Cluster Comput (2017) 20:1247–1257
DOI 10.1007/s10586-017-0793-8

Ant colony optimization with different crossover schemes for global optimization

Zhiqiang Chen[1,2,3] · **Rong-Long Wang**[4]

Received: 1 December 2016 / Revised: 9 February 2017 / Accepted: 13 February 2017 / Published online: 23 February 2017

Abstract Global optimization, especially large scale optimization problems arise as a very interesting field of research, because they appear in many real-world problems. Ant colony optimization is one of optimization techniques for these problems. In this paper, we improve the continuous ant colony optimization (ACO_R) with crossover operator. Three crossover methods are employed to generate some new probability density function set of ACO_R. The proposed algorithms are evaluated by using 21 benchmark functions whose dimensionality is 30–1000. The simulation results show that the proposed ACO_R with different crossover operators significantly enhance the performance of ACO_R for global optimization. In the case the dimensionality is 1000, the proposed algorithm also can efficiently solves them. Compared with state-of-art algorithms, the proposal is a very competitive optimization algorithm for global optimization problems.

✉ Zhiqiang Chen
wilber_chen@hotmail.com

Rong-Long Wang
wang@u-fukui.ac.jp

1 National Research Base of Intelligent Manufacturing Service, Chongqing Technology and Business University, Chongqing, China

2 Chongqing Engineering Laboratory for Detection Control and Integrated System, Chongqing Technology and Business University, Chongqing, China

3 School of Computer Science and Information Engineering, Chongqing Technology and Business University, Chongqing, China

4 The Faculty of Engineering, University of Fukui, Fukui-shi, Japan

Keywords Ant colony optimization · Large scale · Continuous optimization problem · Crossover operator

1 Introduction

Many real-world application problems in engineering, science and technology can be formulated as continuous optimization problems (CnOPs) [1–5]. Meta heuristics (e.g., simulated annealing (SA), evolutionary algorithms (EAs), differential evolution (DE), particle swarm optimization (PSO), ant colony optimization (ACO), estimation of distribution algorithms (EDA), etc.) are a family of optimization techniques that have seen increasingly rapid development and have been applied to CnOPs over the past few years [6]. Although these approaches have shown excellent search abilities when applying to some 30–100 dimensional problems, many of them suffer from the "curse of dimensionality", which implies that their performance deteriorates quickly as the dimensionality of search space increases [6]. Complexity of the problem usually increases with the size of problem and the solution space of the problem increases exponentially with the problem size. Thus more efficient search strategies are required to solve the large scale CnOPs. Zhang et al. proposed a specially tailored EA based on a decision variable clustering method [7] and an approximate non-dominated sorting for evolutionary many-objective optimization [8]. Historically, scaling EAs to large size problems have attracted much interest, including both theoretical and practical studies. The earliest practical approach might be the parallelism of an existing EA. Later, cooperative coevolution appears to be another promising method.

Ant colony optimization is inspired by the ants' foraging behavior and it was first applied to solve discrete optimization problems [4,9,10]. Socha and Dorigo [5] applied ACO

for continuous domains, called ACO_R. The fundamental idea underlying ACO for the continuous domains is the shift from using a discrete probability distribution to using a continuous one, that is, a probability density functions (PDFs). It uses a solution archive as a form of pheromone model for the derivation of a probability distribution over the search space. However, ACO_R concentrates mainly on the small-scale CnOPs, and for the larger CnOPs or multi-modal CnOPs, the results obtained by ACO_R are far from being competitive with the results obtained by the other algorithm.

In this paper, to solve efficiently larger CnOPs, several variants of ant colony optimization (ACO_R) with crossover operations are suggested, which is to keep a proper trade-off mechanism between diversification and intensification. It is also known well that hybridization of EAs with other techniques can greatly improve the efficiency of search [7]. In the proposed algorithm, the operation similar to the crossover in the GA is introduced to generate some new PDF set in the promising space, which is aim at balancing the diversification and intensification. As a result, the pheromone information is enhanced in the promising space, and the global optima can be found efficiently. Additionally, the crossover operation helps the ant colony exploit the correlation information among the design variables.

The proposed algorithms are evaluated by using 21 test functions whose dimension is 30–1000. We compare the results with other continuous optimization methods in the literature. The results show the used crossover operators improve efficiently the performance of the ACO_R and perform better than the compared algorithms.

2 ACO with crossover operators for continuous optimization

2.1 ACO_R

One of the first attempts to apply an ant-related algorithm to the CnOPs is continuous ACO (CACO) [11]. In the CACO, the notion of the nest is introduced, but the CACO does not perform an incremental construction of solutions, which is one of the main characteristic of the ACO metaheuristic. Another ant-related approach to the CnOPs is the API algorithm [12], in which the ants perform their search independently, but starting from the same nest. The third ant-based approach to the CnOPs is continuous interacting ant colony (CIAC) [13]. Other several ant-inspired algorithms for CnOPs were proposed [14–16]. However, as explained in Socha and Dorigo [5], most of these algorithms use search mechanisms different from those used in the ACO metaheuristic [15]. The first algorithm that can be classified as an ACO algorithm for continuous domains is ACO_R [5]. In ACO_R, the discrete probability distributions used in the solution construction by ACO algorithms for combinatorial optimization are substituted by PDFs. ACO_R uses a solution archive for the derivation of these PDFs over the search space. Additionally, ACO_R uses sums of weighted Gaussian functions to generate multimodal PDFs.

ACO_R initializes the solution archive with k solutions that are generated uniformly at random. Each solution is a D-dimensional vector with real-valued components $x_i \in [x_{\min}, x_{\max}]$, with $i = 1, 2, \ldots, D$.

In this paper, we assume that the optimization problems are unconstrained except possibly for bound constraints of the D real-valued variables x_i. The k solutions of the archive are kept sorted according to their quality (from best to worst) and each solution S_j has associated a weight ω_j. This weight ω_i is calculated using a Gaussian function as [15]:

$$\omega_j = \frac{1}{qk\sqrt{2\pi}} e^{\frac{-(rank(j)-1)^2}{2(qk)^2}} \tag{1}$$

where $rank(j)$ is the rank of solution S_j in the sorted archive, and q is a parameter of the algorithm. By computing $rank(j)$, the best solution receives the highest weight. The weights are used to choose probabilistically a guiding solution around which a new candidate solution is generated. The probability of choosing solution S_j as guiding solution is given by Eq. (2) [17]:

$$p_j = \frac{w_j}{\sum_{r=1}^{k} w_r} \tag{2}$$

So that the better the solution, the higher are the chances of choosing it. Once a guiding solution S_{guide} is chosen, the algorithm samples the neighborhood of the i-th real-valued component of the guiding solution s^j_{guide}using a Gaussian PDF with $s^j_{guide} = \mu^j_{guide}$, and σ^j_{guide} equal to

$$\sigma^j_{guide} = \xi \sum_{r=1}^{k} \frac{\left| S^i_r - S^i_{guide} \right|}{k-1} \tag{3}$$

which is the average distance between the value of the i-th component of S^i_{guide} and the values of the i-th components of the other solutions in the archive, multiplied by a parameterx_i [17]. The process of choosing a guiding solution and generating a candidate solution is repeated a total of N_a times (corresponding to the number of "ants") per iteration. Before the next iteration, the algorithm updates the solution archive keeping only the best k of the $k + N_a$ solutions that are available after the solution construction process.

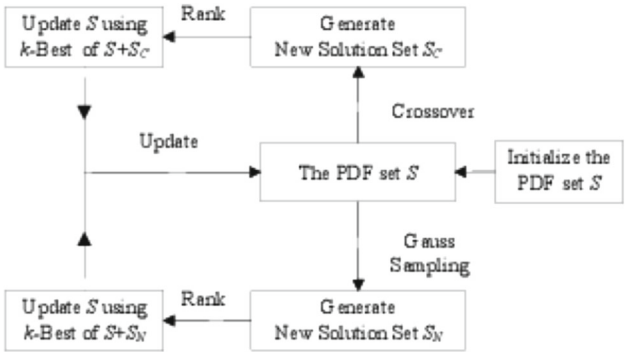

Fig. 1 Updating procedure of PDFs of the $COACO_R$

2.2 The scheme of ACO with crossover operators

Establishing a balance between exploration and exploitation for global search algorithm is important. Some crossover operators could establish an adequate balance between exploration and exploitation, and generate distribution in the exploration and exploitation zones in the correct proportion. For this reason, the crossover is introduced into the ACO. Figure 1 presents the outline of ACO_R with crossover operator $COACO_R$. As shown in Fig. 1, some of PDFs are generated by the crossover operations, and some of the other is from the original way of ACO_R. In the proposed algorithm, the crossover operation in genetic algorithm is employed to improve the search ability of the ant colony by enhancing the pheromone distribution in the promising searching space. A parameter n_{co}, which is the number of the crossover operations, denotes the degree of the crossover operation. As a result, the set of the PDFs become more diversiform and effective, and the ants could find the good solutions efficiently under the improved PDF set.

As for the procedure of the $COACO_R$, initially, the PDF set of the ant colony is filled with randomly generated Gaussian functions. The algorithm iteratively updates the PDF set. The iteration includes two phases, in the first phase, the solutions are constructed according to the PDFs, and in the second phase, the PDF set is updated. In the proposed algorithm, the PDF set consists of not only the PDFs generated by the original ACO_R but also the PDFs generated by the crossover operation. Firstly, m new solutions are built based on the PDF set by each ant independently. Secondly, k PDF vectors are generated by the best k solutions directly. Meanwhile, new $2n_{co}$ PDF vectors are generated by the crossover operations on the newly generated PDFs. After both the two kinds of PDFs were built, the updating of the PDF set is accomplished by adding the new $2n_{co}$ PDF vectors to the PDF set and removing the same number of worst PDF vectors. The outline of the proposed algorithm is shown in Fig. 2.

Algorithm:
Input Parameters: k, m, n_{co}, D, ξ, $\cdots$
Initialize PDFs: $S = S_1, S_2, \cdots, S_k$
$L = (x_{max} - x_{min}))/(2*(m+k))$,
for j = 1 : k
for i = 1 : D
S_j^i = normrnd($x_{min} + 2*i*L$), L),
end
end
while (termination criterion is not satisfied)
//Generate m new solutions $P = (P_1, P_2, \cdots, P_m)$
for j=1 : m
for i = 1 : D
Select guiding solution S^i_{guide} according to weights,
P_j^i = Gaussian Sampling Based on Eq (1),
end
Store and evaluate newly generated solution P_j,
end
Update PDFs S with the best k solutions of $S+P$
// Generate $2n_{co}$ solutions $Z = (Z_1, Z_2, \ldots, Z_{2nco})$
for co=1: n_{co}
Crossovers are performed among selected PDFs
end
Update PDFs S using Z replace the worst $2n_{co}$ solutions of S,
end

Fig. 2 Pseudo code of the $COACO_R$

3 Crossover operators

As mentioned above, crossover operator plays an important role to improve the performance of ACO_R. Three crossover methods (BLX-α [18], UNDX [19] and PNX [20]) are employed for $COACO_R$, which are usually used in genetic algorithm. Based different crossover methods, the presenting $COACO_R$ are named as follows: ACO_{BLX}, ACO_{UNDX}, and ACO_{PNX}, respectively.

3.1 BLX-α

Blend crossover (BLX-α) is a well-known crossover operator proposed by Eshelman and Schaffer [18]. For two parent solutions:

$$
\begin{aligned}
&P_1 = (P_{1,1}, P_{1,2}, \ldots, P_{1,n}),\\
&P_2 = (P_{2,1}, P_{2,2}, \ldots, P_{2,n}),\\
&\min_i = \min\{P_{1,i}, P_{2,i}\},\\
&\max_i = \max\{P_{1,i}, P_{2,i}\}, i = 1, 2, \ldots, n,\\
&I = \max_i - \min_i .
\end{aligned}
$$

the BLX-α randomly picks a solution in the range $[\min_i - \alpha I, \max_i + \alpha I]$. Thus, if u is a random number between 0 and 1, an offspring $\mathbf{C} = (C_1, C_2, \ldots, C_n)$ is created as follows:

$$C_i = (1 - \gamma) P_{1,i} + \gamma P_{2,i} \tag{4}$$

where $\gamma = (1+2\alpha)u-\alpha$. BLX-$\alpha$ has an interesting property: the location of the child solution depends on the difference in parent solutions.

3.2 UNDX

In the unimodal normal distribution crossover (UNDX) [19], multiple parents are used to create two or more offspring solutions around the center of mass of these parents. A small probability is assigned to solutions away from the center of mass. UNDX crossover is formulated as follows:

$$\begin{aligned} c_1 &= m + z_1 e_1 + \sum_{k=2}^{n} z_k e_k, \\ c_2 &= m - z_1 e_2 - \sum_{k=2}^{n} z_k e_k, \end{aligned} \tag{5}$$

$$\begin{aligned} &m = (p_1 + p_2)/2, \\ &z_1 \sim N(0, \sigma_1^2), \quad z_k \sim N(0, \sigma_2^2)(k = 2, \ldots, n), \\ &\sigma_1 = \alpha d_1, \sigma_2 = \beta d_2/\sqrt{n}, \\ &e_1 = (p_2 - p_1)/|p_2 - p_1|, \quad e_i \perp e_j (i, j = 1, \ldots, n, i \neq j) \end{aligned}$$

Here, n is the dimension of the variable. p_1 and p_2 are a pair of parent. d_2 is the distance from p_3 (a parent selected uniformly at random from the mating pool) to $p_1 - p_2$ and $d_1 = |p_1 - p_2|$. α and β are parameters defined by users. The commended settings are $\alpha = 0.5$, and $\beta = 0.35$, respectively.

3.3 PNX

In this work, the parent centric normal crossover (PNX) [20] is also used. This parent-centric crossover is self-adaptive in the sense that the spread of the possible offspring solutions depends on the distance between the parents, which decreases as the population converges. In addition, PNX is an isotropic operator as it does not preferentially search along any particular direction. Another beneficial feature is that PNX has a non-zero probability of generating offspring over the whole search space. In PNX, for each of the offspring c, we proceed as follows to determine its jth gene (c_j). First, we draw a single random number,$\omega \in [0, 1]$, we use the form $y_i^{(1)}$ if $\omega < 0.5$and $y_i^{(2)}$ if $\omega \geq 0.5$. Once this choice is made, the same selected form is used for every component j. The forms are

$$y_j^{(1)} = N\left(x_j^{(1)}, \frac{\left|x_j^{(2)} - x_j^{(1)}\right|}{\eta}\right)$$

$$y_j^{(2)} = N\left(x_j^{(2)}, \frac{\left|x_j^{(2)} - x_j^{(1)}\right|}{\eta}\right)$$

$$c_j = \begin{cases} y_j^{(1)}, & \omega < 0.5 \\ y_j^{(2)}, & \omega \geq 0.5 \end{cases} \tag{6}$$

where $N(\mu, \sigma)$ is a random number drawn from a Gaussian distribution with mean μ and standard deviation σ, x_j^i is the jth component of the ith parent and η is a tunable parameter. The larger is the value of η the more concentrated is the search around the parents. In this paper, η is set to 2.

4 Simulation

In this section, we evaluate the performance of the proposed algorithms. Firstly, we present search abilities when applying to some 30 dimensional problems. Then 500 and 1000 dimensional for each problem is considered to evaluate their performance for large scale global optimization.

4.1 Experimental setup

In order to verify the effectiveness of the proposed algorithm, we use the benchmark test functions ($f_1 \sim f_{21}$) used in the literature [3]. Among the 21 traditional benchmark functions, functions $f_1 \sim f_{10}$ are unimodal (there are some recent evidence that f_4 is a multimodal for D > 3, and the correlation between the variables of Rosenbrock function f_4 is very strong). Functions $f_{11} \sim f_{21}$ are multimodal with the number of local minima increasing exponentially with the problem dimension, and especially the function f_{21} is a strong multi-apex function. Functions $f_1 \sim f_{21}$ are reported as follows:

1. Sphere function (f_1)

$$f(x) = \sum_{i=1}^{n} x_i^2, \quad -5.12 \leq x_i \leq 5.12, x^* = (0, 0, \ldots, 0),$$
$$f(x^*) = 0.$$

2. Ellipsoid function (f_2)

$$f(x) = \sum_{i=1}^{n} (1000^{i-1/n-1} x_i)^2, \quad -5.12 \leq x_i \leq 5.12,$$
$$x^* = (0, 0, \ldots, 0), f(x^*) = 0.$$

3. k-Tablet function (f_3)

$$f(x) = \sum_{i=1}^{k} x_i^2 + \sum_{i=k+1}^{n} (100 x_i)^2, \quad -5.12 \leq x_i \leq 5.12,$$
$$x^* = (0, 0, \ldots, 0), f(x^*) = 0.$$

4. Rosenbrock function f_4

$$f(x) = \sum_{i=2}^{n} (100(x_1 - x_i^2)^2 + (1 - x_i)^2),$$
$$-2.048 \le x_i \le 2.048,$$
$$x^* = (0, 0, \ldots, 0),\ f(x^*) = 0.$$

5. Schewefel problem 3 (f_5)

$$\min_x f(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|, \quad -10 \le x_i \le 10,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

6. Schewefel problem 4 (f_6)

$$\min_x f(x) = \max_x \{|x_i|, 1 \le i \le n\},$$
$$-100 \le x_i \le 100,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

7. Axis parallel hyper ellipsoid (f_7)

$$\min_x f(x) = \sum_{i=1}^{n} i x_i^2, \quad -5.12 \le x_i \le 5.12,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

8. Zakharow's function (f_8)

$$\min_x f(x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} \frac{i}{2} x_i\right)^2 + \left(\sum_{i=1}^{n} \frac{i}{2} x_i\right)^4,$$
$$-5.12 \le x_i \le 5.12,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

9. Exponential problem (f_9)

$$\min_x f(x) = -\exp\left(0.5 \sum_{i=1}^{n} x_i^2\right), \quad -1 \le x_i \le 1,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = -1.$$

10. Ellipsoidal function (f_{10})

$$\min_x f(x) = \sum_{i=1}^{n} (x_i - i)^2, \quad -n \le x_i \le n,$$
$$x^* = (1, 2, \ldots, n)\ and\ f(x^*) = 0.$$

11. Ackley's problem (f_{11})

$$\min_x f(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}\right)$$
$$- \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e, \quad -30 \le x_i \le 30,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

12. Cosine mixture problem (f_{12})

$$\min_x f(x) = \sum_{i=1}^{n} x_i^2 - 0.1 \sum_{i=1}^{n} \cos(5\pi x_i), \quad -1 \le x_i \le 1,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = -0.1n$$

13. Griewank problem (f_{13})

$$\min_x f(x) = 1 + \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right),$$
$$-600 \le x_i \le 600,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

14. Levy and Montalvo problem 1 (f_{14})

$$\min_x f(x) = \frac{\pi}{n} (10 \sin^2(\pi y_1)$$
$$+ \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})]$$
$$\times (y_n - 1)^2),$$
$$where\ y_i = 1 + \frac{1}{4}(x_i + 1),\ -10 \le x_i \le 10,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

15. Levy and Montalvo problem 2 (f_{15})

$$\min_x f(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2$$
$$\times [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2$$
$$\times [1 + \sin^2(2\pi x_n)]), \quad -10 \le x_i \le 10,$$
$$x^* = (0, 0, \ldots, 0)\ and\ f(x^*) = 0.$$

16. Schwefel problem (f_{16})

$$\min_x f(x) = 418.9829 * n - \sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|}),$$
$$-500 \le x_i^{\le} 500,$$
$$x^* = (420.97, \ldots, 420.97)\ and\ f(x^*) = 0.$$

17. Generalized penalized function 1 (f_{17})

$$\min_x f(x) = \frac{\pi}{n}(10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i-1)^2 \times[1+10\sin^2(\pi y_{i+1})] + (y_n-1)^2) + \sum_{i=1}^{n} u(x_i,10,100,4), \quad -10 \le x_i \le 10,$$
$$x^* = (0,0,\ldots,0) \; and \; f(x^*) = 0.$$

18. Generalized penalized function 2 (f_{18})

$$\min_x f(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i-1)^2 \times[1+\sin^2(3\pi x_{i+1})] + (x_n-1)^2) \times[1+\sin^2(2\pi x_n)] + \sum_{i=1}^{n} u(x_i,10,100,4),$$
$$-5 \le x_i \le 5, x^* = (0,0,\ldots,0) \; and \; f(x^*) = 0.$$

In the problem 17 and 18, the penalty function u is given by the following expression:

$$u(x,a,k,m) = \begin{cases} k*pow((x-a),m) & if x > a, \\ -k*pow((x-a),m) & if x < -a, \\ 0 & otherwise. \end{cases}$$

19. Bohachevsky function (f_{19})

$$f(x) = \sum_{i=1}^{n-1}(x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i) - 0.4\cos(4\pi x_{i+1}) + 0.7), -5.12 \le x_i \le 5.12,$$
$$x^* = (0,0,\ldots,0), f(x^*) = 0.$$

20. Schaffer function (f_{20})

$$f(x) = \sum_{i=1}^{n-1}\left[(x_i^2+x_{i+1}^2)^{0.25} \times (\sin^2(50(x_i^2+x_{i+1}^2)^{0.1}) + 1.0)\right], \quad -100 \le x_i \le 100, x^* = (0,0,\ldots,0), f(x^*) = 0.$$

21. Rastrigin function (f_{21})

$$f(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i)), \quad -5.12 \le x_i \le 5.12, x^* = (0,0,\ldots,0), \quad f(x^*) = 0.$$

In general, the comparison of algorithms for CnOP is usually based on the following criterion to evaluate the algorithms: the number of the function evaluations (FEs) to achieve a certain solution quality [9,10]. For all the 21 test functions, we performed 25 independent runs. The stopping criterions are as follows: $|f(s) - f(s*)| < 10^{-7}$(s* is the global optimal solution) or the maximum number of function evaluations (MaxFEs) is set to 10,000*D. It means that if the error accuracy does not reach 10^{-7} within 10,000*D, the simulation run is considered to an unsuccessful run.

Table 1 Parameters setting

Func.	Param.			
	m	k	n_{co}	ξ
$f_1 \sim f_5$, $f_7 \sim f_{10}$, $f_{12} \sim f_{14}$, f_{17}, f_{18}	40	40	5	0.76
f_6, f_{11}, f_{15}, f_{16}, $f_{19} \sim f_{21}$	60	200	5	0.76

4.2 Parameter setting

It is difficult to find the best parameter combination for all the problems because of the multimodality and nonlinearity of different kinds of objective functions. As a result, the robustness of the parameter is very important, and it is a challenge to suggest common fixed values of the parameters. We carried out extensive experiments for the proposed COACO_R algorithm to analyze the parameters.

The same parameters used in the three proposed COACO_R are as follows: the number of ants m, the size of PDF set k, the number of the crossover operations n_{co}, and the evaporation rate ξ of the pheromone. For the sake of fairness, m, k, n_{co} and ξ are set as same as value for three COACO_R and the typical ACO_R. They are set as Table 1. The parameters setting of crossover operators are described in Sect. 3.

4.3 Performance evaluation for 30 dimensionality problems

In this subsection, we firstly present the performance evaluation for 30 dimensionality problems. To investigate the performance of the proposed COACO_R, the convergence properties of the three COACO_R on some typical functions (f_1, f_4, f_{11}, and f_{20}) are analyzed, in comparison with the original ACO_R. The parameters of the ACO_R is set the same as the proposed COACO_R except the n_{co} ($n_{co} = 0$ in ACO_R), because there is no crossover operation in ACO_R. The typical functions include the basic function, the non-separable function which have correlation among the design variables, and the multimodal functions which have a large number of local minima.

Sphere function f_1 is the basic function to evaluate the algorithm. For the non-separable function we choose the

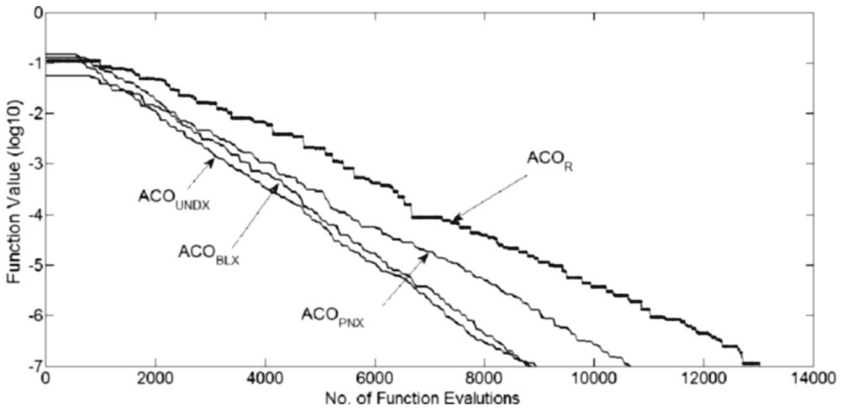

Fig. 3 The convergence process on the sphere f_1, D = 30

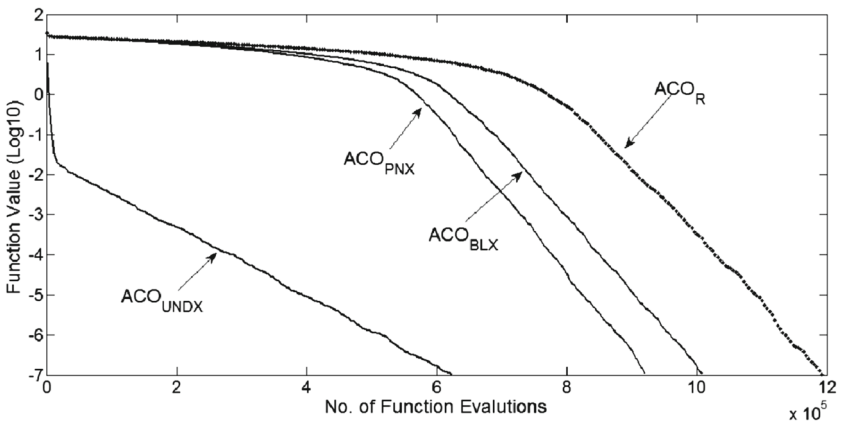

Fig. 4 The convergence process on the sphere f_4, D = 30

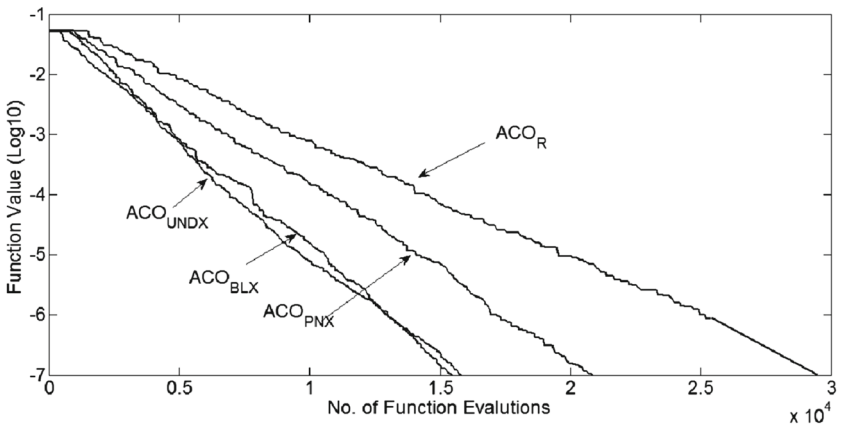

Fig. 5 The convergence process on the sphere f_{11}, D = 30

Rosenbrock function f_4 [21]. In the Rosenbrock function, the variables are correlated and interact between two adjacent variables, and the global minimum is inside a long, narrow, parabolic shaped flat valley. As shown in the Figs. 3 and 4, the proposed algorithm could find the global optimum successfully and faster than the ACO_R for these two.

For the multimodal functions, the Ackley function f_{11} [22] and the Schaffer function f_{20} [23] are chosen. Ackley function has an exponential term that covers its surface with numerous local minima. Schaffer function is made up of a large number of local minima whose value increases with the distance to the global minimum. The number of the local minima is so huge that the ant is trapped in the local minima easily. From the Figs. 5 and 6, we can see that for the Ackley function and the Schaffer function, the proposed three algorithms also faster to reach the required accuracy than the ACO_R. Rastrigin function (f_{21}) is another famous multimodal functions, which is made up of a large number of local minima. As shown in Table 2, the proposed algorithm could find the global optimum very easily while the ACO_R could not find the global optimum for the Rastrigin function.

In order to have an overview of the performance of the proposed algorithms, more extensive experiments are carried out. CMA-ES [22], CGA_R [3] and the differential evolu-

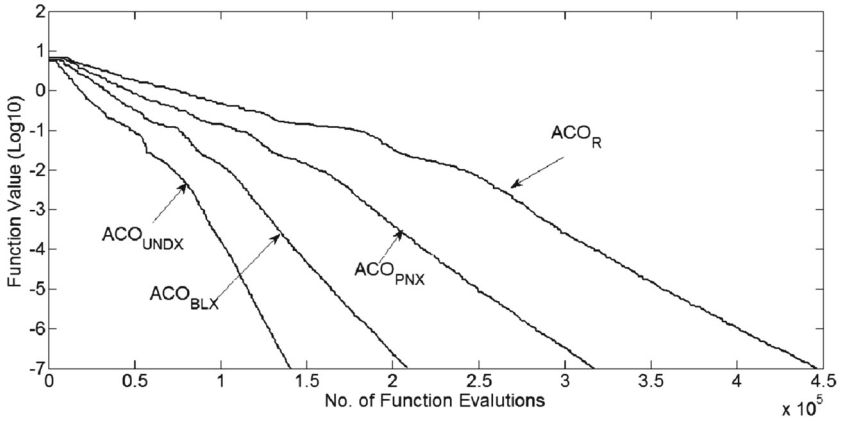

Fig. 6 The convergence process on the sphere f_{20}, D = 30

Table 2 Comparing with other algorithms on 21 benchmark functions, D = 30

Func.	ACO_{BLX}	ACO_{UNDX}	ACO_{PNX}	ACO_R	CGA_R	DE	MMG_{UNDX}
f_1	8.50E+3	**8.15E+3**	1.10E+4	1.84E+4	1.26E+4	4.39E+4	8.03E+4
f_2	**1.35E+4**	2.46E+4	1.74E+4	2.89E+4	3.75E+4	–	–
f_3	**1.22E+4**	2.27E+4	1.54E+4	2.73E+4	4.18E+4	–	–
f_4	3.96E+5	3.85E+5	3.94E+5	–	**3.40E+5**	–	–
f_5	**1.58E+4**	1.76E+4	2.08E+4	3.77E+4	4.86E+4	7.39E+4	–
f_6	2.10E+5	**1.37E+5**	2.28E+5	3.36E+5	2.50E+5	3.75E+5	1.32E+6
f_7	**1.09E+4**	1.14E+4	1.24E+4	1.82E+4	1.99E+4	1.13E+5	3.09E+5
f_8	7.57E+4	**4.78E+4**	9.58E+4	1.89E+5	1.47E+5	–	2.56E+5
f_9	6.54E+3	**6.21E+3**	8.35E+3	1.30 E+4	1.08E+4	3.54E+4	6.11E+4
f_{10}	1.37E+4	**1.34E+4**	1.64E+4	2.72E+4	2.52E+4	–	1.12E+5
f_{11}	**1.96E+4**	2.25E+4	2.51E+4	3.98E+4	7.46E+4	8.40E+4	3.63E+5
f_{12}	2.97E+4	**2.24E+4**	8.10E+4	6.18E+4	2.35E+4	4.18E+4	1.58E+5
f_{13}	**4.40E+4**	**8.53E+3**	9.50E+4	9.50E+4	4.34E+4	5.32E+4	1.04E+6
f_{14}	**6.80E+3**	7.66E+3	8.60E+3	1.39E+4	1.08E+4	3.24E+4	4.45E+5
f_{15}	**2.94E+4**	3.37E+4	3.86E+4	6.64E+4	2.54E+4	4.48E+4	3.58E+5
f_{16}	**1.31E+4**	1.36E+4	1.56E+4	2.54E+5	7.43E+5	5.0E+5	–
f_{17}	**6.65E+3**	7.57E+3	9.15E+3	1.29E+4	1.92E+4	4.40E+4	9.84E+4
f_{18}	**7.70E+3**	8.84E+3	9.65E+3	1.67E+4	7.67E+4	4.50E+4	1.95E+5
f_{19}	3.77E+4	**2.84E+4**	5.38E+4	7.85E+4	3.97E+4	4.93E+4	2.41E+6
f_{20}	2.48E+5	**1.86E+5**	3.72E+5	6.21E+5	5.90E+5	1.93E+5	
f_{21}	4.59E+4	**3.66E+4**	8.54E+4	–	2.40E+5	8.43E+4	4.23E+6

Bold values indicate the best one among all results obtained

tion [24,25] are employed to compare with the proposed $COACO_R$ on 21 benchmark functions ($f_1 \sim f_{21}$).

The CMA-ES is the state-of-the-art algorithm that is useful for continuous optimization. The CGA_R algorithm is a new framework called CGA with an FPDD-LX crossover operator. The DE is another state-of-the-art algorithm that is useful for the real world application, and we select the classical DE approach called DE/rabd/1 to compare with the proposed algorithm. The parameter settings and results of other algorithm is based on the literatures [3,25]. We performed 25 independent runs using the stopping criterions $|f(x) - f(x*)| < 10^{-7}$. The mean numbers of the FEs to achieve the fixed accuracy level 10^{-7} are recorded in Table 2 for the above algorithm.

As shown in Table 2, ACO_{BLX}, ACO_{UNDX} and ACO_{PNX} are all enable to achieve the fixed accuracy level 10^{-7} for 21 problems (D = 30). In this case, ACO_{BLX}, ACO_{UNDX} and CMA-ES obtain 8, 5, and 8 champions among 21 test problems, respectively. ACO_{BLX}, ACO_{UNDX} and ACO_{PNX} are obviously superior to CAM-ES for the multimodal problems. From Table 2, we also known the ACO_R is inferior to three presenting $COACO_R$ when applying to some problems ($D = 30$).

Table 3 Simulation results, D = 500

Func.	ACO_{UNDX}		ACO_{BLX}		ACO_{PNX}		ACO_R	
	Accuracy	FES_{Mean}	Accuracy	FES_{Mean}	Accuracy	FES_{Mean}	Accuracy	FES_{Mean}
f_1	**1.00e−7**	**1.78e+5**	7.92e−2	5.00e+6	7.90e−2	5.00e+6	7.50e−7	5.00e+6
f_2	1.20e+0	5.00e+6	3.22e+3	5.00e+6	1.45e+3	5.00e+6	5.00e+1	5.00e+6
f_3	6.00e−2	5.00e+6	4.25e+2	5.00e+6	2.13e+2	5.00e+6	4.10e−3	5.00e+6
f_4	**1.00e−6**	4.73e+6	5.80e−6	5.00e+6	4.60e+0	5.00e+6	2.12e+1	5.00e+6
f_5	1.60e+0	5.00e+6	1.60e+0	5.00e+6	9.31e−1	5.00e+6	8.51e−5	5.00e+6
f_6	6.10e−1	5.00e+6	1.96e+0	5.00e+6	2.11e+0	5.00e+6	2.12e+1	5.00e+6
f_7	**1.00e−7**	**9.02e+5**	1.53e+1	5.00e+6	6.80e+0	5.00e+6	1.00e−3	5.00e+6
f_8	4.10e−1	5.00e+6	2.16e+2	5.00e+6	6.40e+2	5.00e+6	9.78e+0	5.00e+6
f_9	**1.00e−7**	**1.38e+5**	9.50e−4	5.00e+6	1.06e−3	5.00e+6	**1.00e−7**	3.97e+6
f_{10}	**1.00e−7**	**3.50e+5**	1.58e+6	5.00e+6	1.11e+5	5.00e+6	2.85e+1	5.00e+6
f_{11}	**1.00e−7**	**6.64e+5**	**1.00e−7**	9.34e+5	1.17e−5	5.00e+6	5.91e−1	5.00e+6
f_{12}	**1.00e−7**	**2.99e+5**	**1.00e−7**	4.38e+5	**1.00e−7**	3.38e+6	8.80e−2	5.00e+6
f_{13}	**1.00e−7**	7.41e+5	**1.00e−7**	**5.79e+5**	**1.00e−7**	4.58e+6	1.65e+1	5.00e+6
f_{14}	**1.00e−7**	**1.15e+5**	**3.14e−4**	5.00e+6	1.38e−4	5.00e+6	**1.00e−7**	3.55e+6
f_{15}	**1.00e−7**	**3.03e+5**	**1.00e−7**	4.27e+5	**1.00e−7**	3.28e+6	7.95e−2	5.00e+6
f_{16}	**1.00e−7**	**4.07e+5**	**1.00e−7**	5.82e+5	**1.00e−7**	4.98e+6	6.68e+2	5.00e+6
f_{17}	**1.00e−7**	**1.16e+5**	**2.22e−4**	5.00e+6	3.33e−4	5.00e+6	**1.00e−7**	4.03e+6
f_{18}	**1.00e−7**	**1.65e+5**	**8.51e−3**	5.00e+6	6.25e−3	5.00e+6	**1.00e−7**	4.71e+6
f_{19}	**1.00e−7**	**3.99e+5**	**1.00e−7**	5.62e+5	**1.00e−7**	4.66e+6	5.35e+0	5.00e+6
f_{20}	8.00e+7	5.00e+6	4.50e+2	5.00e+6	7.96e+2	5.00e+6	1.18e+3	5.00e+6
f_{21}	**1.00e−7**	**5.01e+5**	**1.00e−7**	7.83e+5	1.69e−5	5.00e+6	4.34e+0	5.00e+6

Bold values indicate the best one among all results obtained

Table 4 Simulation results, D = 1000

Func.	ACO_{UNDX}		ACO_{BLX}		ACO_{PNX}		ACO_R	
	Accuracy	FES_{Mean}	Accuracy	FES_{Mean}	Accuracy	FES_{Mean}	Accuracy	FES_{Mean}
f_1	**1.00e−7**	**3.78e+5**	4.36e−1	1.00e+7	5.41e−1	1.00e+7	5.68e−4	1.00e+7
f_2	4.60e+0	1.00e+7	6.26e+3	1.00e+7	1.90e+3	1.00e+7	1.02e+2	1.00e+7
f_3	3.32e−2	1.00e+7	2.97e+3	1.00e+7	3.22e+3	1.00e+7	1.19e+1	1.00e+7
f_4	**1.00e−7**	**8.28e+6**	2.24e+1	1.00e+7	4.30e+1	1.00e+7	4.17e+2	1.00e+7
f_5	8.61e+0	1.00e+7	1.24e+1	1.00e+7	1.50e+1	1.00e+7	7.15e−2	1.00e+7
f_6	7.31e−1	1.00e+7	2.31e+0	1.00e+7	2.10e+0	1.00e+7	2.04e+0	1.00e+7
f_7	**1.00e−7**	**2.43e+6**	2.41e+2	1.00e+7	2.77e+2	1.00e+7	2.90e−1	1.00e+7
f_8	3.81e+0	1.00e+7	1.11e+3	1.00e+7	2.27e+3	1.00e+7	1.90e+2	1.00e+7
f_9	**1.00e−7**	**2.82e+5**	8.05e−3	1.00e+7	1.10e−2	1.00e+7	1.03e−1	1.00e+7
f_{10}	**1.00e−7**	**3.50e+5**	5.94e+6	1.00e+7	6.48e+6	1.00e+7	2.27e+4	1.00e+7
f_{11}	**1.00e−7**	**7.35e+5**	**1.00e−7**	2.95e+6	6.65e−3	1.00e+7	8.49e−1	1.00e+7
f_{12}	**1.00e−7**	**1.33e+6**	**1.00e−7**	1.39e+6	3.01e−4	1.00e+7	3.11e−7	1.00e+7
f_{13}	**1.00e−7**	**6.40e+6**	**1.00e−7**	1.70e+6	2.58e−2	1.00e+7	2.25e+0	1.00e+7
f_{14}	**1.00e−7**	**2.23e+5**	**6.02e−4**	1.00e+7	8.81e−4	1.00e+7	2.32e−4	1.00e+7
f_{15}	**1.00e−7**	**6.22e+5**	**1.00e−7**	1.38e+6	6.71e−6	1.00e+7	2.00e−1	1.00e+7
f_{16}	**1.00e−7**	**8.21e+5**	**1.00e−7**	1.91e+6	2.95e−2	1.00e+7	1.08e+3	1.00e+7
f_{17}	**1.00e−7**	**2.22e+5**	3.69e−4	1.00e+7	1.00e−3	1.00e+7	1.17e−4	1.00e+7
f_{18}	**1.00e−7**	**3.32e+5**	7.31e−2	1.00e+7	6.71e−2	1.00e+7	3.70e−3	1.00e+7
f_{19}	**1.00e−7**	**8.35e+5**	**1.00e−7**	2.06e+6	9.11e−3	1.00e+7	1.64e+1	1.00e+7
f_{20}	1.51e+3	1.00e+7	1.38e+3	1.00e+7	1.53e+3	1.00e+7	2.47e+3	1.00e+7
f_{21}	**1.00e−7**	**1.03e+6**	**1.00e−7**	2.45e+6	3.75e−1	1.00e+7	9.74e+1	1.00e+7

Bold values indicate the best one among all results obtained

4.4 Performance evaluation for 500–1000 dimensionality problems

Nowadays, high dimensional optimization problems are an interesting field of research. Although these approaches such as CMA-ES and DE have shown excellent search abilities when applying to some 30–100 dimensional problems, many of them suffer from their performance deteriorates quickly as the dimensionality of search space increases. CMA-ES is an algorithm that uses several operations with a complexity of $O(n^3)$, where n is the dimension value, and although there are versions that try to reduce this problem, it has not been actually resolved [24]. This behavior makes CMA-ES does not scale well for large scale optimization problems. To further challenge and evaluate the proposed algorithms, we apply them to 500–1000 dimensionality problems.

Tables 3 and 4 record the reached accuracy level ($f(x) - f(x^*)$) to terminate before reaching the stopping criterions: $|f(x) - f(x*)| < 10^{-7}$, or the number of function evaluations (FES) is larger than 10,000*D. The mean FES needed in each run to achieve the fixed accuracy level (see the "Accuracy" column in Tables 3, 4) are also indicated in Tables 3 and 4. As shown in Tables 3 and 4, ACO_{UNDX} has excellent performance for large scale optimization problems. There are fourteen 500 dimensionality problems to reach the accuracy level 10^{-7} and one to 10^{-6} when using ACO_{UNDX}. As for 1000 dimensionality problems, ACO_{UNDX} has 15 problems to reach the accuracy level 10^{-7}. Other two algorithms ACO_{BLX} and ACO_{PNX} are also superior to ACO_R for large scale optimization problems. With regard to the mean FES needed in each run, ACO_{UNDX} also obviously less than other algorithms.

5 Conclusion

In this paper, we evaluated the performance of the ACO algorithms with different crossover operations for CnOPs. A large number of simulations on 21 benchmark problems were carried out. Simulation results showed ACO_R with crossover operation is far well the typical ACO_R from 30 to 1000 dimensionality problems. So we can draw a conclusion that crossover operators can effectively improve the global exploration ability of ACO_R. ACO_{BLX} has the most excellent performance for 30 dimensionality problems and it is superior to CAM-ES for the multimodal problems. ACO_{UNDX} has the most excellent performance for large scale optimization problems. The proposal is a very competitive optimization algorithm for large scale problems, both in results and in processing time. In future works, we will focus cooperative coevolution between $ACO_R$and EA.

Acknowledgements This work is supported by Scientific and Technological Research Program of Chongqing Municipal Education Commission [Nos. KJ1500607, KJ1400629], Science Research Fund of Chongqing Technology and Business University [No. 2011-56-05], and the National Natural Science Foundation of China [51375517, 61402063].

References

1. Zhang, X., Tian, Y., Jin, Y.: A knee point driven evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **19**(6), 761–776 (2015)
2. Zhang, X., Tian, Y., Cheng, R., Jin, Y.: An efficient approach to nondominated sorting for evolutionary multi-objective optimization. IEEE Trans. Evol. Comput. **19**(2), 201–213 (2015)
3. Chen, Z.Q., Wang, R.L.: A new framework with FDPP-LX crossover for real-coded genetic algorithm. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E94.A**(6), 1417–1425 (2011)
4. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
5. Socha, K., Dorigo, M.: Ant colony optimization for continuous domains. Eur. J. Oper. Res. **185**(3), 1155–1173 (2008)
6. Tang, K., Yao, X., Suganthan, P.N., MacNish, C., Chen, Y.P., Chen, C.M., Yang, Z.: Benchmark Functions for the CEC'2008 Special Session and Competition on Large Scale Global Optimization. IEEE World Congress on Computational Intelligence (2008), Hong Kong
7. Zhang, X., Tian, Y., Cheng, R., Jin, Y.: A decision variable clustering based evolutionary algorithm for large-scale many-objective optimization. IEEE Trans. Evol. Comput. (2016). doi:10.1109/TEVC.2016.2600642
8. Zhang, X., Tian, Y., Jin, Y.: Approximate non-dominated sorting for evolutionary many-objective optimization. Inf. Sci. **369**(10), 14–33 (2016)
9. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: an autocatalytic optimizing process. Technical Report 91-016 Revised, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1991
10. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybern. B **26**(1), 29–41 (1996)
11. Bilchev, G., Parmee I.C.: The ant colony metaphor for searching continuous design spaces. Selected Papers from AISB Workshop on Evolutionary Computing, vol. 993, pp. 25–39 (1995)
12. Monmarche, N., Venturini, G., Slimane, M.: On how pachycondyla apicalis ants suggest a new search algorithm. Future Gener. Comput. Syst. **16**(8), 937–946 (2000)
13. Dreo, J., Siarry, P.: A new ant colony algorithm using the heterarchical concept aimed at optimization of multiminima continuous functions. Ant Algorithms **2463**, 216–221 (2002)
14. Dréo, J., Siarry, P.: Continuous interacting ant colony algorithm based on dense heterarchy. Future Gener. Comput. Syst. **20**(5), 841–856 (2004)
15. Hu, X.M., Zhang, J., Li, Y.: Orthogonal methods based ant colony search for solving continuous optimization problems. J. Comput. Sci. Technol. **23**, 2–18 (2008)
16. Hu, X.M., Zhang, J., Chung, H.S.H., Li, Y., Liu, O.: SamACO: variable sampling ant colony optimization algorithm for continuous optimization. IEEE Trans. Syst. Man Cybern. B Cybern. **40**, 1555–1566 (2010)
17. Liao, T., Stützle, T.: A unified ant colony optimization algorithm for continuous optimization. Eur. J. Oper. Res. **234**, 597–609 (2014)
18. Eshelman, L.J., Schaffer, J.D.: Real-coded genetic algorithms and interval schemata. In: Whitley, D.L. (ed.) Foundation of Genetic Algorithms II, pp. 187–202. Morgan Kaufmann, San Mateo (1993)

19. Ono, I., Kobayashi, S.: A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover. In: Back, T. (ed.) Proceedings of the Seventh International Conference on Genetic Algorithms, pp. 246–253. Morgan Kaufmann, San Mateo (1997)
20. Ballester, P.J., Carter, J.N.: An effective real-parameter genetic algorithm with parent centric normal crossover for multimodal optimization. In: Deb, K., et al. (eds.) Lecture Notes in Computer Science, vol. 3102, pp. 901–913. Springer, Berlin (2004)
21. Shang, Y.W., Qiu, Y.H.: A note on the extended rosenbrock function. Evol. Comput. **14**, 119–126 (2006)
22. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Trans. Evol. Comput. **3**(2), 82–102 (1999)
23. Rosenbrock, H.H.: An automatic method for finding the greatest or least value of a function. Comput. J. **3**(3), 175–184 (1960)
24. Ortiz-Boyer, D., Hervas-Martinez, C., Garcia-Pedrajas, N.: A crossover operator for evolutionary algorithms based on population features. J. Artif. Intell. Res. **24**, 1–48 (2005)
25. Hansen, N.: The CMA Evolution Strategy: A Tutorial, 2010

Zhiqiang Chen received a B.S. degree from WuHan University of Water-Conservancy and Electric Power, WuHan, China and an M.S. degree from ChongQing University, Chongqing, China in 2001 and 2004 respectively. He received Ph.D degree from Fukui University, Japan in 2011. He is currently an associate professor with the School of Computer Science and Information Engineering, Chongqing Technology and Business University. His main research interests are Computational Intelligence and signal processing.

Rong-Long Wang received a B.S. degree from Hangzhe teacher's college, Zhejiang, China and an M.S. degree from Liaoning University, Liaoning, China in 1987 and 1990, respectively. He received his D.E. degree from Toyama University, Toyama, Japan in 2003. From 1990 to 1998, he was an Instructor in Benxi University, Liaoning, China. In 2003, he joined University of Fukui, Fukui Japan, where he is currently an associate professor in Department of Electrical and Electronics Engineering. His current research interests include genetic algorithm, neural networks, and optimization problems.