Dichotomy Guided Based Parameter Adaptation for Differential Evolution

Xiao-Fang Liu, Zhi-Hui Zhan* (Corresponding Author), Jun Zhang
Department of Computer Science, Sun Yat-sen University, Guangzhou, P. R. China, 510275
Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, P.R. China
Collaborative Innovation Center of High Performance Computing, National Univ. of Defense Tech., China
Engineering Research Center of Supercomputing Engineering Software, Ministry of Education, P.R. China
*zhanzhh@mail.sysu.edu.cn

ABSTRACT
Differential evolution (DE) is an efficient and powerful population-based stochastic evolutionary algorithm, which evolves according to the differential between individuals. The success of DE in obtaining the optima of a specific problem depends greatly on the choice of mutation strategies and control parameter values. *Good* parameters lead the individuals towards optima successfully. The increasing of the success rate (the ratio of entering the next generation successfully) of population can speed up the searching. Adaptive DE incorporates success-history or population-state based parameter adaptation. However, sometimes *poor* parameters may improve individual with small probability and are regarded as successful parameters. The *poor* parameters may mislead the parameter control. So, in this paper, we propose a novel approach to distinguish between *good* and *poor* parameters in successful parameters. In order to speed up the convergence of algorithm and find more "*good*" parameters, we propose a dichotomy adaptive DE (DADE), in which the successful parameters are divided into two parts and only the part with higher success rate is used for parameter adaptation control. Simulation results show that DADE is competitive to other classic or adaptive DE algorithms on a set of benchmark problem and IEEE CEC 2014 test suite.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *Heuristic methods*.

General Terms
Algorithms, Performance, Design, Experimentation

Keywords
Adaptive parameter control; dichotomy-guided; differential evolution; evolutionary optimization

GECCO '15, July 11 - 15, 2015, Madrid, Spain
 ISBN 978-1-4503-3472-3/15/07…$15.00
DOI: http://dx.doi.org/10.1145/2739480.2754646

1. INTRODUCTION
Differential evolution (DE) is a stochastic evolutionary algorithm, which is designed for real parameter optimization problems [1]. Optimization is a challenging area in mathematics. DE has been extensively studied and has been applied to lots of practical problems because of its relative simplicity and competitiveness [2][3]. However, the search performance of DE algorithms depends greatly on control parameter settings [4] such as population size *NP*, scaling factor *F*, and crossover rate *CR*. The optimal settings of these parameters are problem-dependent. A great amount of research work analyzes the effects of these parameters and proposes some suggestion of suitable parameters. The trial-and-error method for tuning the control parameters usually requires tedious optimization trials, so many adaptive or self-adaptive mechanisms have been proposed to update the control parameters during the evolutionary process [5][6][7][8][10][11], with adaptive selection of mutation strategies [9][12][13], and dynamic population size selection [14][15][16][17] to update the control parameters during the evolutionary process. The adaptive and self-adaptive mechanisms outperformed the classical DE algorithms in terms of the rate of convergence for many benchmark problems.

The mechanisms of adaptive parameter control can be categorized into three classes [18] as: deterministic parameter control in which parameters are changed based on deterministic rules such as the time-dependent change of the mutation rates proposed by Holland [19], adaptive parameter control based on learning from evolution process [9], and self-adaptive parameter control in which parameters are associated with individuals and undergo recombination. Adaptive parameter control mechanism updates the parameters according to the historic successful experience [10] or the current population state [20][21]. The successful parameters are believed to direct the individuals towards optima in the next generation. Some success history experience based adaptive parameter control strategies have been proposed.

JADE is a well-known effective DE algorithm which uses a novel mutation strategy called current-*p*best/1 and an external archive for storing the inferior solutions. Each individual has its own parameters *F* and *CR* which are generated by probability distribution of Cauchy and Gaussian based on μ_F and μ_{CR}, respectively. Therein, μ_F and μ_{CR} are updated generation by generation according to the historical accumulation and the *F* and *CR* values of the successful individuals in current generation. Since the excellent performance of JADE, some extension researches on JADE have been proposed. In order to

improve the reliability of JADE, Peng et al. [22] modify the control parameter adaptation strategy of JADE by adding a weighting strategy and apply a "restart with knowledge transfer" strategy to guide the subsequent search. Yang et al. [23] applied JADE to optimize the weight vector in adaptive weighting process for large scale optimization problems. Gong et al. [24] proposed an approach combining strategy adaptation mechanism with JADE for numerical optimization. In order to improve the robustness of JADE, Tanabe et al. [25] proposed Success-History based Adaptive DE (SHADE), an improved version of JADE based on a historical record of successful parameter settings. In each generation, SHADE selects μ_F and μ_{CR} from the memory of a diverse set of parameters randomly and updates the memory according to the successful parameters with a weighting strategy.

JADE updates the control parameters according to all the successful parameters in the last generation, which may include some *poor* parameters. That is, some parameters may be not good themselves, but the individual successes in improving the solution quality, mainly due to other factors rather than parameters. These *poor* parameters, however, may disturb the parameter adaptive control process. The weighting strategy at the ratio of improvement is one way to reduce the noise. Abandoning the *poor* and remaining the *good* is another way. In this paper, we proposed a different approach, Dichotomy-guided adaptive DE (DADE), which is a variant of JADE. DADE divides the successful F and CR into two parts according to the mean μ_F and μ_{CR} of last generation, respectively. Instead of based on all the successful F and CR like in JADE, the update of μ_F and μ_{CR} only related to the part with higher success rate. The population is divided into two parts according to the mean of F and CR, and the success rate is considered as the ratio of the number of individuals successfully entering the next generation to the number of individuals in this part.

The contributions of this work are:

1) Propose the idea of quality discrimination of successful parameters. The *good* parameters are distinguished from the *poor* parameters.

2) Propose a novel dichotomy-guided adaptive approach to select the *good* parameters from all the successful parameter for parameter adaptive process. The *poor* parameters are abandoned.

The rest of the paper is organized as follows. In Section 2, we review the base DE algorithm. Section 3 introduces some adaptive DE algorithms and JADE algorithm, which is the base of our algorithm. Section 4 develops DADE in detail. Experiments are carried out in Section 0 and test results are compared with other algorithms. Finally, conclusions are summarized in Section 6 and future work is highlighted.

2. DIFFERENTIAL EVOLUTION

As a branch of evolutionary algorithm, DE [1], first proposed by Storn and Price, is an effective and efficient global optimization algorithm. Just like genetic algorithm, DE performs mutation, crossover, and selection on population. The initial population $\{x_{i,0} = (x_{1i}, x_{2i}, \ldots, x_{D,i}) | \mathrm{i} = 1, 2, ,\ldots, NP\}$ is randomly generated according to a uniform distribution on the search space as (1), where NP represents the size of population and D is the dimension of problem.

$$x_{j,i} = x_{j,\min} + \mathrm{rand}(0,1) \cdot (x_{j,\max} - x_{j,\min}) \quad (1.)$$

where $x_{j,\min}$ and $x_{j,\max}$ are the predefined lower and upper bounds of the jth dimension, rand(0,1) is a random number in range [0,1]. Then DE generates new population by mutation, crossover, and selection operations repeatedly until reaching the terminal condition. The operations are described below.

Mutation: At each generation g, a mutant vector $v_{i,g}$ is created by mutation operation for each individual $x_{i,g}$. The five frequently used mutation strategies are listed as follows:

DE/rand/1

$$v_{i,g} = x_{r1,g} + F \cdot (x_{r2,g} - x_{r3,g}) \quad (2.)$$

DE/rand/2

$$v_{i,g} = x_{r1,g} + F \cdot (x_{r2,g} - x_{r3,g}) + F \cdot (x_{r4,g} - x_{r5,g}) \quad (3.)$$

DE/best//1

$$v_{i,g} = x_{\mathrm{best},g} + F \cdot (x_{r1,g} - x_{r2,g}) \quad (4.)$$

DE/best/2

$$v_{i,g} = x_{\mathrm{best},g} + F \cdot (x_{r1,g} - x_{r2,g}) + F \cdot (x_{r3,g} - x_{r4,g}) \quad (5.)$$

DE/current-to-best/1

$$v_{i,g} = x_{i,g} + F \cdot (x_{\mathrm{best},g} - x_{i,g}) + F \cdot (x_{r1,g} - x_{r2,g}) \quad (6.)$$

where the indices $r1$, $r2$, $r3$, and $r4$ are distinct random integers selected from [1, 2, …, NP]. The factor F is a positive control parameter for scaling the difference vectors. The base vector $x_{\mathrm{best},g}$ is the individual with best fitness in generation g. The above mutation strategies can be generalized by DE/-/k scheme, where k is a positive integer representing the number of difference vectors. $v_{i,g}$ combines a base vector with k difference vectors multiplied by F.

Crossover: In order to enhance the diversity of the population, a binomial crossover operation forms a trial vector $\mathrm{u}_{i,g} = (u_{1,i,g}, u_{2,i,g}, \ldots, u_{D,i,g})$ as:

$$u_{j,i,g} = \begin{cases} v_{j,i,g}, & \text{if } \mathrm{rand}(0,1) \le CR \text{ or } j = j_{\mathrm{rand}} \\ x_{j,i,g}, & \text{otherwise} \end{cases} \quad (7.)$$

where rand(0,1) is a uniform random number on the interval [0,1], and j_{rand} is a uniform integer selected from [1, 2, …, D], which is used to ensure that trial vector has at least one component different from $x_{i,g}$. The crossover probability $CR \in [0,1]$ is another control parameter corresponding to the average fraction of vector components inherited from trial vector.

Selection: To determine the new individual surviving in the next generation g+1, the trial vector $u_{i,g}$ is compared to the target vector $x_{i,g}$. The vector with better fitness enters the next generation. For example, for a minimization problem, the selected vector is given by:

$$x_{i,g+1} = \begin{cases} u_{i,g}, & \text{if } f(u_{i,g}) \le f(x_{i,g}) \\ x_{i,g}, & \text{otherwise} \end{cases} \quad (8.)$$

where $f(x)$ is the fitness evaluation function.

3. RELATED WORK

This section briefly reviews some adaptive DE algorithms that dynamically update control parameters as the evolutionary search proceeds.

3.1 SaDE

Qin et al. [9] proposed SaDE to simultaneously implement four mutation strategies "DE/rand/1/bin", "DE/rand-to-best/2/bin", "DE/rand/2/bin", and "DE/current-to-rand/1". The probability of applying four mutation strategies to generate trial vector is

adapted based on their success rate in the past LP (a constant) generations. The more suitable mutation strategy for the problem under consideration contributes more at different learning stages. In addition, the scaling factors F are independently generated by normal distribution $N(0.5, 0.3)$ for different individuals in the current population at each generation. It shows that this scheme keeps both the local (with smaller F value) and global search (with larger F value) ability to generate potentially good mutation vectors throughout the evolution process. The crossover probabilities are randomly generated in range [0,1] according to normal distribution with mean *CRm* and standard deviation 0.1 for each 5 generations. The mean *CRm* of different strategies are calculated according to the successful *CR* which are used for the corresponding strategy. At each generation, the *CR* values associated with the trial vectors entering the next generation are recorded. After LP generations, *CRm* is recalculated according to the recorded CR values.

3.2 jDE

Brest et al. [8] proposed a new version of self-adaptive DE, jDE, which uses the classic DE/rand/1/bin strategy. The key idea is that the better values of control parameters lead to better individuals, which are more likely to survive and produce offspring, so that the *good F* and *CR* are propagated to the next generation. The population size is fixed but the control parameters F_i and CR_i of each individual are updated during the optimization. At the beginning, F_i of each individual is set as 0.5 and CR_i as 0.9. Instead of generating F_i and CR_i by normal distributions and updating *CRm* by recorded successful values in SaDE, jDE regenerated F_i and CR_i with probabilities τ_1 by uniform distributions on [0.1,0.9] and [0,1], respectively. The rules for self-adapting control parameters F and CR are quite simple. Experimental results suggest that jDE performs better than DE/rand/1/bin.

3.3 JADE

In this subsection, we describe JADE [10], which is the basis of our algorithm DADE. Its main features are a new mutation strategy current-to-*p*best/1, parameter adaptation, and adoption of external archive.

3.3.1 Currrent-to-pbest/1

Current-to-best/1 strategy uses a greedy mechanism, directing the trial vectors towards the best individual in the population. The greedy strategy leads to the quick convergence to local optimum. Current-to-best/1 strategy performs well on unimodal problems, but poorly on multimodal problems. In view of the fast convergence but less reliable performance of current-to-best strategy, Zhang et al. [10] proposed the current-to-*p*best strategy with *p* greediness as:

$$v_{i,g} = x_{i,g} + F_i \cdot (x_{p\text{best},g} - x_{i,g}) + F_i \cdot (x_{r1,g} - x_{r2,g}) \qquad (9.)$$

where $x_{p\text{best,g}}$ is randomly selected from the top $NP \times p$ ($p \in [0,1]$) individuals in generation g, $r1$ and $r2$ are random integers in range of [1, 2, …, NP] different from i and *p*best. F_i is the F parameter of individual i. Current-to-best is the special case of (9) when p is $1/NP$. For each dimension j, if $v_{j,i,g}$ is outside the bounds of $[x_{j,\min}, x_{j,\max}]$, the component is set to be the middle of the violated bounds and the corresponding component of the target vector as:

$$v_{j,i,g} = (x_{j,i,g} + x_{j,\min})/2, if\ v_{j,i,g} < x_{j,\min}$$
$$v_{j,i,g} = (x_{j,i,g} + x_{j,\max})/2, if\ v_{j,i,g} > x_{j,\max} \qquad (10.)$$

3.3.2 Parameter adaptation

Each individual has its own F_i and CR_i parameters, which are generated by probability distribution as:

$$CR_i = \text{randn}_i(\mu_{CR}, 0.1) \qquad (11.)$$

$$F_i = \text{randc}_i(\mu_F, 0.1) \qquad (12.)$$

where $\text{randn}_i(\mu_{CR},0.1)$, $\text{randc}_i(\mu_F,0.1)$ are the random number generated according to normal distribution of (μ_{CR}, 0.1) and Cauchy distribution of (μ_F,0.1). CR_i is truncated to [0, 1]. If $CR_i > 1$ or < 0, CR_i is set as 1 or 0. F_i is truncated to (0,1]. F_i is truncated as 1 when $F_i > 1$ and regenerated when $F_i \le 0$. Both μ_{CR} and μ_F are initialized to 0.5 at the beginning, and updated at the end of each generation. The F_i and CR_i generated trial vectors entering the next generation successfully at the current generation are recorded in the set S_F and S_{CR} respectively. The uF and uCR are calculated according to the successful F and CR as:

$$\mu_{CR} = (1-c) \cdot \mu_{CR} + c \cdot \text{mean}_A(S_{CR}) \qquad (13.)$$

$$\mu_F = (1-c) \cdot \mu_F + c \cdot \text{mean}_L(S_F) \qquad (14.)$$

where c is a predefined constant between 0 and 1, $\text{mean}_A(S_{CR})$ is the arithmetic mean of CR in S_{CR}, and $\text{mean}_L(S_F)$ is the Lehmer mean as:

$$\text{mean}_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \qquad (15.)$$

3.3.3 External Archive

In order to make use of the promising progress direction information, the inferior solutions $x_{i,g}$ which did not enter the next generation are recorded in an archive A. Define the current population as P. In the current-to-*p*best/1 with archive, $r2$ in Eq.(9) is randomly selected from the union, P ∪ A, of the population and archive. Archive is empty at the beginning, and then the inferior solutions are added into it for each generation. The maximal size of archive is set as NP. If the archive is full, the new member replaces the randomly selected member in archive.

4. DICHOTOMY-GUIDED ADAPTIVE DE

Good F and *CR* guide the individual towards the optimum. JADE updates μ_F and μ_{CR} according to the successful S_F and S_{CR} to move towards *good F* and *CR* values slowly. In order to speed up the parameter update process to get more *good F* and *CR*, we proposed a dichotomy-guided adaptive DE (DADE), a variant of JADE which uses a different parameter adaptation mechanism based on dichotomy-guided. In DADE, the F and CR in S_F and S_{CR} are divide into two parts. In contrast to JADE, which uses all the successful F and CR values to guide parameter adaptation, DADE picks a part of successful parameters with higher success rate to guide control parameter adaptation as search progresses. The individuals of the whole population are divided into two parts according to the mean of F and CR which are used to generate F and CR, the part with higher success rate is selected to update parameter. The success rate is defined as the ratio of the number of individuals successfully entering the next generation to the number of individuals in this part.

ALGORITHM I. PSEUDO CODE OF MEAN OF F AND CR UPDATE IN DADE

Algorithm 1: Update μ_F and μ_{CR} in DADE

c = MINC + (MAXC-MINC)**fes*/*FES*
If $S_{CR}\neq\emptyset$ and $S_F\neq\emptyset$ then
  For i = 1 to $|S_{CR}|$
    If $CR_i \le \mu_{CR}$ then $CR_i \rightarrow L_{SCR}$
    If $CR_i \ge \mu_{CR}$ then $CR_i \rightarrow R_{SCR}$
  For i = 1 to $|S_F|$
    If $F_i \le \mu_F$ then $F_i \rightarrow L_{SF}$
    If $F_i \ge \mu_F$ then $F_i \rightarrow R_{SF}$
  If $\| L_{SCR}|/|L_{PCR}| - |R_{SCR}|/|R_{PCR}| \| > C_{CR}$ then
    If $|L_{SCR}|/|L_{PCR}| \ge |R_{SCR}|/|R_{PCR}|$ then
      $\mu_{CR} = (1-c)\cdot\mu_{CR} + c\cdot \text{mean}_A(L_{SCR})$
    Else
      $\mu_{CR} = (1-c)\cdot\mu_{CR} + c\cdot \text{mean}_A(R_{SCR})$
  Else
    $\mu_{CR} = (1-c)\cdot\mu_{CR} + c\cdot \text{mean}_A(S_{CR})$

  If $\| L_{SF}|/|L_{PF}| - |R_{SF}|/|R_{PF}| \| > C_F$ then
    If $|L_{SF}|/|L_{PF}| \ge |R_{SF}|/|R_{PF}|$ then
      $\mu_F = (1-c)\cdot\mu_F + c\cdot \text{mean}_L(L_{SF})$
    Else
      $\mu_F = (1-c)\cdot\mu_F + c\cdot \text{mean}_L(R_{SF})$
  Else
    $\mu_F = (1-c)\cdot\mu_F + c\cdot \text{mean}_L(S_F)$
Else
  μ_F and μ_{CR} remain unchanged

ALGORITHM II. PSEUDO CODE OF DADE ALGORITHM

Algorithm 2: DADE algorithm

// Initialization phase
g=0; μ_F = 0.5; μ_{CR} = 0.5; $A = \emptyset$
Initialize population $P_0 = \{x_{1,0}, x_{2,0}, \ldots, x_{NP,0}\}$ randomly;
// Main loop
While *the termination criteria are not met* do
  $S_{CR}=\emptyset$, $S_F=\emptyset$, $L_{PF}=\emptyset$, $R_{PF}=\emptyset$, $L_{PCR}=\emptyset$, $R_{PCR}=\emptyset$;
  For i = 1 to *NP* do
    $F_i = \text{randc}_i(\mu_F, 0.1)$;
    $CR_i = \text{randn}_i(\mu_{CR}, 0.1)$;
    If $F_i \le \mu_F$ then $F_i \rightarrow L_{PF}$
    If $F_i \ge \mu_F$ then $F_i \rightarrow R_{PF}$
    If $CR_i \le \mu_{CR}$ then $CR_i \rightarrow L_{PCR}$
    If $CR_i \ge \mu_{CR}$ then $CR_i \rightarrow R_{PCR}$
    Generate trial vector $u_{i,g}$ by current-to-*p*best/1/bin
    If $f(x_{i,g}) \le f(u_{i,g})$ then
      $x_{i,g+1} = x_{i,g}$
    Else
      $x_{i,g+1} = u_{i,g}$, $x_{i,g} \rightarrow A$, $CR_i \rightarrow S_{CR}$, $F_i \rightarrow S_F$
    End if
  End for
  Randomly remove solutions from A so that $|A| \le NP$
  Update μ_F and μ_{CR} (Algorithm 1);
  g ++;
End

4.1 Dichotomy-Guided Parameter Adaption

The key idea of dichotomy guided parameter adaption is to move the mean μ_F and μ_{CR} for F and CR towards the part with higher success rate. In each generation, the control parameters CR_i and F_i used by each individual are generated by Eqs.(10) and (11) according to the procedure described above for JADE in Section 3. As with JADE, S_F and S_{CR} record the successful F and CR in the current generation. All F and CR of the entire population are recorded in P_F and P_{CR}. At the end of each generation, S_F and S_{CR} are divided into two parts by μ_F and μ_{CR}. If F is smaller than μ_F, F is inserted into the L_{SF}. If F is larger than μ_F, F is inserted into R_{SF}. If F is equal to μ_F, F is inserted into both L_{SF} and R_{SF}. Similar to S_F, the S_{CR} are also divided into two parts L_{SCR} and R_{SCR} which record the small and large CR respectively. The same dichotomy is applied on the P_F and P_{CR} of the population, which produces L_{PF} and R_{PF}, L_{PCR} and R_{PCR}, respectively. The update of μ_{CR} and μ_F is as:

$$\mu_{CR} = \begin{cases} (1-c)\cdot\mu_{CR} + c\cdot \text{mean}_A(L_{SCR}), \text{if } |L_{SCR}|/|L_{PCR}| \ge |R_{SCR}|/|R_{PCR}| \\ (1-c)\cdot\mu_{CR} + c\cdot \text{mean}_A(R_{SCR}), \text{otherwise} \end{cases} \quad (16.)$$

$$\mu_F = \begin{cases} (1-c)\cdot\mu_F + c\cdot \text{mean}_L(L_{SF}), \text{if } |L_{SF}|/|L_{PF}| \ge |R_{SF}|/|R_{PF}| \\ (1-c)\cdot\mu_F + c\cdot \text{mean}_L(R_{SF}), \text{otherwise} \end{cases} \quad (17.)$$

$$c = \text{MINC} + (\text{MAXC} - \text{MINC}) * fes / \text{FES} \quad (18.)$$

where MINC and MAXC are constant, which are set as 0.01 and 0.1 in this paper. FES is the max function evaluation number throughout the searching process, and *fes* is the current function evaluation times used at the current time. c is a parameter in [0.01,0.1], increases linearly with increasing of function evaluation number. $|L_{SCR}|$ is the number of elements in L_{SCR}, $\text{mean}_A(L_{SCR})$ is the usual arithmetic mean, mean_L is the Lehmer mean as Eq.(14). Both μ_F and μ_{CR} are initialized to 0.5 at the beginning. The pseudo code of update process is described in Algorithm I. If the success rate of two part of L_{SF} and R_{SF} is near, the selection of anyone part may cause the misleading of F and CR. So in order to lessen the risk of misleading, we add a threshold C_F and C_{CR} to control the selection. Only when the difference of success rate between left and right is larger than the threshold, Eqs.(16) and (17) are carried out to calculate μ_F and μ_{CR}. Otherwise, all the F and CR in S_F and S_{CR} are used to calculate μ_F and μ_{CR} as Eqs.(13) and (14).

4.2 Overall Algorithm

The overall DADE algorithm is shown in Algorithm II. The procedure of DADE is the same as JADE except for the adaptation of parameters. DADE adopts the current-to-*p*best/1 strategy. The individuals of the population are generated randomly in the searching space. After the initialization, in each generation, F_i of individual i is generated by Cauchy distribution of mean μ_F and CR_i is generated by normal distribution of mean μ_{CR}. Both μ_F and μ_{CR} are initialized to 0.5 at the beginning. For each individual, trial vector is generated by crossover and mutation operations. Then the vector with better fitness value between trial and target vectors is selected and enters the next generation. At the end of each generation, the μ_F and μ_{CR} are updated according to the dichotomy approach described in Algorithm I. The parameters are divided into two parts and the part with higher success rate is selected to calculate μ_F and μ_{CR}. DADE regenerates new population repeatedly until reaching the terminal conditions. The population size is fixed throughout the evolutionary process.

5. EXPERIMENTS AND COMPARISIONS

5.1 Experimental Settings

In this section, DADE is applied to minimize a set of 42 scalable benchmark functions of dimensions D = 30 chosen from [26][27][28] and [29], as shown in Table I. The benchmark functions of f_1-f_{12} have an optimal value f^*=0, which have different characteristics. f_1-f_4 are continuous unimodal functions. f_5 is the Rosenbrock function which is unimodal functions for 2 dimension but multimodal for high dimension. f_6 is a discontinuous step function, and f_7 is a noisy quartic function. f_8 is separable multimodal functions. f_9-f_{12} are nonseparable multimodal functions. The last 30 functions are taken from the CEC 2014 competition [29]. F_1-F_3 are rotated unimodal function, F_4-F_{16} are simple multimodal functions, F_{17}-F_{22} are

TABLE I. TEST FUNCTIONS OF DIMENSION *D*

Name	Test functions	Initial range
Sphere	$f_1(x)=\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$
Schewefel 2.22	$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$[-10,10]^D$
Schewefel 1.2	$f_3(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	$[-100,100]^D$
Schewefel 2.21	$f_4=\max_i\{\lvert x_i\rvert\}$	$[-100,100]^D$
Rosenbrock	$f_5(x)=\sum_{i=1}^{D-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	$[-30,30]^D$
Step	$f_6(x)=\sum_{i=1}^{D}\lfloor x_i+0.5\rfloor^2$	$[-100,100]^D$
Noisy Quaric	$f_7(x)=\sum_{i=1}^{D} ix_i^4+rand[0,1)$	$[-1.28,1.28]^D$
Rastrigin	$f_8(x)=\sum_{i=1}^{D}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	$[-5.12,5.12]^D$
Ackley	$f_9=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))+20+e$	$[-32,32]^D$
Griewank	$f_{10}(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$	$[-600,600]^D$
Penalized	$f_{11}(x)=\frac{\pi}{D}\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)^2\}+\sum_{i=1}^{D}u(x_i,10,100,4)$ where $y_i=1+\frac{1}{4}(x_i+1)$ and $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a\le x_i\le a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	$[-50,50]^D$
	$f_{12}=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]$ $+(x_D-1)^2[1+\sin^2(2\pi x_D)]\}+\sum_{i=1}^{D}u(x_{i,}5,100,4)$	$[-50,50]^D$
Shifted and Rotated, hybrid , composition functions	F_1-F_{30}: The 30 functions from the IEEE CEC 2014 special session and competition [29]	$[-100,100]^D$

hybrid functions, and F_{23}-F_{30} are composition functions. Obviously, the last 30 functions are much more complex.

DADE is compared with two other adaptive DE algorithms SaDE and jDE, the classic DE/rand/1/bin. For fair comparison, we set the parameters of DADE to be fixed, p = 0.05 and MINC=0.01, MAXC=0.1, C_F=0.3 and C_{CR}=0.15 in all simulation. We follow the parameter settings in the original paper of JADE [10], SaDE [9], jDE [8]. The parameters of DE/rand/1/bin are set to F = 0.5 and CR = 0.9.

In all simulations, we set the population size NP to be 100 for 30D problems. All algorithms runs in 50 times for each function. The results reported are the mean of 50 runs. For clarity, the results of the best and second best algorithms are marked in **boldface** and *italic*, respectively. In addition, we use Wilcoxon's rank sum test at α = 0.05 to evaluate the statistical significance of the results. The ≈,+, - indicate whether a given algorithm performed significantly not different (≈), better(+) or worse(-) compared to DADE according the Wilcoxon rank-sum test.

5.2 Experimental Results

5.2.1 Classic Functions: f_1-f_{12}

The mean and standard deviation of the results obtained by each algorithm for f_1-f_{12} are summarized in Table II. These statistics are calculated at the end of the optimization. As shown in Table II, DADE performs the best or the second best in all functions. For f_1 and f_2, DADE performs best compared with JADE, jDE, SaDE, and DE. Compare with jDE, DADE performs better on 10 functions except for f_6 and f_9. DADE obtains better results on 7 functions than SaDE, while the results of the other functions are not significantly different. For DE, DADE performs better on 9 functions except for 3 functions which are not significantly different. So, we can conclude that DADE performs better than jDE, SaDE, and DE. Compared with JADE, DADE performs better on functions f_1 and f_2 but worse on 3 functions f_3, f_4, and f_7. DADE takes only part of the successful F and CR values to update the mean μ_F and μ_{CR} while JADE use all successful F and CR. DADE performs better than JADE on f_1 and f_2, which

TABLE II. EXPERIMENTAL RESULTS OF 30-DIMENSIONAL PROBLEMS F_1-F_{12}, AVERAGED OVER 50 INDEPENDENT RUNS

≈,+, - indicates whether a given algorithm performed significantly not different, better or worse compared to DADE according the Wilcoxon rank-sum test

fun	Gen	DADE		JADE		jDE		SaDE		DE/rand/1/bin	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
f1	1500	**1.81E-77**	4.34E-77	*2.58E-59(-)*	1.15E-58	1.26E-28(-)	1.25E-28	5.93E-38(-)	1.13E-37	7.85E-14(-)	9.31E-14
f2	2000	**4.49E-50**	1.18E-49	2.46E-20(-)	1.68E-19	9.02E-24(-)	6.60E-24	5.50E-25(-)	1.23E-24	1.16E-09(-)	7.86E-10
f3	5000	*6.02E-72*	1.29E-71	**2.73E-86(+)**	1.15E-85	8.31E-14(-)	1.35E-13	1.49E-44(-)	1.05E-43	5.54E-11(-)	8.40E-11
f4	5000	*7.73E-56*	3.72E-55	**2.05E-65(+)**	5.16E-65	7.99E-01(-)	9.55E-01	2.44E-19(-)	5.30E-19	5.28E-01(-)	9.63E-01
f5	20000	*1.60E-30*	5.12E-30	7.97E-02(≈)	5.64E-01	1.59E-01(-)	7.89E-01	3.99E-01(-)	1.21E+00	**3.47E-31(≈)**	2.45E-30
f6	1500	**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
f7	3000	*7.56E-04*	2.64E-04	**6.97E-04(+)**	3.68E-04	3.41E-03(-)	8.16E-04	1.88E-03(-)	6.92E-04	4.60E-03(-)	1.23E-03
f8	5000	**0.00E+00**	0.00E+00	**0.00E+00(≈)**	0.00E+00	**0.00E+00(≈)**	0.00E+00	**0.00E+00(≈)**	0.00E+00	6.75E+01(-)	3.22E+01
f9	2000	**3.41E-15**	0.00E+00	**3.41E-15(≈)**	0.00E+00	*7.89E-15(-)*	1.73E-15	**3.41E-15(≈)**	0.00E+00	9.26E-08(-)	3.92E-08
f10	3000	*7.23E-21*	1.87E-20	1.48E-04(≈)	0.00E+00	**1.08E-21(-)**	7.67E-21	2.96E-04(-)	1.46E-03	1.97E-04(≈)	1.39E-03
f11	1500	**1.57E-32**	2.78E-48	**1.57E-32(≈)**	1.66E-47	8.77E-30(-)	1.20E-29	**1.57E-32(≈)**	1.66E-47	7.17E-15(-)	5.56E-15
f12	1500	**1.35E-32**	5.57E-48	**1.35E-32(≈)**	8.29E-48	8.47E-29(-)	1.07E-28	**1.35E-32(≈)**	8.29E-48	4.38E-14(-)	3.46E-14
+(better than DADE)				2		0		0		0	
-(worse than DADE)				3		10		7		9	
≈(no sig.)				7		2		5		3	

TABLE III. SUCCESS RATE AND SEARCH SPEED COMPARISONS ON 30-DIMENSIONAL PROBLEMS $f1$-$f12$

fun	Acceptable accuracy		DADE	JADE	jDE	SaDE
f1	1E-6	SR	**100**	**100**	**100**	**100**
		FEs	**22503**	25580	49996	33106
		Rank	1	2	4	3
f2	1E-6	SR	**100**	**100**	**100**	**100**
		FEs	**35266**	44078	65232	52438
		Rank	1	2	4	3
f3	1E-6	SR	**100**	**100**	**100**	**100**
		FEs	100036	**62324**	289720	111152
		Rank	2	**1**	4	3
f4	1E-6	SR	**100**	**100**	0	**100**
		FEs	73893	**61670**	N/A	172498
		Rank	2	**1**	4	3
f5	1E-6	SR	**100**	98	96	90
		FEs	143366	**104438**	536256	199920
		Rank	**1**	2	3	4
f6	0	SR	**100**	**100**	**100**	**100**
		FEs	**10733**	11588	21942	14722
		Rank	**1**	2	4	3
f7	1E-2	SR	**100**	**100**	**100**	**100**
		FEs	29733	**28744**	103474	53178
		Rank	2	**1**	4	3
f8	1E-6	SR	**100**	**100**	**100**	**100**
		FEs	147996	116920	**102216**	132116
		Rank	4	2	**1**	3
f9	1E-6	SR	**100**	**100**	**100**	**100**
		FEs	**32693**	37716	73174	48982
		Rank	**1**	2	4	3
f10	1E-6	SR	**100**	96	**100**	96
		FEs	**24596**	29172	53458	36431
		Rank	**1**	3	2	4
f11	1E-6	SR	**100**	**100**	**100**	**100**
		FEs	**20543**	24012	44762	29700
		Rank	**1**	2	4	3
f12	1E-6	SR	**100**	**100**	**100**	**100**
		FEs	**22520**	25964	48746	31876
		Rank	**1**	2	4	3
Avg_Rank			**1.38**	1.69	3.23	2.92

shows that some F and CR disturb the parameter adaptive control.

In Table III, we summarize the success rate (SR) of each algorithm and the average number of function evaluations over successful runs (FEs). An experiment is considered as successful if the best solution is found with acceptable accuracy. FEs and SR are useful to compare the convergence rate (in successful runs) and reliability of different algorithms. The ranks in the table are evaluated based on the descending order of the success rates and the ascending order of FEs. From Table III, we find that all the algorithm perform high success rate on most functions. In particular, DADE 100% successfully finds the acceptable solutions on all functions. Only DADE can find achieve the near-global optimum for f_5, while JADE, jDE, and SaDE sometimes fails to obtain the optimum. From the aspect of search speed, DADE and JADE uses a smaller FEs to reach the acceptable solution than jDE and SaDE. DADE ranks the first and is the fastest in most of the functions. For all the multimodal nonseprable functions f_5 and f_9-f_{12}, DADE ranks the first and performs the best. Overall, DADE performs the best on most functions in f_1-f_{12}. The results show that the dichotomy guided parameter adaptive approach speeds up the search process, and move to *good* F and CR quickly.

5.2.2 IEEE CEC 2014 Competition: F_1-F_{30}

Table IV compares on the five algorithms on IEEE CEC 2014 competition function set. All the functions are shifted, rotated, hybrid, or composition functions. These functions are difficult for most of the optimization algorithms. It can be seen that no algorithm is able to obtain a near-global optimum on all functions. DADE obtains better results on more functions. DADE performs better than JADE, jDE, SaDE, and DE on 10, 20, 20, and 20 functions, respectively. Conversely, JADE, jDE, SaDE, and DE perform better than DADE on 5, 4, 7, and 8 functions, respectively. Overall, DADE performs the best. For simple functions F_1-F_{16}, DADE performs better on 6 but worse than JADE on 3 functions. And DADE performs better than JADE on hybrid functions F_{17}-F_{22} especially for F_{18}, F_{20}, and F_{21}. For compose functions, DADE performs worse than JADE on F_{28} and F_{29} but better on F_{26} and an equivalent effect on other functions. DADE outperforms JADE on simple and hybrid functions, but not on composition functions on more occasions.

6. CONCLUSION

This paper proposed DADE, a dichotomy-guided based adaptive DE which extends JADE. DADE takes only higher success rate part of the successful F and CR to update the mean μ_F and μ_{CR} to get more *good* parameters. DADE was shown to outperform JADE, jDE, SaDE, and DE/rand/1/bin on most cases

TABLE IV. EXPERIMENTAL RESULTS OF 30-DIMENSIONAL PROBLEMS F_1-F_{30}, AVERAGED OVER 50 INDEPENDENT RUNS

fun	DADE		JADE		jDE		SaDE		DE/rand/1/bin	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	**2197.59**	2373.15	*2334.58(-)*	1417.61	83957.9(-)	86118.3	87778.9(-)	64379.6	82601.1(-)	92570
F2	1.23E-14	1.43E-14	1.95E-14(≈)	1.33E-14	3.41E-15(+)	9.33E-15	**0(+)**	0	*3.41E-15(+)*	9.33E-15
F3	6.63E-14	2.15E-14	0.000369(-)	0.001803	1.82E-14(+)	2.68E-14	**0(+)**	0	*7.96E-15(+)*	1.99E-14
F4	*2.01E-13*	4.55E-13	**7.47E-14(+)**	2.90E-14	11.8499(-)	23.221	0.945987(-)	5.20477	1.54154(-)	8.92808
F5	*20.309*	0.0448103	**20.2896(+)**	0.030424	20.3635(-)	0.0347005	20.8342(-)	0.0584629	20.913(-)	0.0492376
F6	**0.520859**	1.25853	9.51302(-)	2.11086	8.97849(-)	5.17562	*4.61054(-)*	7.50528	4.78853(-)	2.42099
F7	4.17E-14	5.57E-14	*2.01E-14(≈)*	4.38E-14	9.09E-14(-)	4.59E-14	**0(+)**	0	0.00014 (≈)	0.00104596
F8	**0**	0	**0(≈)**	0	0.0198992(≈)	0.140708	1.08613(-)	1.78133	127.885(-)	23.7552
F9	**18.9327**	3.13409	26.9342(-)	4.59556	44.032(-)	6.77148	120.631(-)	8.98442	178.66(-)	11.0678
F10	**0.00902167**	0.011832	0.009797**(≈)**	0.012034	3.60701(-)	3.54782	395.372(-)	78.1793	3949.23(-)	718.258
F11	**1571.45**	235.326	1656.61(≈)	209.396	2809.69(-)	301.751	5864.01(-)	331.763	6790.67(-)	295.158
F12	*0.29049*	0.0438903	**0.258242(+)**	0.04569	0.497145(-)	0.0705661	1.7974(-)	0.232437	2.3848(-)	0.248515
F13	**0.168534**	0.0232873	*0.21185(-)*	0.03296	0.288649(-)	0.0386808	0.293576(-)	0.0368541	0.36334(-)	0.0380017
F14	**0.226569**	0.034456	*0.236031(≈)*	0.032342	0.305864(-)	0.0385814	0.269278(-)	0.0296651	0.267298(-)	0.0295258
F15	*3.1928*	0.442603	**3.06125(≈)**	0.415325	5.79822(-)	0.564672	11.6375(-)	0.969501	15.5367(-)	1.0396
F16	**9.16391**	0.384313	*9.33009(-)*	0.432232	10.3873(-)	0.345541	12.3589(-)	0.200787	12.6395(-)	0.261038
F17	1240.11	362.124	*1196.01(≈)*	331.88	1530.1(≈)	981.534	**1043.72(+)**	314.034	1474(-)	168.889
F18	80.8564	30.2393	147.674(-)	453.766	**18.4001(+)**	9.85452	90.5753(-)	25.3154	*53.0875(+)*	7.21286
F19	**4.54948**	0.975606	*4.62783(≈)*	0.788838	5.4887(-)	0.715036	5.75106(-)	0.509561	5.03464(-)	0.6568
F20	*22.9696*	77.2491	2793.38(-)	2444.79	**12.3273(-)**	3.56416	37.1735(-)	7.75455	32.9986(-)	7.49414
F21	**294.969**	133.987	15951.5(-)	59467.3	*303.584*(≈)	198.642	786.735(-)	200.888	683.399(-)	150.878
F22	144.494	56.3702	143.033(≈)	61.9006	142.043(≈)	58.9894	*122.325(+)*	45.0506	**75.1894(+)**	68.9857
F23	**315.244**	5.78E-14	**315.244**(≈)	4.02E-13	**315.244**(≈)	4.16E-13	**315.244**(≈)	3.60E-13	**315.244**(≈)	4.02E-13
F24	224.703	2.35238	224.913(≈)	2.15984	225.343(-)	1.85919	*223.865(≈)*	0.902074	**216.895(+)**	10.3466
F25	204.257	1.12724	203.823(≈)	1.10005	203.369(+)	0.646241	*203.275(+)*	1.13723	**202.634(+)**	0.119261
F26	**100.168**	0.027169	102.171(-)	13.9746	100.29(-)	0.0461031	100.294(-)	0.0372848	100.342(-)	0.041282
F27	*327.274*	45.8999	336.618(≈)	46.5475	391.229(-)	24.2412	**321.56(+)**	37.4329	351.65(-)	56.7135
F28	*817.208*	49.724	**786.422(+)**	42.283	825.959(≈)	19.222	835.533(-)	32.5592	823.013(-)	25.3954
F29	**733.661**	16.7342	175687(+)	1.25E+06	830.088(-)	78.4253	760.115(-)	38.1916	699189(+)	2.39E+06
F30	*1663.12*	619.812	1682.65(≈)	699.958	2528.91(-)	921.983	1774.41(≈)	921.505	**1408.09(+)**	751.981
+(better than DADE)			5		4		7		8	
-(worse than DADE)			10		20		20		20	
≈			15		6		3		2	

≈,+, - indicates whether a given algorithm performed significantly not different (≈), better(+) or worse(-) compared to DADE according the Wilcoxon rank-sum test.

of 12 benchmark functions and 30 complex functions of CEC 2014 competition. Although DADE may find the near solution as JADE, DADE uses a smaller function evaluation times than JADE. DADE runs faster on most functions. The dichotomy guided approach moves the parameter towards the part with higher success rate, and speeds up the searching process. The experiment results show that the idea of taking part of successful *F* and *CR* for parameter control is reasonable. Only taking the *good* parameters for the parameter update process, the algorithm speeds up. There are some *poor F* and *CR* of successful individual which may disturb parameter control.

Further research work includes 1) analyzing the parameter control process further; 2) finding other methods to pick the *good F* and *CR* for parameter adaptive control.

7. ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundations of China (NSFC) with No. 61402545, the Natural Science Foundations of Guangdong Province for Distinguished Young Scholars with No. 2014A030306038, the Project for Pearl River New Star in Science and Technology, Guangzhou, China, the Fundamental Research Funds for the Central Universities, the NSFC Key Program with No. 61332002, the NSFC for Distinguished Young Scholars with No. 61125205, and the National High-Technology Research and Development Program (863 Program) of China No.2013AA01A212.

8. 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] Y. L. Li, Z. H. Zhan, Y. J. Gong, W. N. Chen, J. Zhang, and Y. Li, "Differential evolution with an evolution path: A DEEP evolutionary algorithm," *IEEE Trans. on Cybernetics*, DOI:10.1109/TCYB.2014.2360752, 2014.

[3] Z. H. Zhan and J. Zhang, "Enhance differential evolution with random walk," in *Proc. Genetic Evol. Comput. Conf.*, Philadelphia, America, Jul. 2012, pp. 1513-1514.

[4] S. Das and P. N. Suganthan, "Differential Evolution: A Survey of the State-of-the-Art," *IEEE Tran. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, 2011.

[5] M. M. Ali and A. Törn, "Population set based global optimization algorithms: Some modifications and numerical studies," *Comput. Oper. Res.*, vol. 31, no. 10, pp. 1703–1725, 2004.

[6] M. G. H. Omran, A. Salman, and A. P. Engelbrecht, "Self-adaptive differential evolution," *in Proc. Comput. Intell. Security*, Lecture Notes in Artificial Intelligence 3801. 2005, pp. 192–199

[7] Z. H. Zhan and J. Zhang, "Self-adaptive differential evolution based on PSO learning strategy," in *Proc. Genetic Evol. Comput. Conf.*,Portland, America, Jul., 2010, pp. 39-46..

[8] 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 Tran. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, 2006.

[9] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization," *IEEE Tran. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, 2009.

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

[11] S. M. Islam, S. Das, S. Ghosh, S. Roy, 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.

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

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

[14] J. Teo, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Comput. A Fusion Found. Methodol. Applicat.*, vol. 10, no. 8, pp. 673–686, 2006.

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

[16] Z. H. Zhan and J. Zhang, "Co-evolutionary differential evolution with dynamic population size and adaptive migration strategy," in *Proc. Genetic Evol. Comput. Conf.*.,Dublin, Ireland, Jul., 2011, pp. 211-212.9.

[17] R. A. Sarker, S. M. Elsayed, T. Ray, "Differential evolution with dynamic parameters selection for optimization problems" *IEEE Tran. Evol. Comput.*, vol. 18, no. 5, pp. 689-707, Oct. 2014.

[18] A. E. Eiben and J. E. Smith, Introduction to Evolutionary Computing. Berlin, Germany: Springer, 2003.

[19] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: The University of Michigan Press, 1975.

[20] Z. H. Zhan, J. Zhang, Y. Li, and H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics--Part B*, vol. 39, no. 6, pp. 1362-1381, Dec. 2009.

[21] W. J. Yu, M. Shen, W. N. Chen, Z. H. Zhan, Y. J. Gong, Y. Lin, O. Liu, J. Zhang, "Differential Evolution With Two-Level Parameter Adaptation", *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1080-1099, Jul. 2014.

[22] F. Peng, K. Tang, G. Chen, and X. Yao, "Multi-start JADE with knowledge transfer for numerical optimization," in *IEEE CEC*, 2009, pp. 1889–1895.

[23] Z. Yang, J. Zhang, K. Tang, X. Yao, and A. C. Sanderson, "An adaptive coevolutionary differential evolution algorithm for large-scale optimization," in *Proc. IEEE CEC*, 2009, pp. 102–109.

[24] W. Gong, Z. Cai, C. X. Ling, and H. Li, "Enhanced differential evolution with adaptive strategies for numerical optimization," *IEEE Trans. On Systems, Man, and Cybernetics, Part B*, vol. 41, no. 2, pp. 397–413, 2011.

[25] R. Tanabe and A. Fukunaga, "Success-History Based Parameter Adaptation for Differential Evolution," in *IEEE CEC*, 2013, pp. 71–78.

[26] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. on Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[27] Z. H. Zhan, J. Zhang, Y. Li, and Y. H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 6, pp. 832-847, Dec. 2011.

[28] Y. H. Li, Z. H. Zhan, S. J. Lin, J. Zhang, and X. N. Luo, "Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems," *Information Sciences*, vol. 293, no. 1, pp. 370-382, 2015.

[29] J. J. Liang, B. Y. Qu, P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2014.