A new differential artificial bee colony algorithm for large scale optimization problems

Takuya Kagawa[1], Akihide Utani[1], and Hisao Yamamoto[1]

[1]Tokyo City University, Tokyo 158-8557, Japan
(Tel: 81-3-5707-0104)

[1]autani@tcu.ac.jp

Abstract: This paper proposes a novel optimization algorithm based on advanced Artificial Bee Colony (ABC) algorithm that has the good performance on large scale optimization problems named the differential ABC algorithm. In the proposed algorithm, the generation equation of the mutation vector of Differential Evolution (DE) is introduced in advanced ABC algorithm. We evaluate the proposed algorithm through numerical experiments on well-known benchmark functions, such as Rastrigin function, Schwefel function, Ackley function and Griewank function, and discuss its development potential. In numerical experiments performed, the performances of the proposed algorithm are compared with those of the existing optimization ones, such as Particle Swarm Optimization (PSO) algorithm, DE algorithm, ABC algorithm and advanced ABC algorithm.

Keywords: large scale optimization problems, multimodal functions, global optimization, differential evolution, ABC algorithm.

1 INTRODUCTION

The minimization of multimodal functions with many local and global minima is a problem that frequently arises in diverse scientific fields and numerous engineering design problems. This problem is NP-hard in the sense of its computational complexity even in simple cases. As techniques of computing a global minimum of the objective function, many meta-heuristics, which are search algorithms for optimization based on heuristic knowledge, have been proposed. Some well-known representative meta-heuristics are simulated annealing and tabu search which are the traditional optimization algorithms, genetic algorithms and immune algorithms which are classified as evolutionary computation techniques, and ant colony optimization algorithms and particle swarm optimization algorithms [1] which belong to the category of swarm intelligence algorithms.

In meta-heuristics, genetic algorithms and immune algorithms, classified as evolutionary computation techniques, are generally techniques for combination optimization problems. In genetic algorithms and immune algorithms, the variables of continuous type are frequently translated into those of discrete (genetic) type. If there is a dependency between variables, therefore, a promising solution may be destroyed during the solution search process (genetic operation) and the solution search performance may deteriorate. To the contrary, particle swarm optimization algorithm [1] can directly handle the variables of continuous type. Even when there is a dependency between variables, therefore, an efficient and effective solution search can be realized. Recently, particle swarm optimization algorithm is intensively researched because it is superior to the other algorithms on many difficult optimization problems. The ideas that underlie particle swarm optimization algorithm are inspired not by the evolutionary mechanisms encountered in natural selection, but rather by the social behavior of flocking organisms, such as swarms of birds and fish schools. Particle swarm optimization algorithm is a population-based algorithm that exploits a population of individuals to probe promising regions of the search space. The algorithm is simple and allows unconditional application to various optimization problems. However, it has been confirmed that the performance of particle swarm optimization algorithm on large scale optimization problems is not always satisfactory. Therefore, we have proposed a new optimization algorithm based on Artificial Bee Colony (ABC) algorithm [2] that has the good performance on large scale optimization problems [3]. However, for hundreds of dimensional engineering design problems, the improvement of the search performance is moreover required.

This paper proposes a novel optimization algorithm based on advanced ABC algorithm [3] that has the good performance on large scale optimization problems named the differential ABC algorithm. In the proposed algorithm, the generation equation of the mutation vector of Differential Evolution (DE) [4] is introduced in advanced ABC algorithm. We evaluate the proposed algorithm through numerical experiments on well-known benchmark functions, such as Rastrigin function, Schwefel function, Ackley function and Griewank function, and discuss its development potential. In numerical experiments performed, the performances of the proposed algorithm are compared with those of the existing optimization ones. The rest of this paper is organized as follows. In Section 2, the proposed algorithm named the differential ABC algorithm is outlined. In Section 3, the results of numerical experiments are reported. Finally, the paper closes with conclusions in Section 4.

2 PROPOSED ALGORITHM

In the original ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers, and scouts. Half of the bees that construct artificial bee colony consist of the employed bees and the rest constitute the onlookers. At the initial state of ABC algorithm, multiple solution search points are randomly set in multidimensional solution search space. For every solution search point, there is only one employed bee, i.e., the number of employed bees is equal to the number of solution search points.

Each employed bee updates its own allocated solution search point in the solution search process. Each onlooker selects one solution search point from the probability based on the fitness value of each solution search point and updates the solution search point chosen. The employed bee of the solution search point that is not undated during the set search iterations becomes a scout and starts to search a new solution search point randomly.

The original ABC algorithm is summarized as follows:

Step 1 (Initialization)

Generate and evaluate the population of solution search points ($\boldsymbol{x}_i$, i=1, ... , SN), where i indicates the solution search point's index. SN is the number of solution search points.

Step 2 (Update of search points by employed bees)

1. Produce and evaluate the new solution search point ($\boldsymbol{v}_i$) on each solution search point ($\boldsymbol{x}_i$) according to the following formula

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{mj}), \quad i = 1,\ldots,SN \tag{1}$$

where $m \in \{1, 2, \ldots, SN\}$ represents randomly chosen solution search point's index and $j \in \{1, 2, \ldots, D\}$ represents randomly chosen variable's index. D is the number of optimization variables. Although m is determined randomly, it has to be different from i. ϕ_{ij} is a random number, uniformly distributed within the interval [-1,1].

2. Apply the greedy selection process between $\boldsymbol{x}_i$ and $\boldsymbol{v}_i$.

Step 3 (Update of search points chosen by onlookers)

1. Compute the probability (P_i) of each solution search point ($\boldsymbol{x}_i$) according to the following formulas

$$fit_i = \begin{cases} \dfrac{1}{1+f(\boldsymbol{x}_i)}, & f(\boldsymbol{x}_i) \ge 0 \\ 1 + abs(f(\boldsymbol{x}_i)), & f(\boldsymbol{x}_i) < 0 \end{cases} \quad (i = 1,\ldots,SN) \tag{2}$$

$$P_i = fit_i \Big/ \sum_{n=1}^{SN} fit_n \tag{3}$$

where $f(\boldsymbol{x})$ is the objective function. Each onlooker selects one solution search point ($\boldsymbol{x}_i$) depending on the probability (P_i).

2. Produce and evaluate the new solution search point ($\boldsymbol{v}_i$) on the solution search point ($\boldsymbol{x}_i$) chosen by each onlooker, as in Step2.
3. Apply the greedy selection process between $\boldsymbol{x}_i$ and $\boldsymbol{v}_i$.

Step 4 (Update of solution search points by scouts)

If the solution search point that is not undated during the set search iterations exists, replace it with randomly produced solution search point within solution search space.

Step 5 (Update of best solution)

Memorize the best solution search point.

Step 6 (Judgment of end)

Finish the search when the end condition is satisfied. Otherwise, return to Step 2.

The algorithm of [3] is an advanced ABC algorithm. To detect a global optimum solution effectively, ABC algorithm is improved as follows:

1. Improvement of Eq.(2) to compute fitness value (fit_i)
To improve the adaptability to various engineering design problems, in the proposed algorithm, the fitness value (fit_i) of each solution search point is computed as follows:

$$fit_i = \begin{cases} \dfrac{1}{f(\boldsymbol{x}_i) - f_{bound}}, & f(\boldsymbol{x}_i) - f_{bound} \ge f_{accuracy} \\ \dfrac{1}{f_{accuracy}}, & f(\boldsymbol{x}_i) - f_{bound} < f_{accuracy} \end{cases} \quad (i = 1,\ldots,SN) \tag{4}$$

where f_{bound} represents the boundary value of $f(\boldsymbol{x}^+)$ on $\boldsymbol{x}^+$ acceptable as a solution for every problem and $f_{accuracy}$ indicates the exactness of convergence to f_{bound}.

2. Improvement of search point selection by onlookers
To improve the performance of global solution search, in the proposed algorithm, the solution search is divided into two stages. Each onlooker selects one solution search point out of the high-order α solution search points of fitness value (fit_i) randomly during the first stage of the search.
3. Improvement of reference point (m) in Eq.(1)

To improve the update frequency of each solution search point, the reference point (m) in Eq.(1) is randomly chosen out of the high-order α solution search points of the fitness value (fit_i) during the first stage of the search and is chosen by the roulette selection based on the probability (P_i) of each solution search point ($\boldsymbol{x}_i$) during the second stage of the search.

In addition, Step 4 (Update of search points by scouts) in the original ABC algorithm is not executed in the algorithm of [3] because the performance of global solution search is improved by the above 2 (Improvement of search point selection by onlookers).

In the proposed algorithm, the following mutation vector generation equation of Differential Evolution (DE) [4] is introduced for the update of search points by employed bees.

$$\boldsymbol{v}_i = \boldsymbol{x}_{r1} + F(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3}), \quad i = 1, \dots, SN \qquad (5)$$

where $\boldsymbol{x}_{r1}$, $\boldsymbol{x}_{r2}$ and $\boldsymbol{x}_{r3}$ represent the search point's index randomly selected, and F indicates a control parameter determined within the interval [0,1]. The above mutation vector generation equation has the good performance on the global search of multimodal functions. By introducing Eq.(5), it can be expected that the performance of the proposed algorithm is improved.

3 EXPERIMENTAL RESULTS

Through numerical experiments on D dimensional benchmark functions, which are Rastrigin function, Schwefel function, Ackley function and Griewank function, the performances of the proposed algorithm are investigated in detail to verify its effectiveness on large scale optimization problems. In numerical experiments performed, the performances of the proposed algorithm are compared with those of the existing optimization ones.

- Rastrigin function

$$\text{min. } f_1(\boldsymbol{x}) = \sum_{j=1}^{D} \left\{ x_j^2 - 10\,cos(2\pi x_j) + 10 \right\}$$

$$\text{subj. to } -5.12 \le x_j \le 5.12, \quad j = 1,\dots,D$$

$$\boldsymbol{x}^* = (0,\dots,0), \quad f_1(\boldsymbol{x}^*) = 0$$

- Schwefel function

$$\text{min. } f_2(\boldsymbol{x}) = 418.98288727\,D + \sum_{j=1}^{D} -x_j sin(\sqrt{|x_j|})$$

$$\text{subj. to } -512 \le x_j \le 512, \quad j = 1,\dots,D$$

$$\boldsymbol{x}^* = (420.968750,\dots,420.968750), \quad f_2(\boldsymbol{x}^*) = 0$$

- Ackley function

$$\text{min. } f_3(\boldsymbol{x}) = 20 + e - 20\,exp\left(-0.2\sqrt{\frac{1}{D}\sum_{j=1}^{D} x_j^2} \right) - exp\left(\frac{1}{D}\sum_{j=1}^{D} cos(2\pi x_j) \right)$$

$$\text{subj. to } -30 \le x_j \le 30, \quad j = 1,\dots,D$$

$$\boldsymbol{x}^* = (0,\dots,0), \quad f_3(\boldsymbol{x}^*) = 0$$

- Griewank function

$$\text{min. } f_4(\boldsymbol{x}) = \frac{1}{4000}\sum_{j=1}^{D} x_j^2 - \prod_{j=1}^{D} cos\left(\frac{x_j}{\sqrt{j}} \right) + 1$$

$$\text{subj.to } -600 \le x_j \le 600, \quad j = 1,\dots,D$$

$$\boldsymbol{x}^* = (0,\dots,0), \quad f_4(\boldsymbol{x}^*) = 0$$

where $\boldsymbol{x}^*$ of each benchmark function represents a global optimum solution. In **Fig.1**, the landscapes of multimodal functions are illustrated. The settings which were used in numerical experiments performed are shown in **Table1**. In the experimental results, the proposed algorithm is evaluated through a comparison with existing optimization ones.

Table1 Settings used in the experiments performed.

colony size(N)	*60*
employed bees(SN)	*50% of colony size*
onlookers(N - SN)	*50% of colony size*
Number of iterations	*1000*

The experimental results obtained by applying PSO algorithm, DE algorithm, ABC algorithm, Advanced ABC (AABC) algorithm and the proposed algorithm are arranged in **Tables 2, 3, 4** and **5**. In **Fig.2**, the convergence curves on Rastrigin function (D=100) are shown. From these experimental results, it can be confirmed that the proposed algorithm is substantially advantageous for large scale optimization problems.

4 CONCLUSION

In this paper, a novel optimization algorithm named the differential ABC algorithm has been proposed. In numerical experiments performed on some well-known representative benchmark functions, such as Rastrigin function and Schwefel function, the performances of the proposed algorithm were compared with those of the existing optimization ones. The experimental results indicate that the proposed algorithm is superior to the existing optimization ones for large scale optimization problems.

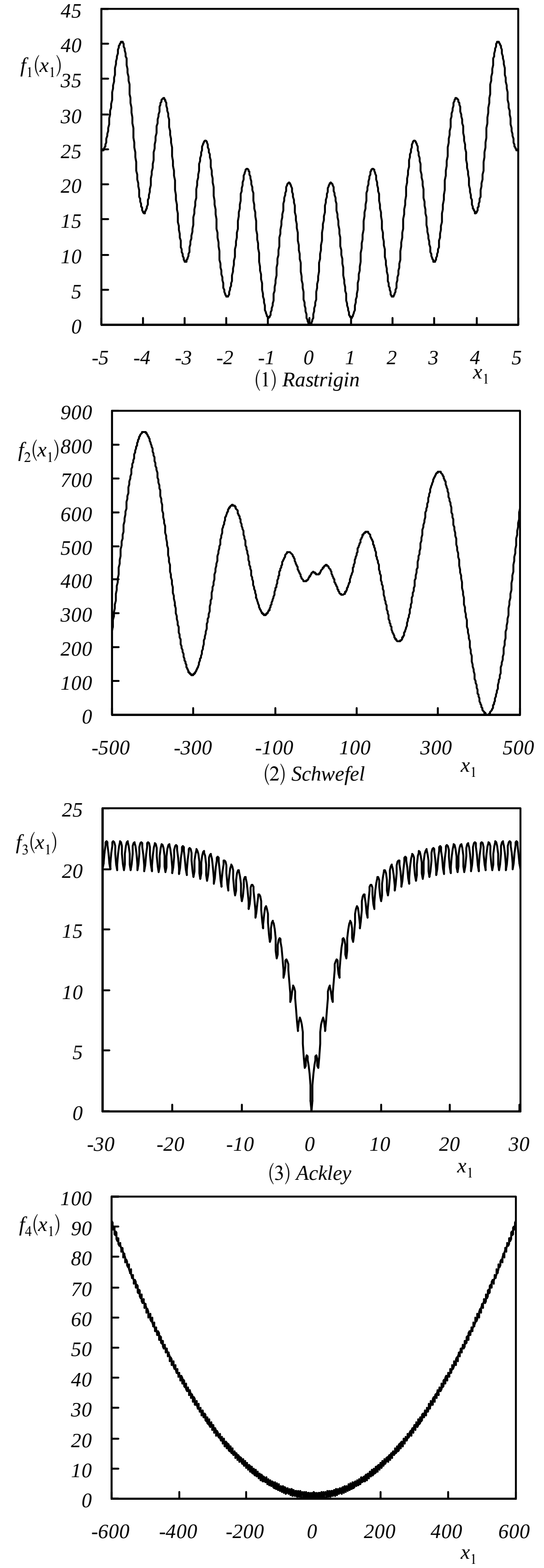

Fig.1. Landscapes of multimodal functions ($D = 1$).

Table2. Experimental results (Rastrigin function).

Method	Dim.	Best	Ave.	Worst
PSO	50	1.05×10^{2}	1.81×10^{3}	3.14×10^{3}
	75	1.58×10^{2}	2.30×10^{3}	3.26×10^{3}
	100	2.72×10^{2}	3.71×10^{3}	6.10×10^{3}
	150	3.54×10^{2}	6.60×10^{3}	9.31×10^{3}
DE	50	7.62×10^{-2}	4.55×10^{0}	4.23×10^{1}
	75	1.53×10^{1}	8.77×10^{1}	1.81×10^{2}
	100	1.54×10^{2}	2.21×10^{2}	3.22×10^{2}
	150	2.98×10^{2}	5.59×10^{2}	7.38×10^{2}
ABC	50	4.51×10^{-4}	3.53×10^{0}	1.25×10^{1}
	75	4.93×10^{0}	2.42×10^{1}	9.82×10^{1}
	100	6.22×10^{1}	7.16×10^{1}	1.02×10^{2}
	150	1.28×10^{2}	2.77×10^{2}	3.22×10^{2}
AABC	50	2.08×10^{-10}	8.10×10^{-1}	2.99×10^{0}
	75	2.80×10^{0}	8.00×10^{0}	1.57×10^{1}
	100	1.21×10^{1}	1.39×10^{1}	2.92×10^{1}
	150	4.67×10^{1}	5.66×10^{1}	9.53×10^{1}
Proposal	50	3.23×10^{-11}	4.01×10^{-1}	1.99×10^{0}
	75	1.28×10^{-5}	2.33×10^{-1}	1.15×10^{0}
	100	9.96×10^{-1}	2.88×10^{0}	9.50×10^{0}
	150	3.66×10^{1}	5.31×10^{1}	6.30×10^{1}

Table3. Experimental results (Schwefel function).

Method	Dim.	Best	Ave.	Worst
PSO	50	5.39×10^{3}	9.78×10^{3}	1.25×10^{4}
	75	9.20×10^{3}	3.48×10^{4}	4.52×10^{4}
	100	1.92×10^{4}	6.92×10^{4}	9.88×10^{4}
	150	3.35×10^{4}	4.07×10^{5}	5.65×10^{5}
DE	50	1.14×10^{3}	2.24×10^{3}	4.62×10^{3}
	75	6.27×10^{3}	9.87×10^{3}	1.11×10^{4}
	100	1.24×10^{4}	3.45×10^{4}	4.10×10^{4}
	150	5.72×10^{4}	6.70×10^{4}	7.54×10^{4}
ABC	50	8.98×10^{2}	1.30×10^{3}	1.97×10^{3}
	75	2.73×10^{3}	6.01×10^{3}	6.56×10^{3}
	100	6.02×10^{3}	9.31×10^{3}	1.00×10^{4}
	150	9.60×10^{3}	1.62×10^{4}	3.20×10^{4}
AABC	50	1.18×10^{2}	2.03×10^{2}	1.18×10^{3}
	75	1.43×10^{3}	2.39×10^{3}	3.06×10^{3}
	100	2.54×10^{3}	3.03×10^{3}	5.26×10^{3}
	150	6.78×10^{3}	9.02×10^{3}	9.91×10^{3}
Proposal	50	0.00	6.31×10^{1}	2.30×10^{2}
	75	1.90×10^{-6}	7.42×10^{1}	3.44×10^{2}
	100	2.03×10^{-2}	1.48×10^{2}	4.27×10^{2}
	150	1.85×10^{3}	2.53×10^{3}	3.49×10^{3}

Table4. Experimental results (Ackley function).

Method	Dim.	Best	Ave.	Worst
PSO	50	1.82×10^{0}	2.11×10^{1}	2.65×10^{1}
	75	2.54×10^{0}	4.05×10^{1}	6.02×10^{1}
	100	4.84×10^{0}	2.40×10^{2}	4.30×10^{2}
	150	5.79×10^{0}	2.61×10^{2}	8.90×10^{2}
DE	50	9.89×10^{-4}	2.44×10^{-3}	3.56×10^{-3}
	75	2.66×10^{-1}	8.52×10^{-1}	$7.25\times10^{.0}$
	100	5.26×10^{0}	7.81×10^{0}	8.01×10^{0}
	150	8.59×10^{0}	9.15×10^{0}	9.45×10^{0}
ABC	50	4.74×10^{-4}	1.71×10^{-3}	2.24×10^{-2}
	75	1.30×10^{-1}	7.96×10^{-1}	1.58×10^{0}
	100	1.89×10^{0}	2.49×10^{0}	4.12×10^{0}
	150	4.57×10^{0}	5.75×10^{0}	6.68×10^{0}
AABC	50	3.30×10^{-9}	5.70×10^{-5}	2.21×10^{-3}
	75	1.85×10^{-5}	2.44×10^{-4}	1.55×10^{-3}
	100	1.51×10^{-3}	7.17×10^{-2}	3.74×10^{-1}
	150	1.48×10^{-1}	1.96×10^{-1}	2.46×10^{0}
Proposal	50	1.65×10^{-9}	1.15×10^{-7}	1.37×10^{-6}
	75	8.62×10^{-6}	4.94×10^{-5}	1.67×10^{-4}
	100	9.76×10^{-4}	7.67×10^{-3}	2.41×10^{-2}
	150	8.78×10^{-2}	1.91×10^{-1}	3.93×10^{-1}

Table5. Experimental results (Griewank function).

Method	Dim.	Best	Ave.	Worst
PSO	50	1.23×10^{0}	1.91×10^{1}	2.13×10^{1}
	75	3.89×10^{0}	4.93×10^{1}	9.83×10^{1}
	100	4.01×10^{0}	7.12×10^{1}	1.51×10^{2}
	150	5.99×10^{0}	7.05×10^{1}	1.87×10^{2}
DE	50	5.42×10^{-6}	9.80×10^{-3}	7.30×10^{-2}
	75	5.72×10^{-2}	1.47×10^{-1}	3.89×10^{-1}
	100	4.19×10^{-1}	6.24×10^{-1}	3.83×10^{0}
	150	2.02×10^{0}	3.08×10^{0}	5.31×10^{0}
ABC	50	8.50×10^{-7}	9.21×10^{-3}	3.73×10^{-2}
	75	4.26×10^{-4}	4.57×10^{-2}	1.28×10^{-1}
	100	8.17×10^{-3}	7.94×10^{-2}	5.34×10^{-1}
	150	1.79×10^{-2}	7.17×10^{-1}	9.97×10^{-1}
AABC	50	1.14×10^{-16}	9.13×10^{-5}	4.56×10^{-3}
	75	7.92×10^{-11}	3.61×10^{-3}	6.84×10^{-2}
	100	4.00×10^{-8}	5.49×10^{-3}	3.76×10^{-2}
	150	1.74×10^{-4}	1.68×10^{-2}	2.56×10^{-1}
Proposal	50	1.12×10^{-16}	2.88×10^{-10}	1.75×10^{-9}
	75	5.50×10^{-11}	7.75×10^{-8}	4.71×10^{-7}
	100	1.63×10^{-8}	5.24×10^{-4}	8.84×10^{-3}
	150	1.07×10^{-4}	8.88×10^{-3}	6.63×10^{-2}

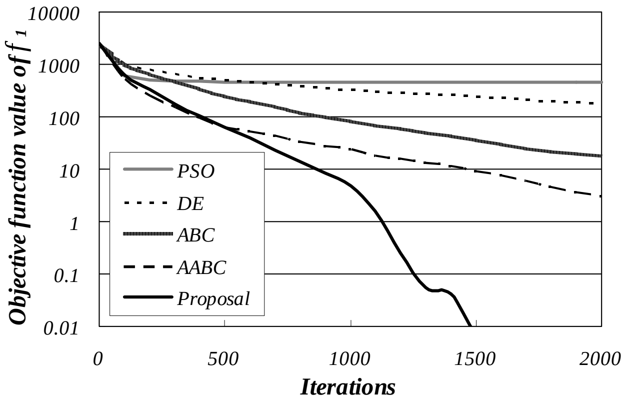

Fig.2. Convergence curves.
[Rastrigin function (D=100)]

REFERENCES

[1] J. Kennedy and R.C. Eberhart, Particle swarm optimization, *Proc. of the IEEE International Conference on Neural Networks*, pp.1942-1948, 1995.

[2] D. Karaboga and B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *J. Global Optimization*, Vol.39, pp.459-471, 2007.

[3] R. Gocho, A. Utani and H. Yamamoto, Improved artificial bee colony algorithm for large-scale optimization problems, *Proc. of the 16th International Symposium on Artificial Life and Robotics*, pp.605-608, 2011.

[4] R. Storn and K. Price, Differential evolution–A simple and efficient heuristic for global optimization over continuous space, *J. Global Optimization*, Vol.11, pp.341-359, 1997.