Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 308250, 7 pages
http://dx.doi.org/10.1155/2013/308250

Research Article

Opposition-Based Animal Migration Optimization

Yi Cao, Xiangtao Li, and Jianan Wang

School of Computer Science and Information Technology, Northeast Normal University, Changchun 130117, China

Correspondence should be addressed to Yi Cao; caoy370@nenu.edu.cn

Received 25 July 2013; Accepted 8 September 2013

Academic Editor: William Guo

AMO is a simple and efficient optimization algorithm which is inspired by animal migration behavior. However, as most optimization algorithms, it suffers from premature convergence and often falls into local optima. This paper presents an opposition-based AMO algorithm. It employs opposition-based learning for population initialization and evolution to enlarge the search space, accelerate convergence rate, and improve search ability. A set of well-known benchmark functions is employed for experimental verification, and the results show clearly that opposition-based learning can improve the performance of AMO.

1. Introduction

Many real world problems can be summarized as optimization problems. Optimization problems play an important role in both industrial application and scientific research. In the past decades, different optimization algorithms have been proposed. Among them, genetic algorithm may be the first and popular algorithm inspired by natural genetic variation and natural selection [1, 2]. Inspired by the social behavior of bird flocking or fish school, particle swarm algorithm was developed by Kennedy and Eberhart in 1995 [3, 4]. Artificial bee colony algorithm was proposed by Karaboga and Basturk in 2005, which simulates the foraging behavior of bee swarm [5, 6]. Ant colony optimization (ACO) which simulates the action of ants was first introduced by Dorigo [7, 8]. Animal Migration Optimization (AMO) as a new optimization algorithm inspired by animal migration behavior was first proposed by Li et al. in [9]. AMO simulates the widespread migration phenomenon in the animal kingdom, through the change of position, replacement of individual, and finding the optimal solution gradually. AMO has obtained good experimental results on many optimization problems.

Optimization algorithms often begin from an initial set of variables which are generated randomly and through iteration to obtain the global optimal solutions or the maximum of the objective function. It is known that the performance of these algorithms is highly related to diversity of particles; it may easily fall into local optima and has slow convergence rate and poor accuracy in the later stage of evolution. In recent years, many efforts have been done to improve the performance of different algorithms.

The concept of opposition-based learning (OBL) was first introduced by Tizhoosh [10]. The main idea is to consider current candidate solution and its opposite candidate in order to enlarge the search scope, and it also uses elite selection mechanism to speed up the convergence speed and find the optimal solution. It has been proved in [11], an opposite learning mechanism has more chance to be closer to the global optimum solution than a random candidate solution. The opposition-based learning idea has been successfully applied on GA [11] that PSO [12–16], DE [17, 18], artificial neural networks (ANN) [19, 20], and ant colony optimization (ACO) [21]; experimental results show that opposition-based learning can improve the search capabilities of the algorithm to some extent.

This paper presents an algorithm to improve the performance of AMO. In the absence of priori information about the population, we employ opposition-based learning during population initialization and population evolution. Through the introduction of opposition-based learning mechanism, it can transform solutions from current search space to a new search space to enlarge the search space. By means of selecting the better solution between current solution and the opposite solution, it will improve search ability and accelerate convergence rate, and it has more chance to find the global optima.

The rest of this paper is organized as follows. Section 2 briefly introduces the animal migration optimization algorithm. Section 3 gives a simple description of

```
For i = 1 to NP
    For j = 1 to D
        If rand > Pa
            x_{i,G+1} = x_{r1,G} + rand · (x_{best,G} − x_{i,G}) + rand · (x_{r2,G} − x_{i,G})
        End if
    End for
End for
```

Algorithm 1

opposition-based learning. Section 4 explains an implementation of the proposed algorithm, opposition-based AMO algorithm. Section 5 presents a comparative study among AMO, OPAMO, and other optimization algorithms on 23 benchmark problems. Finally, the work is concluded in Section 6.

2. Animal Migration Optimization

AMO is a new heuristic optimization algorithm inspired by the behavior of animal migration which is a ubiquitous phenomenon that can be found in all major animal groups, such as birds, mammals, fish, reptiles, amphibians, insects, and crustaceans [9].

In this algorithm, there are mainly two processes. In the first process the algorithm simulates how the groups of animals move from current position to a new position. During this process, each individual should obey three main rules: (1) move in the same direction as its neighbors; (2) remain close to its neighbors; (3) avoid collisions with its neighbors. We select one neighbor randomly and update the position of the individual according to this neighbor, as can be seen in the following formula:

$$x_{i,G+1} = x_{i,G} + \delta\left(x_{\text{neighborhood},G} - x_{i,G}\right), \tag{1}$$

where $x_{\text{neighborhood},G}$ is the current position of the neighborhood, δ is produced using a random number generator controlled by a Gaussian distribution, $x_{i,G}$ is the current position of ith individual, and $x_{i,G+1}$ is the new position of ith individual.

In the following process, the algorithm simulates how new animals are introduced to the group during the migration. During the population updating process, some animals will leave the group and some new animals will join the new population. We assume that the number of available animals is fixed. The animals will be replaced by some new individuals with a probability Pa. The probability is used according to the quality of the fitness. For the best fitness, the probability Pa is 1/NP. For the worst fitness, the probability is 1. This process can be shown in Algorithm 1, where $r_1, r_2 \in [1, \ldots, N_P]$ are randomly chosen integers and $r_1 \neq r_2 \neq i$. After producing a new solution, it will be evaluated and compared with $x_{i,G}$. If the objective fitness of $x_{i,G+1}$ is smaller than the fitness of $x_{i,G}$, $x_{i,G+1}$ is accepted as a new basic solution; otherwise, $x_{i,G}$ would be obtained.

To verify the performance of AMO, 23 benchmark functions chosen from the literature are employed. The results show that the proposed algorithm clearly outperforms some evolution algorithms from the literature.

3. Opposition-Based Learning

In the evolutionary algorithm, algorithm often starts from random initial population until a satisfactory solution is found. If no population prior information is known, the speed of evolution is relation to the distance between the initial particles and the best particle. If we select some initial particles close to the best individual, we can accelerate the convergence of the algorithm to some extent.

The opposition learning algorithm is a new type of reinforcement learning algorithm, and it has been proven to be an effective concept to enhance various optimization approaches [22]. This algorithm utilizes opposition learning mechanism to generate opposite population and employs elite selection to choose the individual closer to the best individual as the member of initial population, thus facilitating the overall evolutionary convergence speed.

For ease of description, we give the definition of the opposition point first.

Definition 1. Let $x \in [a, b]$ be a real number; the opposite point x^* is defined as

$$x^* = a + b - x. \tag{2}$$

Similarly, the definition can be extended to high dimensional space.

Definition 2. Let $x = [x_1, x_2, \ldots, x_D]$ as a point in D-dimension space, where $x_i \in [a_i, b_i]$, for all $i \in \{1, 2, 3, \ldots, D\}$; the point $x^* = (x_1^*, x_2^*, \ldots, x_D^*)$ is defined as the opposition value of x, in which

$$x_i^* = a_i + b_i - x_i. \tag{3}$$

Through the two previous definitions, we can conclude that the opposite learning algorithm is defined as follows.

Definition 3. $x = [x_1, x_2, \ldots, x_D]$ is a candidate solution in D-dimension space; assume $f(x)$ is the fitness function which is used to evaluate the fitness of the candidate. According to the definition of the opposite point, $x^* = (x_1^*, x_2^*, \ldots, x_D^*)$ is the opposite of x. If $f(x^*)$ is better than $f(x)$, then choose x^* as candidate instead of x; otherwise, x is obtained, through evaluating the fitness of x and the opposite x^* to get the better individual.

(1) Begin
(2) Set the generation counter $G = 0$, and randomly initialize with a population of NP animal X_i.
(3) Evaluate the fitness for each individual X_i.
(4) For $i = 1$ to NP do
(5) For $j = 1$ to D do
$OPX_{i,j} = k\left(a_j + b_j\right) - X_{i,j}$
(6) End for
(7) Calculate the fitness value of OPX_i
(8) End for
(9) Select NP fittest individual from $\{X_i\} \cup \{OPX_i\}$ as an initial population;
(10) While stopping criteria is not satisfied do
(11) For $i = 1$ to NP do
(12) For $j = 1$ to D do
(13) $X_{i,G+1} = X_{i,G} + \delta \cdot (X_{\text{neighborhood},G} - X_{i,G})$
(14) End for
(15) If rand < Pa then
(16) For $j = 1$ to D do
(17) $OPX_{i,G+1} = k\left(a_j + b_j\right) - X_{i,G+1}$
(18) End for
(19) End for
(20) Select NP fittest particles from $\{X_G\} \cup \{X_{G+1}\} \cup \{OPX_{G+1}\}$ as current population;
(21) For $i = 1$ to NP
(22) For $j = 1$ to D
(23) Select randomly $r1 \neq r2 \neq i$
(24) If rand > Pa then
(25) $X_{i,G+1} = X_{r1,G} + \text{rand} \cdot (X_{\text{best},G} - X_{i,G}) + \text{rand} \cdot (X_{r2,G} - X_{i,G})$
(26) End if
(27) End for
(28) For $i = 1$ to NP do
(29) Evaluate the offspring $x_{i,G+1}$
(30) If $x_{i,G+1}$ is better than x_i then
(31) $x_i = x_{i,G+1}$
(32) End If
(33) End for
(34) Memorize the best solution achieved so far
(35) End while
(36) End

Algorithm 2: Opposition-based AMO algorithm.

4. Enhanced AMO Using Opposition-Based Learning

4.1. Basic Concept. According to the probability principle, each randomly generated candidate solution has fifty percent probability chance away from the optimal solution compared to its opposition solution; AMO algorithm usually generates candidate solutions randomly in the search space as initial population because of the lack of prior information. During the optimization process, the calculation time of finding optimal solution will be changed according to the distance between the candidate solution and the optimal solution. By applying opposition-based learning, we not only evaluate the current candidate x but also calculate its opposition candidate x^*; this will provide more chance of finding the solution closer to the global optimal value.

Let x be a solution in current search space, $x \in [a, b]$; the new solution x^* in the opposite space is

$$x^* = \Delta - x. \tag{4}$$

From the above definition, we can infer $x^* \in [\Delta - b, \Delta - a]$. Obviously, through the opposite transformation, the center of search space is changed from $(a + b)/2$ to $(2\Delta - a - b)/2$.

Let $\Delta = k(a + b)$, where k is a real number in $[0, 1]$, and it is often generated randomly; this denotes the probability of reverse learning.

4.2. Opposition-Based AMO Algorithm. In this paper, we introduce a novel algorithm which combines opposition learning algorithm with AMO algorithm. Through the introduction of opposition learning mechanism, we consider the candidate solution in both current search space and its opposite space simultaneously. By selecting the better individuals in the two search spaces, it can provide more chance to find the global optimal solution and largely speed up the convergence.

During the process of population initialization and evolution, through the introduction of opposition learning mechanism, more candidate solutions will be considered, and we choose the most likely candidate solutions for evolution.

Table 1: Benchmark functions based on our experimental study.

Test function	D	Range	Optimum
$f_{01} = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$f_{02} = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$f_{03} = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]$	0
$f_{04} = \max_i \{\lvert x_i \rvert, 1 \le i \le D\}$	30	$[-100, 100]$	0
$f_{05} = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	0
$f_{06} = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	30	$[-100, 100]$	0
$f_{07} = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]$	0
$f_{08} = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	$[-500, 500]$	$-418.9829 * n$
$f_{09} = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]$	0
$f_{10} = -20 \exp\left(-0.2\sqrt{(1/D)\sum_{i=1}^{D} x_i^2}\right) - \exp\left((1/D)\sum_{i=1}^{D} \cos 2\pi x_i\right) + 20 + e$	30	$[-32, 32]$	0
$f_{11} = (1/400)\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(x_i/\sqrt{i}) + 1$	30	$[-600, 600]$	0
$f_{12} = (\pi/D)\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_i + 1)] + (yD - 1)^2 + \sum_{i=1}^{D} u(x_i, 10, 100, 4)\}$ $y_i = 1 + ((x_i + 1)/4)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
$f_{13} = 0.1\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_i + 1)] + (yD - 1)^2\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	30	$[-50, 50]$	0
$f_{14} = \left[(1/500) + \sum_{j=1}^{25} \left(1/\left(j + \sum_{i=1}^{2} (x_i - a_{ij})^6\right)\right)\right]^{-1}$	2	$[-65.53, 65.53]$	0.998004
$f_{15} = \sum_{i=1}^{11} [a_i - x_1(b_i^2 + b_i x_i)/(b_i^2 + b_i x_3 + x_4)]^2$	4	$[-5, 5]$	0.0003075
$f_{16} = 4x_1^2 - 2.1x_i^4 + (1/3)x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316285
$f_{17} = (x_2 - (5.1/4\pi^2)x_1^2 + (5/\pi)x_1 - 6)^2 + 10(1 - (1/8\pi))\cos x_1 + 10$	2	$[-5, 10] * [0, 15]$	0.398
$f_{18} = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)\right] \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2)]$	2	$[-5, 5]$	3
$f_{19} = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	$[0, 1]$	-3.86
$f_{20} = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6	$[0, 1]$	-3.32
$f_{21} = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.1532
$f_{22} = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.4029
$f_{23} = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.5364

We described the opposition-based population initial process in detail.

(1) Initialize the population $P(N_P)$ in the search space $[a, b]$ randomly.

(2) Calculate opposition population according to initial population; each dimension is calculated as follows:

$$\text{OP}_{ij} = k\left(a_j + b_j\right) - P_{ij}, \quad i = 1, 2, \ldots, N_P;\ j = 1, 2, \ldots, D, \tag{5}$$

where P_{ij} and OP_{ij} denote the jth variable of the ith vector of the population and opposite population, respectively, and k is a real number in $[0, 1]$ which denotes the probability of reverse learning.

(3) Choose N_P fittest individual as initial population from the union of random population P and opposition population OP according to the value of fitness.

During the evolution process, we still adopt opposition learning method to increase the opportunity of finding the optimal solution. When a new individual is generated or joined, its opposition value is considered; if the fitness of opposite solution is better than the new individual, the opposite solution is adopted; otherwise, the new individual is obtained.

However, opposition learning method could not be suitable for all kinds of optimization problems. For instance, when calculating the opposition candidate, the solution may jump away from the solution space. If this happens,

Table 2: Minimization result of benchmark functions $f1$–$f7$ for different algorithms.

Function	Algorithm	PSO	DE	FA	GSA	ABC	AMO	OP-AMO
$f01$	Mean	3.3340$e-10$	5.6016$e-14$	0.0017	3.3748$e-18$	2.9860$e-20$	8.6464$e-40$	1.4222$E-61$
	Rank	6	5	7	4	3	2	1
$f02$	Mean	6.6598$e-11$	4.7348$e-10$	0.0453	8.92115$e-09$	1.4213$e-15$	8.2334$e-32$	1.5342$E-40$
	Rank	4	5	7	6	3	2	1
$f03$	Mean	2.9847	2.8038$e-11$	0.0182	0.1126	2.4027$e+03$	8.8904$e-04$	1.0303$E-60$
	Rank	6	2	4	5	8	3	1
$f04$	Mean	7.9997	0.2216	0.0554	9.9302$e-10$	18.5227	2.8622$e-05$	0.6132
	Rank	6	4	3	1	7	2	5
$f05$	Mean	46.9202	0.2657	38.1248	20.0819	0.0441	4.1817	2.9114$E-06$
	Rank	7	3	6	5	2	4	1
$f06$	Mean	3.6925$e-10$	4.5028$e-14$	0.0017	3.3385$e-18$	3.0884$e-20$	0	0
	Rank	6	5	7	4	3	1	1
$f07$	Mean	0.0135	0.0042	0.0082	0.0039	0.0324	0.0017	0.0004
	Rank	6	4	5	3	7	2	1
Average rank		5.86	4	5.57	4	4.71	2.28	1.57
Overall rank		**7**	**3**	**6**	**3**	**5**	**2**	**1**

Table 3: Minimization result of benchmark functions $f8$–$f13$ for different algorithms.

Function	Algorithm	PSO	DE	FA	GSA	ABC	AMO	OP-AMO
$f8$	Mean	−8.8278$e+03$	−1.1276$e+04$	−6.2238$e+03$	−3.0499$e+03$	−1.2507$e+04$	−1.2569$e+04$	−1.1562$E+04$
	Rank	5	4	6	7	3	2	1
$f9$	Mean	18.2675	134.6789	23.5213	7.2831	0	0	0
	Rank	5	8	6	4	1	1	1
$f10$	Mean	3.8719$e-06$	7.4739$e-08$	0.0094	1.4717$e-09$	1.1946$e-09$	4.4409$e-15$	1.7183
	Rank	5	4	6	3	2	1	7
$f11$	Mean	0.0168	0	0.0025	0.01265	0	0	1
	Rank	6	1	4	5	1	1	7
$f12$	Mean	0.0083	4.7114$e-15$	8.8694$e-06$	2.0358$e-20$	1.1928$e-21$	1.5705$e-32$	1.5704$E-32$
	Rank	7	5	6	4	3	2	1
$f13$	Mean	4.6694$e-07$	3.1598$e-14$	1.2812$e-04$	5.6991$e-33$	2.2990$e-20$	1.3498$e-32$	1.3496$E-32$
	Rank	6	5	7	1	4	3	2
Average rank		5.67	4.5	5.83	4	2.33	1.67	3.17
Overall rank		**6**	**4**	**5**	**3**	**2**	**1**	**3**

the solution will be invalid; to avoid this case, the transformed candidate is assigned to a random value as follows:

$$x^* = \text{rand}(a, b) \quad \text{if } x < a \parallel x > b, \tag{6}$$

where rand (a, b) is a random number between a and b.

The opposition learning based AMO algorithm is described in Algorithm 2.

5. Experimental Results

To evaluate the performance of our algorithm, we applied it to 23 standards benchmark functions as shown in Table 1. These functions have been widely used in the literature.

The maximum numbers of generations are 1500 for f_1, f_6, f_{10}, f_{12}, and f_{13}, 2000 for f_2 and f_{11}, 3000 for f_7, f_8, and f_9, and 5000 for f_3, f_4,and f_5, 400 for f_{15}, 100 for f_{14}, f_{16}, f_{17}, f_{19}, f_{21}, f_{22}, and f_{23}, 30 for f_{18}, and 200 for f_{20}. The population size is 50 because this algorithm has two phases. The results of the algorithm on the 23 test problems are presented in Tables 2, 3, and 4.

As seen from Table 2 to Table 4, opposition-based AMO algorithm significantly improves the results on functions f_1, f_2, f_3, f_5, f_6, and f_7 and achieves the same performance with AMO on most other functions, and it is better than PSO, DE, ABC, and FA. This denotes that the algorithm can accelerate convergence rate and find better solutions for some optimization problems, and it did not lead to the premature of algorithm.

Figure 1 shows the evolutionary process of test function f_2 by AMO and opposition learning based AMO. The horizontal ordinate denotes the evolution iterations, and the vertical ordinate is the optimal value of objective function. We can see from the graph that the convergence rate of opposition-based AMO is obviously accelerated and the accuracy of optimization is also enhanced.

Table 4: Minimization result of benchmark functions $f14$–$f23$ for different algorithms.

Function	Algorithm	PSO	DE	FA	GSA	ABC	AMO	OP-AMO
$F14$	Mean	0.9980	0.9980	3.0273	5.9533	0.9980	0.9980	0.9980
	Rank	3	2	6	7	1	4	4
$F15$	Mean	$6.5867e-04$	$4.5400e-04$	0.0010	0.0048	$7.4715e-04$	$3.9738e-04$	$3.0748E-04$
	Rank	4	3	6	7	5	2	1
$F16$	Mean	−1.0316	−1.0316	−1.0314	−1.03163	−1.0316	−1.0316	−1.0316
	Rank	4	3	7	1	2	5	5
$f17$	Mean	0.3979	0.3979	0.3979	0.3979	0.3979	0.3979	0.3979
	Rank	5	1	6	1	7	1	1
$f18$	Mean	3.0001	3.0000	3.0123	3.7403	3.0000	3.0018	2.9999
	Rank	4	2	5	7	2	6	1
$f19$	Mean	−3.8628	−3.8628	−3.8613	−3.8625	−3.8628	−3.8628	−3.8629
	Rank	4	3	8	7	4	2	1
$F20$	Mean	−3.2554	−3.2174	−3.2741	−3.3220	−3.3220	−3.3220	−3.3223
	Rank	6	7	5	1	2	3	4
$F21$	Mean	−7.6393	−10.1532	−6.5633	−4.784	−10.1528	−10.0592	−10.1532
	Rank	5	1	6	7	3	4	1
$F22$	Mean	−7.3602	−10.4029	−10.4027	−6.5797	−10.4012	−10.3899	−10.4029
	Rank	6	1	3	7	4	6	1
$F23$	Mean	−8.9611	−10.5364	−10.2297	−8.2651	−10.5339	−10.4990	−10.5364
	Rank	5	1	6	7	3	4	1
Average rank		4.6	2.4	5.8	5.2	3.3	3.7	2
Overall rank		**5**	**2**	**7**	**6**	**3**	**4**	**1**

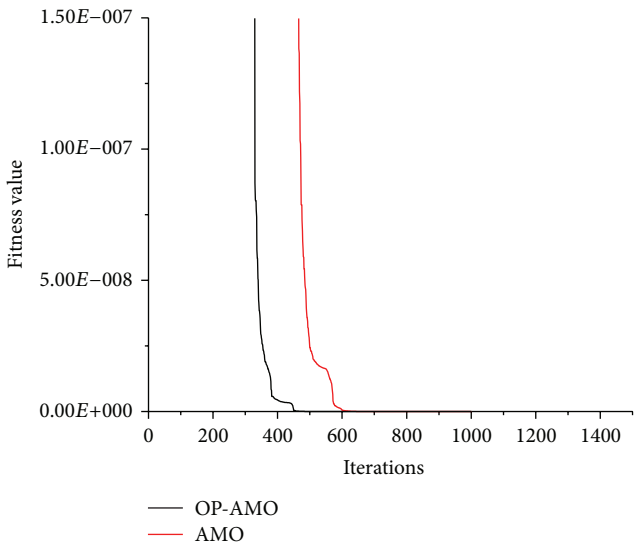

Figure 1: Evolutionary process of the two algorithms on function f_2.

6. Conclusions

A novel opposition-based AMO algorithm is proposed in this paper. This approach can provide more chance to find better solutions by transforming candidate solutions from current search space to a new search space. Experimental results show that, compared with previous AMO, the proposed algorithm is efficient in most of the test functions. However, we can also see from the experimental results that this algorithm is not suitable for all kinds of problems; for some optimization problems, algorithm has no significant improvement. How to improve the algorithm to adapt to a more optimization problem is worth of further study.

Conflict of Interests

The authors declare that they do not have any commercial or associative interest that represents a conflict of interests in connection with the work submitted.

References

[1] M. Melanie, *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, Mass, USA, 1999.

[2] S. N. Sivanandam and S. N. Deepa, *Introduction to Genetic Algorithms*, Springer, Berlin, 2008.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 14, pp. 1942–1948, December 1995.

[4] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, John Wiley & Sons, Hoboken, NJ, USA, 2005.

[5] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[6] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 687–697, 2008.

[7] M. Dorigo, *Optimization, learning and natural algorithms [Ph.D. dissertation]*, Politecnico di Milano, Milano, Italy, 1992.

[8] M.-H. Lin, J.-F. Tsai, and L.-Y. Lee, "Ant colony optimization for social utility maximization in a multiuser communication system," *Mathematical Problems in Engineering*, vol. 2013, Article ID 798631, 8 pages, 2013.

[9] X. Li, J. Zhang, and M. Yin, "Animal migration optimization: an optimization algorithm inspired by animal migration behavior," *Neural Computing and Applications*, 2013.

[10] H. R. Tizhoosh, "Opposition-based learning: a new scheme for machine intelligence," in *International Conference on Computational Intelligence for Modelling, Control and Automation (CIMCA '05)*, pp. 695–701, November 2005.

[11] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition versus randomness in soft computing techniques," *Applied Soft Computing Journal*, vol. 8, no. 2, pp. 906–918, 2008.

[12] Z. Lin and L. Wang, "A new opposition-based compact genetic algorithm with fluctuation," *Journal of Computational Information Systems*, vol. 6, no. 3, pp. 897–904, 2010.

[13] H. Lin and H. Xingshi, "A novel opposition-based particle swarm optimization for noisy problems," in *Proceedings of the 3rd International Conference on Natural Computation (ICNC '07)*, pp. 624–629, Haikou, China, August 2007.

[14] H. Wang, H. Li, Y. Liu, C. Li, and S. Zeng, "Opposition-based particle swarm algorithm with Cauchy mutation," in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 4750–4756, Singapore, September 2007.

[15] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu, and M. Ventresca, "Enhancing particle swarm optimization using generalized opposition-based learning," *Information Sciences*, vol. 181, no. 20, pp. 4699–4714, 2011.

[16] H. Wang, Z. Wu, S. Rahnamayan, and J. Wang, "Diversity analysis of opposition-based differential evolution-an experimental study," in *Proceedings of the International Symposium on Intelligence Computation and Applications*, pp. 95–102, 2010.

[17] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution algorithms," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 2010–2017, Vancouver, Canada, July 2006.

[18] R. S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[19] M. Ventresca and H. R. Tizhoosh, "Improving the convergence of backpropagation by opposite transfer functions," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '06)*, pp. 4777–4784, Vancouver, Canada, July 2006.

[20] M. Ventresca and H. R. Tizhoosh, "Opposite transfer functions and backpropagation through time," in *Proceedings of the IEEE Symposium on Foundations of Computational Intelligence (FOCI '07)*, pp. 570–577, Honolulu, Hawaii, USA, April 2007.

[21] A. R. Malisia and H. R. Tizhoosh, "Applying opposition-based ideas to the Ant Colony System," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '07)*, pp. 182–189, Honolulu, Hawaii, USA, April 2007.

[22] W.-f. Gao, S.-y. Liu, and L.-l. Huang, "Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 11, pp. 4316–4327, 2012.