2009 Ninth International Conference on Intelligent Systems Design and Applications

A Scalability Test for Accelerated DE Using Generalized Opposition-Based Learning

Hui Wang, Zhijian Wu, Shahryar Rahnamayan, and Lishan Kang

***Abstract*—In this paper a scalability test over eleven scalable benchmark functions, provided by the current workshop (Evolutionary Algorithms and other Metaheuristics for Continuous Optimization Problems - A Scalability Test), are conducted for accelerated DE using generalized opposition-based learning (GODE). The average error of the best individual in the population has been reported for dimensions 50, 100, 200, and 500 in order to compare with the results of other algorithms which are participating in this workshop. Current work is based on opposition-based differential evolution (ODE) and our previous work, accelerated PSO by generalized OBL.**

I. Introduction

Differential Evolution (DE) is an effective robust optimization algorithm which was proposed by Price and Storn in 1997 [1]. In this paper, an enhanced DE, based on generalized OBL (GOBL) is proposed to accelerate the convergence rate of classical DE. The GOBL was introduced in our previous work [10] which presented a general model for opposition-based learning (OBL). Accelerated DE by OGBL has been called GODE in this paper. In order to verify the performance of GODE, current work provides a scalability test over 11 benchmark functions, provided by the current workshop, for dimensions 50, 100, 200, and 500.

The rest of the paper is organized as follows. In Section II, the classical DE algorithm is briefly reviewd. The GOBL technique and its analysis are presented in Section III. Section IV gives an implementation of the proposed algorithm, GODE. In Section V, the test functions, parameters besides a comprehensive set of scalability experiments are provided. Finally, the work is summarized in Section VI.

II. A Brief Review of Differential Evolution

DE is a population-based stochastic search algorithm, and has been successfully applied to solve complex problems including linear and nonlinear, unimodal and multimodal functions. It has been shown that DE is faster and more robust on these functions than many other evolutionary algorithms [2].

Hui Wang, Zhijian Wu and Lishan Kang are with the State Key Laboratory of Software Engineering, Wuhan University, Wuhan, 430072 China (e-mail: wanghui_cug@yahoo.com.cn; zjwu9551@sina.com; kang_whu@yahoo.com). **Shahryar Rahnamayan** is with Faculty of Engineering and Applied Science, University of Ontario Institute of Technology (UOIT), 2000 Simcoe Street North, Oshawa, ON L1H 7K4 (e-mail: shahryar.rahnamayan@uoit.ca).

There are several variants of DE [1], where the most popular variant is shown by "$DE/rand/1/bin$" which is called classical version. The proposed algorithm is also based on this DE scheme. Let us assume that $X_i(t)(i = 1, 2, \ldots, N_p)$ is the ith individual in population $P(t)$, where ps is the population size, t is the generation index, and $P(t)$ is the population in the tth generation. The main idea of DE is to generate trial vectors. Mutation and crossover are used to produce new trial vectors, and selection determines which of the vectors will be successfully selected into the next generation.

Mutation–For each vector $X_i(t)$ in Generation t, a mutant vector V is generated by

$$V_i(t) = X_{i_1}(t) + F\left(X_{i_2}(t) - X_{i_3}(t)\right), \qquad (1)$$
$$i \neq i_1 \neq i_2 \neq i_3,$$

where $i = 1, 2, \ldots, N_p$ and i_1, i_2, and i_3 are mutually different random integer indices within $[1, N_p]$. The population size N_p should be satisfied $N_p \geq 4$ because i, i_1, i_2, and i_3 are different. $F \in [0, 2]$ is a real number that controls the amplification of the difference vector $(X_{i_2}(t) - X_{i_3}(t))$.

Crossover–Like genetic algorithms, DE also employs a crossover operator to build trial vectors by recombining two different vectors. The trial vector is defined as follows:

$$U_i(t) = \left(U_{i,1}(t), U_{i,2}(t), \ldots, U_{i,n}(t)\right), \qquad (2)$$

where $j = 1, 2, \ldots, n$ and

$$U_{i,j}(t) = \begin{cases} V_{i,j}(t), & \text{if } rand_j(0,1) \leq CR \vee j = l \\ X_{i,j}(t), & \text{otherwise} \end{cases}. \qquad (3)$$

$CR \in (0, 1)$ is the predefined crossover probability, and $rand_j(0, 1)$ is a random number within $(0, 1)$ for the jth dimension, and $l \in \{1, 2, \ldots, n\}$ is a random parameter index.

Selection–A greedy selection mechanism is used as follows:

$$X_i(t) = \begin{cases} U_i(t), & \text{if } f(U_i(t) < f(X_i(t))) \\ X_i(t), & \text{otherwise} \end{cases} . \quad (4)$$

Without loss of generality, this paper only considers minimization problem. If, and only if, the trial vector $U_i(t)$ is better than $X_i(t)$, then $X_i(t)$ is set to $U_i(t)$; otherwise, the $X_i(t)$ is unchanged.

III. Generalized Opposition-based Learning (GOBL)

A. *Opposition-Based Learning*

Opposition-based Learning (OBL) [18], [19] is a new concept in computational intelligence, and has been proven to be an effective method to EAs in many optimization problems [8], [20]–[24]. When evaluating a solution x to a given problem, simultaneously computing its opposite solution will provide another chance for finding a solution more closer to the global optimum [9].

Opposite Number [20]−Let $x \in [a, b]$ be a real number. The opposite number of x^* is defined by:

$$x^* = a + b - x. \quad (5)$$

Similarly, the definition is generalized to higher dimensions as follows.

Opposite Point [20]−Let $X = (x_1, x_2, ..., x_D)$ be a point in a D-dimensional space, where $x_1, x_2, ..., x_D \in R$ and $x_j \in [a_j,\ b_j]$, $j \in 1, 2, ..., D$. The opposite point $X^* = (x_1^*, x_2^*, ..., x_D^*)$ is defined by:

$$x_j^* = a_j + b_j - x_j. \quad (6)$$

By applying the definition of opposite point, the opposition-based optimization can be defined as follows.

Opposition-based Optimization [20]−Let $X = (x_1, x_2, ..., x_D)$ be a point in a D-dimensional space (i.e., a candidate solution). Assume $f(X)$ is a fitness function which is used to evaluate the candidate's fitness. According to the definition of the opposite point, $X^* = (x_1^*, x_2^*, ..., x_D^*)$ is the opposite of $X = (x_1, x_2, ..., x_D)$. If $f(X^*)$ is better than $f(X)$, then update X with X^*; otherwise keep the current point X. Hence, the current point and its opposite point are evaluated simultaneously in order to continue with the fitter one.

B. *Generalization of OBL Model*

Based on the concept of OBL, we propose a generalized OBL as follows [10]. Let x be a solution in current search space S, $x \in [a, b]$. The new solution x^* in the opposite space S^* is defined by [10]:

$$x^* = \Delta - x, \quad (7)$$

where Δ is a computable value and $x^* \in [\Delta - b, \Delta - a]$. It is obvious that the differences between current search space S and opposite search space S^* are the center positions of search space. Because the size of search range (indicates the size of interval boundaries) of S and S^* are $b - a$, and the center of current search space moves from $\frac{a+b}{2}$ to $\frac{2\Delta-a-b}{2}$ after using GOBL.

Similarly, the definition of GOBL is generalized to a D-dimensional search space as follows.

$$x_j^* = \Delta - x_j, \quad (8)$$

where $j = 1, 2, \ldots, D$.

By applying the GOBL, we not only evaluate the current candidate x, but also calculate its opposite candidate x^*. This will provide more chance of finding solutions closer to the global optimum. So it is important to investigate when the GOBL is beneficial.

Assume that the current candidate is x, and the corresponding candidate in the opposite space is x^*. The opposite candidate x^* is closer to the global optimum x^o than the current candidate x, if and only if

$$|x^* - x^o| < |x - x^o|. \quad (9)$$

Hence

$$\begin{aligned} & (x^* - x^o)^2 - (x - x^o)^2 < 0 \\ & \Rightarrow (x^* + x - 2x^o)(x^* - x) < 0 \\ & \Rightarrow (\Delta - 2x^o)(\Delta - 2x) < 0. \end{aligned} \quad (10)$$

It is obvious that

$$x < \frac{\Delta}{2}, \quad \text{if } x^o > \frac{\Delta}{2}, \quad (11)$$

$$x > \frac{\Delta}{2}, \quad \text{if } x^o < \frac{\Delta}{2}. \quad (12)$$

That is, the opposite candidate x^* is closer to the global optimum x^o than the current candidate x, when x^o and x are located at the different sides of $\frac{\Delta}{2}$.

However, according to No-Free-Lunch theorem [17], the GOBL could not be suitable for all kinds of optimization problems. For instance, the opposite candidate may be far away from the global optimum when solving multimodal problems. To avoid this case, a new elite selection mechanism based on population is used after the opposition. The specific use method of GOBL are described in Section V.

C. Four Different Schemes of Generalized OBL

Let $\Delta = k(a+b)$, where k is a real number. The generalized OBL model is defined by:

$$x^* = k(a+b) - x. \tag{13}$$

So the opposite candidate x^* is closer to the global optimum x^o than the current candidate x, when x^o and x are located at the different sides of $\frac{k(a+b)}{2}$.

Let us consider four typical GOBL schemes with different values of k as follows.

1) $k = 0$ (Symmetrical Solutions in GOBL, GOBL-SS)
The GOBL-SS model is defined by

$$x^* = -x, \tag{14}$$

where $x \in [a,b]$ and $x^* \in [-b,-a]$. The current solution x and opposite solution x^* are on the symmetry of origin.

2) $k = \frac{1}{2}$ (Symmetrical Interval in GOBL, GOBL-SI)
The GOBL-SI model is defined by

$$x^* = \frac{a+b}{2} - x, \tag{15}$$

where $x \in [a,b]$ and $x^* \in [-\frac{b-a}{2}, \frac{b-a}{2}]$. The interval of the opposite space is on the symmetry of origin.

3) $k = 1$ (Opposition-based learning, OBL)
When $k = 1$, the GOBL model is identical to Eq.5, where $x \in [a,b]$ and $x^* \in [a,b]$.

4) $k = rand(0,1)$ (Random GOBL, GOBL-R)
The GOBL-SI model is defined by

$$x^* = k(a+b) - x, \tag{16}$$

where k is a random number within $[0,1]$, $x \in [a,b]$ and $x^* \in [k(a+b)-b, k(a+b)-a]$. The center of the opposite space is at a random position between $-\frac{a+b}{2}$ and $\frac{a+b}{2}$.

For a given problem, it is possible that the opposite candidate may jump out of the definition domain $[X_{min}, X_{max}]$. When this happens, the GOBL will be invalid, because the opposite candidate is infeasible. To avoid this case, the opposite candidate is assigned to a random value as follows.

$$x^* = rand(a,b), \quad \text{If } x^* < X_{min} \mid\mid x^* > X_{max}, \tag{17}$$

where $rand(a,b)$ is a random number within $[a,b]$, and $[a,b]$ is the interval boundaries of current search space.

IV. GOBL-Based Optimization

If the interval of the current search space is symmetric with respect to the origin ($a = -b$), then $\Delta = a+b = 0$. According to Eq.10, the GOBL is beneficial when $x^o \cdot x < 0$ is satisfied. If the global optimum $x^o = 0$, then the GOBL is invalid because there does not exist a candidate x which satisfies $0 \cdot x < 0$. The global optimum x^o is located on the origin in many optimization problems [3], [7], [8], [11]. To have an asymmetric opposition, the interval boundaries $[a_j(t), b_j(t)]$ is dynamically updated according to the size of current search space. The new dynamic GOBL model is defined by [8]

$$X^*_{i,j} = k[a_j(t) + b_j(t)] - X_{i,j} \tag{18}$$

$$a_j(t) = \min(X_{i,j}(t)), \quad b_j(t) = \max(X_{i,j}(t)) \tag{19}$$

$$X^*_{i,j} = rand(a_j(t), b_j(t)), \quad \text{If } X^*_{i,j} < X_{min} \mid\mid X^*_{i,j} > X_{max} \tag{20}$$

$$i = 1,2,\dots,N_p, \quad j = 1,2,\dots,D, \quad k = rand(0,1),$$

where $X_{i,j}$ is the jth vector of the ith candidate in the population, $X^*_{i,j}$ is the opposite candidate of $X_{i,j}$, $a_j(t)$ and $b_j(t)$ are the minimum and maximum values of the jth dimension in current search space respectively, $rand(a_j(t), b_j(t))$ is a random number within $[a_j(t), b_j(t)]$, $[X_{min}, X_{max}]$ is the definition domain, N_p is the population size, $rand(0,1)$ is a random number within $[0,1]$, and $t = 1,2,\dots,$ indicates the generations.

V. Generalized Opposition-Based Differential Evolution (GODE)

In our previous work [10], GOBL was applied to PSO and the experimental results showed that the GOBL model with random k works better than the other three models in many benchmark function problems. So, the proposed approach GODE is also based on the random GOBL model in this paper.

Like ODE, the GODE uses the GOBL method to initialize population and produce new candidates in evolutionary generations. The original DE is chosen as a parent algorithm and the proposed GOBL model is embedded in DE to improve its performance. However, the embedded strategy of GODE is different from ODE. In the ODE, the opposition occurs with a probability, and the classical DE executes every generation. But in the GODE, if $rand(0,1) \leq p_o$, then execute the GOBL; otherwise execute the classical DE.

The pseudo-code of GODE is shown in Algorithm 1, where P is the current population, GOP is the transformed population after using GOBL, P_i is the ith individual in P, GOP_i is the ith individual in GOP, k is a random number within $[0,1]$, p_o is the probability of GOBL, N_p is the population size, n is the dimension size, $a_j(t), b_j(t)$ is the interval boundaries

Algorithm 1: Accelerated Differential Evolution based on generalized opposition-based learning (GODE).

```
1  Randomly initialize each individual in population $P$;
2  double $k = rand(0,1)$;
3  for $i = 1$ to $N_p$ do
4      for $j = 1$ to $n$ do
5          $GOP_{i,j} = k(a_j + b_j) - P_{i,j}$;
6          if $GOP_{i,j}$ is out of the definition domain then
7              $GOP_{i,j} = rand(a_j, b_j)$;
8          end
9      end
10     Calculate the fitness value of $GOP_i$;
11     NE++;
12 end
13 Select $N_p$ fittest individuals from $\{P, GOP\}$ as an initial population $P$;
14 while $NE \leq MAX_{NE}$ and $BFV > VTR$ do
15     if $rand(0,1) \leq p_o$ then
16         Update the dynamic interval boundaries $[a_j(t), b_j(t)]$ in $P$
           according to Eq.19;
17         $k = rand(0,1)$;
18         for $i = 1$ to $N_p$ do
19             for $j = 1$ to $n$ do
20                 $GOP_{i,j} = k[a_j(t) + b_j(t)] - P_{i,j}$;
21                 if $GOP_{i,j}$ is out of the definition domain then
22                     $GOP_{i,j} = rand(a_j(t), b_j(t))$;
23                 end
24             end
25             Calculate the fitness value of $GOP_i$;
26             NE++;
27         end
28         Select $N_p$ fittest individuals from $\{P, GOP\}$ as current
           population $P$;
29     end
30     else
31         for $i = 1$ to $N_p$ do
32             Randomly select 3 parents $P_{i1}$, $P_{i2}$ and $P_{i3}$ from $P$,
               where $i \neq i1 \neq i2 \neq i3$ ;
33             for $j = 1$ to $n$ do
34                 if $rand(0,1) < CR$ then
35                     $V_{i,j} = P_{i1,j} + F(P_{i2,j} - P_{i3,j})$;
36                     $U_{i,j} = V_{i,j}$;
37                 end
38                 else
39                     $U_{i,j} = P_{i,j}$;
40                 end
41             end
42             Calculate the fitness value of $U_i$;
43             NE++;
44             if $f(U_i) < f(P_i)$ then
45                 $P_i^{'} = U_i$
46             end
47             else
48                 $P_i^{'} = P_i$;
49             end
50         end
51     end
52 end
```

of current population, $rand(a_j(t), b_j(t))$ is a random number within $[a_j(t), b_j(t)]$, BFV is the best fitness value, VTR is the value-to-reach [8], NE is the number of evaluations, and MAX_{NE} is the maximum number of evaluations.

VI. Conducted Scalability Tests

A. Experimental Framework

For the experiments, the following eleven scalable benchmark problems have been considered:

1. F1-F6 of the CEC'2008 Special Session and Competition on Large Scale Global Optimization test suite [25].

2. Schwefel's Problem 2.22 (F7), Schwefel's Problem 1.2 (F8), Extended f10 (F9), Bohachevsky (F10), and Schaffer (F11), see [26] for their descriptions.

The requirements on the simulation procedure are the followings [26]:

1. Each algorithm is run 25 times for each test function, and the average error of the best individual of the population is computed. For a solution x, this measure is defined as: $f(x) - f(op)$, where op is the optimum of the function.

2. The study has been made with dimensions 50, 100, 200, and 500. The maximum number of fitness evaluations is $5000 \times D$. Each run stops when the maximal number of evaluations is achieved.

B. Setting Control Parameters

Parameter settings for all conducted experiments are as follows (the same setting has been used in literature cited after of each parameter):

- Population size, $N_p = 100$ [8]
- Differential amplification factor, $F = 0.5$ [8]
- Crossover probability constant, $C_r = 0.9$ [8]
- Probability of GOBL, $p_o = 0.4$ (selected based on our previous experiments)
- Maximum number of function calls, $\text{MAX}_{\text{NFC}} = 5000\times$ D (proposed by the current workshop) [26]
- Mutation Strategy: DE/rand/1/bin (classical DE) [8]

C. Numerical Results

Results for dimensions 50, 100, 200, and 500 are summarized in Tables I, II, III, and IV, respectively. In addition, in order to conduct some statistic tests during the workshop, the author will provide to the organizers an Excel file with all these results.

D. Results Analysis

As seen, for functions F1-F6 (shifted problems), when the dimension of the problems increases the average error ($f(x) - f(op)$) increases as well, which looks logical. But for functions F7-F11 (unshifted problems), GODE performs in a reverse manner. The main reason is that we use the same population size (N_p) for problems with different dimensions. This is unfair to lower dimensional problems, because higher dimensional problems have more maximum generation ($\text{MAX}_{\text{NE}}/N_p$) than lower dimensional ones. Another reason is that GODE is not sensitive to F7-F11 with different dimensions. If given

TABLE I
GODE RESULTS FOR D=50, EACH ALGORITHM IS RUN 25 TIMES FOR EACH TEST FUNCTION, AND THE AVERAGE OF ERROR ($f(x) - f(op)$, op: THE OPTIMUM THE FUNCTION) FOR THE BEST INDIVIDUAL OF THE POPULATION IS COMPUTED.

$Function$	$f(x) - f(op)$
F1	$1.6872e - 014$
F2	20.1736
F3	64.9945
F4	332.358
F5	$6.90133e - 004$
F6	$1.65117e - 008$
F7	$6.76794e - 067$
F8	$6.54375e - 080$
F9	$5.25139e - 032$
F10	$6.04275e - 136$
F11	$5.85362e - 032$

TABLE II
GODE RESULTS FOR D=100, EACH ALGORITHM IS RUN 25 TIMES FOR EACH TEST FUNCTION, AND THE AVERAGE OF ERROR ($f(x) - f(op)$, op: THE OPTIMUM THE FUNCTION) FOR THE BEST INDIVIDUAL OF THE POPULATION IS COMPUTED.

$Function$	$f(x) - f(op)$
F1	$3.12245e - 014$
F2	47.134
F3	252.605
F4	466.972
F5	0.00614746
F6	0.734186
F7	$3.22251e - 140$
F8	$1.9967e - 149$
F9	$5.55917e - 066$
F10	$5.64935e - 268$
F11	$5.83833e - 067$

TABLE III
GODE RESULTS FOR D=200, EACH ALGORITHM IS RUN 25 TIMES FOR EACH TEST FUNCTION, AND THE AVERAGE OF ERROR ($f(x) - f(op)$, op: THE OPTIMUM THE FUNCTION) FOR THE BEST INDIVIDUAL OF THE POPULATION IS COMPUTED.

$Function$	$f(x) - f(op)$
F1	$2.37789e - 010$
F2	71.0062
F3	748.518
F4	411.156
F5	0.0338863
F6	2.92399
F7	$6.1449e - 277$
F8	$1.19597e - 293$
F9	0
F10	0
F11	0

TABLE IV
GODE RESULTS FOR D=500, EACH ALGORITHM IS RUN 25 TIMES FOR EACH TEST FUNCTION, AND THE AVERAGE OF ERROR ($f(x) - f(op)$, op: THE OPTIMUM OF THE FUNCTION) FOR THE BEST INDIVIDUAL OF THE POPULATION IS COMPUTED.

$Function$	$f(x) - f(op)$
F1	0.0229091
F2	89.0436
F3	2688.548
F4	2690.83
F5	0.0317864
F6	12.2297
F7	0
F8	0
F9	0
F10	0
F11	0

enough MAX_{Gen}, it will find more accurate solutions. In order to investigate the mentioned statement, we have changed population size $N_p = 100$ (the constant one) with $N_p = D$ for function F7, results are as follows:

$D = 50,\ N_p = 50,\ f(x) - f(op) = 4.38552e - 173$
$D = 100,\ N_p = 100,\ f(x) - f(op) = 3.22251e - 140$
$D = 200,\ N_p = 200,\ f(x) - f(op) = 1.2945e - 116$

As seen, the average error increases by dimension and these results confirm our reasoning at least for F7. Population size can play a crucial role to obtain more accurate solutions. The overall performance of GODE will be more clear during comparison with other algorithms participating in the workshop.

VII. SUMMARY

In this paper, Differential Evolution Based on generalized opposition-based learning (GODE) is proposed. The GOBL is an enhanced opposition-based learning, which transforms candidates in current search space to a new search space. By simultaneously evaluating solutions in current search space and transformed space, we can provide more chance of finding better solutions. A scalability test over 11 scalable benchmark functions, provided for the current workshop, are conducted. Results for the dimensions 50, 100, 200 and 500 are reported.

REFERENCES

[1] R. Storn and K. Price, "Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimiz.*, vol. 11, pp. 341–359, 1997.
[2] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proc. Congr. Evol. Comput.*, 2004, vol. 2, pp. 1980–1987.

[3] J. Brest, S. Greiner, B. Bos̆ković, M. Mernik, and V. Z̆umer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. Evol. Comput.*, vol. 10, no. 6 pp. 646–657, 2006.

[4] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, vol.9, no. 6, pp. 448–462, 2005.

[5] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proc. Congr. Evol. Comput.*, 2005, vol.2, pp. 1785–1791.

[6] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaption for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, 2009.

[7] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Proc. Congr. Evol. Comput.*, 2008, pp. 1110–1116.

[8] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1 pp. 64–79, 2008.

[9] S. Rahnamayan, H.R. Tizhoosh, M.M.A Salama, "Opposition versus Randomness in Soft Computing Techniques," *Elsevier Journal on Applied Soft Computing*, Volume 8, March 2008, pp. 906-918.

[10] H. Wang, Z. Wu, Y. Liu, J. Wang, D. Jiang, and L. Chen, "Space transformation search: A new evolutionary technique," in *World Summit on Genetic and Evolutionary Computation*, 2009, (in press).

[11] X. Yao, Y. Liu ,and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, pp. 82–102, 1999.

[12] Z. Yang, J. He, and X. Yao, "Making a difference to differential evolution," in *Advance in Metaheuristics for Hard Optimization*, 2008, pp. 397–414.

[13] M. M. Ali and A. Törn, "Population set-based global optimization algorithms: Some modifications and numerical studies," *Comput. Oper. Res.*, vol. 31, no. 10 pp. 1703–1725, 2004.

[14] J. Sun, Q. Zhang, and E. Tsang, "DE/EDA: A new evolutionary algorithm for global optimization," *Info. Sci.*, vol. 169, pp. 249–262, 2004.

[15] H. Y. Fan and J. Lampinen, "A trigonometric mutation operation to differential evolution," *J. Global Optimiz.*, vol. 27, no. 1, pp. 105–129, 2003.

[16] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 107–125, 2008.

[17] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, pp. 67–82, 1997.

[18] H. R. Tizhoosh, "Opposition-based learning: a new scheme for machine intelligence" in *Proc. of Int. Conf. on Computational Intelligence for Modeling Control and Automation*, 2005, pp. 695–701.

[19] H. R. Tizhoosh, "Opposition-Based Reinforcement Learning" in *J. of Advanced Computational Intelligence and Intelligent Informatics*, vol. 10, no. 4, pp. 578–585, 2006.

[20] S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama, "Opposition-based differential evolution algorithms," in *Proc. Congr. Evol. Comput.*, 2006, pp. 2010–2017.

[21] S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama, "Opposition-based differential evolution for optimization of noisy problems," in *Proc. Congr. Evol. Comput.*, 2006, pp. 1865–1872.

[22] H. Wang, Y. Liu, S. Y. Zeng, H. Li, and C. H. Li, "Opposition-based particle swarm algorithm with Cauchy mutation," in *Proc. Congr. Evol. Comput.*, 2007, pp. 4750–4756.

[23] S. Rahnamayan, H.R. Tizhoosh, and M M. A Salama, "Quasi-Oppositional Differential Evolution," in *Proc. Congr. Evol. Comput.*, 2007, pp. 2229–2236.

[24] M. Ventresca, H. R. Tizhoosh, "Numerical Condition of Feedforward Networks with Opposite Transfer Functions," in *IEEE Int. Joint Conf. on Neural Networks*, 2008, pp. 3232–3239.

[25] K. Tang, X. Yao, P. N. Suganthan, C. MacNish, Y. P. Chen, C. M. Chen, Z. Yang, *Benchmark Functions for the CEC'2008 Special Session and Competition on Large Scale Global Optimization*, Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China, http://nical.ustc.edu.cn/cec08ss.php, 2007.

[26] Francisco Herrera, Manuel Lozano, Workshop: Evolutionary Algorithms and other Metaheuristics for Continuous Optimization Problems - A Scalability Test, http://sci2s.ugr.es/programacion/workshop/Scalability.html.