Acta Technica *62* No. 1A/2017, 371–380

A balancing artificial bee colony algorithm for constrained optimization problems[1]

Zhen Wang[2], Yuelin Gao[2]

Abstract. To solve constrained optimization problems, a balancing Artificial Bee Colony (BABC) algorithm is proposed in this paper. It is important to balance constraints and objective function during iterations by population-based algorithm. The feasible rules are the main constraint handling method employed in this paper. To balance the constraints and objective function, the replacement mechanism and the mutation strategy are added. Furthermore, the opposite learning initialization is introduced to make the initial colony scattered evenly on the search area. And the best-lead search equation is used in onlooker bee phase to improve the convergent speed. The BABC algorithm is tested on 18 test functions with 30-D at IEEE CEC2010. The experimental results are compared with other state-of-art algorithms, which suggest that the BABC algorithm outperforms or performs similarly to other algorithms.

Key words. Constrained optimization, Artificial bee colony algorithm, feasibility rule.

1. Introduction

The research to find better optimization algorithms of constrained optimization problems arose from science and engineering has never stopped. Since science and engineering constrained optimization problems are getting complicated, the significant attentions are paid on efficient and effective algorithm. In recent years, population-based algorithms have been widely used to deal with constrained optimization problems. Currently, a population-based algorithm called artificial bee colony (ABC) algorithm is proposed to simulating the behavior of foraging bees, which is widely used for solving many kinds of optimization problems and real world problems including constrained optimization problems [1].

[1]This work is supported by National Nature Science Foundation of China (No. 61650104, No. 61561001), 2016 scientific research project of North Minzu University (No. 2016SXKY03) and the key research projects of North Minzu University (2015KJ10).

[2]Institute of Mathematics and Information Science, North Minzu University, Yinchuan 750021, China

Since most of the population-based algorithms are originally designed for unconstrained optimization problems, one of the main concerns of using these algorithms for solving constrained optimization problems is how to select the promising individuals for solution iteration. In most cases, the individuals in the population with better fitness function values may not be feasible, which makes the constraints handling method combined in the algorithm is much more important. Generally speaking, methods dealing with the constraints can be divided into five groups.

Methods based on transforming unfeasible solutions into feasible ones with some operators. Michalewicz and Schoenauer [2] discussed difficulties connected with solving the general nonlinear programming problem and provide a constraints handling method.

Methods based on penalty functions. Thakur et al. [3] proposed a modified real coded genetic algorithm for constrained optimization with penalty function method. Elsayedet et al. [4] applied the concept of training and testing with a self-adaptive multi-operator based evolutionary algorithm to find suitable parameters with penalty function method.

Methods based on feasibility rules. Deb [5] first proposed the feasibility rules for binary tournaments in Genetic Algorithms. Tuba and Bacanin [6] introduced modifications to the seeker optimization algorithm to control exploitation/exploration balance and hybridized it with elements of the firefly algorithm that improved its exploitation capabilities for constrained optimization problems with feasibility rules.

Methods based on stochastic ranking. Rodrigues [7] presented a new technique to handle constraints in the solution of optimization problems by evolutionary algorithms - the Balanced Ranking Method (BRM).

Other hybrid methods. Gao et al. [8] defined the best fitness value among feasible solutions in current population as gbest and converted the original COPs to multi-objective optimization problems (MOPs) with one constraint. Wang et al. [9] incorporated the objective function information into the feasibility rules for solving constrained optimization problems with evolutionary optimization.

All these methods have their own advantages. Among all the constraints handling methods, the well-known feasibility rule is widely used because of the simplicity. Since feasible solutions are preferred unfeasible ones, the algorithms often lack diversity in population. An effective selection method should balance the information obtained by objective function and constrained functions during searching progress, which can maintain the diversity of the population.

In this paper, a balancing artificial bee colony (BABC) algorithm is proposed to overcome shortcomings mentioned above. In BABC algorithm, a modified ABC algorithm serves as the search engine and the well-known feasibility rule is used as the constraints handling method. In addition, the replacement mechanism and mutation strategy are used to compare individuals based on objective function. Furthermore, the information of objective function has also been used to guide the search. The performance of the BABC algorithm has been tested on 18 benchmark test functions collected in IEEE CEC2010 and compared with other state-of-the-art algorithms [9].

The reminder of the paper is organized as follows. In Section 2, a balancing artificial bee colony algorithm for solving constrained optimization problem is introduced. In Section 3, the proposed algorithm is compared with other algorithms. In Section 4, a conclusion is provided.

2. Methodology

Generally, for minimization problems, a constrained optimization problem can be formulated as follows:

$$\begin{array}{ll} \min f(x), & x = (x_1, x_2, ..., x_n) \in R^n \\ \text{subject to} & l_i \le x_i \le u_i,\ i = 1, ..., n, \\ & g_j(x) \le 0 \text{ for } j = 1, ..., q, \\ & h_j(x) = 0 \text{ for } j = q+1, ..., m. \end{array}$$

The objective function f is defined on an n-dimensional rectangle S in $R^n (S \subseteq R^n)$. Domains of variables are defined by their own upper and lower bounds. A feasible region $F \subseteq S$ is defined by m constraints, and x is defined on the feasible region. If there exists an inequality constraint that satisfies $g_j(x) = 0,\ j = 1, ..., q$ for any $x \in F$, the constraint is called active constraint at point x. All equality constraints are considered active at all points of F.

In the original ABC algorithm, the swarm consists of three kinds of artificial bees: employed bees, onlooker bees and scout bees. Employed bees explore the food source and share the information to the onlooker bees in the hive. Onlooker bees select a food source and exploit it based on the information shared by employed bees. The employed bee whose food source has been abandoned becomes a scout bee to search a new food source. For constrained optimization problems, the feasibility rules are used as the constraints handling method in ABC algorithm. However, in the method based on feasibility rules, the feasible solution prefers unfeasible solution, so that it will cause premature convergence. How to balance the constraints and objective function is a basic problem when solving constrained optimization problems by the ABC algorithm. Denote the food source number as SN, the position of the ith food source as x_i, $(i = 1, \cdots, SN)$, which is a D-dimensional vector. In the following subsections, the BABC algorithm for constrained optimization problems is described in detail to overcome the shortages described above.

2.1. Opposition-based learning initialization

It is important to generate an even initial population of swarm intelligence-based algorithms, which can affect the convergent speed and solution quality. In the initialization, there is no information about the distribution of solution. Therefore, the initial population needs to be scatted evenly on searching space to make sure the algorithm can search the whole space uniformly. In original ABC algorithm, the initial population is generated randomly in the searching space, which cannot ensure the evenly distributed population initialization. To conquer this issue, the initial population is generated by using opposition-based learning initialization. Notice that opposition-based learning method can simultaneously generate a solution and its opposite [10], which can make the initial population be scattered evenly over the searching space. The algorithm for generating an initial population based on opposition-based learning initialization method is given as follows.

Algorithm 1: Generation of initial population.

Step 1: Set the population size SN and the individual counter $i = 1, j = 1$.
Step 2: Randomly generate a population $X(SN)$ by

$$x_{ij} = x_{\min,j} + \text{rand}(0,1)\,(x_{\max,j} - x_{\min,j})\,,$$
$$i \in \{1, 2, \cdots, SN\}\,, j \in \{1, 2, \cdots, D\}\,. \tag{1}$$

Step 3: For $i = 1, \cdots, SN; j = 1, \cdots D$, generate a population $OX(SN)$ by

$$ox_{ij} = x_{\min,j} + x_{\max,j} - x_{i,j}.$$

Step 4: Among the $X(SN)$ and $OX(SN)$, select SN individuals having the smallest costs as the initial population.

2.2. Searching equation

It is well known that the exploration ability and exploitation ability are both crucial for the search equations to find the optima in searching space, and the tow abilities are contradict to each other. It is well known that the search equation is good at exploration and poor at exploitation in original ABC algorithm. In order to balance the exploration ability and exploitation ability, two kinds of searching equations are used in BABC algorithm.

At the employed bee stage, the original searching equation is also used to keep the exploration ability, that is

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj})\,, \tag{2}$$

where k, j are randomly selected indices, $k \in \{1, 2, ..., SN\}$, $k \neq i$, $j \in \{1, 2, ..., N\}$, and $\phi_{ij} \in [-1, 1]$ is a random number.

At onlooker bee stage, the searching equation is modified as best guided searching equation proposed by Zhu et al. [11] to enhance the exploitation ability, that is

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) + \varphi_{ij}(x_{\text{best},j} - x_{lj})\,, \tag{3}$$

where $k, l \in \{1, 2, ..., SN\}$ is a random selected index which is different from i, $j \in \{1, 2, ..., D\}$ is a randomly selected index, $x_{\text{best},j}$ is the global best solution, $\phi_{ij} \in [-1, 1]$ and $\varphi_{ij} \in [0, 1.5]$ are both uniformly distributed random numbers.

At the scout bee stage, the individual is generated randomly within the range of the boundaries of the parameters by (1).

2.3. Replacement mechanism and mutation strategy

By replacing some individuals from the population, the replacement mechanism attends to alleviating the greediness of the feasibility rule. It can maintain the diversity of the population and balance the constraints and objective function. First, divide the population into S parts with the same size and sort it based on the objective function value in descending order. Second, choose the individual with the maximum degree of constraint violation from the first part and the individual with the minimum degree of constraint violation from A, where the set A consists of the individuals that cannot survive at the employed bee and onlooker bee stages. Third, compare the objective function value of the two selected individuals, and store the minimum one into the population, the other stored into A. Repeat the above step on all S parts of population respectively. Therefore, the replacement mechanism can be described as the following algorithm:

Algorithm 2: Replacement mechanism

```
Sort population in descending order according to the objective function values and
divide it into $S$ parts with the same size;
$i = 1$
while $A$ is not empty and $i < S$
    Select the individual $x_a$ with the maximum degree of constraint violation from
    the $i$th part of population;
    Select the individual $x_b$ with the minimum degree of constraint violation from $A$;
    if $f(x_b) < f(x_a)$
        Replace $x_a$ with $x_b$ and delete $x_b$ from $A$;
    end if
    $i = i + 1$
end while
```

Consider the premature issue, a simple mutation strategy is used when all the individuals in the population are unfeasible. The details are as follows:

Algorithm 3: Mutation strategy

```
if all the individuals in the population are unfeasible
    Randomly select an individual $x_c$ from population;
    Generate a random integer number $k$ between 1 and $D$, and let the $k$th dimension
    of $x_c$ be equal to a value randomly chosen from $[L_k, U_k]$. Thus obtain a mutation
    individual $x_d$;
    Choose the individual $x_e$ with the maximum degree of constraint violation in
    population;
    if $f(x_d) < f(x_e)$
        Replace $x_e$ with $x_e$;
    end if
end if
```

2.4. BABC algorithm

In BABC algorithm, the probability of a food source being selected by an onlooker bee is given by:

$$p_i = \begin{cases} 0.5 + \left(\frac{\text{fit}_i}{\sum_{i=1}^{SN} \text{fit}_i}\right) \times 0.5 & \text{if solution is feasible,} \\ \left(1 - \frac{\text{violation}_i}{\sum_{i=1}^{SN} \text{violation}_i}\right) & \text{if solution is unfeasible,} \end{cases} \tag{4}$$

where violation_i is the penalty value of the solution x_i. The ith fitness value fit_i for a minimization problem is defined as

$$\text{fit}_i = \begin{cases} \frac{1}{1+f_i}, & f_i \geq 0, \\ 1 + \text{abs}(f_i), & f_i < 0, \end{cases} \tag{5}$$

where f_i is the objective function value of the ith solution.

Therefore, the solution procedure of the BABC algorithm is as follows:

Algorithm 4: Balancing artificial bee colony algorithm

Step 1: Initialize the food source by algorithm 1 and set all the parameters in the algorithm.

Step 2: The employed bees search the new food source according to (2). Evaluate the objective function value, fitness value and violation value of this food source. Then, use the feasibility rule to make a choice between the new food source and the old one. If the new food source is preferred to the old one, the new one replaces the old one.

Step 3: Calculate the selective probability according to (5) and select a food source for onlooker bees. Then, the onlooker bees search the new food source according to (3). And evaluate the objective function value, fitness value and violation value of this food source. Then, use feasibility rule to make a choice between the new food source and the old one. If the new food source is preferred to the old one, the new one replaces the old one.

Step 4: Replace some individuals according to the replacement mechanism introduced by **Algorithm 2**.

Step 5: Implement the mutation strategy introduced by **Algorithm 3**.

Step 6: If there is an abandoned food source, replace it with a new food source discovered by the scout bee.

Step 7: Check whether the termination criteria are satisfied. If they are, stop searching and output the final food position; otherwise, return to **Step 2**.

3. Result analysis and discussion

In this section, the effectiveness of the BABC algorithm is empirically evaluated on a set of benchmark test functions with 18 benchmark test functions with 30-D developed in IEEE CEC2010. Note that all objective functions of these benchmark test functions should be minimized. The parameters settings of BABC algorithm are as follow: colony size (SN) is 80, maximum cycle number (MCN) is 7500, the value of limit is $0.5 * SN * D$. Thus, the maximum number of FEs is 600000. Experiments were repeated 25 times independently.

Table 1 show the statistical results of original ABC algorithm and BABC algorithm for 18 benchmark test functions. In these tables, Best, Mean and Worst represent the best, mean and worst function value over 25 independent runs; Std. represents the stability of algorithm; Percentage represents the percentage of feasible individuals in population. The results show that the BABC algorithm can obtain the optimal solution stability better than original ABC algorithm. From the last column, the percentage values show that the BABC algorithm also has more feasible individuals in population than original ABC algorithm dose.

To further exam the performance of the proposed algorithm, the best solutions are compared with εDEag, SRS-εDEag, ECHT-DE, AIS-IRP, DyHF, FROFI and CMODE. All the comparison results show in Table 2. According to the mean results, it is shown that BABC algorithm performs equal or better than εDEag, SRS-εDEag, ECHT-DE, AIS-IRP, DyHF, FROFI, CMODE and ABC. The compared results show that the proposed algorithm has the better performance on constrained optimization problems.

It is well known that the performance of any evolutionary algorithm on constrained optimization problem is highly affected by the constraint handling method. In fact, the algorithm using feasibility rules lack of objective function information used in the population. In BABC algorithm, we combine the objective function information through searching procedure, such that the proposed algorithm can efficiently and effectively solve the constrained optimization problems.

4. Conclusion

In this paper, we present a balancing ABC algorithm for constrained optimization problems, called BABC. In BABC, the information of objective function has been used to alleviate the greediness and improve the robustness of the well-known feasibility rule by the replacement mechanism and the mutation strategy. To improve the convergent speed, the opposite learning initialization on initial stage and the best-lead search equation on onlooker bee stage are used. Experiments are performed on benchmark test sets from IEEE CEC2010. The comparative studies indicate that:

i) the opposite learning initialization is a promising way to deal with the diversity of initial population;

ii) the best-lead search equation adding in original ABC algorithm make the algorithm to have the capability to improve the convergent speed during evolution;

iii) the replacement mechanism and mutation strategy can balance the information of both constraints and objective function, and increase the diversity of population.

In conclusion, the BABC algorithm can be efficiently used for solving constrained optimization problems. In the future, there are some directions that should be noticed:

i) to design new constrained handling model for solving more complicated high-dimension and large-scale constrained optimization problems;

ii) to combine with advanced discrete processing methods for discrete constrained optimization problems.

Table 1. Statistical results over 25 independent runs

Fun	Algorithm	Best	Mean	Worst	Std	Percentage
C01	ABC	–5.79E−01	–5.54E−01	–5.30E−01	1.49E−02	1.00
	BABC	–5.64E−01	–4.75E−01	–3.66E−01	4.71E−02	1.00
C02	ABC	2.80E+00	3.94E+00	4.75E+00	5.44E−01	0.33
	BABC	1.89E+00	3.71E+00	4.33E+00	5.18E−01	0.97
C03	ABC	2.63E+02	6.79E+02	1.51E+03	3.49E+02	0.00
	BABC	4.74E+03	3.60E+04	1.08E+05	2.60E+04	0.00
C04	ABC	1.32E+00	2.51E+00	4.79E+00	9.16E−01	0.00
	BABC	3.80E+01	1.87E+04	2.03E+05	4.97E+04	0.00
C05	ABC	6.95E−05	2.43E−03	8.45E−03	1.76E−03	0.00
	BABC	1.97E−03	4.21E+02	5.31E+02	1.11E+02	0.83
C06	ABC	5.53E−04	3.04E−03	7.81E−03	1.89E−03	0.00
	BABC	8.29E−05	4.42E+02	5.61E+02	1.49E+02	0.78
C07	ABC	1.63E−02	1.43E+00	1.51E+01	3.09E+00	1.00
	BABC	1.29E+07	1.48E+09	1.19E+10	3.06E+09	1.00
C08	ABC	2.50E−02	3.93E+00	4.79E+01	1.01E+01	1.00
	BABC	1.69E+09	2.06E+10	4.35E+10	1.21E+10	1.00
C09	ABC	8.40E−05	1.97E+12	4.94E+13	9.87E+12	0.00
	BABC	4.04E−04	1.92E+13	3.61E+13	8.79E+12	0.85
C10	ABC	1.10E−04	4.39E+12	4.25E+13	1.24E+13	0.00
	BABC	1.24E+13	2.19E+13	3.64E+13	6.64E+12	0.92
C11	ABC	1.04E−03	3.40E−01	1.84E+00	4.39E−01	0.00
	BABC	2.67E+04	1.28E+06	1.97E+07	3.91E+06	0.00
C12	ABC	–8.68E+02	–4.21E+02	2.51E+02	3.84E+02	0.07
	BABC	2.78E+01	2.27E+10	2.91E+11	7.22E+10	0.00
C13	ABC	–6.45E+01	–6.18E+01	–6.01E+01	1.16E+00	0.97
	BABC	–5.51E+01	–4.33E+01	–2.40E+01	8.23E+00	1.00
C14	ABC	1.64E+13	6.57E+13	1.19E+14	2.84E+13	1.00
	BABC	1.09E+14	2.07E+14	2.86E+14	4.86E+13	1.00
C15	ABC	1.76E+14	3.26E+14	4.80E+14	7.14E+13	0.60
	BABC	1.16E+14	2.60E+14	3.77E+14	6.50E+13	1.00
C16	ABC	1.13E+00	1.17E+00	1.21E+00	1.97E−02	0.22
	BABC	1.06E+00	1.13E+00	1.17E+00	2.88E−02	1.00
C17	ABC	9.07E+02	1.36E+03	2.32E+03	2.96E+02	0.13
	BABC	7.10E+02	1.13E+03	1.74E+03	2.75E+02	0.97
C18	ABC	1.85E+04	3.02E+04	4.25E+04	5.38E+03	0.37
	BABC	1.56E+04	2.59E+04	3.51E+04	5.08E+03	0.99

Table 2: The best solution obtained by εDEag, SRS-εDEag, ECHT-DE, AIS-IRP and BABC

Fun	εDEag	SRS-εDEag	ECHT-DE	AIS-IRP	BABC
C01	–8.21E–01	–8.21E–01	–8.00E–01	–8.20E–01	–4.75E–01
C02	–2.15E+00	–2.19E+00	–1.99E+00	–2.21E+00	3.71E+00
C03	2.88E+01	2.87E+01	9.89E+01	6.68E+01	3.60E+04
C04	8.16E–03	5.70E–03	–1.03E–06	1.98E–03	1.87E+04
C05	–4.50E+02	–4.63E+02	–1.06E+02	–4.36E+02	4.21E+02
C06	–5.28E+02	–5.29E+02	–1.38E+02	–4.54E+02	4.42E+02
C07	2.60E–15	2.70E–15	1.33E–01	1.07E+00	1.48E+09
C08	7.83E–14	4.90E–14	3.36E+01	1.65E+00	2.06E+10
C09	1.07E+01	2.43E+00	4.24E+01	1.57E+00	1.92E+13
C10	3.33E+01	3.29E+01	5.34E+01	1.78E+01	2.19E+13
C11	–2.86E–04	–2.99E–04	2.60E–03	–1.58E–04	1.28E+06
C12	3.56E+02	2.13E+02	–2.51E+01	4.29E–06	2.27E+10
C13	–6.54E+01	–6.59E+01	–6.46E+01	–6.62E+01	–4.33E+01
C14	3.09E–13	1.04E–13	1.24E+05	8.68E–07	2.07E+14
C15	–8.21E–01	–8.21E–01	–8.00E–01	–8.20E–01	–4.75E–01
C16	–2.15E+00	–2.19E+00	–1.99E+00	–2.21E+00	3.71E+00
C17	2.88E+01	2.87E+01	9.89E+01	6.68E+01	3.60E+04
C18	8.16E–03	5.70E–03	–1.03E–06	1.98E–03	1.87E+04

Table 3: The best solution obtained by FROFI, DyHF, CMODE, ABC and BABC

Fun	FROFI	DyHF	CMODE	ABC	BABC
C01	–8.21E–01	–8.21E–01	–8.21E–01	–5.54E–01	–4.75E–01
C02	–2.00E+00	5.74E–01	9.75E–01	3.94E+00	3.71E+00
C03	2.87E+01	3.03E+12	2.18E+01	6.79E+02	3.60E+04
C04	–3.33E–06	8.25E+00	6.72E–04	2.51E+00	1.87E+04
C05	–4.81E+02	2.95E+12	2.77E+02	2.43E–03	4.21E+02
C06	–5.29E+02	–2.10E+01	–4.96E+02	3.04E–03	4.42E+02
C07	0.00E+00	1.59E–01	5.24E–05	1.43E+00	1.48E+09
C08	0.00E+00	4.72E+00	3.68E–01	3.93E+00	2.06E+10
C09	4.30E+01	1.50E+13	1.72E+13	1.97E+12	1.92E+13
C10	3.13E+01	1.57E+13	1.60E+13	4.39E+12	2.19E+13
C11	–3.92E–04	–1.68E–01	9.50E–03	3.40E–01	1.28E+06
C12	–1.99E–01	–1.59E+01	–3.46E+00	–4.21E+02	2.27E+10
C13	–6.83E+01	–6.61E+01	–3.89E+01	–6.18E+01	–4.33E+01
C14	9.80E–29	2.41E+12	9.31E+00	6.57E+13	2.07E+14
C15	2.16E+01	5.49E+13	1.51E+13	3.26E+14	2.60E+14
C16	0.00E+00	7.41E–01	6.30E–02	1.17E+00	1.13E+00
C17	1.59E–01	6.04E+02	3.12E+02	1.36E+03	1.13E+03
C18	4.87E–01	1.18E+04	7.36E+03	3.02E+04	2.59E+04

References

[1] D. Karaboga, B. Akay: *A modified artificial bee colony (ABC) algorithm for constrained optimization problems.* Applied Soft Computing *11* (2011), No. 3, 3021–3031.

[2] Z. Michalewicz, M. Schoenauer: *Evolutionary algorithms for constrained parameter optimization problems.* IEEE Journal Evolutionary Computation *4* (1996), No. 1, 1–32.

[3] M. Thakur, S. S. Meghwani, H. Jalota: *A modified real coded genetic algorithm for constrained optimization.* Applied Mathematics and Computation *235* (2014), 292 to 317.

[4] S. M. Elsayed, R. A. Sarker, D. L. Essam: *Training and testing a self-adaptive multi-operator evolutionary algorithm for constrained optimization.* Applied Soft Computing *26* (2015), 515–522.

[5] K. Deb: *An efficient constraint handling method for genetic algorithms.* Computer Methods in Applied Mechanics and Engineering *186* (2000), Nos. 2–4, 311–338.

[6] M. Tuba, N. Bacanin: *Improved seeker optimization algorithm hybridized with firefly algorithm for constrained optimization problems.* Neurocomputing *143* (2014), 197 to 207.

[7] M. de Castro Rodrigues, B. S. L. M. P. de Lima, S. Guimarães: *Balanced ranking method for constrained optimization problems using evolutionary algorithms.* Information Sciences *327* (2016) 71–90.

[8] L. Gao, Y. Zhou, X. Li, Q. Pan, W. Yi: *Multi-objective optimization based reverse strategy with differential evolution algorithm for constrained optimization problems.* Expert Systems with Applications *42* (2015), No. 14, 5976–5987.

[9] Y. Wang, B. C. Wang, H. X. Li, G. G. Yen: *Incorporating objective function information into the feasibility rule for constrained evolutionary optimization.* IEEE Transactions on Cybernetics *46* (2016), No. 12, 2938–2952.

[10] Q. Xu, L. Wang, N. Wang, X. Hei, L. Zhao: *A review of opposition-based learning from 2005 to 2012.* Engineering Applications of Artificial Intelligence *29* (2014) 1–12.

[11] G. Zhu, S. Kwong: *Gbest-guided artificial bee colony algorithm for numerical function optimization.* Applied Mathematics and Computation *217* (2010), No. 7, 3166–3173.

Received June 6, 2017