Journal of Computational Information Systems 8: 7 (2012) 2741–2750
Available at http://www.Jofcis.com

A CoEvolutionary Algorithm Based on Elitism and Gravitational Evolution Strategies

Yang LOU [1], Junli LI [1,*], Linpeng JIN [2], Gang LI [2]

[1] College of Computer Science, Sichuan Normal University, Chengdu 610066, China

[2] Information Science and Engineering College, Ningbo University, Ningbo 315211, China

Abstract

A global optimization algorithm based on elitism and gravitational evolution strategies is proposed, namely Elitism and Gravitational Evolution based CoEvolutionary Algorithm (EGCoEA). The search agents are divided into two subpopulations with the subpopulation of elites and the common subpopulation, and they updated via three methods. The values of Gravitational Measurement (GM) are used to define the relationships of the elites and the common individuals. The experimental study is carried out to test EGCoEA, compared with Maximal Gravitation Optimization Algorithm (MGOA) and M-Elite Coevolutionary Algorithm (MECA) by a series of typical benchmark functions, including both low-dimensional and high-dimensional problems. The results show EGCoEA performs better than the other two algorithms in solving these problems.

Keywords: Elitist Strategy; Gravitational Measurement; Gravitational Optimization; Coevolutionary Algorithm

1 Introduction

Evolutionary algorithms (EAs) are a series of problem-solving methods that based on simulation of the natural evolution system, and the development can be traced back to the 1950s. Compared with the classical optimization methods, evolutionary algorithms have many advantages, such as unconstrained by the search space limitations or the function types, and function gradient information is not essential, etc. EAs have been widely used for optimization problem. The classical evolutionary algorithms, such as Genetic Algorithms [1-2], Differential Evolution [3-5] etc. have some common disadvantages, i.e. a slow convergence speed, and might be trapped in local optimum value.

Evolution algorithms are population-based methods, which are constituted of individuals. The high fitness individuals, namely elite individuals play important roles to promoting of the evolution [6]. The elite strategy is a process that the one or several individuals with the best fitness are always remained into the next generation of evolution unconditionally. Elite strategy help the

*Corresponding author.
Email address: li.junli@vip.163.com (Junli LI).

excellent characteristics of elite individual pass to the offspring generation, which accelerates the convergence speed. For instance, M-Elite Coevolutionary Algorithm (MECA) [7] keeps M (e.g. M=20) elites of population to improve optimization performance; while the fireworks algorithm [8] uses the strategy that keep one elite individual, in order to maintain the most excellent character of the population throughout the evolution.

Wolpert and Macready proposed the "No Free Lunch theorems" (NFL) [9] in 1997, which has given a proof that any two kind of algorithms for solving all the optimization problems could meet an equivalent performance in average. Then in 2005, their research results show that cooperative evolution (coevolution) is not constrained by the NFL theorem [10]. For all the optimization problems, two types of coevolutionary algorithms may meet, the average performance of one may be better than the other, which implies that the performance of the coevolutionary algorithm has great potential for ascension.

The Maximal Gravitation Optimization Algorithm (MGOA) was firstly proposed by Jin and Li [11-12] in 2010. The MGOA updates individuals via the "gravitational grouping" and " gravitational elimination", and it was used to solve the low-dimensional optimization problems without constraint, but was unsuitable to high-dimensional problems. In this paper, we proposed the CoEvolutionary Algorithm Based on Elitism and Gravitational Evolution Strategies (EGCoEA), which inherits the gravitational frameworks, and the population is divided into two subpopulations, the elite subpopulation and the common subpopulation. The relations between individuals are calculated as a value, and then three update operations, i.e. two types of mutual updates and an ego update. The update processes follow the rule that "elite individuals inhibit common individuals; elite individuals and common individuals cooperate to evolution". Compared with MGOA, EGCoEA removes the time-consuming process of individual clustering, simplifies some operation, and more efficient updating strategies are used. Therefore, EGCoEA performs better in solving the optimization problems than MGOA, especially for the high-dimensional problems.

2 EGCoEA Algorithm Description

An unconstrained optimization problem can be generally described as follows:

Minimize $f(\boldsymbol{x}), \boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in S$ where $S \subseteq R^n$ is the search space, and $f(\cdot)$ is a single objective function, the vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ meet $\underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}}$ represents an individual. Vectors $\underline{\boldsymbol{x}} = (\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_n)$ and $\overline{\boldsymbol{x}} = (\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)$ represent the lower bound and upper bound of the search space, respectively.

Without loss of generality, we consider minimization problem only, and then the individual's fitness is defined as $Fitness(\boldsymbol{x}) = -f(\boldsymbol{x})$.

2.1 Elite strategy

The population is denoted as $Pop = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{NP})$, with a population size NP. In the initial stage of each iteration, the population is divided into two subpopulations based on individuals' fitness. M individuals with better fitness constitute the elite subpopulation $PopE = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M)$. The elite individuals should always meet relationship of $Fitness(\boldsymbol{x}_i) > Fitness(\boldsymbol{x}_j), 1 \leq i < j \leq M$. The other individuals constitute the common subpopulation $PopC$, which contains $NP - M$ individuals and need not be sorted through.

Figure 1 shows the elite individuals on the left of the diagram, and they are ordered by fitness. The right is common individuals, satisfied that any of the common individuals' fitness is worse than the worst elite individual's fitness.

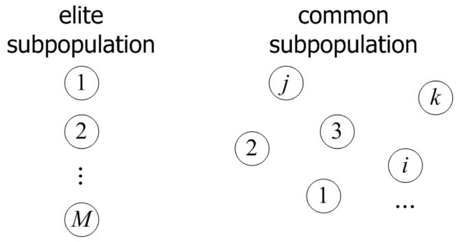


Fig. 1: Elitist strategy illustration

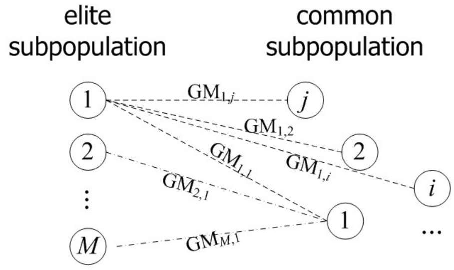


Fig. 2: An example of gravitational measurement

2.2 Gravitational optimization strategy

Gravitation is the attractive force between two objects, which exists between any two objects with masses. The value of gravitation is in proportion to the mass of either object, while in inverse proportion to the distance of them. Calculation of gravitation can be described as follows:

$$F = G\frac{m_1 \cdot m_2}{r^2} \tag{1}$$

where G stands for gravity constant, m_1 and m_2 stand for the mass of two objects, respectively, and represents the distance of objects. The idea of solving optimization problems with gravitational model is based on the fact that if the gravitational value is large, there may be two interpretations for it, thus either or both objects have big masses, or the distance of the objects is very small. Obviously the former case is what we hope, while the other is that should be avoid as far as possible, because which may lead to convergence in a local optimal. Inspired by formula (1), we present the values that describe the relationship of two individuals belong to two different subpopulations, namely "Gravitational Measurement" (GM) values, which are measured only between individuals of two different subpopulations, while individuals in the same subpopulation have no GM values.

In figure 2, the GM values of elite individual 1 and any common individual are calculated ($GM_{1,i}, 1 \leq i \leq NP - M$). Then, other elite individuals need to calculate GM values with the common individuals as well. Similarly, that is for each common individual, a GM value with any elite individual is calculated. In figure 2 there is an example, where the dot dash lines describe the relationship of elite 1 and any common individual. Gravitational measurement values are calculated as the following formula:

$$GM_{i,j} = \frac{Fitness(\boldsymbol{x}_i) \cdot Fitness(\boldsymbol{x}_j)}{r_{i,j} + K}, \boldsymbol{x}_i \in PopE, \boldsymbol{x}_j \in PopC \tag{2}$$

where,$r_{i,j}$ says the Euclidean distance between elite individual $\boldsymbol{x}_i$ and common individual $\boldsymbol{x}_j$. A constant $K \geq 1$ ensures that the denominator is nonzero. A larger K value offsets the influence of $r_{i,j}$, which ruled out the " a large GM value caused by the tiny distance " case. Theoretically, the best value of K should be:

$$K = K_0 = \sqrt{\sum_{i=1}^{n}(\overline{x}_i - \underline{x}_i)^2} \tag{3}$$

Generally, the random initializations make individuals equivalent, thus, K_0 represents the farthest distance that any two individuals may reach in the search space. However, experience shows that when $K \geq 1$, the influence of $r_{i,j}$ is fairly small, and not local-optimal-oriented. In reference [11], the authors firstly introduced K in order to guarantee the denominator nonzero, which performs well in optimizing low-dimensional problems.

2.3 Update methods

EGCoEA proposes three types of update methods: an ego update of elite individuals themselves, a mutual compulsory update of common individuals and a cooperation update of common individuals.

2.3.1 The ego update of elite individuals

The binary difference strategy [13] is used for an ordered subpopulation, and it performs well in the MECA algorithm [7], where binary difference is named as CCOII operator.

As shown in figure 3, new individuals are generated via "two-individual combination".

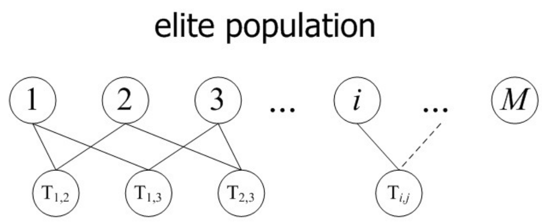


Fig. 3: Binary differential evolution

The ego updates of elite individuals themselves introduces a probability parameters R_T into classic binary difference, thus, the current temporary individual current dimension is generated in a probability of R_T, or it is kept. The purpose of introducing R_T is to better keep excellent features of elite individuals, and it could be described as follows:

$$\begin{cases} T_{i,j,k} = x_{i,k} \pm U_k(0,1) \cdot (x_{i,k} - x_{j,k}), \; if \;\; U_k(0,1) < R_T, 1 \leq i < M, i < j \leq M, 1 \leq k \leq n \\ T_{i,j,k} = x_{i,k} \qquad\qquad\qquad\qquad\qquad\qquad\quad , \; otherwise \end{cases} \tag{4}$$

where $T_{i,j,k}$ is the $kth(1 \leq k \leq n)$ dimension of the temporary individual $T_{i,j}$, which is generated by elite individual $\boldsymbol{x}_i$ and common individual $\boldsymbol{x}_j$, $U(0,1)$ stands for a uniform distribution random number between 0 and 1.

The worse individual of temporary individual $T_{i,j}$ and the worst elite is eliminated:

$$\begin{cases} x_M = T_{i,j}, & if \;\; Fitness(T_{i,j}) > Fitness(x_M) \\ x_M = x_M, & otherwise \end{cases} \tag{5}$$

2.3.2 Common individuals update

Common individuals have two types of update methods: compulsory update and cooperation update. In the evolutionary process, gravitational measurement values are used to describe the relationship between individuals belong to the two subpopulations.

For each elite individual, a common individual that has the minimum gravitational measurement value with the elite individual is selected, and compulsory update is implemented on the selected common individual. The compulsory updated individual will be completely replaced: either replaced by the elite has the greatest distance with it, or by a randomly selected elite individual.

Supposed that a common individual $c_j[i]$ has the minimum gravitational measurement value with the elite $\boldsymbol{x}_i$, and then it is to be mandatorily updated:

$$c_j' = \begin{cases} \max r_{i,j}(c_j[i]), & if \;\; U(0,1) < R_R, 1 \le i \le M, 1 \le j \le NP - M, 1 \le rand \le M \\ x_{rand} \qquad\qquad , & otherwise \end{cases} \tag{6}$$

where c_j' stands for the replacement of the individual c_j, and $\max r_{i,j}(c_j[i])$ is the individual has the greatest distance the elite $c_j[i]$, x_{rand} is the randomly selected elite. R_R is the replacement probability parameter.

For those common individuals are not mandatorily updated, a cooperation update is performed. First, search the elite of the largest GM value with it, and then a cooperation update by the method of binary difference is implemented. The newly produced individuals via the cooperation update still remain part genes of their parent, i.e. the non-mandatory-updated common individuals, and this is an incompletely replace process, and it is distinguished to the mandatorily updated (a completely replace process). Supposed that a non-mandatory-updated common individual c_j has the largest GM value with elite individual x_i , and then a binary difference is implemented:

$$c_{j,k}' = x_{i,k} \pm U_k(0,1) \cdot (x_{i,k} - c_{j,k}), 1 \le i \le M, 1 \le j \le NP - M, 1 \le k \le n \tag{7}$$

where $c_{j,k}'$ is the $kth(1 \le k \le n)$ dimension of the newly produced c_j', and other symbols with stated above.

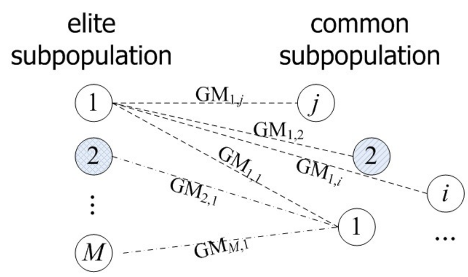


Fig. 4: An example of the update process of the common-individuals

Figure 4 shows an example, the elite individual 1 with its smallest GM value is common individual 2 (shadowed), hence the common 2 will be mandatorily updated. Then after every elite individual has eliminated a common individual, the rest common individuals are survived. We take another example that the common individual 1 is survived after the mandatory update, and then common 1 obtains the right to choose the elite individual 2 (shadowed), which has the maximal GM value with it. Common 1 updates itself through the cooperation of elite 2. Need to explain, mandatory update of common individuals does not equivalent to be deleted, and survived individual indicates non-mandatory-updated. When the population is divided into two subpopulations, and common subpopulation size is always greater than the elite, therefore there must survived individuals being.

2.4 Algorithm flowchart

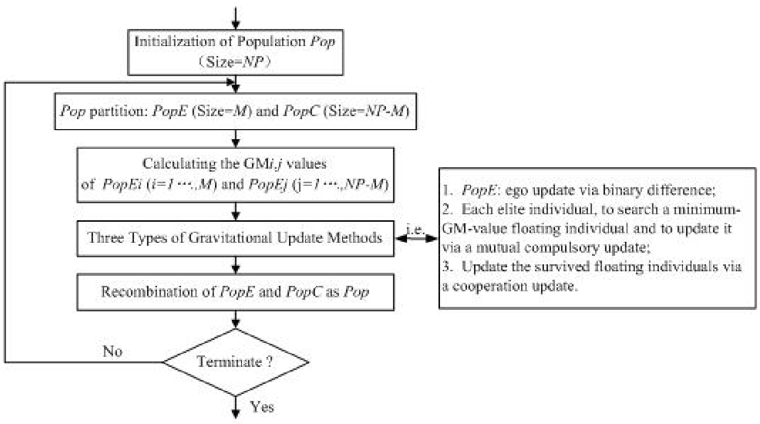


Fig. 5: The flow chart of EGCoEA

3 Simulation Experiment

3.1 EGCoEA and MGOA

Table 1 gives the experimental results of two algorithms in optimizing the low-dimensional problems. The benchmark functions are several typical problems selected from the reference [11], including the Shekel's Foxholes Function, Shubert Function, Trap Function and Easom's Problem.

The parameters setting of MGOA algorithm are in accordance with reference [11]. To be fair, EGCoEA uses similar settings: the population scale $NP = 30$, elite subpopulation scale $M = 10$, and $R_T = R_R = 0.5$. The halting conditions are with reference [11]: supposed fcur is current searched optimal solution, if $|fcur - fmin| < \varepsilon_0$ (in this paper $\varepsilon_0 = 10^{-6}$), or the current function evaluations exceed 150 000 times, the calculation stops. Each benchmark function takes 25 independently random experiments.

It can be inferred from Table 1 that for four results out of the five functions, EGCoEA performs better than MGOA does, while for function f1 the two algorithms get equivalent results. The

Table 1: Performances comparison between EGCoEA and MGOA

Benchmark Functions	fmin	*Mean Function Value*		*Standard Deviation*		*Mean Numbers of Function Evaluations*		*Success Rate*	
		MGOA	EGCoEA	MGOA	EGCoEA	MGOA	EGCoEA	MGOA	EGCoEA
f1	0.988 004	0.998 003 838	0.998 003 838	E-10	E-10	7247	3410	25/25	25/25
f2(min)	-186.730 909	-186.730 908 201	-186.730 908 565	E-7	E-7	**23285**	**1345**	**24/25**	25/25
f2(max)	-210.482 294	-210.482 291 687	-210.482 292 464	E-7	E-6	26459	3229	25/25	25/25
f3	-25	-24.949 913 035	-24.999 999 996	E-2	E-9	18799	4944	25/25	25/25
f4	-1	-0.999 999 993	-0.999 999 996	E-9	E-9	19197	1381	25/25	25/25

proposed algorithm get a more accurate standard in calculation of f3, while a relatively imprecise in the calculation of the f2 (max) than MGOA, and in the calculation of the rest problems, the two algorithms get equivalent results. EGCoEA algorithm requires less average number function evaluations than that of MGOA, especially for f2 (min), EGCoEA requires 1/10 number of function evaluation than MGOA requires. In addition, EGCoEA achieves an all-success results in the experiment (that is, to stop calculation when the conditions $|fcur - fmin| < 10^{-6}$ meet), relatively better than MGOA gets.

3.2 EGCoEA and MECA

Table 2 gives the experimental results of two algorithms in optimizing the high-dimensional problems. The benchmark functions contain 10 typical problems selected from the reference [7], including: f3∼f8, f11∼f13 and f15, and there rearranged in proper order in this paper as f1∼f10. Among the 10 benchmark functions, f1∼f3 are unimodal functions; f4 is a one step function; f5 is a quartic function with noise; f6∼f10 are multimodal function. In the experiment, functions f1∼f9 are 30-dimensional, and f10 is 100-dimensional.

The parameters setting of MECA algorithm are in accordance with reference [7]. To be fair, EGCoEA uses similar settings: the population scale $NP = 100$, elite subpopulation scale $M = 20$, and $R_T = R_R = 0.5$. The halting conditions are with reference [7]: when the current function evaluations exceed 300 000 times, the calculation stops. Each benchmark function takes 25 independently random experiments.

From Table 2, it is known that EGCoEA obtains far better results than MECA in calculation of f1, f2 and f7, that the results of EGCoEA can achieve its theory precision. For functions f8 and f9, reference [7] expounds the existing truncation errors of computers make it can achieve a limit precision only, even if the algorithm found the theory precision. For the 100-dimensional f10, two although get equivalent results, but the standard deviation of EGCoEA is 0, which means the proposed algorithm performs very stable. However, EGCoEA cannot always correctly calculate f3, and the statistical results show that in 25 times independent calculations, the results are 16 times of zero (optimal solutions), and 9 times of nonzero (incorrect solutions).

The average function evaluations of EGCoEA are slightly larger than that of MEGA, and it is caused by the iterative mechanism. Without regard to function f3, EGCoEA obtains an average number function evaluations of 300 612.4 times, which is slightly larger than 300 060.78 times of MECA by a difference of 551.62 times. And the proposed algorithm obtains the average results

Table 2: Performances comparison between EGCoEA and MECA

Benchmark Functions	fmin	*Mean Function Value*		*Standard Deviation*		*Mean Numbers of Function Evaluations*	
		EGCoEA	MECA	EGCoEA	MECA	EGCoEA	MECA
f1	0	**0**	3.274E-95	0	2.313E-94	300 555	300 057
f2	0	**0**	5.124E-2	0	9.732E-2	300 697	300 063
f3	0	**5.417**	7.973E-2	10.835	5.638E-1	300 842	300 055
f4	0	0	0	0	0	300 695	300 055
f5	0	6.710E-4	4.083E-4	2.277E-3	3.800E-4	300 478	300 060
f6	-12569.5	-125 69.486 6	-125 69.486 6	5.453E-12	7.350E-12	300 613	300 055
f7	0	**0**	3.844E-3	0	7.130E-3	300 687	300 066
f8	0	1.570E-32	1.571E-32	0	5.529E-48	300 672	300 068
f9	0	1.349E-32	1.350E-32	8.210E-48	1.106E-47	300 690	300 062
f10	-78.33236	-78.332 331 4	-78.332 331 4	0	1.005E-13	300 425	300 061

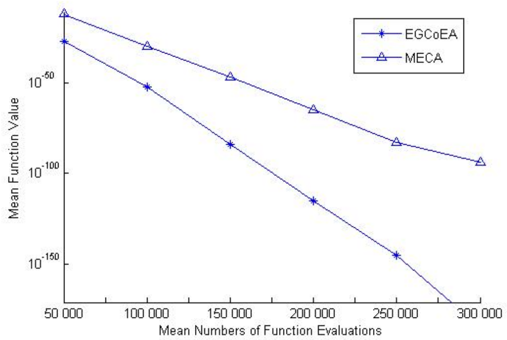
of all problems without regard to function f3 is 6.71 $\times 10^{-4}$, and that of MECA is 5.55 $\times 10^{-2}$. We defined two values to describe the data above, where I_1 is a value represents the improvement of precision, and I_2 stands for the increase of function evaluations.

$$I_1 = (5.55 \times 10^{-2} - 6.71 \times 10^{-4})/5.55 \times 10^{-2} = 0.988$$

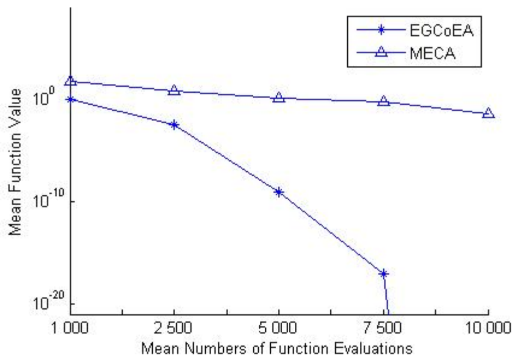
$$I_2 = (300612.4 - 300060.78)/300060.78 = 1.84 \times 10^{-3}$$

That is EGCoEA improves the precision by 98.8%, but increase only 0.184% in the number of average evaluation, thus the slightly increase of function evaluations in fact can be neglected.

Fig.6 shows the average convergence processes of EGCoEA and MECA in 25 times the average of the independent calculations of two 30-dimensional functions (f1 and f7). The figures reflect the proposed algorithm has higher convergence speed, especially for the f7, the new algorithm can calculate it within 10 000 evaluations, which is in accordance with the data in Table 2 that EGCoEA calculates more accurate than MECA in calculation of f7.



(a) f1



(b) f7

Fig. 6: The convergence process of solving (a) f1 and (b) f7 with two algorithms

4 Conclusions

Inspired by the universal gravitational phenomenon, combined with the elite strategy, we proposed a coevolutionary algorithm based on elitism and gravitational evolution strategies. Firstly, the population is divided into two subpopulations by elite strategy; the values of gravitational measurement are utilized to describe relationships of the individuals in different subpopulations. Then three types of update methods, i.e. an ego update of elite individuals, a mutual compulsory update and a cooperation update of common individuals are used for optimization. Compared with MGOA in low-dimensional optimization experiments and with MECA in high-dimensional optimization experiments, the experimental results show that EGCoEA algorithm has superior performance both in solving low-dimensional and high-dimensional problems

Furthermore, the proposed algorithm need to be further studied as well, such as the improvement of updating strategy; gravitational measurement method, etc. In addition, establish a discrete model of EGCoEA to fit the 0/1 knapsack problem, TSP, etc. are as well our next t work to do.

Acknowledgement

This research was supported by the Natural Science Foundation of China under Grant No. 60832003, the Natural Science Foundation of Zhejiang Province under Grant No.Y1100076, K.C. Wong Magna Fund in Ningbo University, and the Scientific Research Foundation of Graduate School of Ningbo University.

References

[1] J. H. Holland. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, 1975.

[2] Y. G. Xi, T. Y. Chai and W.M. Yun. Survey on Genetic Algorithm. Control Theory and Applications, 1996, 13(6): 697-708.

[3] R. Storn. Differential Evolution Research-Trends and Open Questions. Studies in Computational Intelligence. Advances in Differential Evolution. 2008, 143: 1-32.

[4] R. Storn. On the Usage of Differential Evolution for Function Optimization. Proc. of the Biennial Conference of the North American on Fuzzy Information Processing Society. USA. 1996: 519-523.

[5] Q. W. Yang, L. Cai and Y. C. Xue. A survey of Differential Evolution Algorithms. Pattern Recognition and Artificial Intelligence. 2008, 21(4): 506-513.

[6] C. W. Ahn, and R.S. Ramakrishna. Elitism-Based Compact Genetic Algorithms. IEEE Tans. on Evolutionary Computation. 2003, 7(4): 367-385.

[7] C. H. Mu, L. C. Jiao and Y. Liu. M-Elite CoEvolutionary Algorithm for Numerical Optimization. Journal of Software, 2009, 20(11): 2925-2938.

[8] Y. Tan and Y. C. Zhu. Fireworks Algorithm for Optimization. Lecture Notes in Computer Science, 2010, 6145: 355-364.

[9] D. H. Wolpert and W. G. Macready. No Free Lunch Theorems for Optimization. IEEE Trans. on Evolutionary Computation. 1997, 1(1): 67-82.

[10] D. H. Wolpert and W. G. Macready. CoEvolutionary Free Lunches. IEEE Trans. on Evolutionary Computation. 2005, 9(6): 721-735.

[11] L. P. Jin, J. L. Li, P. Wei and G. Chen. Maximal Gravitation Optimization Algorithm for Function Optimization. Pattern Recognition and Artificial Intelligence. 2010, 23(5): 653-662.

[12] Y. N. Qiu, J. L. Li and L. P. Jin. A Medical Image Registration Method Based on Probability and Gravity Optimization Model. Chinese Journal of Biomedical Engineering. 2010, 29(3): 345-352.

[13] Y. Lou, J. L. Li, Y. S. Wang. A Binary-Differential Evolution Algorithm Based on Ordering of Individuals. Proc. of the International Conference on Natural Computation. 2010, 5: 2207-2211.

[14] F. J. Solis and R. T. Wets. Minimization by Random Search Techniques. Mathematics of Operations Research. 1981, 6(1): 19-30.