International Journal of Computational Intelligence Systems, Vol. 6, No. 5 (September, 2013), 849-861

Gravitational Co-evolution and Opposition-based Optimization Algorithm

Yang Lou [1, 2]
[1.] *Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University*
Suzhou, 215123, China
[2.] *College of Computer Science, Sichuan Normal University*
Chengdu, 610066, China
lou.yang@hotmail.com

Junli Li
College of Computer Science, Sichuan Normal University
Chengdu, 610066, China
li.junli@vip.163.com

Yuhui Shi
Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University
Suzhou, 215123, China
yuhui.shi@xjtlu.edu.cn

Linpeng Jin
Suzhou Institute of Nano-Tech and Nano-Bionics, Chinese Academy of Sciences
Suzhou, 215123, China
lpjin2012@sinano.ac.cn

Received 19 June 2012

Accepted 12 December 2012

Abstract

In this paper, a Gravitational Co-evolution and Opposition-based Optimization (GCOO) algorithm is proposed for solving unconstrained optimization problems. Firstly, under the framework of gravitation based co-evolution, individuals of the population are divided into two subpopulations according to their fitness values (objective function values), i.e., the elitist subpopulation and the common subpopulation, and then three types of gravitation-based update methods are implemented. With the cooperation of opposition-based operation, the proposed algorithm conducts the optimizing process collaboratively. Three benchmark algorithms and fifteen typical benchmark functions are utilized to evaluate the performance of GCOO, where the substantial experimental data shows that the proposed algorithm has better performance with regards to effectiveness and robustness in solving unconstrained optimization problems.

Keywords: Gravitation; Evolution algorithm; Co-evolution; Opposition-based; Optimization.

1. Introduction

Evolutionary algorithms are a series of problem solving methods based on simulating the natural evolving system, the development of which can be traced back as early as 1950s. Compared with the traditional numerical optimization methods, evolutionary algorithms have many advantages, such as, being unconstrained by the search space limitations or/and the function types, as

well function gradient information being unnecessary for optimizing, etc. Evolutionary algorithms have been widely applied to solving optimization problems. The canonical evolutionary algorithms, including genetic algorithms (GA)[1], differential evolution (DE)[2][3], etc., have some common disadvantages, for example, the relatively slow convergence speed, the tendency of being trapped in local optimum/optimums, and the unstable robustness for various of problems. Therefore, these canonical methods need to be improved, since the problems to be solved in the both the real word and the theoretical scope are increasingly complicated.

Evolutionary algorithms are population-based algorithms, and the population usually consists of several to hundreds of individuals, depending on different strategies and applications. The individuals with high fitness values, namely elitist individuals or elites, play important roles in promoting the evolution process [4]. Elitist strategy is a method that keeps one or several individuals with the best fitness into offspring generation, unconditionally, which helps the excellent attributes or genes pass down to offspring, to accelerate the convergence speed. The elitist strategy has been applied to many evolutionary algorithms, for instances, the M-elite co-evolutionary algorithm (MECA)[5] keeps 20 elites, while the fireworks algorithm[6] keeps only an elite to maintain the currently most excellent character of the population throughout the evolution. Inspired by the elitist strategy, the population sorting strategy[7] was used to differentiate individuals with different fitness, which could be seen as an enhanced strategy of elitism.

D. H. Wolpert and W. G. Macready[8] proposed the no free lunch (NFL) theorem in 1997, which has given us a proof that any two algorithms for solving all the optimization problems could meet an equivalent performance in average. However, their work in 2005 shows that cooperative evolution (co-evolution) is not constrained by the NFL theorem [9], thus, for all the optimization problems, two types of co-evolutionary algorithms may meet that, the average performance of one may be better than the other, which implies that the performance of the co-evolutionary algorithms are of great potential for improvement.

Opposition-based differential evolution (ODE)[10][11] was firstly proposed by S. Rahnamayan, H. Tizhoosh, and et al. in 2006, which based on the principle of opposition-based learning, collaborated with DE. Since proposed in CEC'2006[11], ODE was widely and profoundly researched. In 2007 S. Rahnamayan and H. R. Tizhoosh added a variable jumping rate[12] into it, and in the same year proposed quasi-oppositional differential[13]. J. Tang and X. Zhao proposed an improved opposition-based DE[14] in 2010, which studied on the generating method of individuals. In 2011, A. Esmailzadeh and S. Rahnamayan presented the strategy of protective generation jumping[15], i.e., the generation jumping was stopped, iff it did not take effect.

As an improved version of the binary difference gravitational evolution (BDGE), the elitism and gravitational evolution based co-evolutionary algorithm (EGCoEA)[16] highlighted the status of the elitist individuals in the process of evolution, and gave up the process of individuals clustering. In EGCoEA, the search agents were divided into two subpopulations, thus, a subpopulation of elites and another common subpopulation, and they updated via three methods, included two types of mutual updates and an ego update. The evaluation of gravitational measurement (GM) is used to define the relationships of the elites and the common individuals. The update processes followed the following two principles: 1).the elitist individuals dominate the common individuals; 2).the elitist individuals and the common individuals cooperate to evolve.

In this paper, we proposed the gravitational co-evolution and opposition-based optimization (GCOO) algorithm that inherits the general gravitation based frameworks, including the subpopulations division, the calculation of GM values, etc., but with some important regulation, includes using probability parameters to control the update rate. Three essential update operations via gravitational co-evolution are used to update the agents, as well as an operation based on the so called opposition individuals. In order to integrate the gravitational and opposition-based models organically, the parameters and strategies are adjusted based on a series of trial-and-error experiments.

The rest of this paper is organized as follows: Section 2 gives the description of the two critical strategies, and in Section 3 the proposed algorithm is interpreted in detail, followed by a comprehensive series of experimental studies provided in Section 4. Finally, the work is

concluded in Section 5, and Appendix I lists all the benchmark functions.

2. Preliminaries: Description of Optimization Strategies

2.1. *Problem Definition*

An unconstrained optimization problem can be generally described as follows.

$$\text{Minimize } f(x),\ x=(x_1,x_2,\cdots,x_n)\in S$$

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

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

2.2. *Gravitational Co-evolution and Opposition-based Strategies*

A population is denoted as $Pop=(x_1,x_2,\cdots,x_{NP})$, with the population size *NP*. At the initial stage of an iteration, the population is divided into two subpopulations based on individuals' fitness values. *M* individuals with relatively better fitness constitute the elitist subpopulation, denoted as $PopE=(x_1,x_2,\cdots,x_M)$. The elitist individuals should always meet relationship of $Fitness(x_i)\ge Fitness(x_j)$, $1\le i<j\le M$. The rest individuals constitute the common subpopulation *PopC*, which keeps $NP-M$ individuals and need no sorting operation for individuals.

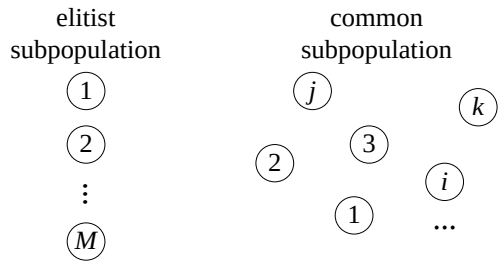


Fig. 1 Elitist Strategy Illustration

Fig. 1 shows a population that is partitioned into two subpopulations. The elitist individuals are on the left of the figure, and they are ordered according to their fitness values. The right part shows the common individuals, satisfied that any common individual's fitness is worse than the worst elitist individual's fitness.

2.2.1 Gravitational Co-evolution Strategy

Gravitation means 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 $F=G\cdot m_1\cdot m_2/r^2$, where *G* stands for gravity constant, m_1 and m_2 stand for the mass of two objects, respectively, and *r* represents the distance between the two objects.

The idea of solving optimization problems with gravitational model is based on the fact that if the calculated gravitational value is large, there may be two reasons for it, 1) either one or both objects have big masses; 2) the distance between the two objects is small. Obviously the first case is profitable, while the second not only brings benefit for the accuracy, but brings some risk as well, because it may lead to convergence in a local optimal. Inspired by the calculation of gravitation, i.e., $F=G\cdot m_1\cdot m_2/r^2$, we present the values that describe the relationship of two individuals belonging to two different subpopulations, namely the *Gravitational Measurement* (GM) values, which are calculated only between individuals from two different subpopulations, while individuals in the same subpopulation do not have the GM values.

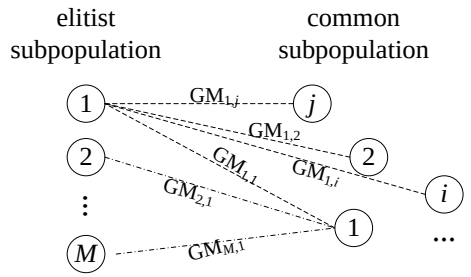


Fig. 2 An Example of Gravitational Measurement

In Fig. 2 the dotted lines, connecting the elitist individual circle 1 with all other common individuals, represent the GM values of elite circle 1 to the common individuals, which are calculated and denoted as $GM_{1,j}$, $1 \le j \le NP - M$. The other elites calculate the GM values to the common individuals in the same way, denoted as $GM_{i,j}$, $1 \le i \le M$, $1 \le j \le NP - M$. Inversely, for each common individual, a GM value to any elitist individual is also calculated. In Fig. 2 we take the common individual circle 1 as an example, where the dot dash lines describe the relationship of common circle 1 with any elitist individual. The relationships established by the gravitational measurements can be seen in two perspectives that no matter the leading roles are the elites or the commons individuals. Gravitational measurement values are calculated by Equation 1.

$$GM_{i,j} = \frac{Fitness(x_i) \cdot Fitness(x_j)}{r_{i,j} + K} \tag{1}$$

where, $x_i \in PopE$, $x_j \in PopC$, $r_{i,j}$ is the Euclidean distance between elitist individual x_i and common individual x_j . A constant $K \ge 1$ ensures that the denominator is nonzero. A larger K value will offset the influence of $r_{i,j}$, which may rule out the large GM value caused by the tiny distance case. Theoretically, the optimal value of K should be:

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

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, empirical setting shows that when $K \ge 1$, the influence of $r_{i,j}$ is fairly small, and not local-optimal-oriented. In [17], the authors firstly introduced K in order to guarantee the denominator nonzero, which performs well in optimizing low-dimensional problems.

2.2.2 Opposition-based Strategy

There are particles and antiparticles, i.e., opposite particles in the field of physics, subject and object in philosophy, subject and object in philosophy of science, good and evil in animism, and Yin and Yang symbols in the *Taichi*, where black and white represent Yin and Yang, respectively, belonging to the Chinese traditional culture. It seems the footprints of the opposition concept can be observed in many areas around us.

The opposition individual in evolutionary algorithms is defined as follows. Suppose a vector $x = (x_1, x_2, \cdots, x_D)$ represents an individual of the D-dimensional space and every dimension of $x_1, x_2, \cdots, x_D$ meets $x_i \in [\underline{x}_i, \bar{x}_i]$, $i = 1, 2, \cdots, D$. Then, the opposition individual of x is denoted as $x^\vee$, $x^\vee = (x_1^\vee, x_2^\vee, \cdots, x_D^\vee)$, each dimension of which is generated according to the Equation 2.

$$x_i^\vee = \underline{x}_i + \bar{x}_i - x_i \tag{2}$$

As shown in Fig. 3, there are three simplest examples of opposition individuals, including one, two and three dimensional cases. In the first case, individual and its opposite are symmetric about the central point. In the second case, opposition of $P(x_1, x_2)$ is $P^\vee$ with its ordinates of $(x_1^\vee, x_2^\vee) = (\underline{x}_1 + \bar{x}_1 - x_1, \underline{x}_2 + \bar{x}_2 - x_2)$. And in the last case, there are two pairs of individuals, thus, P_1 vs. $P_1^\vee$, and P_2 vs. $P_2^\vee$.

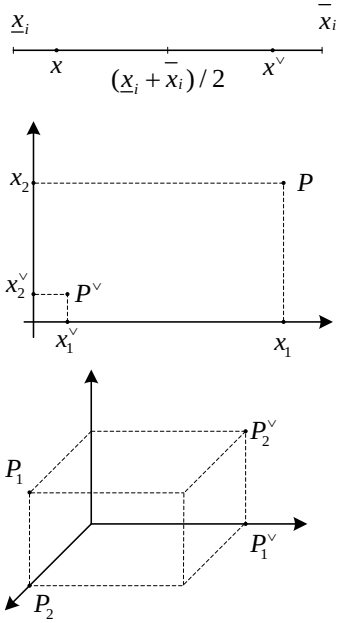


Fig. 3 Illustration of Individuals and the Corresponding Oppositions

3. Gravitational Co-evolution and Opposition-based Optimization Algorithm

There are three gravitational-based update methods and one opposition-based operation in the proposed algorithm. For the sake of efficiency improvement, not all the updates or operations are implemented all the time. We bring two methods to control the utilities of updates. The first is for different individuals, the elitists and the commons, different update methods are used; and the second is that a jumping rate is introduced to control the probability of the opposition-based operation.

3.1. *GCOO Algorithm Description*

GCOO proposes three types of update methods, which are inherited from the EGCoEA in framework, but with some modifications, including parameters and strategies in order to adapt to the new algorithm. The three update methods include an ego update of elitist individuals themselves, a mutual compulsory update of common individuals and a cooperation update of common individuals.

3.1.1 The Ego Update of Elitist Individuals

The binary difference strategy [16][18] is used for an ordered elitist subpopulation, and a similar idea is presented in the MECA algorithm, as the cuboid crossover operator II (CCOII) operator[5]. As shown in Fig. 4, newly temporary individuals are generated by a two-candidate-mutation, but there are always more than three candidates in the mutation strategies of canonical differential evolution.

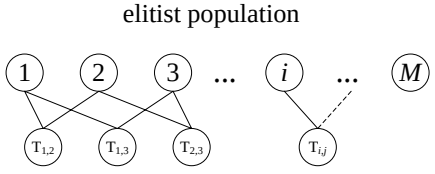


Fig. 4 Binary Differential Evolution

The ego updates of elitist individuals themselves introduces a probability parameters R_T into binary difference proposed in [16] and [18], thus, the current dimension of the temporary individual is generated within a probability of R_T, otherwise it is not updated and the original dimension is kept. The purpose of introducing R_T is to keep excellent features of elitist individuals better, and it can be described as Equation 3.

$$T_{i,j,k} = \begin{cases} x_{i,k} \pm U_k(0,1) \cdot (x_{i,k} - x_{j,k}), \text{ if } U_k(0,1) < R_T \\ x_{i,k}, \text{ otherwise} \end{cases} \quad (3)$$
$$1 \le i < M,\ i < j \le M,\ 1 \le k \le n$$

where $T_{i,j,k}$ is the kth$(1 \le k \le n)$ dimension of the temporary individual $T_{i,j}$, which is generated by elitist individuals x_i and x_j, $U(0,1)$ stands for a uniformly distributed random decimal between 0 and 1.

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

$$x_M = \begin{cases} T_{i,j}, \ if \ \ Fitness(T_{i,j}) > Fitness(x_M) \\ x_M, \ otherwise \end{cases} \quad (4)$$

3.1.2. Common Individuals Update

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

For each elitist individual, a common individual that has the minimum gravitational measurement value to the elitist 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 which has the greatest distance from it, or by a randomly selected elitist individual.

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

$$c'_j = \begin{cases} \max r_{i,j}(c_j[i]), \text{ if } U(0,1) < R_R \\ x_{rand}, \text{ otherwise} \end{cases} \quad (5)$$
$$1 \le j \le NP - M,\ 1 \le i \le M,\ 1 \le rand \le M$$

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

parameter, the analysis of which can be found in section 4.1, as well as the temporary probability parameter R_T .

For a common individual whom is not to be mandatorily updated, a cooperation update is performed. First, the elite which has the largest GM value from the common individual is searched, 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 means an incompletely (the mandatory updated is a complete one) replace process, and it is distinguished to the mandatorily updated. Supposed that a non-mandatory-updated common individual c_j has the largest GM value with the elitist individual x_i , and then a binary difference is implemented according to the Equation 6.

$$c'_{j,k} = x_{i,k} \pm U_k(0,1)\cdot(x_{i,k} - c_{j,k}),\quad 1 \le i \le M,\ 1 \le j \le NP - M,\ 1 \le k \le n \tag{6}$$

where $c'_{j,k}$ is the kth$(1 \le k \le n)$ dimension of the newly produced c'_j, and other symbols are as stated above.

Fig. 5 shows an example that the elite individual circle 1 with its all calculated GM values $GM_{1,j}$, (j=1,2,…,*NP-M*), among which the smallest is $GM_{1,2}$, i.e., the value between the common individual circle 2 (shadowed) and the elite individual circle 1 calculated according to the Equation 1. Then the common circle 2 will be mandatorily updated according to the Equation 5. After that every elitist individual has eliminated a common individual, there still leave some common individuals survived.

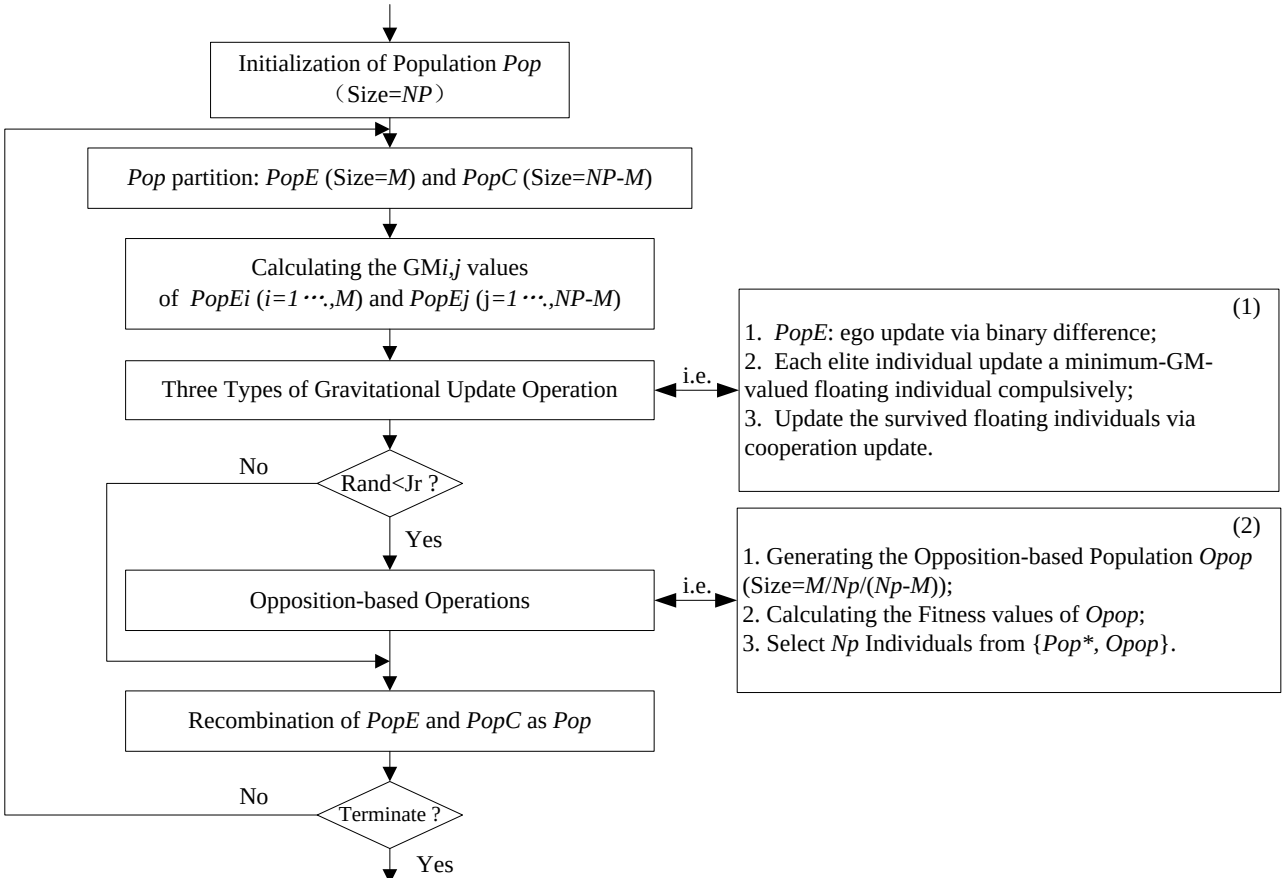


Fig. 6 Flowchart of Gravitational Co-evolution and Opposition-based Optimization Algorithm

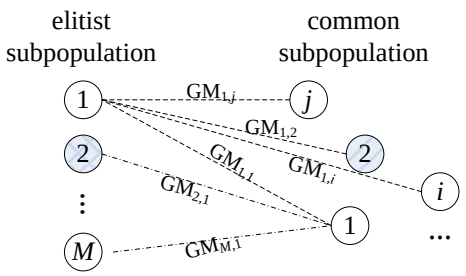


Fig. 5 An Example of the Update Process

In the reverse way, we take another example from the view that focuses on common individuals, where suppose the common individual circle 1 survives after the mandatory update, and then common circle 1 obtains the right to choose the elitist individual which has the largest GM value, i.e., elitist circle 2 (shadowed) in Fig. 5, which has the maximal GM value, i.e., $GM_{2,1}$ among all the $GM_{i,1}$, (i=1,2,…,M) values. Common circle 1 updates itself through the cooperation with elite circle 2 according to the Equation 6. Note that mandatory updates of common individuals are not equivalent to "be deleted", and the survived individual indicates non-mandatory-updated. Since the population is divided into two subpopulations, and common subpopulation size is always greater than the elites, there are always survived individuals in the common subpopulation.

3.1.3 Opposition-based Operation

The opposition-based operation differential evolution is simple and effective, where a constantly jumping rate is used to control in the current generation whether the opposition-based operation is to be implemented. The operation is as well a simple one and we take the elitist subpopulation *PopE* as an example to illustrate the opposition-based operation. Once the condition is satisfied, and the elitist individuals are ready for the operation. The calculation of opposite individuals can be described as Equation 7.

$$PopE^{\vee}_{i,j} = \underline{x}_j + \bar{x}_j - PopE_{i,j} \qquad 1 \le i \le M,\ 1 \le j \le D \tag{7}$$

where $PopE^{\vee}_{i,j}$ and $PopE_{i,j}$ denote the jth variable of the ith vector of the elitist subpopulation *PopE* and the opposite elitist subpopulation $PopE^{\vee}$, respectively. Consequently, we have a double-sized temporary population { *PopE* , $PopE^{\vee}$ }, and the next step is to select M fittest individuals from{ *PopE* , $PopE^{\vee}$ } to form the updated *PopE* .

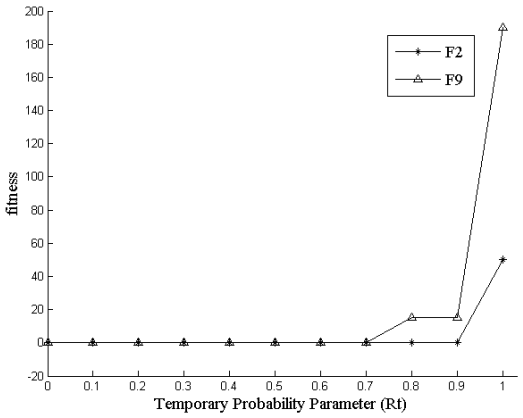
3.2. *GCOO Flowchart*

Fig.6 shows the flowchart of GCOO algorithm. The module (1) shows the three update methods which are based on gravitational co-evolution, and the module (2) is the opposition-based operation. Note that in the Step 3 of module (2), the $Pop*$ represents that the population could be *Pop* , *PopE* or *PopC* , depending on which opposition-based strategy is employed, and it will be discussed in Section 4.2.

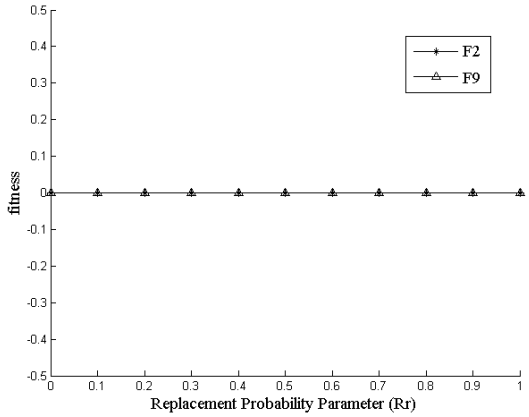
4. Experimental Studies

4.1. *Probability Parameters Analysis*

The temporary probability parameter for the update of the elitist individuals and the replacement probability parameter for common are introduced to control the performance of the update method in the gravitation based co-evolution. The experiments focus on the alterations that affect the performance of optimization, using the functions F2 and F9 in Appendix I. The data is of the average of 50 independently random trials, and each trial stops when it reaches 300,000 times objective function evaluations.



(a) R_T



(b) R_R

Fig. 7 Analysis of Probability Parameters

As shown in Fig. 7(a), the average performance of F9 declines as $R_T > 0.7$, and when $R_T > 0.9$ the result seems unsatisfied. A similar situation happens to F2, where when $R_T > 0.9$ the result starts to become bad. In Fig. 7(a), the overlapped two plots suggest that the change of R_R has no influence on optimization of F2 and F9. According to these, we can find that GCOO algorithm is not that sensitive to the change of the probability parameters when R_T is less than 0.7, hence, we may empirically set the two parameters as $R_T = R_R$ =0.5, which is acceptable for most of the optimization problems we meet.

4.2. *Operation Strategy Analysis*

As opposition-based operation is introduced into the population which consists of two subpopulations, we can easily find there are three possible strategies that the opposition-based operation can impact the performance. First one is that only the elitist individuals undergo the opposition-based operation, and we denote it as strategy S1. The second strategy should be that only the common individuals are operated by opposition-based operation, which is denoted as S2. The last one is both the two subpopulations undergo the opposition-based operation, and it is S3. In addition, we set a control group which does not include any opposition operation and we denote it as S0. Under the same of all the conditions, we compare the four strategies, using the functions F5, F9 and F11 in Appendix I.

Obviously, strategy S1 wins in the competitions of all, while S3 is better than S2. The results that strategy S3 obtains are worse than that of S0, which implies it is not workable to apply the opposite operation on the common individuals.

Table 1 Analysis of Four Types of Opposition-Based Operation Strategies

Benchmark Functions		F5	F9	F14
Dimension		30	30	100
Minimum		0	0	-99.60
Mean Function Values (MFV)	S0	5.417	2.387	-48.751
	S1	***3.610E-31***	***0***	***-77.415***
	S2	18.354	3.174	-45.810
	S3	1.691	***0***	-75.816

4.3. *Benchmark Test Settings*

A real valued function set F={F1, F2, ..., F15} including 15 well-known benchmark functions is employed to evaluate the performance of GCOO. These 15 functions were used as evaluation tool in [5] and partly in [16]. The mathematical forms of these functions can be found in these references. Among the 15 benchmark functions, F1~F5 are unimodal functions; F6 is a one-step function; F7 is a quartic function with noise; F8~F15 are multimodal functions. In the experiment, functions F1~F13 are 30-dimensional, while F14 and F15 are 100-dimensional.

To evaluate the impact of the proposed algorithm, we compare the experimental results found by GCOO with other three benchmark evolutionary algorithms. The designs and parameters settings of algorithms are summarized below.

1) GCOO: there are three parameters to set in GCOO, i.e., the population size $NP = 100$, the elitist subpopulation size $M = 20$, the jumping rate of opposition operation $Jr = 0.5$, and $R_T = R_R$ =0.5, with the opposition-based operation strategy S1.

2) MECA: the parameters of MECA are set in accordance with that in [5], i.e., the population size $NP = 100$, the elitist subpopulation size $M = 20$, and the probability of cuboid crossover $Pcu = 0.3$.

3) EGCoEA: the parameters of MECA are set in accordance with that in [16], i.e., the population scale

Table 2 Mean Function Values (MFV) and Standard Deviations (SD) of 50 Independently Random Experiments F1~F5

Benchmark Functions		F1	F2	F3	F4	F5
Dimension		30	30	30	30	30
Minimum		0	0	0	0	0
GCOO	MFV	0	0	0	3.610E-31	0
	SD	0	0	0	6.217E-58	0
EGCoEA	MFV	0	0	6.291E-270	5.417	0
	SD	0	0	0	10.835	0
MECA	MFV	4.228E-183	1.845E-110	3.274E-95	7.973E-2	5.124E-2
	SD	0	3.113E-110	2.313E-94	5.638E-1	9.732E-2
ODE	MFV	0	0	0	28.796 481 3	0
	SD	0	0	0	6.858E-2	0

Table 3 Mean Function Values (MFV) and Standard Deviations (SD) of 50 Independently Random Experiments F6~F10

Benchmark Functions		F6	F7	F8	F9	F10
Dimension		30	30	30	30	30
Minimum		0	0	-12 569.5	0	0
GCOO	MFV	0	5.755E-5	-12 569.486 6	0	4.441E-16
	SD	0	5.159E-5	0	0	0
EGCoEA	MFV	0	5.755E-5	-12 569.486 6	2.387	4.441E-16
	SD	0	5.159E-5	0	8.097	0
MECA	MFV	0	4.083E-4	-12 569.486 6	0	0
	SD	0	3.800E-4	7.350E-12	0	0
ODE	MFV	0	1.963E-4	-	0	4.441E-16
	SD	0	9.137E-5	-	0	0

Table 4 Mean Function Values (MFV) and Standard Deviations (SD) of 50 Independently Random Experiments F11~F15

Benchmark Functions		F11	F12	F13	F14	F15
Dimension		30	30	30	100	100
Minimum		0	0	0	-99.60	0
GCOO	MFV	0	1.350E-32	1.571E-32	-77.415	-78.332 331 4
	SD	0	6.817E-48	1.915E-47	3.021E-2	0
EGCoEA	MFV	0	1.350E-32	1.571E-32	-48.751	-78.332 331 4
	SD	0	6.817E-48	1.915E-47	5.930	0
MECA	MFV	3.844E-3	1.350E-32	1.571E-32	-98.709 489 1	-78.332 331 4
	SD	0	1.106E-47	5.529E-48	1.450E-1	1.005E-13
ODE	MFV	0	6.473E-1	7.125E-2	-	-
	SD	0	5.045E-1	4.058E-2	-	-

$NP = 100$, the elitist subpopulation scale $M = 20$, and $R_T = R_R = 0.5$.

4) ODE: the parameters of ODE are set in accordance with that in [10], i.e., the population size $NP = 100$, the differential scale factor $F = 0.5$, crossover rate $Cr = 0.9$, and jumping rate constant $Jr = 0.3$, with the mutation strategy of DE/rand/1/bin.

The stop criterion is, if the accumulated number of function evaluations exceeds 300,000 times, the iteration stops. For each benchmark function, we carry out 50 independently random experiments on searching for its optimum.

4.4. *Benchmark Test Results Analysis*

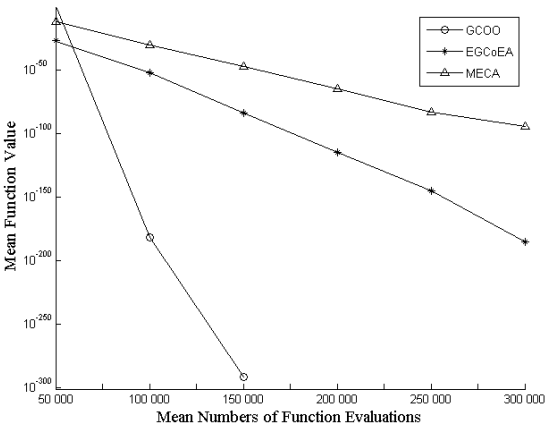
Tables 2 to 4 list the experimental results of benchmark test, where the data including the mean function values (MFV) and standard deviations (SD) of the optimization results are collected from an average of 50 independently random experiments.

As can be seen from Table 2, GCOO and ODE find the theoretical minimum values of F1, F2, F3 and F5, performs better than the other algorithms. As for F4, only GCOO finds a relatively satisfied result that the MFV reaches 10^{-31}, and the SD reaches 10^{-58}.

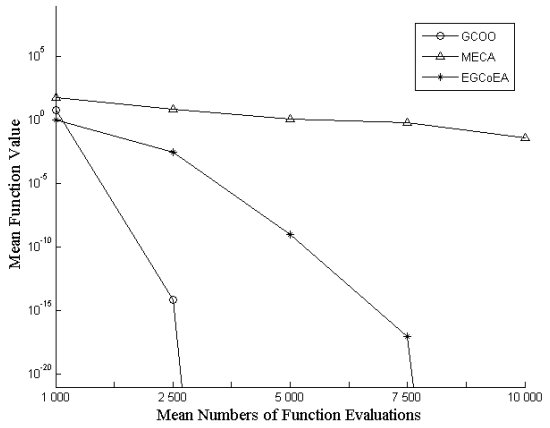
In Table 3, GCOO obtains the lead results of F6~F9, of which EGCoEA does unsatisfactory for F9, MECA does unsatisfactory for F7 and F8, and ODE for the MFV of F7. For F10, only MECA gets the theoretical minimum value, but the other three can finds a relative satisfied mean result reaching 10^{-16}.

In Table 4, where lists the relatively difficult problems to solve, both GCOO and EGCoEA find the relatively best minimum values of F11, F12, F13 and F15. As for F14, MECA find the best solution and GCOO ranks in the second place.

Fig. 8 shows the convergence processes of GCOO, EGCoEA and MECA of 50-trial-averaged data of functions F3 and F11. As can be seen form the figure, the proposed algorithm has much higher convergence speed compared with EGCoEA and MECA. As for F3, the GCOO algorithm can calculate it within 150,000 objective function evaluations; while for F11 which requires less than 3,000 evaluations. The converging figures of F3 and F11 are also partly reflected by the data in Tables 2 and 4, where GCOO finds better average results than MECA, while the plots of other comparison cannot be seen from tables because both GCOO and EGCoEA get the same average result when the terminal condition meets.



(a) F3



(b) F11

Fig. 8 The Convergence Process of Solving (a) F3 and (b) F11 with Two Algorithms

In summary, we tested GCOO, EGCoEA and MECA by 15 functions and tested ODE by 12 functions. The proposed GCOO ranks in the first of all, because the algorithm finds the most satisfying minimums in 13 out of 15 functions. The solutions that GCOO finds in F10 and F14 both rank the second place. This shows the effectiveness and stability of GCOO, which achieves a distinct improvement of both gravitation-based and opposition-based algorithms.

5. Conclusions

In this paper, the gravitational co-evolution and opposition-based optimization algorithm was proposed. With the framework of gravitational-based co-evolution, individuals were divided into two subpopulations according to their fitness values, and three types of gravitational-based updates were implemented. With the opposition-based operation, both gravitational-based and opposition-based methods collaborated to find the optimization. Three benchmark algorithms and fifteen benchmark functions were employed to evaluate the performance of GCOO, and the substantial experimental data has shown that the proposed algorithm has better effectiveness and robustness in solving unconstrained optimization problems.

Possible directions for future work include applying the algorithm into other optimization fields, e.g., multi-objective optimization and large-scaled problems, and improving the usage of cooperative strategies and parameters as well.

Appendix I

1. Sphere Model

$$F_1 = \sum_{i=1}^{D} x_i^2$$

where $x \in [-100,100]^D$

$F_{1\min} = F_1(0,\cdots,0) = 0$

2. Schwefel's Problem 2.22

$$F_2 = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} |x_i|$$

where $x \in [-10,10]^D$

$F_{2\min} = F_2(0,\cdots,0) = 0$

3. Schwefel's Problem 1.2

$$F_3 = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_i)^2$$

where $x \in [-100,100]^D$

$F_{3\min} = F_3(0,\cdots,0) = 0$

4. Schwefel's Problem 2.21

$$F_4 = \max_i \{|x_i|,\ 1 \le i \le D\}$$

where $x \in [-100,100]^D$

$F_{4\min} = F_4(0,\cdots,0) = 0$

5. Generalized Rosenbrock's Function

$$F_5 = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$$

where $x \in [-30,30]^D$

$F_{5\min} = F_5(1,\cdots,1) = 0$

6. Step Function

$$F_6 = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$$

where $x \in [-100,100]^D$

$F_{6\min} = F_6(0,\cdots,0) = 0$

7. Quartic Function i.e. Noise

$$F_7 = \sum_{i=1}^{D} i x_i^4 + random[0,1)$$

where $x \in [-1.28,1.28]^D$

$F_{7\min} = F_7(0,\cdots,0) = 0$

8. Generalized Schwefel's Problem 2.26

$$F_8 = -\sum_{i=1}^{D} (x_i \sin(\sqrt{|x_i|}))^2$$

where $x \in [-500,500]^D$

$F_{8\min} = F_8(420.9687,\cdots,420.9687) = -12569.5$

9. Generalized Rastrigin Function

$$F_9 = \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$$

where $x \in [-5.12,5.12]^D$

$F_{9\min} = F_9(0,\cdots,0) = 0$

10. Ackley's Function

$$F_{10} = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} z_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi z_i)) + 20 + e$$

where $x \in [-32,32]^D$

$F_{10\min} = F_{10}(0,\cdots,0) = 0$

11. Generalized Griewank Function

$$F_{11} = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$$

where $x \in [-600,600]^D$

$F_{11\min} = F_{11}(0,\cdots,0) = 0$

12. Generalized Penalized Function I

$$F_{12} = \frac{\pi}{D}\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2 \cdot [1 + 10\sin^2(\pi y_{i+1})]$$

$$+(y_D - 1)^2\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$$

where,

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(x_i - a)^m, & x_i < -a \end{cases}$$

$$y_i = 1 + \frac{1}{4}(x_i + 1)$$

and $x \in [-50,50]^D$

$F_{12\min} = F_{12}(1, \cdots, 1) = 0$

13. Generalized Penalized Function II

$$f_{13} = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2 \cdot [1 + \sin^2(3\pi x_{i+1})]$$

$$+(x_D - 1)^2 \cdot [1 + \sin^2(2\pi x_D)]\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$$

where,

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(x_i - a)^m, & x_i < -a \end{cases}$$

where $x \in [-50,50]^D$

$F_{13\min} = F_{13}(1, \cdots, 1) = 0$

14. Function 14

$$F_{14} = -\sum_{i=1}^{D} \sin x_i \cdot \sin^{20}\left(\frac{i \times x_i^2}{\pi}\right)$$

where $x \in [0,\pi]^D$

$F_{14\min} = -99.60$

15. Function 15

$$F_{15} = \frac{1}{N}\sum_{i=1}^{D}(x_i^4 - 16x_i^2 + 5x_i)$$

where $x \in [-5,5]^D$

$F_{15\min} = -78.33236$

Acknowledgements

This paper is partially supported by National Natural Science Foundation of China under Grant Numbers 60975080, 61273367, 60832003; Natural Science Foundation of Ningbo under Grant No.2012A610047; and Sichuan Science and Technology Support Plan under Grant No. 2013SZ0085.

References

1. J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.
2. R. Storn. Differential Evolution Research–Trends and Open Questions. Studies in Computational Intelligence. *Advances in Differential Evolution*. 2008, 143: 1–32.
3. 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.
4. C. W. Ahn and R. S. Ramakrishna. Elitism-Based Compact Genetic Algorithms. *IEEE Tansactions on Evolutionary Computation*. 2003, 7(4): 367–385.
5. C. H. Mu, L. C. Jiao and Y. Liu. M-Elite CoEvolutionary Algorithm for Numerical Optimization. *Journal of Software*, 2009, 20(11): 2925–2938.
6. Y. Tan and Y. C. Zhu. Fireworks Algorithm for Optimization. *Lecture Notes in Computer Science*, 2010, 6145: 355–364.
7. Y. Lou and J. L. Li. A Differential Evolution Algorithm Based on Ordering of Individuals. *Proc. of the 2010 International Conference on Industrial Mechatronics and Automation*, China. 2010, vol. 2: 105–108.
8. D. H. Wolpert and W. G. Macready. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*. 1997, 1(1): 67–82.
9. D. H. Wolpert and W. G. Macready. CoEvolutionary Free Lunches. *IEEE Transactions on Evolutionary Computation*. 2005, 9(6): 721–735.
10. S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama. Opposition-Based Differential Evolution. *IEEE Transactions on Evolutionary Computation*, 2008, 12(1): 64–79.
11. S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama. Opposition-Based Differential Evolution Algorithms. *Proc. of IEEE Congress on Evolutionary Computation*, 2006: 2010–2017.
12. S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama. Opposition-Based Differential Evolution (ODE) with Variable Jumping Rate. *Proc. of IEEE Symposium on Foundations of Computational Intelligence*, 2007: 81–88.
13. S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama. Quasi-Oppositional Differential Evolution. *Proc. of IEEE Congress on Evolutionary Computation*, 2007: 2229–2236.
14. J. Tang and X. J. Zhao. On the Improvement of Opposition-Based Differential Evolution. *Proc. of International Conference on Natural Computation*, 2010, 6: 2407–2411.

15. A. Esmailzadeh and S. Rahnamayan. Opposition-Based Differential Evolution with Protective Generation Jumping. *Proc. of IEEE Symposium on Differential Evolution*, 2011:1–8.
16. Y. Lou, J. L. Li, L. P. Jin and G. Li. A CoEvolutionary Algorithm Based on Elitism and Gravitational Evolution Strategies. *Journal of Computational Information Systems*, 2012, 8(7): 2741–2750.
17. 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.
18. Y. Lou, J. L. Li, and 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.