Advances in Computer Science Research, volume 80
International Conference on Computer Science, Electronics and Communication Engineering (CSECE 2018)

An Improved Differential Evolution for Constrained Optimization Problems

Liechao Zhang and Lin Shang*
Wuhan Technical College of Communications, Wuhan 430065, China
*Corresponding author

***Abstract*—A fast and robust differential evolution based on orthogonal design (ODE) is proposed, and then it is used to solve constrained optimization problems. The ODE combines the conventional DE (CDE), which is simple and efficient, with the orthogonal design, which can exploit the optimum offspring. The ODE has some features. 1) It uses a robust crossover based on orthogonal design and an optimal offspring is generated with the constrained statistical optimal method. 2) To decrease the number of the orthogonal design and make the algorithm converge faster, decision variable fraction strategy is applied here. 3) It uses simple diversity rules to handle the constraints and maintain the diversity of the population; 4) A multi-parent hybrid adaptive-crossover-mutation operator based on the non-convex theory is proposed, which can enhance the non-convex search ability. 5) The ODE simplifies the scaling factor F of the CDE, which can reduce the parameters of the algorithm and make it easy to use for engineers. We execute the proposed algorithm to solve 13 benchmark functions with linear or/and nonlinear constraints. Through comparison with some state-of-the-art evolutionary algorithms, the experimental results demonstrate that the performance of the ODE outperforms other evolutionary algorithms in terms of the quality of the final solution and the stability; and its computational cost (measured by the average number of fitness function evaluations) is lower than the cost required by the other techniques compared.**

Keywords—differential evolution; orthogonal design; simple diversity rules; hybrid adaptive-crossover-mutation; function optimization

I. INTRODUCTION

Evolutionary algorithm is an adaptive, self-organizing and implicitly parallel algorithm based on groups of heuristic searching algorithm, which utilizes individuals from groups to search in the solution domain. In the process of evolution, individuals in groups produce new individuals by genetic operators, maintain the diversity of the population and search in the new space to get the optimization. For now, there has been a many branches of EA, including genetic algorithm, evolutionary programming, evolution strategies, genetic programming and so on.

In real engineering design, some complex function optimization problems are often met, which are always nonlinear, incontinous, indifferentiable, multimodal, multi-extremum, no-convex and numerical algorithms(such as simplex method, conjugate gradient method) to get ideal optimization. In recent years, many scholars apply evolution algorithms to complex function optimization and get ideal results. In function optimization[2,3,5,6], the objective function often comes with a lot of equation and inequality constraints, which bring challenges to optimization method. In constrained global optimization problems, penalty functions are usually used to deal with constrained functions .That is, punish the constrained function with a penalty factor, and then add the penalty term to the objective function, which convert the problem to a global optimization without constraints. Although the method has been widely used, its drawbacks is that it's difficult to set and regulate the penalty factor, which means too large or too small penalties will make it simple feasibility rules to deal with the constrained functions and get good results[8].Recently, some scholars have the feasibility of using a few simple rules to deal with constraint function[8,10,11], good results have been achieved.

Differentia evolution[2] is a fast evolutionary algorithm, which uses differential information between individuals to direct search. DE performed well on the International Contest on Evolutionary Computation and get widely used. Compared with other evolutionary algorithm its different characteristics, K.Price and R.Storm designed ten different strategies of DE for users to choose. However, CDE converges slowly in evolution of the late when dealing with the global optimal solution approaches the boundary of feasible and infeasible region. it will be difficult to get the optimal solution or slowing down of convergence will appear.

In this paper, an improved version of the differential evolution based on the orthogonal design(ODE) is presented. Its feature is: a new hybrid adaptive crossover operator is presented to enhance the global searching ability of the algorithm and non-convex search; in the meantime, to ensure better individuals will come out in the evolution, orthogonal crossover operator and constraint statistical optimal method are adopted, combined with decision variable fraction strategy in order to decrease the number of the orthogonal crossover operator and constraint statistical optimal method are adopted, combined with decision variable fraction strategy in order to decrease the number of the orthogonal design and make the algorithm converge faster; It uses simple diversity rules to handle the constraint. This algorithm is called CDE for short. To make it simple to use, the ODE simplifies the scaling factor, and take consistent random numbers in the interval $(0,1]$ .Though 13 standard functions, the superior performance of ODE is verified.

II. PROBLEM DEFINITION

Without loss of generality, take the constrained minimization function optimization for example, which can be described as follows:

$$\text{Minimize } f(X),\ X=(x_1,x_2,...,x_N)\in R^N \tag{1}$$

to satisfy:

$$\begin{cases} g_i(X)\le 0, & i=1,2,...p \\ h_j(X)=0, & j=1,2,...,q \end{cases} \tag{2}$$

among which $X\in S\cap\Gamma$. $S\subseteq R^N$ is called search space, Γ is feasible region, $f(X)$ is objective function, X is a N-dimensional decision vector, $X=[x_1,x_2,\cdots,x_N]^T$,each independent variable x_i to satisfy certain constraint:

$$l_i\le x_i\le u_i,\ i=1,2,...,N \tag{3}$$

p is the number of Inequality constraints, and q is the number of inequity constraints. The constraints can be linear or nonlinear.

III. ORTHOGONAL DESIGN

In industry, agriculture and scientific research, multi-factor influence on products is often taken into consideration. If arrange comprehensive experiments with each factors of different level of mutual collecation, it will be difficult or even impossible to realize. Orthogonal design can get optimization, determine the effect of index factors and levels first, then select the appropriate orthogonal array, according to which the experiment can be arranged. At the end, Analysis of the test results and get a good level.

The orthogonal array has the following properties:

1. For the factor in any column, every level occurs equal times.

2. For the two factors in any two columns, every combination of two levels occurs equal times to represent the experiments.

3. Any two experiments are not the same, so their results cannot be compared directly.

4. If any two columns of an orthogonal array are swapped, the resulting array is still an orthogonal array.

5. If some columns are taken away from an orthogonal array, the result is still an orthogonal array with a smaller number of factors.

IV. ODE: ORTHOGONAL DE

In order to get rapid convergence as the region of global minimum, make it easy to manipulate and get better solutions, we modify the CDE. The enhancements of ODE are as follows:

A. Orthogonal Crossover Operator

1.Design of the orthogonal array

To design a minimal orthogonal array, in the research, we use the two level orthogonal array $L_{2^J}(2^{2^J-1})$, $R=2^J$ denotes the number of the rows of orthogonal array, and $C=2^J-1$ denotes the number of the columns. The orthogonal array needs to find a proper J to satisfy

$$\begin{aligned} &Minimize: R=2^J \\ &St: C=2^J-1\ge N \end{aligned} \tag{4}$$

Where N is the number of the variables. In this study, we adopt the algorithm described in Ref.[5] to construct an orthogonal array. In particular we use $L(R,C)$ to indicate the orthogonal array; and $L(i,j)$ denotes the level of the jth factor in the ith combination in $L(i,j)$.

2. Generation of the orthogonal sub-population

After construction a proper orthogonal array, we select two parents randomly, $X_1=(x_{11},x_{12},...,x_{1N})$ and $X_2=(x_{21},x_{22},...,x_{2N})$, to generate the orthogonal sub-population O(R,N) for two level orthogonal crossovers.

3. Direct statistical optimal method with constraints

In order to choose a better individual as offspring from the orthogonal group, Ref.[5] chooses the offspring according to the fitness value. However, we know that any two experimental results cannot be compared directly from the third property of orthogonal array. As a result, in order to generate the best offspring from the orthogonal array, the algorithm presents a direct statistical optimal method with constraints based on traditional statistics to satisfy the constraint conditions, which is similar to Taguchi method in Ref.[6] and statistical optimal method in Ref.[7], where constraint conditions are added based on intuitive method to verify the diversity of the population.

Calculation of the value of the kth level of jth factor $E(j,k)$ is

$$\begin{cases} E(j,k)=\sum_{L(i,j)=k} X_i.f \\ CE(j,k)=\sum_{L(i,j)=k} X_i.cf \end{cases} \tag{5}$$

where Xi. f is the fitness value of the ith combination's constrained function. In this way, generate the best offspring to each factors in Algorithm 2.

B. Decision Variable Fraction Strategy

If the dimension of the variable is higher, it needs to design a larger orthogonal array with more rows. For example, if N=30,R=32;N=10,R=128.When we use Algorithm 2 to generate an offspring, it may evaluate too many times, leaving the efficiency low. Therefore, ODE uses decision variable fraction strategy to divide the variable into groups(fractions can be equal or not equal. For simple calculation, fractions are equally spaced) to reduce the number of factors. For example, divide 30-dimensional variable into 2 groups, in the way the orthogonal array(R=4) can satisfy the requirements.

C. Simple Diversity Rules

To maintain the diversity of population, 2 rules are put forward to compare 2 individuals when handling constrained optimization. Formula(6) calculates the target fitness value of individuals; formula(7) calculates the sum of constrained functions; formula(8) compares two individuals

$$X.f = f(X) \tag{6}$$

$$X.cf = \sum_{i=1}^{q} \max\{0, g_i(X)\} + \sum_{j=1}^{p} \max\{0, |h_j(X)| - \eta\} \tag{7}$$

$$Better(X_1, X_2) = \begin{cases} X_1.cf < X_2.cf, & 1 \\ X_1.cf > X_2.cf, & 0 \\ (X_1.cf == X_2.cf) \cap (X_1.f < X_2.f), & 1 \\ (X_1.cf == X_2.cf) \cap (X_1.f == X_2.f), & 2 \\ (X_1.cf == X_2.cf) \cap (X_1.f > X_2.f), & 0 \end{cases} \tag{8}$$

where $|h(X)| - \eta$ denotes transforming equality constraint into inequality by deducting a small positive(namely equation tolerance). In formula (8), if return values is 1, individual X1 precedes X2, X1 will be accepted in evolution; if return values is 0, individual X2 precedes X1, X2 will be accepted; if return values is 2, X2 and X1 have the same performance , the 2 individual will be accepted with the same probability in the evolution. In this way, it can be ensured that some infeasible individuals are in the population and thus pressure is relieved and population's diversity is kept.

D. Hybrid Adaptive Crossover Mutation Operators

To enhance the algorithm's no-convex search ability, ODE designs mutil-parent non convex crossover operators, as the formula (9) show:

$$P_1 = \sum_{i=1}^{p} a_i * X_i \tag{9}$$

where $a_i \in [-0.5, 1.5]$ is a random number, and $\sum_{i=1}^{P} a_i = 1$,p denotes the number of chosen parents. formula (9) is mentioned in Ref.[3],and a_i choosing random number from [-0.5,1.5] can enhance the algorithm's no-convex search ability.

ODE generated individuals respectively by formula (9) and DE/rand/1/bin strategy, then choose a better one as offspring to compare with the parent individual. If the offspring precedes the parent, it will replace the parent; otherwise the offspring will be abandoned.

E. Simplifying the Scaling Factor

The scaling factor F is generated uniform randomly from $F \in (0,1]$ in the proposed ODE to make is simple and easy to use without inaccuracy caused manual set.

V. LABORATORY FINDING

A. Experimental Environment And Parameter Setting

For all experiments in ODE, we used the following parameters:

Population size: M=100;probability of crossover: CR=0.9; DE strategy: DE/rand/1/bin; largest number of fitness evaluation: NFFE=240 000; number of parent : P=6; Halting precision: $\varepsilon = 1e-30$; decision variable fraction: if dimension of variable N>8,F=2,otherwise F=N; equality constraint tolerance: $\eta = 0.0001$.

B. Benchmark Functions

In order to test the performance of ODE, 13 benchmark function Ref.[9] were used,11 of which are first proposed in Ref [12],2 of which are added by [9]. All of the 13 are benchmark functions for constrained optimization problems and are widely used in may Refs. Functions g02, g03, g08, g12 are for the max of objective functions and the other are for the min. Properties of the functions have been described in Ref [9].

C. Experimental Results

Every function operates independently 30 times, the results are shown as follows: 1)best function results Best; 2) medium results Media; 3) mean best results Mean Best; 4) worst results Worst: 5) standard deviation Std. Dev; 6) mean function fitness evaluation MFFE; 7) success rate Sr(if $|f(X^*) - f(X_{best})| \le 1e-4$ or best results precede global optimal solution, the operation will be successful); 8) mean time of operating time T. Table 1 shows the results of ODE experiments. In order to test the performance of ODE, the author compare its results with other evolution, which are HM[12], SR[9], ASCHEA[13], CHDE[11] and SMES[8]. Table 2 is comparison of best results between ODE and HM, SR, ASCHEA, CHDE, SMES; Table 3 is comparison of mean results between ODE and HM, SR, ASCHEA, CHDE, SMES; Table 4 is comparison of worst results between ODE and HM, SR, ASCHEA, CHDE and SMES.

TABLE I. STATISTICAL RESULTS OF ODE(RESULTS IN BOLD DENOTES BEST RESULTS PRECEDE GLOBAL OPTIMAL SOLUTION)

Problem	Optimal	Best	Media	Mean Best	Worst	Std. Dev.	MFFE	Sr	*T* (s)
g01	-15	-15	-15	-15	-15	0	41,752	100%	1.265
g02	0.803619	0.803619	0.801252	0.801981	0.792609	0.003518	108,678	80%	4.906
g03	1	**1.0005**	1.0005	1.0005	1.0005	0	209,260	100%	6.359
g04	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	0	33,999	100%	0.953
g05	5126.498	**5126.4967**	5126.4967	5126.902	5127.896	0.4786	102,230	56.67%	3.359
g06	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814	0	12,385	100%	0.344
g07	24.306	24.306	24.306	24.306	24.306	0	71,710	100%	2.593
g08	0.095825	0.095825	0.095825	0.095825	0.095825	0	2,515	100%	0.078
g09	680.63	680.63	680.63	680.63	680.63	0	21,028	100%	0.687
g10	7049.25	**7049.248**	7049.248	7049.248	7049.24802	2.945×10^{-16}	125,890	100%	3.031
g11	0.75	**0.7499**	0.7499	0.7499	0.7499	0	32,965	100%	0.921
g12	1	1	1	1	1	0	1,998	100%	0.984
g13	0.05395	**0.05387**	0.4384	0.3230	0.4384	0.1762	212,169	33.33%	6.906

VI. DISCUSSIONS AND ANALYSIS

We can see from table 1 that ODE can find all global optimal solutions to benchmark functions. And the best results of g03, g05, g10, g11 and g13 are better than the global optimal solutions, which depends on the magnitude of the equation tolerance η when transforming the equality constraint into inequality constraint. MFFEs of all functions is smaller than largest number of evolution, ten of which are smaller than 100000, Success rate is 100% except for g02, g05 and g13, which denotes the absolute values of difference between final result and optimal solution are smaller or equal compared with 1e-4 or optimal solutions during the 30 ODE operations. Moreover, all the functions give small standard deviation, nine(g01, g03, g04, g06, g07, g08, g09, g11 and g12) of which analyses are zero. Analyses above indicate that ODE has outstanding performance when handle constrained optimization problems, which can give high accuracy, more stable solution quality and faster convergence speed. However, ODE doesn't perform well handling g13 with the success rate of 33.33% , and other 20 operations get the local optimal solution 0.4383652659. Thus the algorithm doesn't work well with the equality constraint and needs to be improved, which doesn't go against the "No-Free-Lunch" theory proposed in Ref [14].

TABLE II. COMPARISON OF BEST RESULTS BETWEEN ODE AND HM,SR,ASCHEA,CHDE SEMS (RESULTS IN BOLD ARE BETTER OR GLOBAL OPTIMAL SOLUTION)

Problem	Optimal						
		HM	SR	ASCHEA	CHDE	SMES	ODE
g01	-15	-14.7886	**-15**	**-15**	**-15**	**-15**	**-15**
g02	0.803619	0.79953	0.803515	0.785	**0.803619**	0.803601	**0.803619**
g03	1	0.9997	1	1	1	1	**1.0005**
g04	-30665.539	-30664.5	**-30665.539**	-30665.5	**-30668.539**	**-30665.539**	**-30665.539**
g05	5126.498	--	5126.497	5126.5	5126.497	5126.599	**5126.4967**
g06	-6961.814	-6952.1	**-6961.814**	-6961.81	**-6961.814**	**-6961.814**	**-6961.814**
g07	24.306	24.620	24.307	24.3327	**24.306**	24.327	**24.306**
g08	0.095825	**0.095825**	**0.095825**	**0.095825**	**0.095825**	**0.095825**	**0.095825**
g09	680.63	680.91	**680.630**	680.632	**680.630**	680.632	**680.63**
g10	7049.25	7147.9	7054.316	70561.13	7049.24802	7051.903	**7049.248**
g11	0.75	0.75	0.75	0.75	**0.749**	0.75	0.7499
g12	1	0.99999	**1**	*A. NA*	**1**	**1**	**1**
g13	0.05395	NA	0.053957	*B.* NA	0.053866	0.053986	0.05387

1.Comparison Between ODE and HM

ODE get better best results than HM in 11 functions (g01, g02, g03, g04, g05, g06, g07, g09, g10, g11 and g12). In g08, the two best results are global optimal solution. In the meantime, mean and worst results of ODE are better than HM's in 12 functions(g01,g02, g03, g04, g05, g06, g07, g08, g09, g10, g11 and g12), HM doesn't test function g13, so it doesn't need any comparisons.

2.Comparison Between ODE and SR

Comparison with SR, ODE gets better best result in g02, g03, g05, g07, g10, g11 and g13. For other 6 functions, both algorithm get best results equal to optimal solution. For g02, g03, g05, g06, g07, g09, g10 and g11,ODE gets better mean results and worst results. But mean results and worst results of SR are better than ODE for g13. For other functions, mean results and worst results of both algorithm are optimal solution.

3.Comparison Between ODE and ASCHEA

Comparison with ASCHEA, ODE gets best results in nine functions, except that the two 2 algorithms get the same best results. And ODE get bets better mean results than ASCHEA, except for the same mean results in functions g01 and g08. ASCHEA doesn't test function g12 and g13, doesn't get worst results, so it doesn't need comparison.

TABLE III. COMPARISON OF MEAN RESULTS BETWEEN ODE AND HM,SR,ASCHEA,CHDE SEMS (RESULTS IN BOLD ARE BETTER OR GLOBAL OPTIMAL SOLUTION)

Problem	Optimal						
		HM	SR	ASCHEA	CHDE	SMES	ODE
g01	-15	-14.7082	**-15**	-14.84	-14.792134	**-15**	**-15**
g02	0.803619	0.79671	0.781975	0.59	0.746236	0.785238	**0.801981**
g03	1	0.9989	1	0.99989	0.640326	1	**1.0005**
g04	-30665.539	-30665.5	**-30665.539**	-30665.5	-30592.154435	**-30665.539**	**-30665.539**
g05	5126.498	--	5128.881	5141.65	5218.729114	5174.492	**5126.902**
g06	-6961.814	-6342.6	-6875.940	-6961.81	-6367.575424	-6961.284	**-6961.814**
g07	24.306	24.826	24.374	24.66	104.599221	24.475	**24.306**
g08	0.095825	0.0891568	**0.095825**	**0.095825**	0.091292	**0.095825**	**0.095825**
g09	680.63	681.16	680.656	680.641	692.472322	680.643	**680.63**
g10	7049.25	8163.6	7559.192	7193.11	8442.656946	7253.047	**7049.248**
g11	0.75	0.75	0.75	0.75	0.761823	0.75	**0.7499**
g12	1	0.99913	**1**	*C. NA*	**1**	**1**	**1**
g13	0.05395	*D. NA*	**0.057006**	*E. NA*	0.747227	0.166385	0.3230

TABLE IV. COMPARISON OF WORST RESULTS BETWEEN ODE AND HM,SR,ASCHEA,CHDE SEMS (RESULTS IN BOLD ARE BETTER OR GLOBAL OPTIMAL SOLUTION)

Problem	Optimal						
		HM	SR	ASCHEA	CHDE	SMES	ODE
g01	-15	-14.6154	**-15**	*NA*	-12.743044	**-15**	**-15**
g02	0.803619	0.79119	0.726288	*NA*	0.302179	0.751322	**0.792609**
g03	1	0.9978	1.00	*NA*	0.029601	1	**1.0005**
g04	-30665.539	-30645.9	**-30665.539**	*NA*	-29986.2144	**-30665.539**	**-30665.539**
g05	5126.498	--	5142.472	*NA*	5502.410392	5304.167	**5127.896**
g06	-6961.814	-5473.9	-6350.262	*NA*	-2236.950336	-6952.482	**-6961.814**
g07	24.306	25.069	24.642	*NA*	1120.541494	24.843	**24.306**
g08	0.095825	0.0291438	**0.095825**	*NA*	0.027188	**0.095825**	**0.095825**
g09	680.63	683.18	680.763	*NA*	839.782911	680.719	**680.63**
g10	7049.25	9659.3	8835.655	*NA*	15580.370333	7638.366	**7049.24802**
g11	0.75	0.75	0.75	*NA*	0.870984	0.75	**0.7499**
g12	1	0.99195	**1**	*NA*	**1**	**1**	**1**
g13	0.05395	*F. NA*	**0.216915**	*NA*	2.259875	0.468294	0.4384

4.Comparison Between ODE and CHDE

Comparison with CHDE, ODE gets better best results for function g03, g05 and g10. But for function g13, best results of CHDE are slightly better ODE's. The 2 algorithms get the same best results for other functions. Except for the same results for function g12, ODE gets better mean and worst results for other 12 functions.

5. Comparison Between ODE and SEMS

Best results of ODE are better than SEMS for 8 functions(and). And the 2 algorithms get the same best results

equal to optimal solutions for the other 5 functions. Mean and worst results of ODE are better than SEMS for 8 functions (and). Only worst results of SMES for function are better than ODE's . The 2 algorithms get the same mean and worst results for the other 3 functions.

Judging from the evaluations, ODE can converge fast for all functions with less evaluation times. In all algorithms compared above, the evaluation times of HM, SR, ASCHEA, CHDE and SMES are 1 400 000, 35 000, 1 500 000, 348 000 and 240 000. As has been shown in Table 1, MFFE of ODE is less than other 5 algorithms. That denotes fast convergence, which is of great significance in practical engineering optimization.

VII. CONCLUSION AND FUTURE WORK

For faster speed and stronger speed, this paper has improved CDE and applied orthogonal design to the algorithm, which generates offspring by 2 levels crossover operators and statistical optimal method. In the meantime, ODE utilizes hybrid adaptive crossover operators with non-convex search ability to enhance the non-convex search ability. Through decision variable fraction strategy, it can decrease the number of orthogonal design factors. In that way it will be possible to design a smaller orthogonal array to decrease fitness evaluation is used to handle to constrained functions. Tested by 13 benchmark functions of different kinds and compared with other algorithms, ODE doesn't only speed up convergence on the basis of CDE, but also have improvement on the accuracy and stability of solutions. Moreover, ODE performs better than other 5 algorithms. However, ODE has the drawback of local convergence when handling multi-function optimization with equality constraint, which we will improve in the future. Besides, applying the algorithm to other optimization problems(such as dynamic function optimization, noise function and multi-objective optimization) will be another research area.

ACKNOWLEDGMENT

This work was supported by the key project of "12th Five-Year" planning of Education Science in Hubei province under Grant No.2013A052 and Wuhan Technical College Of Communications Scientific Research and Innovation Team Grant No. CX2018A02 ,No.CX2018B02.

REFERENCES

[1] T. Bäck, H.-P. Schwefel, An overview of evolutionary algorithms for parameter optimization [J]. Evolutionary Computation, 1 (1): 1-23, 1993.

[2] R. Storn, K. Price, Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces [J]. Journal of Global Optimization, 11: 341–359, 1997.

[3] Guo Tao, Kang Li-shan. A new Evolutionary Algorithm for Function Optimization [J]. Wuhan University Journal of Nature Sciences, 4(4): 409-414, 1999.

[4] KaiTai Fang, ChangXin Ma. Orthogonal And Uniform Experimental Design [M]. Science Press, 2001.

[5] Y. W. Leung, Y. Wang. An Orthogonal Genetic Algorithm with Quantization for Global Numerical Optimization [J]. IEEE Transactions on Evolutionary Computation, 5(1), 41-53, 2001.

[6] J. T. Tsai, T. K. Liu and J. H. Chou. Hybrid Taguchi-Genetic Algorithm for Global Numerical Optimization [J]. IEEE Transactions on Evolutionary Computation, 8(4), 365-377, 2004.

[7] SanYou Zeng, Wei Wei, LiSan Kang ect. Multi-objective Evolution Algorithm Based On Orthogonal Design [J]. Chinese Journal of Computers, 28(7): 1153-1162, 2005.

[8] Mezura-Montes, E., Coello Coello, C.A. A Simple Evolution Strategy to Solve Constrained Optimization Problems [J]. IEEE Transactions on Evolutionary Computation, 9(1): 1-17, 2005.

[9] Runarsson, T.P., Yao, X. Stochastic Ranking for Constrained Evolutionary Optimization [J]. IEEE Transactions on Evolutionary Computation, 4: 284–294, 2000.

[10] Deb, K. An Efficient Constraint Handling Method for Genetic Algorithms [J]. Computer Methods in Applied Mechanics and Engineering, 186: 311–338, 2000.

[11] E. Mezura-Montes, A. Coello Coello, and I. Tun-Morales. Simple Feasibility Rules and Differential Evolution for Constrained Optimization [J]. Proceedings of the 3rd Mexican International Conference on Artificial Intelligence, pages 707-716, 2004.

[12] Koziel, S., Michalewicz, Z. Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization [J]. Evolutionary Computation 7: 19–44, 1999.

[13] S. B. Hamida and M. Schoenauer, ASCHEA: New results using adaptive segregational constraint handling [J]. In Proc. Congress on Evolutionary Computation, vol. 1, May 2002, pp. 884–889.

[14] David H. Wolpert, William G. Macready. No Free Lunch Theorems for Optimization [J]. IEEE Transactions on Evolutionary Computation, 1(1): 67-82, 1997.

[15] K. Price and R. Storn, Home Page of Differential Evolution, Available Online at: http://www.ICSI.Berkeley.edu/ storn/code.html, 2008.