The comparison results for the welded beam design problem are listed in Table 13, which shows that CACDE consistently obtains optimal results. It obtains better results than those of CDE in terms of 'Best', those of CDE and CVI-PSO in terms of 'Mean', and those of CDE, CVI-PSO and CCiALF in terms of 'Worst'. CACDE clearly finds optimal results in approximately 10,000 FEs.

Table 14 lists the comparison results for the tension/compression spring design problem. All the approaches except CDE found optimal results in terms of 'Best'. Moreover, in terms of 'Worst', CACDE obtained better results than those of CDE, rank-iMDDE, COMDE and CVI-PSO and worse results than those of DELC, rank-iMDDE and CCiALF.

The comparison results for the speed reducer design problem are listed in Table 15, where it can be observed that all the approaches except PSO-DE achieved similar statistical results.

The comparison results for the three-bar truss design problem are listed in Table 16. The table shows that all the approaches obtained similar statistical results. CACDE performs slightly worse than the other approaches in terms of 'Mean' and 'Worst'.

Table 17 lists the comparison results for the pressure vessel design problem. On this problem, CACDE obtains better results than those of CDE in terms of 'Worst' and better results than those of CVI-PSO in terms of 'Mean' and 'Worst'.

Table 12
Experimental comparison between CACDE with other COEAs from CEC2010 at $n = 30$.

Prob.	Criteria	CACDE	DE/HMO/1[8]	FROFI[43]	MS-ECHT-DE[44]	IABC-MAL[21]	ICTLBO[48]	CAIS[50]
C01	Mean	−8.20E−01	−8.05E−01	−8.21E−01	−6.98E−01	−8.20E−01	−8.18E−01	−8.20E−01
	Std	2.67E−03	9.22E−03	2.36E−03	7.37E−02	4.20E−03	2.97E−03	3.25E−04
C02	Mean	−2.01E+00	−2.28E+00	−2.00E+00	−1.61E+00	−1.96E−01	−3.83E−01	−2.21E+00
	Std	7.78E−02	1.74E−03	4.35E−02	4.91E−01	2.88E−01	1.72E+00	2.84E−03
C03	Mean	3.08E+01	1.25E−09	2.87E+01	3.48E+01	2.87E+01	2.13E+11	6.68E+01
	Std	3.50E+01	1.25E−09	6.24E−08	3.06E+01	5.66E−04	3.35E+11	4.26E+02
C04	Mean	3.54E+00	−3.28E−06	−3.33E−06	8.01E−02	−3.33E−06	1.59E−01	1.98E−03
	Std	7.62E+00	6.98E−08	4.13E−10	2.76E−01	6.10E−07	3.24E−01	1.61E−03
C05	Mean	−3.41E+02	−4.84E+02	−4.81E+02	−1.46E+02	−4.82E+02	−5.98E+01	−4.36E+02
	Std	8.69E+01	5.08E−04	2.84E+00	4.87E+01	3.30E+00	8.86E+00	2.51E+01
C06	Mean	−5.22E+02	−5.31E+02	−5.29E+02	−1.34E+02	−5.29E+02	−4.64E+02	−4.54E+02
	Std	2.92E+00	2.94E−04	5.71E−01	5.53E+01	6.22E−01	9.19E+01	4.79E+01
C07	Mean	9.57E−01	1.25E−09	0.00E+00	6.38E−01	0.00E+00	2.88E+01	1.07E+00
	Std	1.74E+00	1.25E−09	0.00E+00	1.49E+00	0.00E+00	4.68E+01	1.61E+00
C08	Mean	9.76E+00	2.54E−10	0.00E+00	1.89E+02	0.00E+00	1.01E+02	1.65E+00
	Std	3.20E+01	3.05E−10	0.00E+00	4.27E+02	0.00E+00	1.22E+02	6.41E−01
C09	Mean	9.23E+03	7.24E−10	4.30E+01	5.80E+02	7.41E−10	1.01E+07	1.57E+00
	Std	1.26E+04	8.99E−10	3.27E+01	1.87E+03	4.35E−10	3.52E+07	1.96E+00
C10	Mean	8.20E+10	3.60E−11	3.13E+01	1.96E+03	3.13E+01	6.01E+09	1.78E+01
	Std	3.91E+11	5.04E−11	3.27E+01	4.29E+03	3.46E+01	2.31E+10	1.88E+01
C11	Mean	2.99E−03	−3.92E−04	−3.92E−04	6.47E−03	−3.92E−04	−3.65E−04	−1.58E−04
	Std	7.14E−03	6.67E−07	2.64E−06	9.34E−03	2.98E−04	4.98E−05	4.67E−05
C12	Mean	−1.99E−01	−1.99E−01	−1.99E−01	−1.99E−01	−1.99E−01	−1.99E−01	4.29E−06
	Std	2.35E−04	2.14E−07	1.42E−06	4.02E−04	1.30E−02	6.15E−05	4.52E−04
C13	Mean	−6.77E+01	−6.78E+01	−6.83E+01	−6.03E+01	−6.84E+01	−6.81E+01	−6.62E+01
	Std	6.88E−01	3.91E−01	1.95E−01	2.36E+00	1.10E−01	7.78E−01	2.27E−01
C14	Mean	7.37E−26	2.75E−09	9.80E−29	1.99E+03	0.00E+00	8.02E+00	8.68E−07
	Std	1.79E−25	3.17E−09	4.90E−28	7.03E+03	0.00E+00	8.69E+00	3.14E−07
C15	Mean	2.17E+01	4.48E−10	2.16E+01	5.37E+01	2.16E+01	2.91E+01	3.41E+01
	Std	2.45E−01	1.41E−09	8.03E−05	1.19E+02	5.27E−03	3.63E+01	3.82E+01
C16	Mean	6.03E−04	0.00E+00	0.00E+00	7.68E−03	0.00E+00	0.00E+00	8.21E−02
	Std	3.02E−03	0.00E+00	0.00E+00	2.57E−02	0.00E+00	0.00E+00	1.12E−01
C17	Mean	8.24E−01	7.25E−11	1.59E−01	4.40E−01	1.20E+00	3.29E+01	3.61E+00
	Std	6.85E−01	7.50E−11	3.82E−01	3.28E−01	1.55E+00	1.35E+02	2.54E+00
C18	Mean	2.35E−05	3.99E−09	4.87E−01	1.00E−01	1.58E−18	8.82E−04	4.02E+01
	Std	8.46E−05	5.76E−09	1.25E+00	4.50E−01	2.21E−18	3.22E−03	1.80E+01

Table 13
Result of welded beam problem.

Algorithm	Best	Median	Mean	Worst	SD	FES
CACDE	1.72485232	1.72485237	1.72485255	1.72485235	6.12E−08	10,000
CDE[13]	1.733461	NA	1.768158	1.824105	2.20E−02	240,000
DELC[41]	1.724852	1.724852	1.724852	1.724852	4.10E−13	20,000
COMDE[32]	1.724852	1.724852	1.724852	1.724852	1.20E−12	20,000
rank-iMDDE [12]	1.724852309	NA	1.724852309	1.724852309	7.71E−11	15,000
CCiALF[11]	1.724852	NA	1.724852	1.724854	5.11E−07	10,000
NDE[31]	1.724852309	1.724852309	1.724852309	1.724852309	3.73E−12	8000
MVDE[26]	1.7248527	NA	1.7248621	1.7249215	7.88E−06	15,000
PSO-DE[20]	1.724852309	NA	1.724852309	1.724852309	6.70E−16	66,600
CVI-PSO[25]	1.724852	NA	1.725124	1.727665	6.12E−04	25,000
CB-ABC[3]	1.724852	NA	1.724852	NA	2.38E−11	15,000
ICTLBO[48]	1.724852309	NA	1.724852309	1.724852309	1.75E−11	15,000

The aforementioned comparison results indicate that the proposed CACDE is a potential COEA for solving constrained mechanical design problems.

5. Further discussions

The CACDE algorithm involves two main mechanisms: adaptive DE and the cluster-replacement-based feasibility rule). To investigate whether these mechanisms definitely improve the search capability of DE for COPs, this section describes some extra experiments conducted to demonstrate the effectiveness and rationality of the proposed CACDE mechanisms.

Table 14
Result of spring design problem.

Algorithm	Best	Median	Mean	Worst	SD	FES
CACDE	0.01266523	0.01266523	0.01266524	0.01266523	7.45E−10	24,000
CDE[13]	0.0126702	NA	0.012703	0.01279	2.70E−05	240,000
DELC[41]	0.012665233	0.012665233	0.012665267	0.012665575	1.30E−07	20,000
COMDE[32]	0.012665233	0.012665423	0.012667168	0.012676809	3.09E−06	24,000
rank-iMDDE [12]	0.012665233	NA	0.012665297	0.01266743	8.48E−07	10,000
CCiALF[11]	0.012665233	NA	0.012665251	0.012665233	9.87E−08	5000
NDE[31]	0.012665232	0.012665423	0.012668899	0.012687092	5.38E−06	24,000
MVDE[26]	0.012665272	NA	0.012667324	0.012719055	2.45E−06	10,000
PSO-DE[20]	0.012665233	NA	0.012665233	0.012665233	4.90E−12	42,100
CVI-PSO[25]	0.0126655	NA	0.012731	0.0128426	5.58E−05	25,000
CB-ABC[3]	0.012665	NA	0.012671	NA	1.42E−05	15,000

Table 15
Result of speed reducer design problem.

Algorithm	Best	Median	Mean	Worst	SD	FES
CACDE	2994.471303	2994.471644	2994.472199	2994.471604	2.33E−04	18,000
DELC[41]	2994.471066	2994.471066	2994.471066	2994.471066	1.90E−12	30,000
COMDE[32]	2994.471066	2994.471066	2994.471066	2994.471066	1.54E−12	21,000
rank-iMDDE [12]	2994.471066	NA	2994.471066	2994.471066	7.93E−13	19,920
CCiALF[11]	2994.471066	NA	2994.471066	2994.471066	2.31E−12	10,000
NDE[31]	2994.4710661	2994.4710661	2994.4710661	2994.4710661	4.17E−12	18,000
MVDE[26]	2994.471066	NA	2994.471066	2994.471069	2.82E−07	30,000
PSO-DE[20]	2996.348165	NA	2996.348165	2996.348166	1.00E−07	70,100
CB-ABC[3]	2994.471066	NA	2994.471066	NA	2.48E−07	15,000
ICTLBO[48]	2994.471066	NA	2994.471066	2994.471066	4.64E−13	19,920

Table 16
Result of three-bar truss design problem.

Algorithm	Best	Median	Mean	Worst	SD	FES
CACDE	263.8958442	263.895873	263.8959624	263.8958683	2.64E−05	4000
DELC[41]	263.8958434	263.8958434	263.8958434	263.8958434	4.34E−14	10,000
COMDE[32]	263.8958434	263.8958434	263.8958434	263.8958434	5.34E−13	7000
rank-iMDDE [12]	263.8958434	NA	263.8958434	263.8958434	0.00E+00	4920
CCiALF[11]	263.89584337	NA	263.89584337	263.89584337	4.23E−14	5000
NDE[31]	263.8958434	263.8958434	263.8958434	263.8958434	0.00E+00	4000
MVDE[26]	263.8958434	NA	263.8958434	263.8958548	2.58E−07	7000
PSO-DE[20]	263.89584338	NA	263.89584338	263.89584338	1.20E−10	17,600

Table 17
Result of pressure vessel design problem.

Algorithm	Best	Median	Mean	Worst	SD	FES
CACDE	6059.714335	6059.714335	6059.714335	6059.714335	4.11E−10	20,000
CDE[13]	6059.7340	NA	6085.2303	6371.0455	4.30E+01	240,000
DELC[41]	6059.7143	6059.7143	6059.7143	6059.7143	2.10E−11	30,000
COMDE[32]	6059.714335	6059.714335	6059.714335	6059.714335	3.62E−10	30,000
rank-iMDDE [12]	6059.714335	NA	6059.714335	6059.714335	7.57E−07	15,000
CCiALF[11]	6059.714335	NA	6059.714335	6059.714335	1.01E−11	12,000
NDE[31]	6059.714335	6059.714335	6059.714335	6059.714335	4.56E−07	20,000
MVDE[26]	6059.714387	NA	6059.997236	6090.533528	2.91E+00	15,000
PSO-DE[20]	6059.714335	NA	6059.714335	6059.714335	1.00E−10	42,100
CVI-PSO[25]	6059.7143	NA	6292.1231	6820.4101	2.88E+02	25,000
CB-ABC[3]	6059.714335	NA	6126.623676	NA	1.14E+02	15,000
ICTLBO[48]	6059.714335	NA	6059.714335	6059.714335	9.28E−13	15,000

5.1. Analysis of using multiple mutation strategies

In our CACDE, the Mpool contains three mutation strategies. To address whether CACDE still works well using only a single strategy, we tested 3 modified versions of CACDE that use only a single mutation strategy as follows:

(1) CACDE-01: DE/current-to-best/1 is used;
(2) CACDE-02: DE/current-to-rand/1 is used;
(3) CACDE-03: DE/rand/2 is used.

Table 18
Experimental comparison between single and multiple strategies based CACDE on 22 instances.

Prob.	Mean ± Std (Success rate) [Feasible rate]			
	CACDE	CACDE-01	CACDE-02	CACDE-03
G02	−0.8036 ± 1.28E−08 (100%) [100%]	−0.8009 ± 5.17E−03 (76%) [100%]	−0.7701 ± 2.36E−02 (0%) [100%]	−0.7938 ± 2.80E−03 (0%) [100%]
G03	−1.0005 ± 4.00E−16 (100%) [100%]	−1.0005 ± 2.15E−07 (100%) [100%]	−1.0005 ± 7.14E−16 (100%) [100%]	−0.9543 ± 1.89E−02 (0%) [100%]
G06	−6961.8139 ± 0.00E+00 (100%) [100%]	−6961.8139 ± 0.00E+00 (100%) [100%]	−6961.8139 ± 0.00E+00 (100%) [100%]	−6961.4408 ± 1.36E−01 (0%) [100%]
G07	24.3062 ± 5.33E−15 (100%) [100%]	24.3107 ± 1.41E−03 (0%) [100%]	24.3102 ± 6.03E−03 (12%) [100%]	24.3469 ± 1.85E−02 (0%) [100%]
G09	680.6301 ± 3.36E−13 (100%) [100%]	680.6305 ± 1.30E−04 (0%) [100%]	680.6301 ± 2.29E−13 (100%) [100%]	680.6346 ± 1.84E−03 (0%) [100%]
G10	7049.2480 ± 3.76E−12 (100%) [100%]	7355.4358 ± 7.87E+01 (0%) [100%]	7062.7465 ± 1.00E+01 (0%) [100%]	7714.4498 ± 1.66E+02 (0%) [100%]
G11	0.7499 ± 1.60E−11 (100%) [100%]	0.7499 ± 1.01E−05 (100%) [100%]	0.7499 ± 1.13E−16 (100%) [100%]	0.7501 ± 1.25E−04 (40%) [100%]
G13	0.0539 ± 1.37E−12 (100%) [100%]	0.0693 ± 7.70E−02 (96%) [100%]	0.0539 ± 2.89E−17 (100%) [100%]	0.6309 ± 5.21E−01 (0%) [0%]
G14	−47.7649 ± 2.55E−14 (100%) [100%]	−47.7649 ± 7.70E−09 (100%) [100%]	−48.6368 ± 5.27E+00 (0%) [0%]	−47.7649 ± 1.29E−08 (100%) [100%]
G17	8853.5339 ± 0.00E+00 (100%) [100%]	8853.5339 ± 1.07E−11 (100%) [100%]	8868.7614 ± 3.03E+01 (72%) [100%]	8856.4962 ± 1.48E+01 (96%) [100%]
G18	−0.8660 ± 2.27E−17 (100%) [100%]	−0.8554 ± 4.34E−02 (0%) [100%]	−0.8660 ± 0.00E+00 (100%) [100%]	−0.8647 ± 1.20E−03 (16%) [100%]
G21	193.7245 ± 2.03E−11 (100%) [100%]	266.0986 ± 6.80E+01 (0%) [0%]	214.4338 ± 4.01E+01 (76%) [100%]	373.3301 ± 1.62E+02 (0%) [0%]
G23	−400.0551 ± 1.47E−13 (100%) [100%]	−388.0548 ± 6.00E+01 (96%) [100%]	−314.8094 ± 2.21E+02 (0%) [0%]	−400.0549 ± 1.15E−04 (32%) [100%]

Table 19
Experimentalcomparison between feasibility rule and cluster based feasibility rule on 22 instances.

Prob.	Mean ± Std (Success Rate) [Feasible Rate]		Prob.	Mean ± Std (Success Rate) [Feasible Rate]	
	CACDE	CACDE-04		CACDE	CACDE-04
G02	−0.8036 ± 0.00E+00 (100%) [100%]	−0.7990 ± 6.38E−03 (56%) [100%]	G13	0.0539 ± 0.00E+00 (100%) [100%]	0.3464 ± 4.07E−01 (40%) [100%]
G17	8853.5339 ± 0.00E+00 (100%) [100%]	8859.4868 ± 2.06E+01 (92%) [100%]	G21	193.7245 ± 0.00E+00 (100%) [100%]	198.9636 ± 2.62E+01 (96%) [100%]

For each CACDE variant, 25 independent runs were executed on 22 instances from CEC2006, and the final statistical results are summarized in Table 18. It can be observed from the table that CACDE performs better than CACDE-01, CACDE-02 and CACDE-03 on most instances. More specifically, CACDE obtained slightly better results on 9, 8 and 13 instances compared to CACDE-01, CACDE-02 and CACDE-03, respectively. These results indicate that the full CACDE using multiple strategies works better than variants that use only a single strategy.

5.2. Analysis of adaptive mechanism for updating selection probabilities

In CACDE, the selection probability for each element in every candidate pool is adaptively adjusted based on the previous search information. To better understand the evolution process, Figs. 8–9 depict the selection probabilities of an offspring being produced by a particular mutation strategy, scale factor and crossover rate versus the fitness evaluation number on instances G02 and G21. As Fig. 8 shows, the DE/current-to-best/1 and DE/rand/2 strategies have an advantage over the DE/current-to-rand/1 strategy during the first 0.6×10^5 FEs; subsequently, however, their selection probabilities decrease gradually. This result means that, for G02, the most suitable mutations in the initial stage are DE/current-to-best/1 and DE/rand/2 and that DE/current-to-rand/1 becomes the most appropriate strategy later in the process. Regarding the scale factor, $F = 0.4$ has an advantage over the other values. Regarding CR, initially, a value of 0.2 is used more frequently; later, however, there are no remarkable differences among all 5 values. When G21 is solved, DE/current-to-rand/1 has the best ranking, followed by DE/current-to-best/1 and DE/rand/2. An F value of 0.4 is initially remarkably advantageous. Meanwhile, a CR value of 1.0 is remarkably advantageous compared to other values according to the selection probability values.

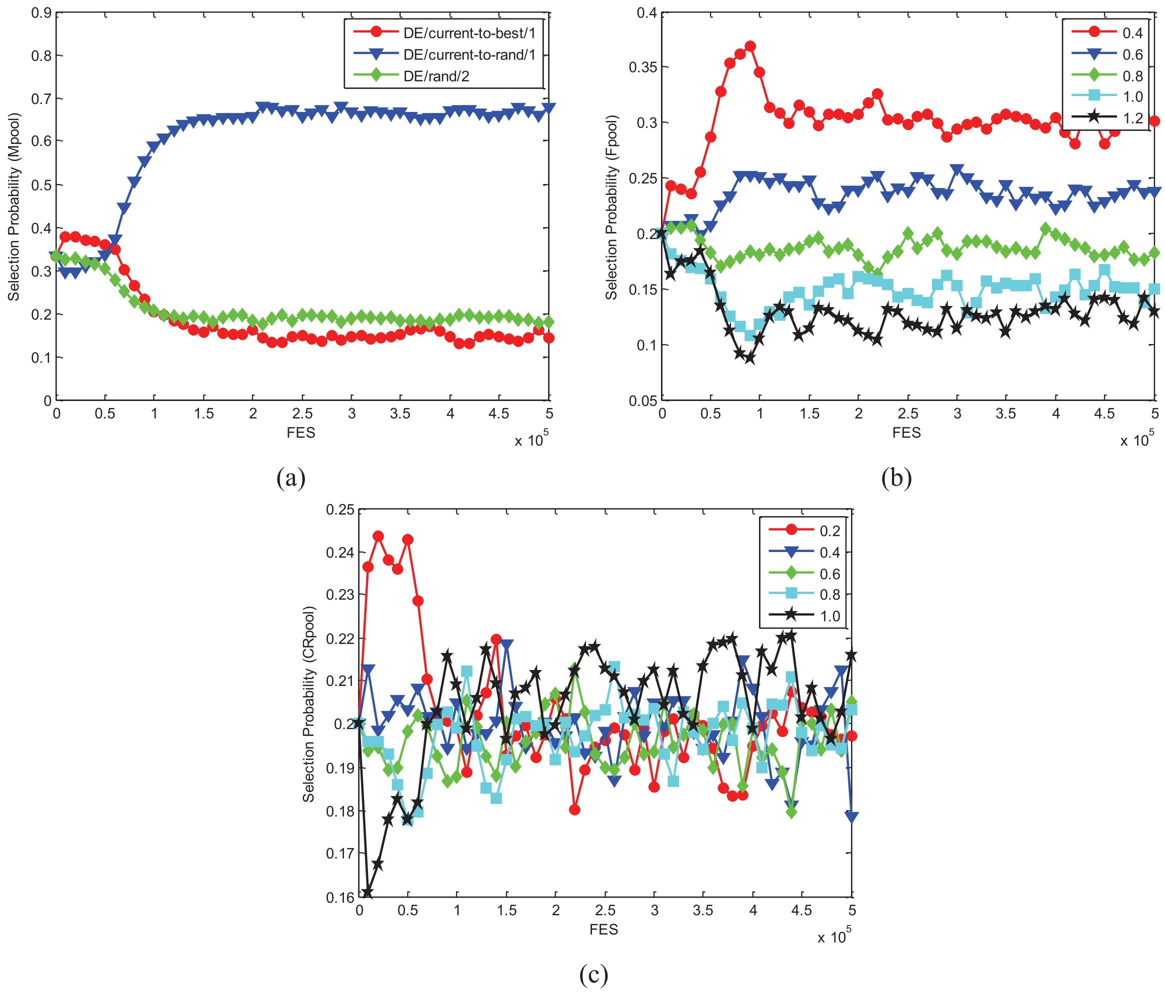

Fig. 8. Selection probability versus fitness evaluation number for G02. (a) mutation strategy; (b) scale factor; (c) crossover rate.

Table 20
Mean objective value, success rate, and feasible rate on test functions G02, G17, G21, and G23 with different η values.

η	Mean ± Std (Success rate) [Feasible rate]			
	G02	G17	G21	G23
0.01	−0.8036 ± 1.28E−08 (100%) [100%]	8853.5339 ± 0.00E+00 (100%) [100%]	193.7245 ± 2.03E−11 (100%) [100%]	−400.0551 ± 1.47E−13 (100%) [100%]
0.02	−0.8032 ± 3.33E−03 (96%) [100%]	8853.5339 ± 0.00E+00 (100%) [100%]	193.7245 ± 1.82E−11 (100%) [100%]	−373.8953 ± 1.31E+02 (96%) [96%]
0.05	−0.8032 ± 2.20E−03 (96%) [100%]	8853.5339 ± 0.00E+00 (100%) [100%]	193.7245 ± 2.32E−11 (100%) [100%]	−400.0454 ± 4.80E−02 (92%) [100%]
0.1	−0.8036 ± 6.91E−08 (100%) [100%]	8859.6701 ± 2.13E+01 (92%) [100%]	204.2028 ± 3.63E+01 (92%) [100%]	−394.5957 ± 1.12E+01 (64%) [100%]
0.2	−0.8036 ± 1.01E−06 (100%) [100%]	8867.5456 ± 3.25E+01 (72%) [100%]	200.7844 ± 2.45E+01 (92%) [100%]	−88.6265 ± 2.66E+02 (12%) [56%]
0.3	−0.8036 ± 2.54E−05 (96%) [100%]	8876.0893 ± 3.89E+01 (12%) [76%]	299.6760 ± 2.62E+02 (44%) [84%]	−52.7816 ± 2.62E+02 (0%) [64%]
0.4	−0.8032 ± 2.12E−03 (92%) [100%]	8891.2578 ± 4.75E+01 (8%) [40%]	383.8610 ± 3.37E+02 (20%) [64%]	−153.4858 ± 5.87E+02 (0%) [48%]
0.5	−0.8032 ± 1.81E−03 (80%) [100%]	8868.4427 ± 7.79E+01 (4%) [4%]	496.6128 ± 4.54E+02 (8%) [36%]	−349.7247 ± 5.95E+02 (0%) [52%]

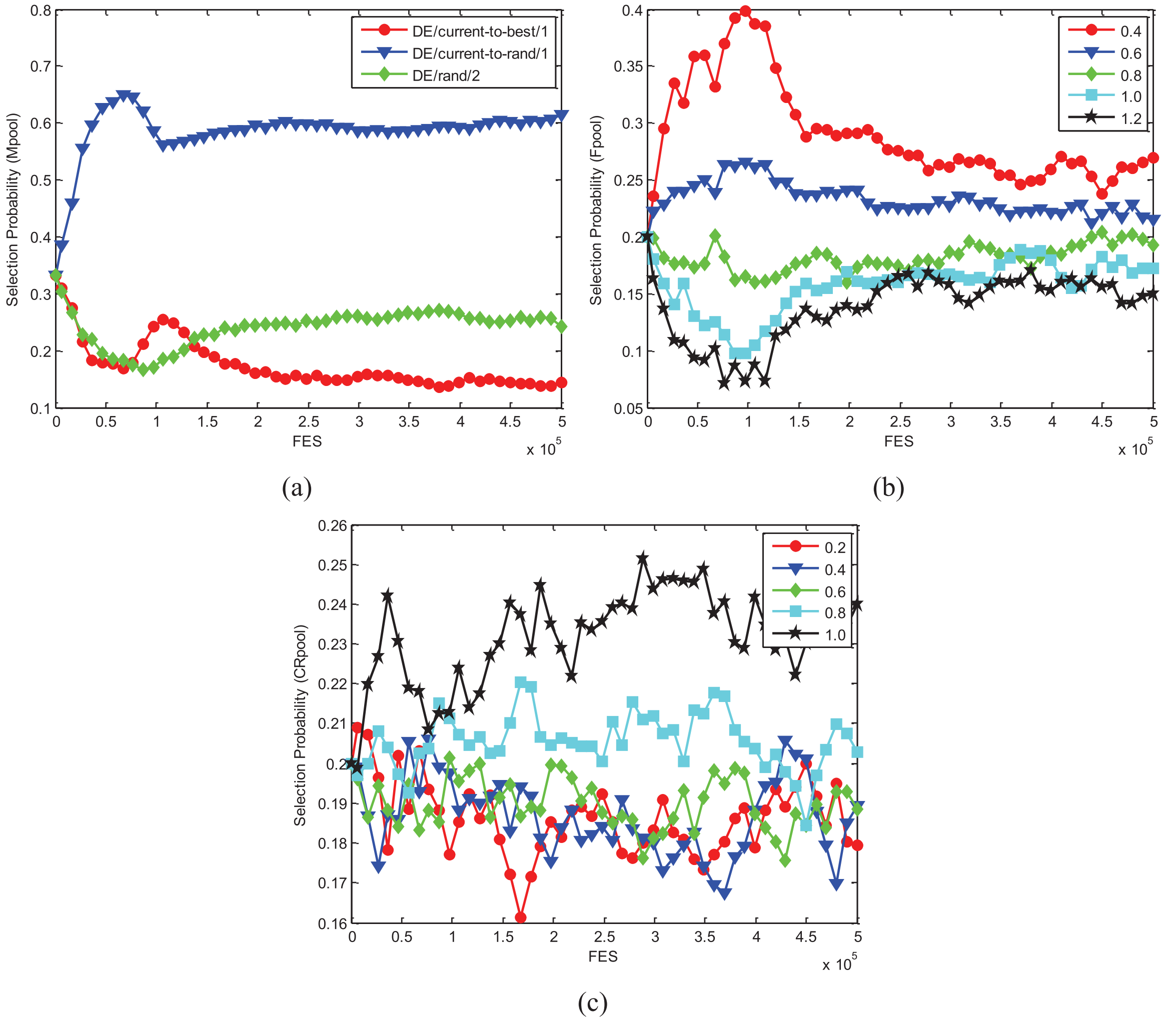

Fig. 9. Selection probability versus fitness evaluation number for G21. (a) mutation strategy; (b) scale factor; (c) crossover rate.

5.3. Analysis of the cluster-replacement-based feasibility rule

To validate the effectiveness of the cluster-replacement-based feasibility rule, our adaptive DE is first combined with the original feasibility rule (CACDE-04) and then compared with CACDE on 22 instances. The results are listed in Table 19 shows that CACDE obtained results similar to CACDE-04 except on 4 instances (i.e., G02, G13, G17 and G21). These 4 instances are relatively difficult for COEAs. Moreover, it can be observed from Table 19 that CACDE achieves slightly better results in terms of the mean, success rate and feasible rate.

5.4. Analysis of parameter η

In our adaptive DE, a learning rate (η) is introduced to control the updating process for the selection probabilities. To analyze its impact on the performance of CACDE, we tested 8 different settings (i.e., 0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.4, and 0.5). Table 20 lists the experimental results for 4 highly constrained test instances (i.e., G02, G17, G21, and G23).

In Table 20, the performance of CACDE degrades as η increases, particularly when η reaches a relatively large value, e.g., $\eta > 0.1$. These results indicate that a small η value works better with CACDE than does a large one. Thus, we choose $\eta = 0.01$ for our CACDE.

6. Conclusions

The performance of a COEA depends heavily on the search engine and CHT used. In this paper, a new constrained optimization evolutionary algorithm that combines adaptive differential evolution with a cluster-replacement-based feasibility rule is proposed for solving constrained problems. To enhance the search capability of DE, a mutation pool (Mpool), a scale factor pool (Fpool) and a crossover rate pool (CRpool) with diverse characteristics are incorporated into DE to act as candidates; then, a selection probability updating mechanism is designed to determine the most proper trial vector generation strategies and the corresponding control parameters for each vector in the main population at different generations. Meanwhile, to alleviate the excessive greediness of the feasibility rule, a cluster-based-replacement operator is proposed that replaces some vectors with an infeasible vector with a low objective. The proposed CACDE algorithm is compared with some state-of-the-art DE-based and non-DE-based approaches on 42 artificial constrained numerical benchmark problems and 5 widely used constrained mechanical design problems. The results show that the proposed CACDE method has a better or comparable performance on selected instances in terms of most performance indicators, such as the error value, feasible rate, success rate and success performance. Finally, the effectiveness and benefits of the new adaptive DE and cluster-replacement-based feasibility rule are experimentally investigated and evaluated through multiple experiments. The numerical results indicate that the proposed adaptive DE and cluster replacement strategy are helpful for enhancing DE's search capability when solving COPs.

In future work, we would like to improve the current version of CACDE for problems with nonseparable/separable objectives and constraints. We would also like to determine how to approximate the global feasible solutions with fewer fitness evaluations using surrogate models.

Acknowledgments

The authors would like to express their sincere thanks to the editors and reviewers for their valuable suggestions and comments. This work was supported by National Natural Science Foundation of China under Grants 61703268 and 51605277, by the Fundamental Research Funds for the Central Universities under Grant 222201717006, and by the Natural Science Foundation of Jiangsu Province under Grant BK20160540.

References

[1] M.M. Ali, W.X. Zhu, A penalty function-based differential evolution algorithm for constrained global optimization, Comput. Optim. Appl. 54 (3) (2013) 707–739.
[2] Y. Ao, H. Chi, An adaptive differential evolution algorithm to solve constrained optimization problems in engineering design, Engineering 2 (1) (2010) 65–77.
[3] I. Brajevic, Crossover-based artificial bee colony algorithm for constrained optimization problems, Neural Comput. Appl. 26 (7) (2015) 1587–1601.
[4] J. Brest, V. Zumer, M.S. Maucec, Self-adaptive differential evolution algorithm in constrained real-parameter optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC, 2006, pp. 215–222.
[5] X. Chen, W. Du, F. Qian, Multi-objective differential evolution with ranking-based mutation operator and its application in chemical process optimization, Chemom. Intell. Lab. Syst. 136 (16) (2014) 85–96.
[6] K. Deb, An efficient constraint handling method for genetic algorithms, Comput. Methods Appl. Mech. Eng. 186 (2–4) (2000) 311–338.
[7] S.M. Elsayed, R.A. Sarker, D.L. Essam, Multi-operator based evolutionary algorithms for solving constrained optimization problems, Comput. Oper. Res. 38 (12) (2011) 1877–1896.
[8] S.M. Elsayed, R.A. Sarker, D.L. Essam, Self-adaptive differential evolution incorporating a heuristic mixing of operators, Comput. Optim. Appl. 54 (3) (2013) 771–790.
[9] Á. Fialho, L. Da Costa, M. Schoenauer, M. Sebag, Analyzing bandit-based adaptive operator selection mechanisms, Ann. Math. Artif. Intell. 60 (1) (2010) 25–64.
[10] W.F. Gao, G.G. Yen, S.Y. Liu, A dual-population differential evolution with coevolution for constrained optimization, IEEE Trans. Cybern. 45 (5) (2015) 1094–1107.
[11] B. Ghasemishabankareh, X. Li, M. Ozlen, Cooperative coevolutionary differential evolution with improved augmented lagrangian to solve constrained optimisation problems, Inf. Sci. 369 (2016) 441–456.
[12] W. Gong, Z. Cai, D. Liang, Engineering optimization by means of an improved constrained differential evolution, Comput. Methods Appl. Mech. Eng. 268 (4) (2014) 884–904.
[13] F.Z. Huang, L. Wang, Q. He, An effective co-evolutionary differential evolution for constrained optimization, Appl. Math. Comput. 186 (1) (2007) 340–356.
[14] G. Iacca, F. Caraffini, F. Neri, Multi-strategy coevolving aging particle optimization, Int. J. Neural Syst. 24 (1) (2014) 1450008.
[15] S. Iliya, F. Neri, Towards artificial speech therapy: a neural system for impaired speech segmentation, Int. J. Neural Syst. 26 (6) (2016) 1650023.
[16] G. Jia, Y. Wang, Z. Cai, Y. Jin, An improved (mu+lambda)-constrained differential evolution for constrained optimization, Inf. Sci. 222 (4) (2013) 302–322.
[17] X. Kong, H. Ouyang, X. Piao, A prediction-based adaptive grouping differential evolution algorithm for constrained numerical optimization, Soft Comput. 17 (12) (2013) 2293–2309.
[18] S. Kukkonen, J. Lampinen, Constrained real-parameter optimization with generalized differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC, 2006, pp. 207–214.
[19] J.J. Liang, T.P. Runarsson, E. Mezura-Montes, M. Clerc, P.N. Suganthan, C.A. Coello Coello, K. Deb, Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization, Technical Report, Nanyang Technological University, Singapore, 2006.
[20] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Appl. Soft Comput. 10 (2) (2010) 629–640.
[21] W. Long, X. Liang, S. Cai, J. Jiao, W. Zhang, An improved artificial bee colony with modified augmented lagrangian for constrained optimization, Soft Comput. (2017) 1–22.
[22] R. Mallipeddi, P.N. Suganthan, Ensemble of constraint handling techniques, IEEE Trans. Evolut. Comput. 14 (4) (2010) 561–579.
[23] R. Mallipeddi, P.N. Suganthan, Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization, Technical Report, Nanyang Technological University, Singapore, 2010.
[24] R. Mallipeddi, P.N. Suganthan, Q.K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2) (2011) 1679–1696.
[25] I. Mazhoud, K. Hadj-Hamou, J. Bigeon, P. Joyeux, Particle swarm optimization for solving engineering problems: a new constraint-handling mechanism, Eng. Appl. Artif. Intell. 26 (4) (2013) 1263–1273.

[26] V.V.D. Melo, G.L.C. Carosio, Investigating multi-view differential evolution for solving constrained engineering design problems, Expert Syst. Appl. 40 (9) (2013) 3370–3377.
[27] E. Mezura-Montes, O. Cetina-Domínguez, Empirical analysis of a modified artificial bee colony for constrained numerical optimization, Appl. Math. Comput. 218 (22) (2012) 10943–10973.
[28] E. Mezura-Montes, C.A. Coello Coello, Constraint-handling in nature-inspired numerical optimization: Past, present and future, Swarm Evolut. Comput. 1 (4) (2011) 173–194.
[29] E. Mezura-Montes, M.E. Miranda-Varela, R.d.C. Gómez-Ramón, Differential evolution in constrained numerical optimization: an empirical study, Inf. Sci. 180 (22) (2010) 4223–4262.
[30] E. Mezura-Montes, J. Velázquez-Reyes, C.A. Coello Coello, Promising infeasibility and multiple offspring incorporated to differential evolution for constrained optimization, in: Proceedings of Genetic and Evolutionary Computation Conference, GECCO, Washington DC, USA, 2005, pp. 225–232.
[31] A.W. Mohamed, A novel differential evolution algorithm for solving constrained engineering optimization problems, J. Intell. Manuf. (2017) 1–34.
[32] A.W. Mohamed, H.Z. Sabry, Constrained optimization based on modified differential evolution algorithm, Inf. Sci. 194 (5) (2012) 171–208.
[33] F. Neri, V. Tirronen, Recent advances in differential evolution: a survey and experimental analysis., Artif. Intell. Rev. 33 (1–2) (2010) 61–106.
[34] I. Poikolainen, F. Neri, F. Caraffini, Cluster-based population initialization for differential evolution frameworks, Inf. Sci. 297 (2015) 216–235.
[35] F. Qian, B. Xu, R. Qi, H. Tianfield, Self-adaptive differential evolution algorithm with α-constrained-domination principle for constrained multi-objective optimization, Soft Comput. 16 (8) (2012) 1353–1372.
[36] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evolut. Comput. 13 (2) (2009) 398–417.
[37] R.V. Rao, V. Patel, An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems, Int. J. Ind. Eng. Comput. 3 (4) (2012) 535–560.
[38] T.P. Runarsson, X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Trans. Evolut. Comput. 4 (3) (2000) 284–294.
[39] R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[40] M.F. Tasgetiren, P.N. Suganthan, A multi-populated differential evolution algorithm for solving constrained optimization problem, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC, 2006, pp. 33–40.
[41] L. Wang, L.P. Li, An effective differential evolution with level comparison for constrained engineering design, Struct. Multidiscip. Optim. 41 (6) (2010) 947–963.
[42] Y. Wang, Z. Cai, Constrained evolutionary optimization by means of (mu+lambda)-differential evolution and improved adaptive trade-off model, Evolut. Comput. 19 (2) (2011) 249–285.
[43] Y. Wang, B.C. Wang, H.X. Li, G.G. Yen, Incorporating objective function information into the feasibility rule for constrained evolutionary optimization., IEEE Trans. Cybern. 46 (12) (2015) 2938–2952.
[44] W.H. Wei, J.H. Wang, M. Tao, Constrained differential evolution with multiobjective sorting mutation operators for constrained optimization, Appl. Soft Comput. 33 (3) (2015) 207–222.
[45] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evolut. Comput. 1 (1) (1997) 67–82.
[46] D. Xia, Y. Li, W. Gong, G. He, An adaptive differential evolution algorithm for constrained optimization problems, Acta Electronica Sinica 44 (10) (2016) 2535–2542.
[47] B. Xu, L. Tao, W. Cheng, A self-adaptive differential evolution algorithm with multiple strategies and its application, CIESC J. 67 (12) (2016) 5190–9198.
[48] K. Yu, X. Wang, Z. Wang, Constrained optimization based on improved teaching-learning-based optimization algorithm, Inf. Sci. 352;C353 (C) (2016) 61–78.
[49] H. Zhang, G.P. Rangaiah, Self-adaptive differential evolution with taboo list for constrained optimization problems and its application to pooling problems, Comput. Aided Chem. Eng. 29 (2) (2011) 572–576.
[50] W. Zhang, G.G. Yen, Z. He, Constrained optimization via artificial immune system., IEEE Trans. Cybern. 44 (2) (2014) 185–198.