Adaptive Differential Evolution With Sorting Crossover Rate for Continuous Optimization Problems

Yin-Zhi Zhou, Wen-Chao Yi, Liang Gao, *Member, IEEE*, and Xin-Yu Li

***Abstract*—Differential evolution (DE) is one of the best evolutionary algorithms (EAs). The effort of improving its performance has received great research attentions, such as adaptive DE (JADE). Based on the analysis on the aspects that may improve the performance of JADE, we introduce a modified JADE version with sorting crossover rate (*CR*). In JADE, *CR* values are generated based on mean value and Gaussian distribution. In the proposed algorithm, a smaller *CR* value is assigned to individual with better fitness value. Therefore, the components of the individuals, which have better fitness values, can appear in the offspring with higher possibility. In addition, the better offspring generated from last iteration are supposed to have better schemes, hence these schemes are preserved in next offspring generation procedure. This modified version is called as JADE algorithm with sorting *CR* (JADE_sort). The experiments results with several excellent algorithms show the effectiveness of JADE_sort.**

***Index Terms*—Adaptive differential evolution (JADE), scheme retention mechanism, sorting crossover rate (*CR*).**

I. Introduction

DIFFERENTIAL evolution (DE) proposed by Storn and Price [1] is an effective and efficient evolutionary algorithms (EAs). It has been widely used in real-world applications, including constrained optimization problems [2], [3], multiobjective optimization problems [4]–[6], engineering applications [7]–[9], etc. The performance of DE is strongly affected by the parameter settings or choice of mutation strategies. Many related works improve the performance of DE significantly. We will briefly review some papers in the following paragraphs.

Three parameters need to be set in the initialization phase: 1) mutation factor (F); 2) crossover rate (CR); and 3) number of population (NP). The relationship between algorithm's performance and the settings are still under study. Usually, the chosen of NP is related to the complexity of the problem and determined without changing during the iteration. There are many works presenting proper parameter setting [10]–[12], but these suggestions which are quite different from each other indicate that no fixed parameters setting can be omnipotent for all kinds of problems. The influence of different parameters setting to the performances of DE has been studied. Ali [13] analyzed the probability density function (pdf) of points generated by mutation operation. This process shows that a larger F value results in unnecessary points being generated outside the search region, which obviously slows down the efficiency. Based on Ali's work [12], Wang and Huang [14] restricted the points in the search range. They proved that as F increases, the probability distribution of offspring tends to be uniform distribution; on the other hand, a small F cannot change pdf remarkably, rendering the mutation operation meaningless in this case. Zaharie [15] analyzed the influence of binomial and exponential crossover operation. For exponential crossover, small range of CR values (usually [0.9, 1]) are better. In the case of binomial crossover, the use of a uniform discretization of [0, 1] is appropriate.

In other literature, dynamically adapting parameter strategies had been designed to avoid the trouble of finding out which parameter settings are suitable for the characteristics of problems. Brest *et al.* [16] proposed a parameters setting method with self-adapting strategy. The values of F and CR are randomly chosen within certain range. Parameters are updated only if satisfying probability constraints. Qin *et al.* [17] proposed SaDE that updated both mutation strategies and control parameters. Pan *et al.* [18] followed the successful application of SaDE and developed a different learning mechanism. In the proposed SspDE [18], the selection of mutation strategies for each individual is determined from the experience of each individual, while, in SaDE, the strategy for

Manuscript received February 11, 2016; revised February 9, 2017; accepted February 20, 2017. Date of publication March 10, 2017; date of current version August 16, 2017. This work was supported in part by the Natural Science Foundation of China under Grant 51435009, Grant 51375004, Grant 51421062, and Grant 61232008, and in part by the Open Research Fund Program of the State Key Laboratory of Digital Manufacturing Equipment and Technology, HUST, China. This paper was recommended by Associate Editor F. Herrera. *(Corresponding author: Xin-Yu Li.)*

Y.-Z. Zhou is with the State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China, and also with the School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore 639798 (e-mail: leitezhou@gmail.com).

W.-C. Yi is with the State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China, and also with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: janety1989@gmail.com).

L. Gao and X.-Y. Li are with the State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: gaoliang@mail.hust.edu.cn; lixinyu@mail.hust.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2017.2676882

each individual is chosen according to the success rate of each strategy. In parameter updating, SaDE uses successful rate to generate mean value of normal distribution for each *CR*; *F* are independent from the previous experience. In SspDE, *CR* and *F* are either randomly chosen from the list storing successful parameters or randomly generated. EPSDE proposed by Mallipeddi *et al.* [19] also used the adapting strategy that the parameters and mutation strategies which generate better trail vectors surviving into next iteration. The parameters are chosen from a fixed range pool of discrete values with step 0.1, different from using continuous values in SaDE and SspDE.

Zhang and Sanderson [20] used *p*best individual, which is randomly chosen from the top 100**p*% individuals set, in mutation strategy. This so called adaptive DE (JADE) algorithm updates *CR* and *F* of each individual combining the information of successful experience of all individuals and parameters of this individual in the last generation. Based on JADE, Islam *et al.* [21] proposed an improved version called MDE-pBX. They modified the crossover operation by replacing the target vector with a randomly chosen vector from top *p* individuals, where *p* is gradually decreased from (*NP*/2) to 1. Instead of arithmetic mean and Lehmer mean used in JADE, MDE-pBX use power mean with power equals to 1.5. Tanabe and Fukunaga [22] proposed success-history-based parameter adaptation DE algorithm, which updates the mean value of control parameters according to the success information. Gong *et al.* [23] repaired the *CR* by modifying *CR* to the real generated probability, which is the ratio between the number of dimensions chosen from mutant vector and the sum of dimensions.

There are some other improvements omitted in this part, such as DDEBQ [24] and DEEP [25]. We also omit several algorithms based on self-adaptive population, such as the works in Teo [26] and Brest and Maucec [27]. The details of other state-of-the-art researches can be referred to two surveys [28], [29].

Based on current literature, most algorithms listed above did not use DE/rand/1 strategy alone. Some strategies try to add the information of best individuals in the population, such as DE/current-to-best/1 and DE/best/1. These strategies are effective for unimodal problems; however, they may lead to premature convergence, especially solving multimodal problems. Some mechanisms can help these strategies avoiding mentioned disadvantages and exert their strengths. In SaDE [17], SspDE [18], CoDE [30], and EPSDE [19], mixed mutation strategies are used. In addition, proper parameter setting can achieve the same effect, such as in JADE.

Different from the above researches, in this paper, a modified version of JADE is proposed. Its motivation is to propagate better scheme by assigning smaller *CR* values to better individuals and proposing the better scheme retention mechanism based on JADE. To balance the exploration and exploitation ability, an adaptive *p* setting is developed.

In the first strategy, the randomly assignment of *CR* values in JADE is replaced with better mechanism. Individuals with better fitness values are supposed to contain better scheme. Assigning smaller *CR* values result in smaller crossover possibilities for fitter individuals and it has better chance to propagate their better scheme. In the second strategy, the result that an offspring can replace its parent means that the scheme comes from the trail vector may be better than the corresponding component combination. This scheme should be retained into its offspring in the next generation. The proposed algorithm is compared with several famous EAs to verify its effectiveness.

The rest of this paper is organized as follows. Section II presents the traditional DE algorithm. Section III introduces the proposed JADE algorithm with sorting *CR* (JADE_sort). The computational results and analyses of comparison between JADE_sort and other DE algorithms are provided in Section IV. Finally, Section V concludes the proposed algorithm.

II. Classical DE Algorithms

The classical DE algorithms are briefly presented in this section. DE is a population-based algorithm; it consists of *NP* individuals $x_{i,G}$, where *G* denotes the number of iteration. Each individual $x_{i,G}$ consists of *D* variables constrained within the search range of $[x\min_j, x\max_j], j = 1, \ldots, D$. Usually, the initial individuals are randomly generated. Mutation, crossover, and selection operations are three main operators. These operations will be discussed in detail as follows.

A. Mutation Operation

DE/rand/1 is the most commonly used mutation strategy. The mutation vector $v_{i,G+1}$, $i = (1, 2, \ldots, NP)$, is generated with three randomly chosen target vectors $x_{r1,G}$, $x_{r2,G}$, $x_{r3,G}$. The formula is represented as

$$v_{i,G+1} = x_{r1,G} + F\left(x_{r2,G} - x_{r3,G}\right), r1 \neq r2 \neq r3 \neq i \quad (1)$$

where *F* is a mutation control parameter.

DE/current-to-best/1 is another frequently used strategy

$$v_{i,G+1} = x_{i,G} + F\left(x_{\text{best},G} - x_{i,G}\right) + F\left(x_{r1,G} - x_{r2,G}\right) \quad r1 \neq r2 \neq i \quad (2)$$

where $x_{\text{best},G}$ is the best individual in the population.

B. Crossover Operation

Crossover operation generate trial vector by picking the components from the target vector or its offspring mutant vector with probability. Usually binomial crossover can be described as

$$u^j_{i,G+1} = \begin{cases} v^j_{i,G+1}, & \text{if } r(j) \leq CR \text{ or } j = n_j \\ x^j_{i,G}, & \text{otherwise} \end{cases} \quad (3)$$

where $u^j_{i,G+1}$ means the *j*th number of trial vector $u_{i,G+1}$; $r(j)$ is a random number between [0, 1]; n_j is a randomly generated dimension to make sure that at least 1-D of the trial vector is chosen from the mutant vector; and *CR* is crossover parameter.

C. Selection Operation

The selection operation determines whether the trail vector or the target vector can be saved into the next iteration by using greedy selection strategy

$$x_{i,G+1} = \begin{cases} u_{i,G+1} & \text{if } f(u_{i,G+1}) \le f(x_{i,G}) \\ x_{i,G} & \text{otherwise} \end{cases} \quad (4)$$

where $f(u_{i,G+1})$ and $f(x_{i,G})$ are the objective values of $u_{i,G}$ and $x_{i,G}$, and $f(u_{i,G+1}) \le f(x_{i,G})$ is used for solving minimization problems.

III. Adaptive Differential Evolution Algorithm With Sorting CR

In this part, a JADE algorithm with sorting *CR* is proposed based on JADE. The main innovations of this paper are as follows.

1) An adaptive p setting is proposed, instead of a fixed p value in original JADE. Setting the p value larger at the beginning of the iteration can enhance the global search ability, and then adaptively decreasing the value helps to strengthen the local search ability.
2) Sorting *CR* mechanism is added. After sorting the *CR* set and individuals set according to their values and fitness, respectively, the better individual is assigned with smaller *CR* value, which helps to maintain the better scheme that the better individual contains.
3) A scheme retention mechanism is proposed to maintain the components from the mutant vector of the better offspring. It can improve JADE's local search ability by providing the deep search around the individuals.

The proposed algorithm will be described in detail in the following sections.

A. DE/Current-to-pbest/1/ Strategy

DE/rand/1 and DE/current-to-best/1 are two commonly applied mutation strategies. The latter strategy uses the best individual to guide the search direction, which speeds up algorithm's convergence. However, fast convergence may lead to the lack of global exploration ability. The algorithm gets stuck into local optimum for multimodal optimization problems. Therefore, a current-to-*p*best strategy proposed by Zhang and Sanderson [20] is adopted here

$$v_{i,G+1} = x_{i,G} + F_i\left(x^p_{\text{best},G} - x_{i,G}\right) + F_i\left(x_{r1,G} - x_{r2,G}\right) \quad (5)$$

where $x^p_{\text{best},G}$ is a randomly chosen individual with top $100p\%$ objective value. Therefore, the population will not convergent to the same best individual. This strategy has proved to be able to overcome the shortcoming of premature convergence of DE/current-to-best/1 strategy and accelerate convergence speed of DE/rand/1 strategy. x_{r1} and $x_{r2,}$ are randomly chosen individuals that are not equal to $x^p_{\text{best},G}$ and x_i.

In the original JADE, the p value keeps small during the whole evolutionary process. We use an adaptive setting strategy in this paper. At the beginning of the evolutionary process, p is set to be large to keep the diversity of the population. As the individuals converge, p should decrease to improve the local search ability. Therefore, p is reduced generationally using the following strategy:

$$p = \max\left(\left\lfloor \frac{NP}{2}\left(1 - \frac{G}{G_{\max}}\right)\right\rfloor, 2\right) \quad (6)$$

where $\lfloor x \rfloor$ means the minimum integer larger than x; G represents the current iteration number, while $G_{\max}$ gives the maximum iteration number. When G equals to 0, the value of p is $\lfloor NP/2 \rfloor$, the algorithm can therefore enhance its exploration ability. Then, p is gradually reduced to 2.

B. Archive Set

In JADE, DE/current-to-*p*best/1 strategy with archive can be described as

$$v_{i,G+1} = x_{i,G} + F_i\left(x^p_{\text{best},G} - x_{i,G}\right) + F_i\left(x_{r1,G} - \bar{x}_{r2,G}\right) \quad (7)$$

where $\bar{x}_{r2,G}$ is a random individual chosen from archive set. The archive set is a pool containing the current population and a constant number, usually the population size, of individuals replaced in former iterations. The archive is kept within a limited size by randomly removing some solutions.

A randomly selected item in archive set has worse fitness value with high possibility than the one selected from current population. Adding a vector with individuals in archive set can guide the search direction into promising area.

C. Parameters Adaption Strategy

In JADE, F_i and CR_i are independently generated according to successful parameters in last generation. The formula of generating CR_i is as follows:

$$CR_i = \text{rand}n_i(\mu_{CR}, 0.1) \quad (8)$$

where $\text{rand}n_i(\mu_{CR}, 0.1)$ generates a random value with a uniformly distribution. 0.1 is standard deviation; μ_{CR} is mean value, which is calculated using the following formula:

$$\mu_{CR} = (1-c)\mu_{CR} + c \cdot \text{mean}_A(S_{CR}) \quad (9)$$

where S_{CR} is the set of all successful crossover probabilities in last generation; $\text{mean}_A(x)$ is the usual arithmetic mean; and c is a positive constant between 0 and 1.

The adaption of F_i is similar to CR_i with the following formula:

$$\begin{cases} F_i = \text{rand}c_i(\mu_F, 0.1) \\ \mu_F = (1-c)\mu_F + c \cdot \text{mean}_L(S_F) \\ \text{mean}_L(S_F) = \sum_{S_F} F^2 / \sum_{S_F} F \end{cases} \quad (10)$$

where $\text{rand}c_i(x, y)$ means Cauchy distribution with location parameter x and scale parameter y; $\text{mean}_L(x)$ means Lehmer mean; and S_F is the set of successful mutation factors.

D. CR Sorting Mechanism

Crossover operation is an important strategy related to the diversity of the population. As stated in the above part, the *CR* value in JADE are randomly generated and assigned to individuals. It neglects the relationship between the *CR* values and the fitness values of the individuals.

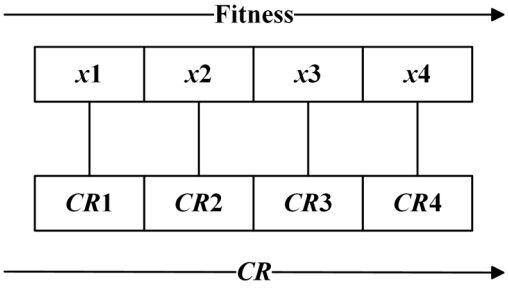


Fig. 1. Description of *CR* sorting mechanism.

Typically, an individual with better fitness value contains better scheme. In the crossover operation, retaining the better scheme into its offspring is more likely to generate better offspring. Therefore, for an individual with relatively better fitness value among the population should be assigned a relatively small corresponding *CR* value.

In the proposed sorting *CR* mechanism, both *CR* set and population are sorted according to their values. The smaller *CR* value is assigned to individual with better fitness value; the larger *CR* value is then assigned to individual with poorer fitness value. The entire process is summarized in Fig. 1.

E. Better Scheme Retention Mechanism

The binomial crossover is adopted from Neri and Tirronen [28]. We first set the $b^j_{i,G}$ as the binary string used in the binomial crossover. $b^j_{i,G}$ is defined as the jth component in ith offspring generated from parent or mutation vector in the Gth generation. Thus its value equals to 1, if the condition is met; otherwise, its value equals to 0. Hence, binomial crossover can be given as follows:

$$b^j_{i,G} = \begin{cases} 1, & \text{if rand} \geq CR \text{ or } j = j\text{rand} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

$$u^j_{i,G} = b^j_{i,G} v^j_{i,G} + \left(1 - b^j_{i,G}\right) x^j_{i,G}. \tag{12}$$

Based on (12), the binary string is stochastically related to crossover parameter *CR*. The components that inherit from the mutant vector $v_{i,G}$ of the trail vector $u_{i,G}$ are directly related to binary string.

During the iteration, some offspring can replace their parents. It means the components taken from the offspring are better than the origin components in the parent. Keeping these components into next generation is more likely to generate better offspring. Thus in this process, the randomly string used in the next iteration is not generated by calculating the conditions, but generated by converting $b^j_{i,G}$. $b^j_{i,G} = 0$ means this component taken from mutation vector. In the next generation, $b^j_{i,G+1}$ should be converted to 1. The value of $b^j_{i,G+1}$ is calculated as

$$b^j_{i,G+1} = 1 - b^j_{i,G}. \tag{13}$$

The process of scheme retention mechanism can be shown in Fig. 2. In the Gth generation, the first and fourth components of trail vector are selected from mutant vector. Assume the trial vector is better than the target vector in fitness, it becomes the target vector for next iteration. Therefore, according to the retention mechanism, in $G+1$th generation, the components selected from target vector should be first and fourth components.

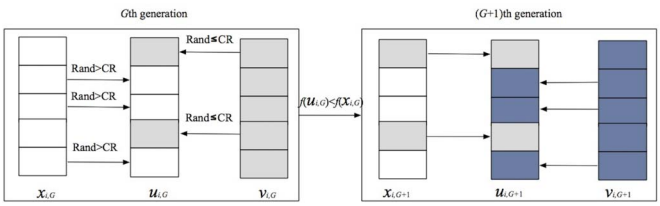


Fig. 2. Description of better scheme retention mechanism.

F. Procedure of JADE_Sort

In summary, in proposed JADE_sort, individuals and *CR* values are both sorted according to their values. Therefore, individual with better fitness value is assigned smaller *CR* value for crossover operation and vice-versa. The scheme from offspring that is better than its parent is contained into next offspring.

The procedure of JADE_sort is present in below.

Step 1: Set $G = 0$; randomly generate initial population; set μ_F and μ_{CR}.
Step 2: Calculate the fitness values of the population.
Step 3: Calculate the value of p; rank the population according to their fitness; and choose the top p individuals into $p$best set.
Step 4: Generate the F_i and CR_i for each individual according to (11)–(13).
Step 5: Sort CR_i according to their values.
Step 6: Use current-to-$p$best /1/bin strategy to generate trial vectors u_i; calculate the fitness values of trial vectors; use selection process to choose the better individuals into next generation.
Step 7: Update archive set; update μ_F and μ_{CR}.
Step 8: Set $G = G + 1$; if G meets the stopping criteria, stop the iteration; otherwise, turn to step 3.

IV. Experimental Results

The 25-benchmark functions are used to test the effectiveness of the proposed JADE_sort algorithm. These functions are presented for CEC-2005 competition. The detail description of functions can be found in [34]. According to the classification, F01–F05 belong to unimodal functions group; F06–F12 are basic multimodal functions; F13 and F14 are expanded multimodal functions; and F15–F25 are hybrid composition functions.

In the following sections, we first study the setting of F and *CR* for the proposed algorithm in Section IV-B. In Section IV-C, the proposed algorithm with three improved mechanisms is compared with the three versions with only one improved mechanism. It is compared with several classical DE algorithms in Section IV-D. Then the detailed comparisons are made among the JADE variants that share many similarities with the proposed algorithm in Section IV-E. In Section IV-F,

TABLE I
COMPARISONS WITH DIFFERENT PARAMETER SETTING

30D	JADE1		JADE1_sort		JADE2		JADE2_sort
1	2.88E-21±2.43E-21	-	0.00E+00±0.00E+00		0.00E+00±0.00E+00	=	0.00E+00±0.00E+00
2	1.98E+00±7.06E-01	-	1.09E-27±4.35E-28		6.09E-16±9.66-16	+	1.35E-14±1.84-14
3	2.55E+06±7.65E+05	-	1.13+04±1.54E+04		8.83E+04±5.48E+04	=	8.73E+04±5.38E+04
4	4.14E+01±1.33E+01	-	5.07E-16±1.45E-15		6.96E-06±1.23E-05	=	4.55E-06±6.66E-06
5	2.02E+01±1.74E+01	-	2.64E-11±1.21E-11		3.85E+02±4.43E+02	=	4.45E+02±3.64E+02
6	5.79E+00±5.76E-01	-	0.00E+00±0.00E+00		1.59E+00±1.99E+00	=	6.38E-01±1.49E+00
7	4.19E-09±2.67E-09	-	2.96E-04±1.48E-03		9.55E-03±9.89E-03	=	1.04E-02±7.14E-03
8	2.09E+01±3.47E-02	-	2.06E+01±3.84E-01		2.09E+01±6.66E-02	=	2.09E+01±3.70E-02
9	5.42E+01±6.33E+00	-	3.86E+01±1.53E+01		5.24E+00±1.73E+00	-	2.96E-04±1.48E-03
10	1.68E+02±1.26E+01	-	2.01E+01±6.58E+00		5.98E+01±8.47E+00	-	3.33E+01±8.00E+00
11	3.74E+01±1.03E+00	-	7.56E+00±3.83E+00		2.84E+01±1.04E+00	-	2.56E+01±1.91E+00
12	2.28E+03±2.23E+03	=	3.16E+03±4.02E+03		1.52E+03±1.57E+03	+	1.32E+03±1.20E+03
13	1.29E+01±9.29E-01	-	4.38E+00±1.81E+00		3.62E+00±3.26E-01	-	2.27E+00±1.66E-01
14	1.33E+01±1.38E-01	-	1.29E+01±2.20E-01		1.26E+01±2.51E-01	-	1.24E+01±2.39E-01
15	3.88E+01±7.26E+01	=	3.84E+02±6.88E+01		3.72E+02±5.42E+01	=	3.64E+02±5.68E+01
16	2.07E+02±6.04E+01	-	6.00E+01±7.27E+01		1.13E+02±1.04E+02	-	7.32E+01±7.17E+01
17	2.26E+02±4.81E+01	-	1.08E+02±9.57E+01		1.28E+02±3.22E+01	=	1.30E+02±8.76E+01
18	9.05E+02±9.50E-01	-	9.04E+02±2.20E-01		9.02E+02±2.14E+01	=	9.06E+02±1.77E+00
19	9.05E+02±8.89E-01	-	9.02E+02±2.13E+01		9.06E+02±1.70E+00	=	9.02E+02±2.13E+01
20	9.05E+02±1.02E+00	-	9.04E+02±2.79E-01		9.06E+02±1.51E+00	=	9.06E+02±2.37E+00
21	5.00E+02±1.16E-13	=	5.00E+02±8.37E-14		5.00E+02±1.01E-13	=	5.12E+02±6.00E+01
22	8.84E+02±1.31E+01	-	8.69E+02±1.70E+01		8.91E+02±1.64E+01	=	8.86E+02±1.13E+01
23	5.34E+02±4.00E-08	-	5.34E+02±1.83E-04		5.34E+02±3.44E-13	=	5.34E+02±4.25E-13
24	2.00E+02±2.90E-14	=	2.00E+02±2.90E-14		2.00E+02±2.90E-14	=	2.00E+02±2.90E-14
25	2.10E+02±4.13E-01	-	2.09E+02±3.93E-01		2.11E+02±6.14E-01	=	2.11E+02±7.30E-01
+	0				2		
=	4				17		
-	21				6		

the proposed algorithm is compared with recently proposed competitive algorithms. At last, Section IV-G concludes the experimental part.

A. Parameters Setting

The following parameters setting for JADE_sort algorithm are used, unless explicitly stated. Population size (NP) is set to be 100; initial distribution parameters are given as follows: $\mu_F = 0.5$ and $\mu_{CR} = 0.5$ weight c equals to 0.1. Maximal number of fitness evaluations G equals to $D \times 10000$, where D represents the dimension of each function. Wilcoxon's rank sum test and t-test statistical analysis are used in the following sections. The significance levels are 0.05 for both statistical analysis methods. Wilcoxon's rank sum test is used in Sections IV-B–IV-E. t-test statistical analysis is used in Section IV-F. The symbol "=," "−," and "+" in tables separately means the result of comparison algorithms equals to, worse than and better than the last column in the tables.

B. Discussion on Setting of F and CR

In JADE, the mean value is updated with the mean value and the successful values in last iteration. In this part, we consider the following two different setting strategies.

1) Using constant value throughout the iteration. Usually the F value should be set between [0.1, 0.9], and the CR value should be close to 1. In this part, we set $\mu_F = 0.7$ and $\mu_{CR} = 0.9$ in all the generation. This strategy is called JADE1.

固定uF,uCR的值

2) Using randomly generated value in each iteration. In the beginning of every generation, μ_F and μ_{CR} are randomly generated in the range [0, 1]. This strategy is called JADE2.

uF,uCR在[0,1]之间随机生成

Table I gives the results of two strategies and its modified versions with sorting CR. The performance of JADE, JADE_sort, and JADE1_sort are summarized in Table II.

Based on the comparisons results of JADE, JADE1, JADE2, and three modified algorithms, by using the CR sorting strategy, the performance of the algorithms can be improved significantly. By adding CR sorting strategy, JADE1_sort algorithm can become no worse than JADE in 20 functions. Compared with JADE_sort, its performance is competitive as well. The results show that CR sorting strategy is effective.

C. Comparison With Single Improvement Algorithm

In this part, we study the JADE algorithm with only one of the three main improved mechanisms. The compared algorithms are as follows.

1) JADE with adaptive p setting, denoted as JADE-IM1.
2) JADE with sorting CR mechanism, denoted as JADE-IM2.
3) JADE with better scheme retention mechanism, denoted as JADE-IM3.

The experimental results of three algorithms and the proposed algorithm are given in Table III. From the summarized results, JADE_sort algorithm obtains the best performance. The proposed JADE_sort algorithm outperforms JADE-IM1 algorithm in 14 functions, the JADE-IM2 algorithm in 15 functions and the JADE-IM3 algorithm in

TABLE II
COMPARISON OF JADE, JADE_SORT, AND CONSTANT SETTING STRATEGY

30D	JADE		JADE_sort		JADE1_sort
1	0.00E+00±0.00E+00	=	0.00E+00±0.00E+00	=	0.00E+00±0.00E+00
2	1.19E-28±8.88E-29	+	5.77E-29±6.53E-29	+	1.09E-27±4.35E-28
3	6.34E+03±4.31E+03	=	1.21E+04±7.24E+03	=	1.13+04±1.54E+04
4	4.96E-16±1.16E-15	=	3.12E-08±1.41E-07	-	5.07E-16±1.45E-15
5	1.92E-06±9.58E-06	-	1.21E-06±3.25E-06	-	2.64E-11±1.21E-11
6	7.87E+00±1.88E+01	-	6.38E-01±1.49E+00	=	0.00E+00±0.00E+00
7	6.31E-03±6.36E-03	-	4.04E-03±4.92E-03	-	2.96E-04±1.48E-03
8	2.09E+01±5.59E-02	-	2.00E+01±2.51E-03	+	2.06E+01±3.84E-01
9	0.00E+00±0.00E+00	+	1.25E+01±3.33E+00	+	3.86E+01±1.53E+01
10	2.38E+01±3.83E+00	-	1.21E+01±3.08E+00	+	2.01E+01±6.58E+00
11	2.54E+01±1.53E+00	-	2.58E+01±5.16E+00	-	7.56E+00±3.83E+00
12	5.86E+03±5.05E+03	-	2.63E+03±2.34E+03	=	3.16E+03±4.02E+03
13	1.49E+00±9.64E-02	+	1.32E+00±8.94E-02	+	4.38E+00±1.81E+00
14	1.23E+01±3.00E-01	+	1.15E+01±5.85E-01	+	1.29E+01±2.20E-01
15	3.72E+02±7.92E+01	=	3.72E+02±5.42E+01	=	3.84E+02±6.88E+01
16	1.05E+02±1.40E+02	=	4.86E+01±7.54E+01	+	6.00E+01±7.27E+01
17	1.52E+02±1.32E+02	=	4.83E+01±2.15E+01	+	1.08E+02±9.57E+01
18	9.04E+02±1.06E+00	-	9.04E+02±3.66E-01	-	9.04E+02±2.20E-01
19	9.04E+02±7.35E-01	=	9.04E+02±1.12E+00	=	9.02E+02±2.13E+01
20	9.04E+02±9.66E-01	-	9.04E+02±7.55E-01	-	9.04E+02±2.79E-01
21	5.00E+02±1.16E-13	=	5.00E+02±4.92E-13	+	5.00E+02±8.37E-14
22	8.59E+02±1.82E+01	=	8.53E+02±2.53E+01	+	8.69E+02±1.70E+01
23	5.34E+02±5.93E-13	+	5.34E+02±2.20E-05	-	5.34E+02±1.83E-04
24	2.00E+02±3.33E-14	=	2.00E+02±6.03E-13	=	2.00E+02±2.90E-14
25	2.11E+02±8.30E-01	-	2.11E+02±6.93E-01	-	2.09E+02±3.93E-01
+	5		10		
=	10		7		
-	10		8		

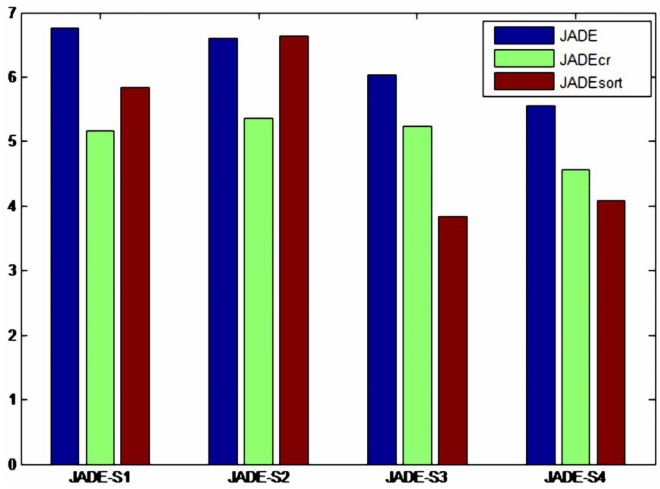


Fig. 3. Average ranking of 12 algorithms when $D = 30$.

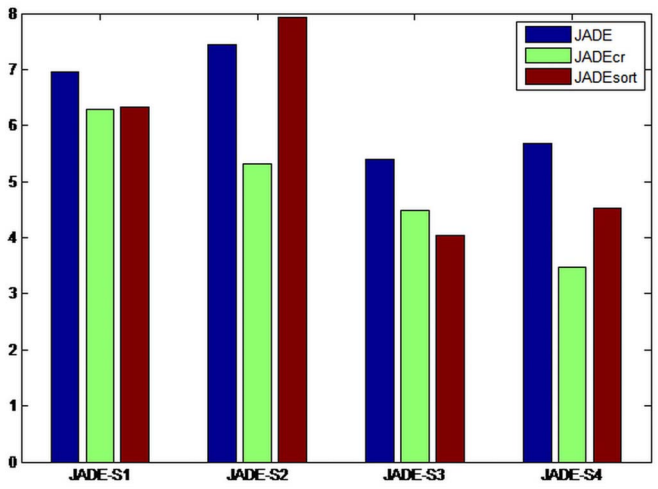


Fig. 4. Average ranking of 12 algorithms when $D = 50$.

12 functions. The hybrid of the three improvements is more effective than only using single one.

D. Comparison With Improved DE Algorithms

In this part, jDE [16], SaDE [17], EPSDE [19], JADE [20], and CoDE [30] are used for comparison in this experiment. The results of these algorithms are calculated by using the code provided by Wang *et al.* [30].

Table IV shows the direct comparison result with $D = 30$ and Table V shows the direct comparison result with $D = 50$. The bold number means the best value obtained in all six algorithms. The comparison results with $D = 30$ is shown in the box plot in Figs. 5 and 6. The results of 12 functions are given.

In summary, for the comparison results with $D = 30$, there is no function that jDE and SaDE is better than JADE_sort. Compared with EPSDE, JADE_sort is better or equally good in first 14 functions, except for F01 and F09. JADE is better than JADE_sort in three unimodal functions. The convergent speed of JADE_sort is slightly reduced by setting adaptive p value with a big initial value (equals to half of the population). However, JADE_sort is better in basic multimodal functions and expanded multimodal functions. CoDE is proposed with better global search ability, but compared with JADE_sort, it is worse in local search ability, which is obviously shown

TABLE III
COMPARISONS WITH SINGLE IMPROVEMENT ALGORITHM

30D	JADE-IM1		JADE-IM2		JADE-IM3		JADE sort
1	0.00E+00±0.00E+00	=	0.00E+00±0.00E+00	=	0.00E+00±0.00E+00	=	0.00E+00±0.00E+00
2	1.02E-28±9.52E-29	=	1.05E-28±1.21E-28	=	9.52E-29±9.01E-29	=	5.77E-29±6.53E-29
3	7.57E+03±4.39E+03	+	5.87E+03±3.77E+03	+	1.52E+04±9.87E+03	+	1.21E+04±7.24E+03
4	3.24E-09±1.06E-08	-	2.69E-13±1.22E-12	+	4.32E-14±2.01E-13	+	3.12E-08±1.41E-07
5	1.82E-07±9.01E-07	+	9.46E-07±3.00E-06	+	9.05E-07±3.93E-06	+	1.21E-06±7.24E+03
6	6.38E-01±1.49E+00	=	2.24E+01±2.91E+01	-	4.13E-13±1.78E-12	=	6.38E-01±1.49E+00
7	6.00E-03±8.10E-03	-	9.76E-03±1.50E-03	-	8.57E-03±8.79E-03	-	4.04E-03±4.92E-03
8	2.00E+01±2.03E-10	+	2.09E+01±1.91E-01	-	2.09E+01±1.92E-01	-	2.00E+01±2.51E-03
9	1.99E+01±5.72E+00	-	0.00E+00±0.00E+00	+	0.00E+00±0.00E+00	+	1.25E+01±3.33E+00
10	1.46E+01±3.14E+00	-	2.41E+01±3.87E+00	-	2.52E+01±4.08E+00	-	1.21E+01±3.08E+00
11	2.59E+01±2.86E+00	+	2.53E+01±1.29E+00	+	2.51E+01±1.71E+00	+	2.58E+01±5.16E+00
12	2.03E+03±3.37E+03	=	1.09E+04±3.86E+03	-	2.38E+03±2.82E+03	-	2.63E+03±2.34E+03
13	1.47E+00±6.00E-01	-	1.48E+00±1.04E-01	-	1.44E+00±1.30E-01	-	1.32E+00±8.94E-02
14	1.15E+01±6.20E-01	+	1.24E+01±2.89E-01	-	1.22E+01±3.19E-01	-	1.15E+01±5.85E-01
15	3.45E+02±8.97E+01	=	3.02E+02±5.84E+01	+	3.14E+02±1.40E+02	+	3.72E+02±5.42E+01
16	7.37E+01±1.17E+02	-	8.40E+01±9.82E+01	-	9.57E+01±1.30E+02	-	4.86E+01±7.54E+01
17	1.09E+02±1.17E+02	-	7.39E+01±1.23E+01	-	1.39E+02±1.21E+02	-	4.83E+01±2.15E+01
18	9.04E+02±9.11E-01	-	9.05E+02±1.09E+00	-	9.04E+02±5.40E-01	-	9.04E+02±3.66E-01
19	9.04E+02±7.55E-01	-	9.04E+02±8.42E-01	-	9.04E+02±3.82E-01	-	9.04E+02±1.12E+00
20	9.05E+02±1.03E+00	-	9.04E+02±5.84E-01	-	9.04E+02±6.77E-01	=	9.04E+02±7.55E-01
21	5.00E+02±2.15E-13	=	5.00E+02±9.91E-14	=	5.00E+02±7.06E-14	=	5.00E+02±4.92E-13
22	8.57E+02±2.64E+01	-	8.66E+02±1.86E+01	-	8.69E+02±2.07E+01	-	8.53E+02±2.53E+01
23	5.34E+02±5.15E-01	-	5.34E+02±2.20E-05	-	5.34E+02±1.71E-04	=	5.34E+02±2.20E-05
24	2.00E+02±1.27E-12	-	2.00E+02±2.90E-14	=	2.00E+02±2.90E-14	=	2.00E+02±6.03E-13
25	2.11E+02±1.03E+00	-	2.11E+02±6.85E-01	-	2.11E+02±7.62E-01	-	2.11E+02±6.93E-01
+	5		6		6		
=	6		4		7		
-	14		15		12		

TABLE IV
DIRECT COMPARISON RESULTS WITH DIMENSION EQUALS TO 30

30D	jDE		SaDE		EPSDE		JADE		CoDE		JADE sort
1	**0.00E+00±0.00E+00**	=	**0.00E+00±0.00E+00**	=	**0.00E+00±0.00E+00**	=	**0.00E+00±0.00E+00**	=	**0.00E+00±0.00E+00**	=	**0.00E+00±0.00E+00**
2	7.52E-07±1.13E-06	-	1.62E-05±5.26E-05	-	1.03E-24±4.93E-24	-	1.19E-28±8.88E-29	-	1.53E-15±2.28E-15	-	**5.77E-29±6.53E-29**
3	1.81E+05±8.90E+04	-	5.24E+05±2.53E+05	-	1.31E+06±6.11E+07	-	**6.34E+03±4.31E+03**	+	1.06E+05±5.16E+04	-	1.21E+04±7.24E+03
4	3.31E-02±6.98E-02	-	1.09E+02±1.42E+02	-	7.70E+01±2.75E+02	-	**4.96E-16±1.16E-15**	+	7.22E-03±7.72E-02	-	3.12E-08±1.41E-07
5	4.39E+02±3.74E+02	-	3.14E+03±4.57E+02	-	1.46E+03±6.24E+02	-	**1.92E-06±9.58E-06**	+	5.48E+02±4.54E+02	-	1.21E-06±3.25E-06
6	2.03E+01±2.20E+01	-	5.25E+01±3.11E+01	-	3.19E-01±1.10E+00	-	7.87E+00±1.88E+01	-	4.25E-10±1.11E-09	-	**6.38E-01±1.49E+00**
7	1.25E-02±7.09E-03	-	1.21E-02±8.21E-03	-	1.93E-02±1.47E-02	-	6.31E-03±6.36E-03	=	1.06E-02±1.04E-02	-	**4.04E-03±4.92E-03**
8	2.10E+01±5.59E-02	-	2.09E+01±7.16E-02	-	2.10E+01±4,20E-02	-	2.09E+01±5.59E-02	-	2.01E+01±9.45E-02	-	**2.00E+01±2.51E-03**
9	**0.00E+00±0.00E+00**	+	**1.59E-01±4.70E-01**	+	**0.00E+00±0.00E+00**	+	**0.00E+00±0.00E+00**	+	**0.00E+00±0.00E+00**	+	1.25E+01±3.33E+00
10	5.43E+01±9.94E+00	-	4.71E+01±1.28E+01	-	4.58E+01±9.08E+00	-	2.38E+01±3.83E+00	-	4.11E+01±1.15E+01	-	**1.21E+01±3.08E+00**
11	2.79E+01±1.65E+00	-	1.67E+01±2.97E+00	+	3.53E+01±3.77E+00	-	2.54E+01±1.53E+00	+	**1.18E+01±2.66E+00**	+	2.58E+01±5.16E+00
12	1.18E+04±8.69E+03	-	4.01E+03±3.41E+03	-	3.95E+04±5.50E+03	-	5.86E+03±5.05E+03	-	**1.47E+03±1.88E+03**	+	2.63E+03±2.34E+03
13	1.70E+00±1.36E-01	-	3.99E+00±3.48E-01	-	1.92E+00±1.46E-01	-	1.49E+00±9.64E-02	-	1.58E+00±3.37E-01	-	**1.32E+00±8.94E-02**
14	1.29E+01±2.93E-01	-	1.27E+01±2.77E-01	-	1.36E+01±2.47E-01	-	1.23E+01±3.00E-01	-	1.23E+01±4.29E-01	-	**1.15E+01±5.85E-01**
15	3.40E+02±1.18E+02	=	3.60E+02±5.78E+01	=	**2.23E+02±4.88E+01**	+	3.72E+02±7.92E+01	=	3.84E+02±5.54E+01	-	3.72E+02±5.42E+01
16	7.72E+01±1.20E+01	-	1.03E+02±9.33E+01	-	1.25E+02±1.09E+02	-	1.05E+02±1.40E+02	-	7.44E+01±3.30E+01	-	**4.86E+01±7.54E+01**
17	1.40E+02±3.24E+01	-	6.57E+01±1.30E+01	-	1.82E+02±9.85E+01	-	1.52E+02±1.32E+02	-	7.32E+01±3.30E+01	-	**4.83E+01±2.15E+01**
18	9.04E+02±6.27E-01	=	8.74E+02±6.18E+01	+	**8.21E+02±4.32E+00**	+	9.04E+02±1.06E+00	=	9.05E+02±9.24E-01	-	9.04E+02±3.66E-01
19	9.04E+02±1.23E+00	=	8.54E+02±6.23E+01	+	**8.22E+02±5.16E+00**	+	9.04E+02±7.35E-01	=	9.05E+02±1.07E+00	-	9.04E+02±1.12E+00
20	9.04E+02±9.32E-01	=	8.84E+02±5.98E+01	+	**8.21E+02±4.52E+00**	+	9.04E+02±9.66E-01	=	9.04E+02±7.20E-01	=	9.04E+02±7.55E-01
21	**5.00E+02±8.61E-14**	=	**5.00E+02±1.16E-13**	=	8.34E+02±1.00E+02	-	**5.00E+02±1.16E-13**	=	**5.00E+02±4.88E-13**	=	**5.00E+02±4.92E-13**
22	8.76E+02±1.70E+01	-	9.38E+02±2.02E+01	-	**5.10E+02±7.58E+00**	+	8.59E+02±1.82E+01	=	8.63E+02±1.27E+01	-	8.53E+02±2.53E+01
23	**5.34E+02±3.10E-04**	=	**5.34E+02±4.80E-03**	=	8.55E+02±6.64E+01	-	**5.34E+02±5.93E-13**	+	**5.34E+02±4.56E-04**	-	**5.34E+02±2.20E-05**
24	2.00E+02±2.90E-14	=	2.00E+02±2.90E-14	=	2.13E+02±1.72E+00	-	2.00E+02±3.33E-14	=	2.00E+02±2.90E-14	+	2.00E+02±6.03E-13
25	2.10E+02±7.23E-01	+	2.13E+02±1.84E+00	-	2.13E+02±3.01E+00	-	2.11E+02±8.30E-01	=	2.11E+02±7.98E-01	=	2.11E+02±6.93E-01
+	2		5		6		6		4		
=	8		5		1		10		3		
-	15		15		18		9		18		

in unimodal functions, and slightly poorer in global search ability, while JADE_sort outperforms in basic multimodal functions and expanded multimodal functions.

For the comparison results with $D = 50$, JADE_sort is the best algorithm among all six algorithms. The algorithm remains its strength in global search ability.

E. Comparison With JADE and JADE_cr Variants Algorithms

In this part, considering that the proposed algorithm shares many similarities with JADE_cr and JADE algorithms, a detailed discussion of JADE_sort with JADE and JADE_cr [35] algorithms is presented. JADE_cr algorithm

TABLE V
DIRECT COMPARISON RESULTS WITH DIMENSION EQUALS TO 50

50D	jDE		SaDE		EPSDE		JADE		CoDE		JADE_sort
1	**0.00E+00±0.00E+00**	=	2.02E-30±1.01E-29	=	5.05E-30±1.41E-29	=	0.00E+00±0.00E+00	=	8.08E-30±4.04E-29	=	**0.00E+00±0.00E+00**
2	1.04E-02±1.01E-02	-	1.22E-01±1.89E-01	-	1.56E-17±7.82E-17	-	**3.19E-27±1.49E-27**	+	6.40E-09±8.07E-09	-	5.10E-21±5.49E-21
3	4.27E+05±1.48E+05	-	9.62E+05±3.47E+05	-	1.94E+07±3.16E+07	-	**1.73E+04±6.87E+03**	+	1.94E+05±7.16E+04	-	3.48E+04±1.64E+04
4	4.22E+02±4.28E+02	-	6.61E+03±3.01E+03	-	8.25E+03±1.47E+04	-	**3.41E-01±5.87E-01**	+	3.86E+02±2.61E+02	-	2.43E+01±3.57E+01
5	3.06E+03±6.22E+02	-	8.23E+03±1.14E+03	-	4.38E+03±1.12E+03	-	1.39E+03±5.38E+02	=	3.17E+03±5.89E+02	-	**1.10E+03±4.33E+02**
6	4.86E+01±3.12E+01	-	8.26E+01±4.18E+01	-	1.59E+00±1.99E+00	=	3.22E+00±9.62E+00	=	**8.68E-01±1.50E+00**	=	1.43E+00±1.95E+00
7	**3.05E-03±7.03E-03**	=	6.10E-03±1.11E-02	=	5.81E-03±6.92E-03	=	3.15E-03±6.56E-03	+	3.64E-03±5.95E-03	=	4.73E-03±7.37E-03
8	2.11E+01±4.86E-02	-	2.11E+01±5.48E-02	-	2.11E+01±4.72E-02	-	2.11E+01±2.30E-01	-	2.01E+01±1.14E-01	-	**2.00E+01±4.83E-04**
9	**0.00E+00±0.00E+00**	+	1.79E+00±1.07E+00	+	3.98E-02±1.99E-01	+	**0.00E+00±0.00E+00**	+	3.98E-01±5,74E-01	+	1.01E+01±1.49E+01
10	9.49E+01±1.17E+01	-	1.23E+02±2.20E+01	-	1.57E+02±4.15E+01	-	4.74E+01±1.02E+01	-	7.42E+01±1.44E+01	-	**3.18E+01±6.19E+00**
11	5.43E+01±2.02E+00	+	3.91E+01±3.19E+00	+	6.95E+01±3.82E+01	-	5.16E+01±2.55E+00	+	**2.88E+01±4.50E+00**	+	5.44E+01±3.61E+00
12	1.54E+04±1.24E+04	=	1.42E+04±1.19E+04	=	3.16E+05±5.68E+04	-	1.26E+04±8.95E+03	=	**1.17E+04±8.07E+03**	=	1.30E+04±1.67E+04
13	3.06E+00±1.70E-01	-	8.81E+00±1.48E+00	-	6.30E+00±5.55E-01	-	2.78E+00±1.96E-01	-	3.38E+00±5.20E-01	-	**2.48E+00±2.19E-01**
14	2.26E+01±2.71E-01	-	2.22E+01±3.25E-01	-	2.34E+01±2.50E-01	-	2.17E+01±3.38E-01	-	2.21E+01±4.72E-01	-	**2.09E+01±4.42E-01**
15	3.12E+02±9.71E+01	=	3.93E+02±3.79E+01	-	**2.48E+02±5.23E+01**	+	2.72E+02±9.36E+01	+	3.76E+02±6.63E+01	-	3.30E+02±9.55E+01
16	8.18E+01±1.01E+01	-	9.60E+01±6.83E+01	-	1.49E+02±1.10E+02	-	9.48E+01±1.00E+02	-	7.19E+01±2.84E+01	-	**3.23E+01±2.02E+01**
17	1.82E+02±2.68E+01	-	1.52E+02±1.41E+02	-	2.47E+02±7.64E+01	-	1.16E+02±8.70E+01	-	7.33E+01±2.25E+01	-	**7.83E+01±7.29E+01**
18	9.20E+02±3.19E+01	=	9.85E+02±1.36E+01	-	**8.46E+02±3.46E+00**	+	9.22E+02±3.27E+00	=	9.16E+02±2.44E+01	=	9.24E+02±4.47E+00
19	9.20E+02±2.82E+00	+	9.78E+02±1.43E+01	-	**8.46E+02±3.77E+00**	+	9.22E+02±3.46E+00	=	9.21E+02±4.07E+00	=	9.19E+02±2.50E+01
20	9.20E+02±2.54E+00	+	9.77E+02±7.21E+02	-	**8.47E+02±3.07E+00**	+	9.22E+02±8.37E+00	=	9.17E+02±2.46E+01	=	9.23E+02±3.90E+00
21	6.62E+02±2.40E+02	-	7.21E+02±3.29E+02	-	7.29E+02±3.80E+00	-	5.20E+02±1.01E+02	+	7.24E+02±2.58E+02	-	**5.00E+02±6.79E-13**
22	9.04E+02±1.37E+01	-	9.85E+02±1.17E+01	-	**5.00E+02±8.49E-02**	+	8.93E+02±2.04E+01	=	9.06E+02±2.10E+01	-	8.89E+02±1.88E+01
23	8.80E+02±2.17E+02	-	7.63E+02±3.10E+02	-	7.34E+02±3.85E+00	-	5.96E+02±1.57E+02	=	7.67E+02±2.42E+02	-	**5.40E+02±1.46E+00**
24	**2.00E+02±1.43E-12**	=	5.53E+02±4.80E+02	-	2.41E+02±2.77E+01	-	2.32E+02±1.59E+02	-	**2.00E+02±6.03E-13**	=	**2.00E+02±1.83E-14**
25	**2.16E+02±1.26E+00**	=	2.23E+02±4.72E+00	-	2.38E+02±1.37E+01	-	2.19E+02±2.24E+00	-	2.17E+02±1.94E+00	=	2.17E+02±2.02E+00
+	4		2		6		8		2		
=	7		3		3		9		9		
-	14		20		16		8		14		

TABLE VI
RESULTS OF JADE_SORT IN FOUR SITUATIONS WHEN $d = 30$

30D	JADE sort-s1	JADE sort-s2	JADE sort-s3	JADE sort-s4
1	0.00E+00±0.00E+00	0.00E+00±0.00E+00	0.00E+00±0.00E+00	0.00E+00±0.00E+00
2	1.09E-19±2.38E-19	8.37E-19±1.27E-18	5.77E-29±6.53E-29	3.51E-28±2.28E-28
3	4.93E+04±3.28E+04	5.64E+04±3.31E+04	1.21E+04±7.24E+03	2.17E+04±1.62E+04
4	3.00E-04±6.60E-04	5.59E-03±1.30E-02	3.12E-08±1.41E-07	6.98E-07±1.65E-06
5	1.65E+02±3.72E+02	2.12E+02±3.63E+02	1.21E-06±3.25E-06	1.78E-04±3.72E-04
6	1.75E+00±2.02E+00	1.28E+00±1.89E+00	6.38E-01±1.49E+00	3.19E-01±1.10E+00
7	1.57E-02±1.53E-02	1.53E-02±1.42E-02	4.04E-03±4.92E-03	6.20E-03±8.13E-03
8	2.00E+01±9.93E-06	2.00E+01±7.33E-05	2.00E+01±2.51E-03	2.00E+01±4.65E-04
9	1.38E+01±4.37E+00	2.46E+01±7.50E+00	1.25E+01±3.33E+00	2.24E+01±8.71E+00
10	1.59E+01±3.27E+00	2.44E+01±7.48E+00	1.21E+01±3.08E+00	2.07E+01±6.68E+00
11	2.61E+01±3.19E+00	1.62E+01±3.69E+00	2.58E+01±5.16E+00	1.47E+01±2.81E+00
12	1.57E+03±1.74E+03	2.61E+03±2.71E+03	2.63E+03±2.34E+03	8.31E+02±1.18E+03
13	1.34E+00±9.74E-02	3.13E+00±6.81E-01	1.32E+00±8.94E-02	3.28E+00±7.28E-01
14	1.19E+01±5.73E-01	1.14E+01±9.90E-01	1.15E+01±5.85E-01	1.11E+01±7.58E-01
15	3.60E+02±8.16E+01	3.44E+02±5.06E+01	3.72E+02±5.42E+01	3.24E+02±5.22E+01
16	3.06E+01±4.65E+00	6.90E+01±1.15E+02	4.86E+01±7.54E+01	9.85E+01±1.42E+02
17	4.76E+01±6.00E+00	6.00E+01±7.40E+01	4.83E+01±2.15E+01	6.04E+01±7.42E+01
18	9.06E+02±1.43E+00	9.07E+02±1.70E+00	9.04E+02±3.66E-01	9.04E+02±7.60E-01
19	9.06E+02±1.38E+00	9.07E+02±1.85E+00	9.04E+02±1.12E+00	9.04E+02±6.14E-01
20	9.06E+02±1.20E+00	9.06E+02±1.73E+00	9.04E+02±7.55E-01	9.04E+02±9.17E-01
21	5.00E+02±1.99E-13	5.00E+02±2.06E-13	5.00E+02±4.92E-13	5.00E+02±1.87E-13
22	8.68E+02±2.44E+01	8.67E+02±2.61E+01	8.53E+02±2.53E+01	8.43E+02±2.73E+01
23	5.34E+02±3.99E-03	5.34E+02±3.57E-04	5.34E+02±2.20E-05	5.34E+02±4.34E-04
24	2.00E+02±1.55E-12	2.00E+02±1.58E-12	2.00E+02±6.03E-13	2.00E+02±2.90E-14
25	2.10E+02±8.35E-01	2.12E+02±1.19E+00	2.11E+02±6.93E-01	2.11E+02±8.57E-01

is a modified version of JADE. It uses real crossover probability to replace calculated CR value. The real crossover probability is

$$CR_i = \frac{\sum_{j=1}^{D} b_i^j}{D} \tag{14}$$

where b_i^j equals to 1 if $\text{rand}_j \leq CR_i$ and equals to 0 if $\text{rand}_j > CR_i$.

In the comparisons, we consider two situations: $D = 30$ and $D = 50$. Four mutation strategies are considered. The first and second mutation strategies do not include archive set. The second and fourth mutation strategies use randomly selected base vector.

DE/current-to-$p$best/1 without archive (denoted as s1)

$$v_{i,G+1} = x_{i,G} + F_i\left(x^p_{\text{best},G} - x_{i,G}\right) + F_i\left(x_{r1,G} - x_{r2,G}\right). \tag{15}$$

TABLE VII
RESULTS OF JADE_SORT IN FOUR SITUATIONS WHEN $d = 50$

50D	JADE_sort-s1	JADE_sort-s2	JADE_sort-s3	JADE_sort-s4
1	0.00E+00±0.00E+00	1.26E-29±4.19E-29	0.00E+00±0.00E+00	1.26E-29±4.19E-29
2	4.49E-11±1.17E-10	7.20E-11±5.90E-11	5.10E-21±5.49E-21	2.98E-18±8.11E-18
3	6.16E+04±2.37E+04	6.53E+04±1.81E+04	3.48E+04±1.64E+04	4.17E+04±3.43E+04
4	2.60E+02±3.57E+02	5.41E+02±5.28E+02	2.43E+01±3.57E+01	3.59E+01±4.58E+01
5	2.57E+03±4.60E+02	2.65E+03±4.71E+02	1.10E+03±4.33E+02	1.64E+03±4.87E+02
6	1.75E+00±2.02E+00	2.90E+00±2.23E+00	1.43E+00±1.95E+00	1.28E+00±1.90E+00
7	1.31E-02±1.61E-02	8.94E-03±1.42E-02	4.73E-03±7.37E-03	4.04E-03±5.99E-03
8	2.00E+01±4.86E-04	2.00E+01±1.03E-03	2.00E+01±4.83E-04	2.00E+01±7.45E-04
9	1.76E+01±2.01E+00	4.71E+01±9.74E+00	1.01E+01±1.49E+01	4.37E+01±9.80E+00
10	4.93E+01±8.02E+01	5.52E+01±1.25E+01	3.18E+01±6.19E+00	4.01E+01±1.18E+01
11	5.49E+01±3.59E+00	3.85E+01±4.10E+00	5.44E+01±3.61E+00	3.66E+01±4.29E+00
12	1.94E+04±1.65E+04	1.37E+04±1.18E+04	1.30E+04±1.67E+04	7.66E+03±6.32E+03
13	2.43E+00±2.39E-01	6.62E+00±1.12E+00	2.48E+00±2.19E-01	5.64E+00±1.09E+00
14	2.15E+01±8.09E-01	2.13E+01±1.03E+00	2.09E+01±4.42E-01	2.03E+01±1.09E+00
15	3.52E+02±8.81E+01	3.53E+02±8.52E+01	3.30E+02±9.55E+01	3.18E+02±8.81E+01
16	3.37E+01±7.45E+00	4.58E+01±1.90E+01	3.23E+01±2.02E+01	7.30E+01±1.00E+02
17	9.89E+01±9.91E+01	8.49E+01±1.06E+02	7.83E+01±7.29E+01	6.23E+01±7.83E+01
18	9.27E+02±2.80E+01	9.31E+02±2.81E+01	9.24E+02±4.47E+00	9.23E+02±3.39E+00
19	9.28E+02±2.76E+01	9.41E+02±8.21E+00	9.19E+02±2.50E+01	9.22E+02±3.50E+00
20	9.27E+02±2.74E+01	9.40E+02±7.27E+00	9.23E+02±3.90E+00	9.21E+02±4.69E+00
21	5.00E+02±8.95E-13	5.00E+02±1.01E-12	5.00E+02±6.79E-13	5.00E+02±6.81E-13
22	9.21E+02±2.57E+01	9.21E+02±2.42E+01	8.89E+02±1.88E+01	8.84E+02±2.06E+01
23	5.42E+02±3.22E+00	5.59E+02±7.56E+01	5.40E+02±1.46E+00	5.41E+02±3.29E+00
24	2.00E+02±1.73E-12	2.00E+02±1.69E-12	2.00E+02±1.83E-14	2.00E+02±1.83E-14
25	2.19E+02±2.14E+00	2.21E+02±3.41E+00	2.17E+02±2.02E+00	2.17E+02±2.18E+00

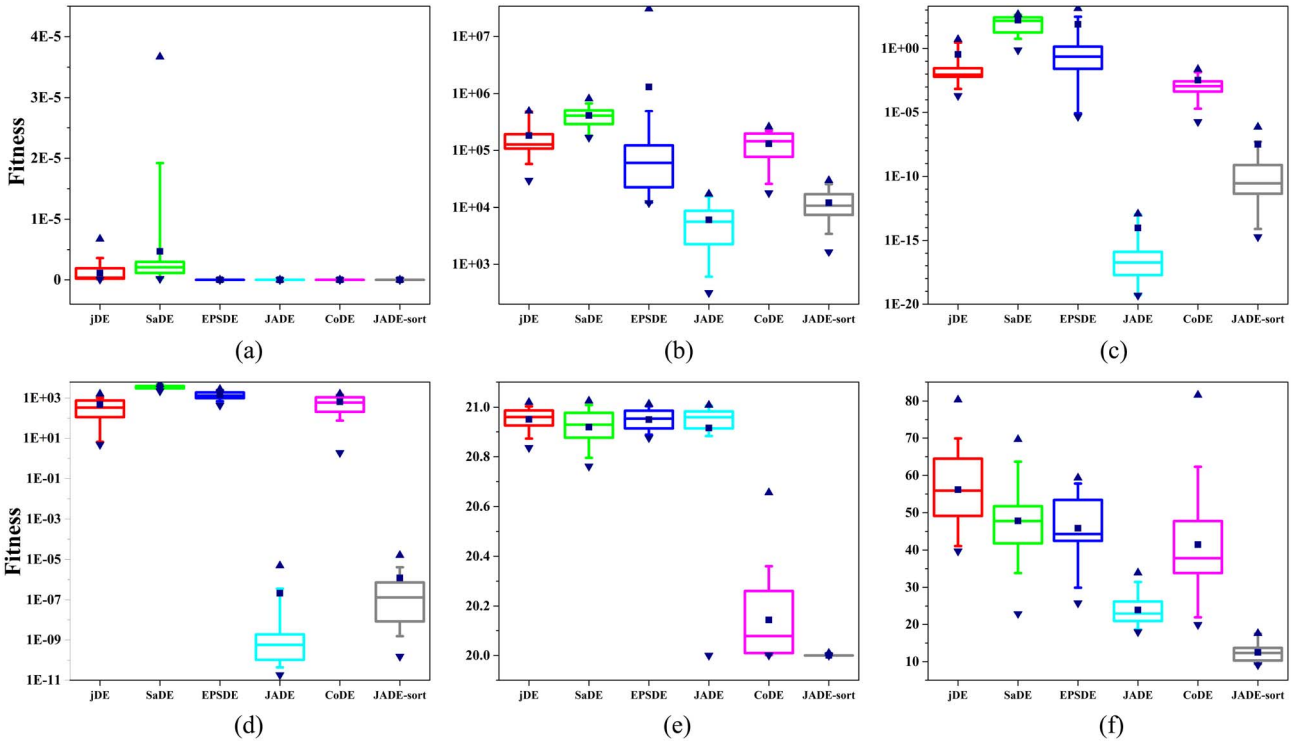


Fig. 5. Box plot of jDE, SaDE, EPSDE, JADE, CoDE, and JADE_sort in six test functions. (a) F02. (b) F03. (c) F04. (d) F05. (e) F08. (f) F10.

DE/rand-to-*p*best/1/bin without archive (denoted as s2)

$$v_{i,G+1} = x_{i,G} + F_i\left(x^p_{\text{best},G} - x_{r1,G}\right) + F_i\left(x_{r2,G} - x_{r3,G}\right). \quad (16)$$

DE/current-to-*p*best/1/bin with archive (denoted as s3)

$$v_{i,G+1} = x_{i,G} + F_i\left(x^p_{\text{best},G} - x_{i,G}\right) + F_i\left(x_{r1,G} - \bar{x}_{r2,G}\right). \quad (17)$$

DE/rand-to-*p*best/1/bin with archive (denoted as s4)

$$v_{i,G+1} = x_{i,G} + F_i\left(x^p_{\text{best},G} - x_{r1,G}\right) + F_i\left(x_{r2,G} - \bar{x}_{r3,G}\right). \quad (18)$$

The results of JADE_sort with four format are given in Tables VI and VII. The comparison results of JADE and JADE_cr are adopted from Yi *et al.* [36]. Fig. 3 is the average ranking of 12 algorithms in 25 test functions. The

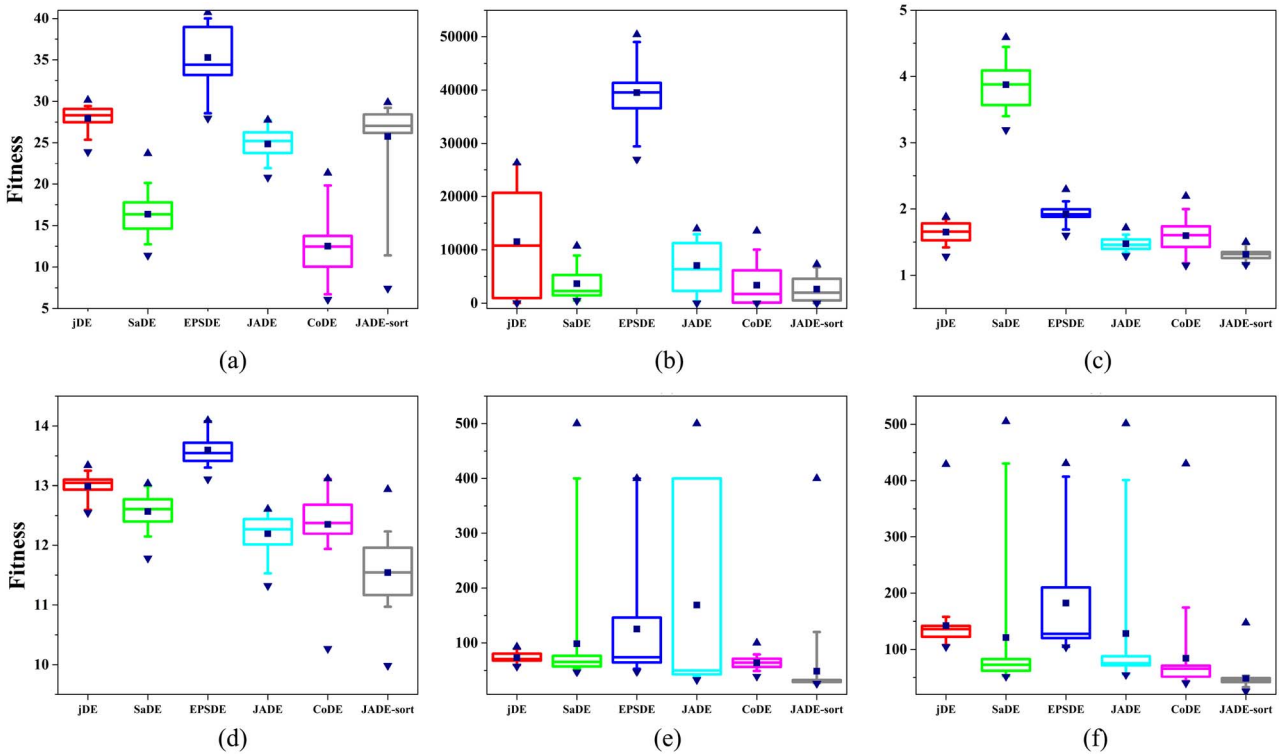


Fig. 6. Box plot of jDE, SaDE, EPSDE, JADE, CoDE, and JADE_sort in six test functions. (a) F11. (b) F12. (c) F13. (d) F14. (e) F16. (f) F17.

TABLE VIII
COMPARISONS WITH OTHER STATE-OF-THE-ART META-HEURISTIC ALGORITHMS

30D	LdDE	SPAM	DMPSADE	JADE-SI
1	-	=	=	=
2	-	=	-	=
3	-	+	-	+
4	=	-	-	=
5	=	-	-	=
6	=	-	=	=
7	-	+	+	=
8	-	=	-	-
9	-	+	+	+
10	-	-	-	-
11	=	+	-	=
12	=	+	-	-
13	-	-	-	=
14	+	-	-	-
15	=	+	+	=
16	NA	-	=	=
17	NA	-	-	=
18	NA	=	+	=
19	NA	=	+	=
20	NA	=	+	=
21	NA	=	=	=
22	NA	-	-	=
23	NA	=	-	=
24	NA	=	=	=
25	NA	-	-	NA
+	1	6	6	2
=	6	9	5	18
-	8	10	14	4

symbols on x-axis represent the JADE-based algorithms with first, second, third, and fourth mutation strategy, denoted as JADE-S1, JADE-S2, JADE-S3, and JADE-S4, respectively. The comparison standard is the mean value. In first and second mutation strategies, JADE_sort is worse than JADE_cr. In third and fourth mutation strategies, JADE_sort is better than JADE and JADE_cr. The best algorithm is JADE_sort with third mutation strategies, namely JADE_sort-s3.

Fig. 4 is the average ranking of 12 algorithms when D is equals to 50. The symbols on x-axis have the same meaning as in Fig. 3. Sum it up, in the first mutation strategies, JADE_sort-s1 is same as JADE_cr-s1 and better than JADE-s1. In the second and fourth mutation strategies, JADE_sort is worse than JADE_cr. In the third mutation strategies, JADE_sort-s3 is better than JADE_cr-s3 and JADE-s3. Comparing all 12 functions, the best algorithm is JADE_cr with fourth mutation strategies, namely JADE_sort-s3, JADE_sort with third mutation strategy ranks second with minor difference.

F. Comparison With Other State-of-the-Art Algorithms

In this part, other four algorithms (LdDE [37], SPAM [38], DMPSADE [39], and JADE-SI [40]) are used for comparison. JADE_sort-s3 is selected as the representative to be compared; we denoted it as JADE_sort for short. The results of the other four state-of-the art algorithms are adopted from their original paper. The symbol $=$, $-$, and $+$, respectively, means the result of the JADE_sort has no difference with, has difference and better than, has difference and worse than the compared algorithm. NA in Table VIII indicates that the results are not available in the original paper. From Table VIII, JADE_sort is better than other four algorithms. Compared with LdDE, JADE_sort is better in eight test functions and equal in six test functions. Compared with SPAM, JADE_sort

is better in ten test functions, and 14 test functions while compared with DMPSADE. JADE_sort is better in four test functions and equal in 18 functions while compared with JADE-SI. Thus, JADE_sort shows better global and local search ability than other state-of-the-art meta-heuristic algorithms.

G. Conclusion of the Experimental Results

In this chapter, nine algorithms are used for comparison with the proposed algorithm. The comparison with five successful DE variants and four state-of-the-art algorithms show the proposed algorithm is a competitive algorithm. Detailed discussions with JADE and JADE_cr in four mutation strategies also illustrate the effectiveness of proposed mechanism.

V. Conclusion

We propose a *CR* selection mechanism and add it into the flowchart of JADE to strength the search ability of DE and JADE algorithm. By setting better parents with smaller *CR* values, the mechanism can provide higher chance for these individuals to retain their scheme into their offspring. Combining with adaptive p setting strategy and scheme retention mechanism, the proposed algorithm can enhance the global search ability of JADE, which is verified through comparison tests.

However, as shown in the comparison, the proposed JADE_sort algorithm failed to provide better performance on hybrid composition functions. Although most algorithms cannot solve these hybrid composition functions effectively, it will be an interesting work to improve the frame of JADE to improve its global search ability for hybrid composition functions.

Besides, we only discuss the influence of *CR* in JADE. *F* has an important influence on the performance of DE. Designing a selection method considering both *F* and *CR* may be a practical way for further improvement. The proper setting of μ_F and μ_{CR} is another research topic. Using constant value in experiment *E* show it can improve the search ability of some functions. It can be helpful to find out more efficient adaptive method. In addition, JADE_sort can be used for real applications, such as engineering optimization problems [41], [42], etc.

References

[1] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[2] W.-F. Gao, G. G. Yen, and S.-Y. Liu, "A dual-population differential evolution with coevolution for constrained optimization," *IEEE Trans. Cybern.*, vol. 45, no. 5, pp. 1108–1121, May 2015.

[3] Y. Wang and Z. X. Cai, "Combining multiobjective optimization with differential evolution to solve constrained optimization problems," *IEEE Trans. Evol. Comput.*, vol. 16, no. 1, pp. 117–134, Feb. 2012.

[4] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 666–685, Oct. 2013.

[5] M. Ali, P. Siarry, and M. Pant, "An efficient differential evolution based algorithm for solving multi-objective optimization problems," *Eur. J. Oper. Res.*, vol. 217, no. 2, pp. 404–416, Mar. 2012.

[6] X. Qiu, J.-X. Xu, K. C. Tan, and H. A. Abbass, "Adaptive cross-generation differential evolution operators for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 232–244, Apr. 2016.

[7] Q.-K. Pan, L. Wang, L. Gao, and W. D. Li, "An effective hybrid discrete differential evolution algorithm for the flow shop scheduling with intermediate buffers," *Inf. Sci.*, vol. 181, no. 3, pp. 668–685, Feb. 2011.

[8] W. F. Sacco, A. A. de Moura Meneses, and N. Henderson, "Some studies on differential evolution variants for application to nuclear reactor core design," *Progr. Nuclear Energy*, vol. 63, pp. 49–56, Mar. 2013.

[9] A. R. Yildiz, "A new hybrid differential evolution algorithm for the selection of optimal machining parameters in milling operations," *Appl. Soft Comput.*, vol. 13, no. 3, pp. 1561–1566, Mar. 2013.

[10] R. Gamperle, S. D. Muller, and P. Koumoutsakos, "A parameter study for differential evolution," in *Proc. Adv. Intell. Syst. Fuzzy Syst. Evol. Comput.*, Interlaken, Switzerland, 2002, pp. 11–15.

[11] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Comput.*, vol. 9, no. 6, pp. 448–462, Jun. 2005.

[12] R. Storn, "Differential evolution research—Trends and open questions," in *Advances in Differential Evolution*, U. K. Chakraborty, Eds. Heidelberg, Germany: Springer, 2008, pp. 1–31.

[13] M. M. Ali, "Differential evolution with preferential crossover," *Eur. J. Oper. Res.*, vol. 181, no. 3, pp. 1137–1147, Sep. 2007.

[14] L. Wang and F.-Z. Huang, "Parameter analysis based on stochastic model for differential evolution algorithm," *Appl. Math. Comput.*, vol. 217, no. 7, pp. 3263–3273, Dec. 2010.

[15] D. Zaharie, "Influence of crossover on the behavior of differential evolution algorithms," *Appl. Soft Comput.*, vol. 9, no. 3, pp. 1126–1138, Jun. 2009.

[16] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.

[17] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

[18] Q.-K. Pan, P. N. Suganthan, L. Wang, L. Gao, and R. Mallipeddi, "A differential evolution algorithm with self-adapting strategy and control parameters," *Comput. Oper. Res.*, vol. 38, no. 1, pp. 394–408, Jan. 2011.

[19] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 1679–1696, Mar. 2011.

[20] J. Q. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[21] S. M. Islam, S. Das, S. Ghosh, S. Roy, and P. N. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 482–500, Apr. 2012.

[22] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Cancun, Mexico, 2013, pp. 71–78.

[23] W. Y. Gong, Z. H. Cai, and Y. Wang, "Repairing the crossover rate in adaptive differential evolution," *Appl. Soft Comput.*, vol. 15, pp. 149–168, Feb. 2014.

[24] S. Das, A. Mandal, and R. Mukherjee, "An adaptive differential evolution algorithm for global optimization in dynamic environments," *IEEE Trans. Cybern.*, vol. 44, no. 6, pp. 966–978, Jun. 2014.

[25] Y. L. Li *et al.*, "Differential evolution with an evolution path: A DEEP evolutionary algorithm," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1798–1810, Sep. 2015.

[26] J. Teo, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Comput.*, vol. 10, no. 8, pp. 673–686, Jun. 2006.

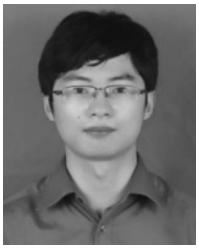
[27] J. Brest and M. S. Maucec, "Population size reduction for the differential evolution algorithm," *Appl. Intell.*, vol. 29, no. 3, pp. 228–247, Dec. 2008.

[28] F. Neri and V. Tirronen, "Recent advances in differential evolution: A survey and experimental analysis," *Artif. Intell. Rev.*, vol. 33, no. 1, pp. 61–106, Feb. 2010.

[29] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

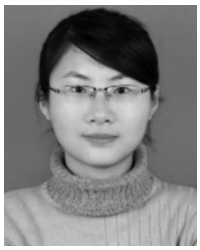
[30] Y. Wang, Z. X. Cai, and Q. F. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 55–66, Feb. 2011.

[31] P. Kaelo and M. M. Ali, "Differential evolution algorithms using hybrid mutation," *Comput. Optim. Appl.*, vol. 37, no. 2, pp. 231–246, Mar. 2007.
[32] S. I. Birbil and S. C. Fang, "An electromagnetism-like mechanism for global optimization," *J. Glob. Optim.*, vol. 25, no. 3, pp. 263–282, Mar. 2003.
[33] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553, Jun. 2009.
[34] P. N. Suganthan *et al.* (2005). *Problem Definitions and Evaluation Criteria for the CEC2005 Special Session on Real-Parameter Optimization.* [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan
[35] W. Y. Gong, Z. H. Cai, and Y. Wang, "Repairing the crossover rate in adaptive differential evolution," *Appl. Soft Comput.*, vol. 15, pp. 149–168, Feb. 2014.
[36] W. C. Yi, Y. Z. Zhou, L. Gao, X. Y. Li, and J. H. Mou, "An improved adaptive differential evolution algorithm for continuous optimization," *Expert Syst. Appl.*, vol. 44, pp. 1–12, Feb. 2016.
[37] N. D. Jana and J. Sil, "Levy distributed parameter control in differential evolution for numerical optimization," *Nat. Comput.*, vol. 15, no. 3, pp. 371–384, Sep. 2016.
[38] F. Caraffini, F. Neri, and L. Picinali, "An analysis on separability for memetic computing automatic design," *Inf. Sci.*, vol. 265, pp. 1–22, May 2014.
[39] Q. Q. Fan and X. F. Yan, "Self-adaptive differential evolution algorithm with discrete mutation control parameter," *Expert Syst. Appl.*, vol. 42, no. 3, pp. 1551–1572, Feb. 2015.
[40] Y. L. Xu, J.-A. Fang, W. Zhu, X. P. Wang, and L. D. Zhao, "Differential evolution using a superior–inferior crossover scheme," *Comput. Optim. Appl.*, vol. 61, no. 1, pp. 243–274, May 2015.
[41] J.-Q. Li, Q.-K. Pan, and P.-Y. Duan, "An improved artificial bee colony algorithm for solving hybrid flexible flowshop with dynamic operation skipping," *IEEE Trans. Cybern.*, vol. 46, no. 6, pp. 1311–1324, Jun. 2016.
[42] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2016.2554622.



Yin-Zhi Zhou received the B.Sc. and Ph.D. degrees in industrial engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2011 and 2015, respectively.

He is currently a Research Fellow with the School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore. His current research interests include evolutionary computation, scheduling, and simulation.



Wen-Chao Yi received the B.Sc. and Ph.D. degrees in industrial engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2011 and 2016, respectively.

She is currently a Research Fellow with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. Her current research interests include evolutionary algorithms, multiobjective optimization, scheduling, and simulation.

Liang Gao (M'08) received the B.Sc. degree in mechatronic engineering from Xidian University, Xi'an, China, in 1996, and the Ph.D. degree in mechatronic engineering from the Huazhong University of Science and Technology (HUST), Wuwhan, China, in 2002.

He is a Professor and the Head of the Department of Industrial and Manufacturing System Engineering, State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, HUST. He had published over 100 refereed papers. His current research interests include operations research and optimization and scheduling.

Xin-Yu Li received the Ph.D. degree in industrial engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2009.

He is an Associate Professor with the Department of Industrial and Manufacturing Systems Engineering, State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, HUST. He had published over 40 refereed papers. His current research interests include intelligent algorithm and scheduling.