Hybridizing artificial bee colony (ABC) algorithm with differential evolution for large scale optimization problems

Nadezda Stanarevic

Abstract**—**Artificial bee colony (ABC) and differential evolution (DE) are two metaheuristics used for hard optimization problems. In this paper, a novel method called DEM-ABC is proposed to improve the exploitation process in ABC algorithm. The method combines differential evolution mutation strategies with original ABC algorithm for improving its convergence and performance. The proposed approach was tested by using a set of well-known large-scale unconstrained benchmarks problems. Comparisons show that DEM-ABC outperforms or performs similarly as the original ABC algorithms in terms of the quality of the resulting solutions.

Keywords**—** Artificial bee colony, Differential evolution, Large-scale optimization problems, Swarm intelligence.

I. Introduction

MANY recent years many nature inspired algorithms have been introduced due to the fact that many real-world optimization problems have become increasingly large, complex and dynamic. The size and complexity of the problems nowadays require the development of methods and solutions whose efficiency is measured by their ability to find acceptable results within a reasonable amount of time [1]. An ant colony, a flock of birds or an immune system are typical examples of a swarm system [2]. Swarm intelligence mimics the intelligent behavior of groups of individuals with a very limited intellectual capacity. The exact same principles of swarm intelligence in nature can be used in optimization algorithms. These algorithms can be classified into different groups depending on the criteria being considered, such as population based, iterative based, stochastic, deterministic, etc. An artificial swarm consists of a set of cooperating autonomous individuals, called agents. These agents satisfy their own objectives through cooperation with other agents. Communication and coordination are usually limited to a certain range, so that cooperation between agents only takes place locally [3]. Optimization is the process of finding the best way to use available resources, while at the same time not violating any of the required conditions. In simple terms, with optimization we attempt to maximize desirable properties and minimize undesirable characteristics. Users generally demand that a practical minimization technique should fulfill several requirements [4]:

- Ability to handle different type of problems.;
- Ease of use with few control variables.
- Good convergence mechanism to the global minimum in consecutive independent trials.

The optimization algorithms which are inspired by intelligent behavior of honey bees are among the most recently introduced techniques. Several approaches have been proposed to model the specific intelligent behaviors of honey bee swarms and were applied to solving different type of problems [5], [6], [7], [8], [9]. Since its invention by Karaboga in 2005, ABC algorithm has been successfully applied to many kinds of problems [10], [11], [12], [13], [14]. According to the various applications mentioned above, ABC algorithm confirmed its good performances, but we noticed an insufficiency in exploitation process. Inspired by differential evolution (DE) [4], we modified exploitation process by applying different DE mutation strategies. We name the modified ABC algorithm as Differential Evolution Mutation ABC (DEM-ABC).

The rest of this paper is organized as follows. In Sections 2 and 3, ABC and DE are briefly introduced. The modified ABC algorithm called DEM-ABC algorithm is presented in Section 4. Section 5 presents and discusses the experimental results. Finally, the conclusion is drawn in Section 6.

II. ABC algorithm

Generally, all modern heuristic algorithms work as follows: a population of individuals is randomly initialized and each individual represents a potential solution to the problem. The quality of each solution is evaluated by using a fitness function. A selection process is applied during each iteration of an EA in order to form a new population. The selection process is directed towards the fitter individuals to increase their chances of being included in the new population. This procedure is repeated until convergence is reached. The best solution found is expected to be a near-optimum solution [15], [16], [17].

Manuscript received December 25, 2011.

The research was supported by the Ministry of Science, Republic of Serbia, Project No. III-44006

N. Stanarevic is with the Faculty of Mathematics, University of Belgrade, Belgrade, Serbia, e-mail: srna@stanarevic.com

Artificial bee colony (ABC) is a relatively new member of swarm intelligence. ABC tries to model natural behavior of real honey bees in food foraging. In the ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. The number of employed bees is equal to the number of food sources and an employed bee is assigned to one of the sources.

There are three main parts in the ABC algorithm: sending the employed bees onto the food sources and measuring their nectar amounts; selecting the food sources by the onlookers; determining the scout bees and finding new possible food sources. At the initialization stage, a set of food source positions are randomly selected by the bees.

In the ABC algorithm, the position of a food source represents a possible solution of the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution.

Short pseudo-code of the ABC algorithm is given below [4]:

```
Initialize the population of solutions
Evaluate the population
Produce new solutions for the employed bees
Apply the greedy selection process
Calculate the probability values
Produce the new solutions for the onlookers
Apply the greedy selection process
Determine the abandoned solution for the scout, and replace
it with a new randomly produced solution
Memorize the best solution achieved so far
```

An onlooker bee chooses a food source depending on the probability value associated with that food source, pi, calculated by the following expression

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{1}$$

where fiti is the fitness value of the solution i which is proportional to the nectar amount of the food source in the position i.

In order to produce a candidate food position from the old one in memory, the ABC uses the following expression

$$\upsilon_{i,j} = x_{i,j} + \phi_{i,j}(x_{i,j} - x_{k,j}) \tag{2}$$

where k ∈ {1, 2,.., SN} and j ∈ {1, 2,...,D} are randomly chosen indexes. A greedy selection mechanism is employed as the selection operation between the old one and the candidate [4]. If the new food has equal or better nectar than the old source, it replaces the old one in the memory. Otherwise, the old one remains. Providing that a position cannot be improved further through a predetermined number of cycles, the food source is assumed to be abandoned.

The value of predetermined number of cycles is an important control parameter of the ABC algorithm, which is called "limit" for abandonment. In the ABC, the parameter limit is calculated using the formula SN*D, where SN is the number of solutions and D is the number of variables of the problem.

ABC algorithm combines four different selection processes: 1) for discovering promising regions a global selection process is used by the artificial onlooker bees, 2) a local selection process carried out in a small area by the employed and the onlookers bees for determining a neighbour food sources around the current source, 3) a greedy selection process (which is also local selection process) carried out by all bees if the nectar amount of the new source is better than the best memorized food source achieved so far. Otherwise, the bee keeps the present one in the memory. 4) a random selection process, exploration, carried out by scouts [4].

III. DE ALGORITHM

A particular EA that has been used for global optimization over continuous spaces is differential evolution (DE). DE is a simple yet powerful evolutionary algorithm proposed by Price and Storn that has been successfully used for solving single-objective optimization problems. Like other metaheuristic methods, two fundamental processes drive the evolution of a DE population: the variation process, which enables exploring different regions of the search space, and the selection process, which ensures the exploitation of previous knowledge based on the fitness values.

Differential evolution is a parallel direct search method which utilizes NP D-dimensional parameter vectors xi,G, i = 1, 2, … , NP as a population for each generation G. Since the parameter vectors are likely to be changed over different generations, we may adopt the following notation for representing the ith vector of the population at the current generation as

$$X_{iG}=[x_{1,i,G}, x_{2,i,G}, x_{3,i,G},\ldots x_{D,i,G},]. \tag{3}$$

NP does not change during the minimization process. DE algorithm starts with an initial population vector, which is randomly generated with no preliminary knowledge about the solution space. The initial population should cover the entire search space constrained by the lower and upper bounds as much as possible.

DE generates new parameter vectors by adding the weighted difference between two population vectors to a third vector – mutation [18]. A candidate replaces the parent only if it is better than its parent.

DE guides the population towards the vicinity of the global optimum through repeated cycles of mutation, crossover and selection. General procedure and main steps (mutation, crossover, and selection) of the DE algorithm are shown in Fig.1.

During the mutation process for each individual X in generation t an associated mutant individual Y can be created by using one of the mutation strategies [19], [20] which are explained later in this section.

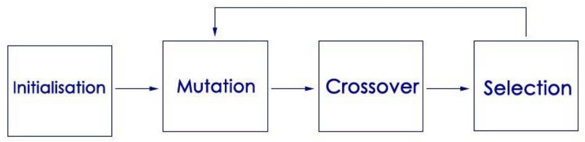

Fig. 1. General Evolutionary Algorithm Procedure

The basic DE algorithm initializes individual solution with random positions in the search-space. For each solution *x* three solutions *xr[1], xr[2], xr[3]* have to be picked from the population at random. Selected solutions must be distinct from each other as well as from current solution *x*. The mutation proces adds more genetic material into the population in order to avoid being trapped in a local optimum.

To increase the diversity of the population DE algorithm utilizes a crossover operation after mutation process. The DE family of algorithms can use two kinds of crossover schemes: exponential and binomial [3], [20], [23]. The main objective of crossover is to explore new areas of the search space.

To keep the population size constant over subsequent generations, the next step of the algorithm calls for selection to determine whether the base or the mutant vector survives to the next generation. If *f(x)* is the function to be minimized, selection process can be described as follows:

$$X_{iG+1} = \begin{cases} \text{mutant vector} & \text{if } f(\text{mutant vector}) \leq f(\text{base vector}) \\ \text{base vector} & \text{if } f(\text{mutant vector}) > f(\text{base vector}) \end{cases} \quad (4)$$

where XiG+1 is the vector of the population at the next generation. In general, selection tends to reduce the diversity of a population, while mutation increases it.

A general notation for DE algorithm is DE/x/y/z where x specifies the base vector to be mutated, y is the number of difference vectors, and z denotes the crossover scheme.

The basic mutation strategy is shown as rand/1 :

$$rand/1: y_{i,j}^t = x_{r[1],j}^t + F * (x_{r[2],j}^t - x_{r[3],j}^t) \quad (5)$$

where *r[k] k∈{1, 2, . . . 5}* is a uniformly distributed random integer number in the range [1, NP], j∈{1, 2, ... n}, F∈[0,2] is an amplification factor.

In the next equation best/1 mutant solution y is created by using two randomly selected solutions $x_{r[1]}$, $x_{r[2]}$, and the best individual $x_{best,j}^t$ in the population at generation t:

$$best/1: y_{i,j}^t = x_{best,j}^t + F * (x_{r[1],j}^t - x_{r[2],j}^t) \quad (6)$$

Highly beneficial methods *currenttobest/1*, *best/2*, and *rand/2* use two additional solutions compared to the basic mutation strategy *rand/1*. The usage of additional different solution best seems to improve the diversity of the population which affects the search direction which is guided by best solution.

$$currenttobest/1: \\ y_{i,j}^t = x_{i,j}^t + F(x_{best,j}^t - x_{i,j}^t) + F(x_{r[1],j}^t - x_{r[2],j}^t) \quad (7)$$

$$best/2: \\ y_{i,j}^t = x_{best,j}^t + F(x_{r[1],j}^t - x_{r[2],j}^t) + F(x_{r[3],j}^t - x_{r[4],j}^t) \quad (8)$$

$$rand/2: \\ y_{i,j}^t = x_{r[1],j}^t + F(x_{r[2],j}^t - x_{r[3],j}^t) + F(x_{r[4],j}^t - x_{r[5],j}^t) \quad (9)$$

Compared to rand/1 and rand/2, mutation strategies such as best/1, currenttobest/1 and best/2 benefit from their fast convergence by incorporating best solution information in the mutation strategies. The main problem with adding the information about the best solution in the mutation strategy is a premature convergence, since the population diversity is decreased. In view of the fast but less aggressive convergence performance currenttobest/1 mutation strategy uses parent solution xi,j (current solution) in the process of creating the associated mutant solution y.

The amplification factor F is a positive real number, and most suggested to be randomized within a range of (0, 1+) [22]. While there is no upper limit on F, effective values are rarely greater than 1.0 [3]. In the classical DE algorithm factor F is pre-defined, and it doesn't change during the evolution.

Although the DE algorithm has been shown to be a powerful evolutionary algorithm for optimization, users are still faced with the problem of hand-tuning the main parameters [24], [25]. As a solution, new self-adaptive methods for automatically and dynamically adaptation of evolutionary parameters such as crossover rates and mutation rates have been proposed. Self adaptation can be described as a process of adaptation to any general class of problems by refiguring evolution strategy without any user interaction [26], [27], [28], [29]. In literature, self-adaptation is usually applied to the F control parameter. The efficiency and robustness of the DE algorithm are sensitive to the setting of F.

With new versions of DE algorithm, researchers proposed several rules for calculating the control parameter F automatically. Ali and Torn [28], Qin and Suganthan [30], and Salman [31] introduced different methods for the self-adapting parameter F.

During the evolution, parameter settings should be gradually self-adapted according to the learning experience. Since the amplification factor F controls the rate at which the population evolves [32], the value of F has to be high enough while the population has not converged to the promising areas (a wide scope search is required) and to limit the algorithm search scope when the population set is distributed in the most feasible areas. Bad choices of the amplification factor F may cause the DE to stagnate before finding a globally optimal solution.

IV. DEM-ABC ALGORITHM

In optimization algorithms, the exploration refers to the ability to investigate the unknown regions while the exploitation refers to the ability to apply the knowledge of the previous good solutions to find better solutions [20]. According to the (2) the new candidate solution is generated by moving the old solution towards (or away from) another solution selected randomly from the population. The randomly selected solution can be a good or a bad one, so the new candidate solution is not necessarily a better solution than the previous one.

In this paper, we propose a new way of extending ABC to be suitable for solving large-scale unconstrained optimization problems. In order to improve the exploitation we have replaced expression (2) with one of the mutation strategies mentioned in the Section 3.

We compared different strategies for creating new solution in onlooker and employed bee phase. Although DEM-ABC uses mutation strategies "borrowed" from DE algorithm, it is still much closer to the ABC algorithm. DEM-ABC uses same random process to initialize population, same expression for calculating probabilities (1), and same scout mechanism as ABC algorithm.

The use of the best solution in onlooker and employed bee phase can drive the new candidate solution towards the global best solution; therefore, the exploitation of ABC algorithm can be increased. Note that the parameter F in Fig. 1 plays an important role in balancing the exploration and exploitation of the candidate solution search. During the first implementation of the mutation strategies mentioned in the Section 3, amplification factor F was self-adaptive, but it turned out that the computation time was too long, thus, we decided to experimentally determined F for each mutation strategy as shown in Table II.

V. EXPERIMENTAL STUDY

A. Functions

Eight unconstrained benchmark test functions are used to validate the proposed DEM-ABC algorithm: Sphere, Rosenbrock, Griewank, Rastrigin, Schwefel, Step, Dixon-Price and Ackly.

Sphere function is continuous, convex and unimodal. Global minimum value for this function is 0 and optimum solution is x=(0, 0, . . . , 0). Surface plot is shown in Fig. 2.

Definition:

$$\sum_{i=1}^{n} X_i^2 \tag{10}$$

where x is in the interval of [-100, 100].

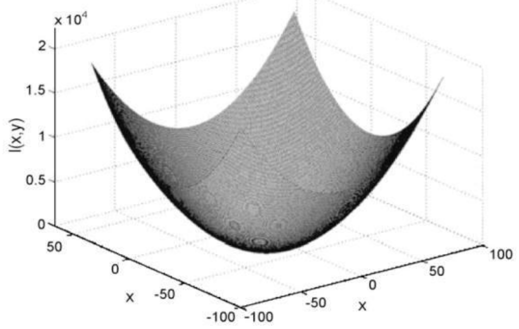

Fig. 2. Sphere function

Rosenbrock function has the global optimum inside a long, narrow, parabolic shaped flat valley. Global minimum value for this function is 0 and optimum solution is x = (1,1,…,1).

Definition:

$$f(x) = \sum_{i=1}^{n-1}\left[100\left(x_i^2 - x_{i+1}\right)^2 + \left(x_i - 1\right)^2\right] \tag{11}$$

where *x* is in the interval of [-50, 50]. Surface plot is shown in Fig. 3.

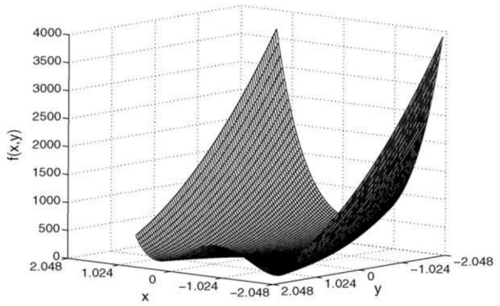

Fig. 3 Rosenbrock function

Griewank`s value is 0, and its global minimum is (0,0,…,0). Since the number of local optima increases with the dimensionality, this function is strongly multimodal. The multimodality disappears for sufficiently high dimensionalities (n > 30) and the problem seems unimodal.

Definition:

$$f(x) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos(x_i/\sqrt{i}) + 1 \tag{12}$$

where x is in the interval of [-600, 600]. Surface plot is shown in Fig. 4.

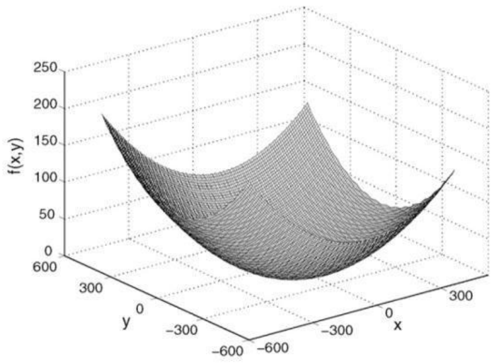

Fig. 4 Griewank function

Rastrigin function is based on Sphere function with the addition of cosine modulation to produce many local minima. Thus the function is multimodal. The global minimum value for this function is 0 and the corresponding global optimum solution is x = (0,0,…,0) .

Definition:

$$f(x) = 10n + \sum_{i=1}^{n}(X_i^2 - 10\cos(2\pi X_i)) \qquad (13)$$

where x is in the interval of [-5.12, 5.12]. Surface plot is shown in Fig. 5.

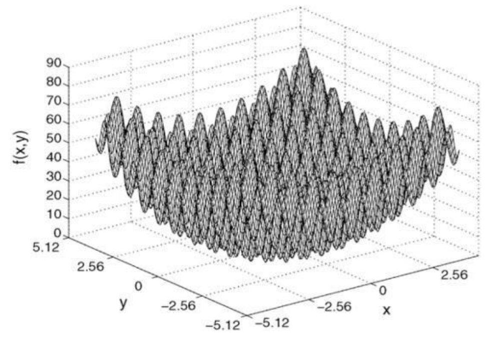

Fig. 5 Rastrigin function

Schwefel function has several local minima. The global minimum value for this function is –418.9829 and the corresponding global optimum solution is x = (1,1,…,1) . The surface of Schwefel function is composed of a great number of peaks and valleys. The function has a second best minimum far from the global minimum where many search algorithms are trapped.

Definition:

$$f(x) = \sum_{1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right) \qquad (14)$$

where x is in the interval of [-500, 500]. Surface plot is shown in Fig. 6.

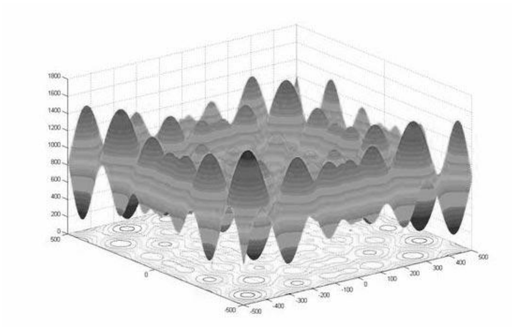

Fig. 6 Schwefel function

Step function has one minimum 0, and it is a discontinuous function which represents the problem of flat surfaces. Flat surfaces are obstacles for optimization algorithms which do not have variable step sizes, because they do not give any information as to which direction is favourable [33].

Definition:

$$f(x) = \sum_{i=1}^{n} \lfloor x_i + 0.5 \rfloor^2 \qquad (15)$$

where xi is in the interval of [-100, 100]. Surface plot is shown in Fig. 7.

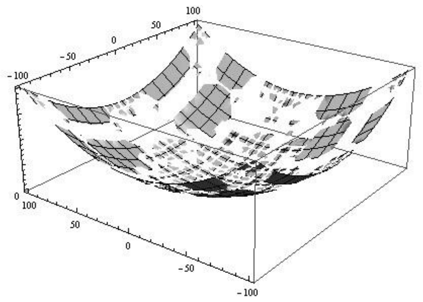

Fig. 7 Step function

Dixon-Price is polynomial, strongly convex function. Global minimum value for this function is 0.

Definition:

$$f(x) = (x1-1)^2 + \sum_{i=2}^{n}(i*(2x_i^2 - x_{i-1})^2) \qquad (16)$$

where x is in the interval of [-10, 10]. Surface plot is shown in Fig. 8.

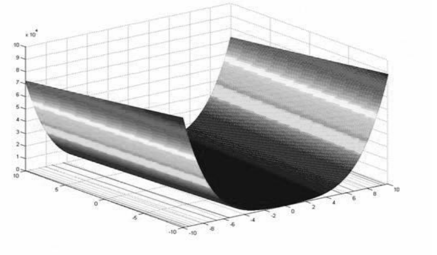

Fig. 8 Dixon-Price function

Ackley function is a widely used multimodal test function. This function has an exponential term that covers its surface with numerous local minima. Global minimum value for this function is 0 and optimum solution is x=(0, 0, . . . , 0).

$$f(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) \qquad (17)$$

$$-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$$

where x is in the interval of [-32, 32]. Surface plot is shown in Fig. 9.

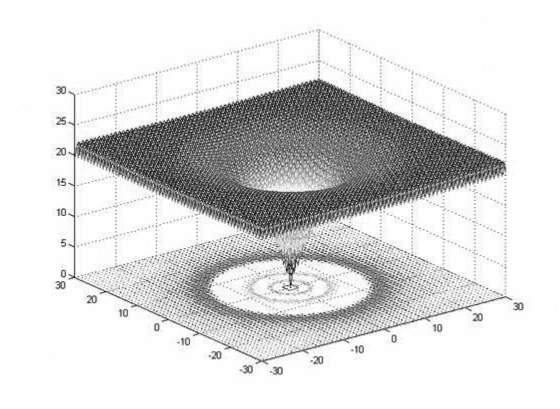

Fig. 9 Ackley function

Testing results will be presented in the following section.

B. Test and results

For all benchmark functions we set the parameters as shown in Table I:

TABLE I
PARAMETERS VALUES FOR DEM-ABC

Parameter	Value
Max eval. fun. calls	1000000
Population size NP	100
Limit	*Sn*D*0.5*

The values of amplification factor (F) are shown in Table II.

TABLE II
VALUES OF AMPLIFICATION FACTOR (F) FOR DIFFERENT MUTATION STRATEGIES

Mutation strategy	F
1	1
2	0.6
3	0.6
4	0.6
5	0.2

As we noted in previous Section these values are based on empirical experiments. The proposed DEM-ABC algorithm is coded in C++ and run on a Pentium Core2Duo, 3-GHz computer with 6 GB RAM memory and Microsoft Visual Studio 2010.

The parameters used by DEM-ABC are the following:

1) NP is number of bees in the colony (employed bees plus onlooker bees) was set to 100..
2) Limit controls the number of trials to improve certain food source. If a food source could not be improved within defined number of trial, it is abandoned by its employed bee. Limit is is equal to Sn*D*0.5
3) Max evaluation function calls: maximum number of objective function calls was set to 1000000 for all functions

Each of the experiments was conducted 30 times using different random seeds.

Comparison has been made between DE [23], PSO [34], original ABC algorithm [35] and our modified ABC algorithm for dimensions D = 10, 100 and 500. In experiments in this paper, the proposed DEM-ABC algorithm uses five mutation strategies: „rand/1“, „best/1“, „currenttobest/1“, „rand/2“, „best/2“. The best results for modified ABC algorithm (DEM-ABC) are presented in Tables III-V for all five mutation strategies. Common parameters such as population number, maximum evaluation number were set at same values for all algorithms.

Eight benchmark test functions are used to validate the proposed DEM-ABC algorithm. Tables III-V show the best optimization results of the Sphere, Schaffer, Rosenbrock, Rastrigin, Griewank, Step, Dixon-Price and Ackley functions respectively. These test cases include various types: uni-modal (Sphere, Rosenbrock, Step), multi-modal (Schwefel, Dixon-Price, Rastrigin, Griewank, Ackley), separable (Sphere, Step, Rastrigin), non-separable (Schwefel, Rosenbrock, Dixon-Price, Griewank, Ackley), non-symmetric (Dixon-Price) problems.

Table III lists the best solutions found on unconstrained benchmark problems for D=10. All versions of our proposed algorithm DEM-ABC produced better or highly similar results as the original ABC algorithm. DEM-ABC obtained better solutions than DE, PSO and original ABC, for all function except Rosenbrock function. In fact, the proposed DEM-ABC algorithm performed marginally better solution than with mutation strategy 4. Comparing between DEM-ABC1, DEM-ABC4 and DEM-ABC5, the results were highly similar except that the DEM-ABC4 produced better results for Rosenbrock function. Thus, in terms of the overall best solution found, DEM-ABC4 produced highly competitive results compared to the DE, PSO and conventional ABC algorithm and other DEM-ABC mutation strategies.

Table IV lists the best solutions found on unconstrained benchmark problems for D=100. The results again indicate that DEM-ABC was able to perform similar or better results compared to DE, PSO and original ABC algorithm when the number of function variables is increased. Again, DEM-ABC4 produced the most favorable best solutions for all function. For

Rosenbrock function DEM-ABC3 performed significantly worse results, while for Rastrigin and Dixon-Price function, DEM-ABC3 and DEM-ABC5 performed slightly worse results than original ABC algorithm.

When the dimension is incremented to 500, the DEM-ABC algorithm again produces better or similar results to original ABC. Table V lists the best solutions found on unconstrained benchmark problems for D=500. These results indicate that the performance of DEM-ABC algorithms are quite stable compared to the conventional ABC algorithm.

TABLE III
BEST RESULTS OF PSO, DE, ABC AND DEM-ABC ALGORITHMS ON UNCONSTRAINED LARGE-SCALE BENCHMARK PROBLEMS FOR D=10

D = 10								
Function	**PSO**	**DE**	**ABC**	**DEM-ABC1**	**DEM-ABC2**	**DEM-ABC3**	**DEM-ABC4**	**DEM-ABC5**
Sphere	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Schwefel	-2654.03	-4177.99	-4189.83	-240.830	-240.830	-240.830	-240.830	-240.830
Rosenbrock	0.426	0.000	0.013	0.021	0.779	0.073	0.006	0.024
Rastrigin	7.363	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Griewank	0.059	0.004	0.000	0.000	0.000	0.000	0.000	0.000
Step	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Dixon-Pric	0.666	0.666	0.000	0.000	0.000	0.000	0.000	0.000
Ackley	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

TABLE IV
BEST RESULTS OF PSO, DE, ABC AND DEM-ABC ALGORITHMS ON UNCONSTRAINED LARGE-SCALE BENCHMARK PROBLEMS FOR D=100

D = 100								
Function	**PSO**	**DE**	**ABC**	**DEM-ABC1**	**DEM-ABC2**	**DEM-ABC3**	**DEM-ABC4**	**DEM-ABC5**
Sphere	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Schwefel	-20100.4	-31182.50	-41898.30	-2408.30	-2408.301	-2408.301	-2408.301	-2408.301
Rosenbrock	113.144	132.349	0.055	0.002	0.015	91,131	0.001	0.000
Rastrigin	148.249	133.114	0.000	0.000	0.000	42.232	0.000	94.000
Griewank	0.049	0.001	0.000	0.000	0.000	0.000	0.000	0.000
Step	1.700	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Dixon-Pric	2.076	0.666	1.26E-06	1.26E-06	1.1E-06	2.1E-06	1.1E-06	8.4E-05
Ackley	0.732	0.000	0.000	0.000	0.000	0.000	0.000	0.000

TABLE V
BEST RESULTS OF PSO, DE, ABC AND DEM-ABC ALGORITHMS ON UNCONSTRAINED LARGE-SCALE BENCHMARK PROBLEMS FOR D=500

D = 500								
Function	**PSO**	**DE**	**ABC**	**DEM-ABC1**	**DEM-ABC2**	**DEM-ABC3**	**DEM-ABC4**	**DEM-ABC5**
Sphere	181.160	20.330	8.7E-7	1.2E-07	2.8E-11	2.8E-08	2.7E-07	7.4E-13
Schwefel	-98.1E+03	-13.8E+04	-19.1E+04	-12.1E+03	-12.1E+03	-12.1E+03	-12.1E+03	-12.1E+03
Rosenbrock	10.9E+05	87.2E+09	10.1E+02	2.2E+03	1.7E+03	1.2E+03	1.9E+02	5.8E02
Rastrigin	1033.040	594.690	87.960	1.9E-03	2.9E-01	2.5E-02	7.8E-01	7.1E-05
Griewank	2.200	0.645	0.000	0.000	0.000	0.000	0.000	0.000
Step	1621.000	1998.030	0.000	0.000	0.000	0.000	0.000	0.000
Dixon-Price	15315.110	2636.320	309.600	0.052	0.001	0.002	0.001	0.028
Ackley	3.690	13.000	0.058	0.000	0.000	0.000	0.000	0.000

The increment of dimension does not appear to adversely affect the performance of ABC-DEM algorithm; in fact DEM-ABC preserves its robustness. DEM-ABC4 produced the best solutions for all functions. As shown in Table V, all DEM-ABC algorithms achieved better results than PSO, DE and ABC algorithm for all the functions except for the Rosenbrock function, where mutation 1, 2 and 3 reached slightly worse results tham ABC algorithm.

It can be observed that the performances of DEM-ABC algorithm with mutation strategy 4 are superior to ABC algorithm for all values of the parameter D. From the results, it can be said that, as the number of function variables increases, the performance of the ABC algorithm stands out much more.

The superiority of the ABC algorithm can be explained by its structure which combines explorative and exploitative processes in a balanced manner.

VI. Conclusion

In this paper, we compared the performance of our proposed DEM-ABC algorithm with the original ABC, DE and PSO on a set of 8 large-scaled unconstrained benchmark functions. These algorithms are chosen because they are also swarm intelligence and population based algorithms as the ABC algorithm. Our suggested modification improves the performance of ABC algorithm in terms of improving the exploitation process in employed and onlooker phase. The experimental results of different types of mutation strategies showed that the DEM-ABC algorithm is effective and powerful algorithm for unconstrained large-scaled optimization problems

References

[1] Chiong R, Nature-Inspired Algorithms for Optimisation, Springer, pp. 536, 2009.
[2] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, Inc., NY, 1999.
[3] R. Storn, K. Price, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization, Vol. 11, Issue 4 ,1997, pp. 341–359.
[4] Karaboga D, Basturk B, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. of Global Optimization, Volume 39, No. 0925-5001, 2007, pp. 459-471.
[5] Tereshko, V., Loengarov, A.: Collective Decision-Making in Honey Bee Foraging Dynamics. Computing and Information Systems, Vol.9 No. 3, 2005, pp. 1–7
[6] R. M. Idris, A. Khairuddin, M. W. Mustafa, Optimal Allocation of FACTS Devices in Deregulated Electricity Market Using Bees Algorithm, WSEAS Transactions On Power Systems, Vol. 5, ISSN: 1790-5060, 2010, pp. 108-119
[7] R. Mohamad Idris, A. Khairuddin and M. W. Mustafa, Optimal Allocation of FACTS Devices in Deregulated Electricity Market Using Bees Algorithm, WSEAS Transactions on Power Systems, Vol. 5, Issue 2, 2010, pp. 108-119.
[8] L. Jiann-Horng, H. Li-Ren, Chaotic bee swarm optimization algorithm for path planning of mobile robots, Proceedings of the 10th WSEAS international conference on evolutionary computing, No. 978-960-474-067-3, 2009, pp. 84-89.
[9] Drias, H., Sadeg, S., Yahi, S., Cooperative bees swarm for solving the maximum weighted satisfiability problem, Computational Intelligence and Bioinspired Systems. Vol. 3512/2005, DOI 10.1007/11494669_39, 2005, pp.417-448.
[10] Karaboga D, Basturk B, Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems, Advances in Soft Computing: Foundations of Fuzzy Logic and Soft Computing, Volume 4529/2007, No. 0302-9743, 2007, pp.789–798.
[11] Karaboga D, Akay B, Ozturk C, Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks, Modeling Decisions for Artificial Intelligence, Volume 4617/2007, No. 0302-9743, 2007, pp. 318-329.
[12] [Chunfan Xu, Haibin Duan, Artificial bee colony (ABC) optimized edge potential function (EPF) approach to target recognition for low-altitude aircraft, Pattern Recognition Letters, Vol. 31, No. 0167-86552010, pp. 1759 – 1772,
[13] Efrén Mezura-Montes and Ramiro Ernesto Velez-Koeppel, Elitist Artificial Bee Colony for Constrained Real-Parameter Optimization, Evolutionary Computation (CEC),No. 978-1-4244-6909-3, DOI10.1109/CEC.2010.5586280, 2010, pp. 1-8,
[14] [Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs, Springer, pp. 387, 1996.
[15] [Ayed Salman, Andries P. Engelbrecht, Mahamed G.H. Omran, Empirical analysis of self-adaptive differential evolution, European Journal of Operational Research, Vol. 183, Issue 2, pp. 785-804, Elsevier, 2007, doi:10.1016/j.ejor.2006.10.020,
[16] Shahryar Rahnamayan, G. Gary Wang, Solving Large Scale Optimization Problems by Opposition-Based Differential Evolution (ODE), WSEAS Transactions On Computers, Vol. 7, Issue. 10, 2008, pp. 1792-1804
[17] H Liu, Cai Z, Wang Y, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Applied Soft Computing, Vol. 10, Issue 2, pp. 629–640, 2009.
[18] Mezura-Montes Efren; Edith Miranda-Varela Mariana; del Carmen Gomez-Ramon Rubi, Differential evolution in constrained numerical optimization: An empirical study, Information Sciences, vol. 180, pp. 4223-4262, DOI: 10.1016/j.ins.2010.07.023, 2010
[19] S. Selvi, D. Manimegalai, Scheduling jobs on computational grid using Differential Evolution algorithm, Proceedings of the 12th WSEAS international conference on Networking, VLSI and signal processing, No. 978-960-474-162-5, 2010, pp.118-123.
[20] R. Storn, K. V. Price, J. Lampinen, Differential Evolution–A Practical Approach to Global Optimization. Springer- Verlag, 2005.
[21] I. C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters, Vol.85, No. 6, DOI: 10.1016/S0020-0190(02)00447-7, 2003, pp. 317–325.
[22] Mezura-Montes Efren; Gabriela Palomeque-Ortiz Ana, Parameter Control in Differential Evolution for Constrained Optimization, IEEE Congress on Evolutionary Computation Location, Vol. 1-5, pp. 1375-1382 DOI: 10.1109/CEC.2009.4983104, 2009.
[23] Storn, R., Price, K. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. International Computer Science Institute Berkley, 1995.
[24] J. Teo. Exploring dynamic self-adaptive populations in differential evolution. Soft Computing - A Fusion of Foundations, Methodologies and Applications, Vol. 10, Issue 82005, pp. 673-686 2006, DOI: 10.1007/s00500-005-0537-1.
[25] Brest, J.; Zumer, V.; Maucec, M.S., Self-Adaptive Differential Evolution Algorithm in Constrained Real-Parameter Optimization, Evolutionary Computation, ISBN: 0-7803-9487-9, pp. 215 – 222, 2006, doi: 10.1109/CEC.2006.1688311.
[26] T. Back. Adaptive Business Intelligence Based on Evolution Strategies: Some Application Examples of Self-Adaptive Software. Information Sciences, 148:113–121, 2002.
[27] T. Back, D. B. Fogel, and Z. Michalewicz, editors. Handbook of Evolutionary Computation. Institute of Physics Publishing and Oxford University Press, 1997.
[28] A. E. Eiben and J. E. Smith. Introduction to Evolutionary Computing. Natural Computing. Springer-Verlag, Berlin, 2003.

[29] M. M. Ali and A. Torn. Population Set-Based Global Optimization Algorithms: Some Modifications and Numerical Studies. Computers & Operations Research, Vol. 31, pp.1703–1725, 2004. doi:10.1016/S0305-0548(03)00116-3
[30] A. K. Qin and P. N. Suganthan. Self-adaptive Differential Evolution Algorithm for Numerical Optimization. In The 2005 IEEE Congress on Evolutionary Computation CEC2005, Vol. 2, pp. 1785–1791. 2005. DOI: 10.1109/CEC.2005.1554904.
[31] A. Salman, A.P. Engelbrecht, M.G.H. Omran, Empirical analysis of self-adaptive differential evolution, European Journal of Operational Research, Vol. 183, 2007, pp.785–804. doi:10.1016/j.ejor.2006.10.020
[32] Montano, A.A.; Coello Coello, C.A.; Mezura-Montes, E., MODE-LD+SS: A novel Differential Evolution algorithm incorporating local dominance and scalar selection mechanisms for multi-objective optimization, Evolutionary Computation (CEC), 2010 IEEE Congress, ISBN: 978-1-4244-6909-3, DOI: 10.1109/CEC.2010.5586137,pp. 1-8, 2010
[33] Digalakis, J. G., & Margaritis, K. G. An experimental study of benchmarking functions for genetic algorithms. International Journal of Computer and Mathematics, Vol. 79, pp. 403–416, 2002, DOI: 10.1080/00207160210939
[34] Kennedy, J., Eberhart, R., Particle swarm optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks Vol. 4, 1995, pp. 1942–1948.
[35] Akay B, Karaboga D, Artificial bee colony algorithm for large-scale problems and engineering design optimization, Journal of Intelligent Manufacturing, DOI: 10.1007/s10845-010-0393-4, Vol. 19, No. 09565515, 2010L. Jiann-Horng, H. Li-Ren, Chaotic bee swarm optimization algorithm for path planning of mobile robots, Proceedings of the 10th WSEAS international conference on evolutionary computing, 2009, pp. 84-89.

Nadezda Stanarevic received B.S. and M.S. degrees in mathematics and computer science from Faculty of Mathematics, University of Belgrade, Serbia. She is currently Ph.D. student at Faculty of Mathematics, Computer Science department, University of Belgrade. She is the author or coauthor of seven papers. Her current research interest includes nature inspired metaheuristics.

Mrs. Stanarevic participated in WSEAS conferences**.**