Improved Artificial Bee Colony Algorithm for Constrained Problems

Ivona BRAJEVIC
Faculty of Mathematics
University of Belgrade
Studentski trg 16
SERBIA
ivona.brajevic@googlemail.com

Milan TUBA
Faculty of Computer Science
University Megatrend Belgrade
Bulevar umetnosti 29
SERBIA
tubamilan@ptt.rs

Milos SUBOTIC
Faculty of Computer Science
University Megatrend Belgrade
Bulevar umetnosti 29
SERBIA
milos.subotic@gmail.com

Abstract: - In this paper an improved version of the Artificial Bee Colony (ABC) algorithm adjusted for constrained optimization problems is presented. It has been implemented and tested on several engineering benchmarks which contain discrete and continuous variables. Our results were compared to the results obtained by Simple Constrained Particle Swarm optimization algorithm (SiC-PSO) which showed a very good performance when it was applied to the same problems. Our results are of the comparable quality with faster convergence.

Key-Words: - Constrained optimization, Evolutionary computing, Artificial Bee Colony

1 Introduction

Constrained Optimization problems have numerous applications. Engineering design is one of the scientific fields in which constrained optimization problems frequently arise [1]. These types of problems normally have mixed (continuous and discrete) design variables, nonlinear objective functions and nonlinear constrains. Constrains are very important in engineering design problems, since they are usually imposed in the statement of the problems and sometimes are very hard to satisfy, which makes the search difficult and inefficient.

Different deterministic as well as stochastic algorithms have been developed for solving constrained optimization problems. Deterministic approaches such as sequential quadratic programming methods and generalized reduced gradient methods [2] are inflexible to adapt the solution algorithm to a given problem. Generally a given problem is modelled in such a way that a classical algorithm can handle it [3]. This generally requires making several assumptions which might not be easy to justify in many situations. Therefore their applicability is limited. On the other hand, stochastic optimization algorithms such as Genetic Algorithms, Evolution Strategies, Evolutionary Programming and Particle Swarm Optimization (PSO) do not make such assumptions and they have been successfully applied for solving constrained optimization problems during the past few years [1].

Karaboga has described an Artificial Bee Colony (ABC) algorithm based on the foraging behaviour of honey bees for numerical optimization problems [4]. Karaboga and Basturk have compared the performance of the ABC algorithm with the performance of other well-known modern heuristic algorithms such as Genetic Algorithm (GA), Differential Evolution (DE), Particle Swarm Optimization (PSO) on unconstrained and constrained problems [5[, [6]. It has been shown that the ABC algorithm can be efficiently used for solving unconstrained and constrained optimization problems. In this work, our approach to the ABC algorithm for constrained optimization problems called SC-ABC (Simple Constrained ABC) was applied to real engineering problems existing in the literature and its performance was compared with the performance of Simple Constrained Particle Swarm Optimizer (SiC-PSO) [1]. SiC-PSO algorithm showed a very good performance when it was applied to several engineering design optimization problems.

This paper is organized as follows. Section 2 describes the ABC algorithm for constrained problems. Section 3 presents our proposed approach. Section 4 describes three benchmark problem formulations. Section 5 presents the experimental setup adopted and provides an analysis of the results obtained from our empirical study. Our conclusions and some possible plans for future research are provided in Section 6.

2 The ABC Algorithm for Constrained Optimization Problems

General constrained optimization (CO) problem is to find *x* so as to

minimize *f(x)*, $x = (x_1,...,x_n) \in R^n$ where $x \in F \subseteq S$

The objective function f is defined on the search space $S \subseteq R^n$ and the set $F \subseteq S$ defines the feasible region. The search space S is defined as an n-dimensional rectangle in R^n. The variable domains are limited by their lower and upper bounds:

$$l_i \le x_i \le u_i\,,\ 1 \le i \le n$$

whereas the feasible region $F \subseteq S$ is defined by a set of *m* additional constraints ($m \ge 0$):

$$g_j(x) \le 0,\ \text{for}\ j = 1,\dots,q$$
$$h_j(x) \le 0,\ \text{for}\ j = q+1,\dots m$$

In order to handle the constraints of this problem, the ABC algorithm employs Deb's rules [7], which are used instead of the greedy selection employed between v_i and x_i in the original version of the ABC [4]. Deb's method uses a tournament selection operator, where two solutions are compared at a time by applying the following criteria:

- Any feasible solution satisfying all constraints is preferred to any infeasible solution violating any of the constraints
- Among two feasible solutions, the one having better fitness value is preferred
- Among two infeasible solutions, the one having the smaller constraint violation is preferred

Scout phase of the algorithm provides a diversity mechanism that allows new and probably infeasible individuals to be in the population. Beside of Deb's rules, the second change in ABC for CO problems is in order to produce a candidate food position from the old one in memory. The adapted ABC algorithm uses the following expression:

$$v_{ij} = \begin{cases} x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}), if & R_j \le MR \\ x_{ij} \qquad , & otherwise \end{cases} \qquad (1)$$

instead the expression in ABC algorithm:

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}) \qquad (2)$$

where $k \in \{1,2,\dots,SN\}$ and $j \in \{1,2,\dots,D\}$ are randomly chosen indexes where *k* has to be different from *i* and φ_{ij} is a random number between [-1, 1]. *SN* denotes the number of food source positions, *D* is the number of optimization parameters and R_j, $j \in \{1,2,\dots,D\}$, is a randomly chosen real number in the range [0,1]. *MR*, the modification rate, is a control parameter that controls whether the parameter x_{ij} will be modified or not. In adapted ABC algorithm, artificial scouts are produced at a predetermined period of cycles for discovering new food sources randomly. This period is another control parameter called scout production period (*SPP*) of the algorithm. At every *SPP* cycle, it is checked if there is an abandoned food source or not. If there is, a scout production process is carried out.

3 Proposed Algorithm: SC-ABC

In our proposed approach (called Simple Constrained Artificial Bee Colony, or SC-ABC) as in the ABC for constrained problems, algorithm uses Deb's rules instead of the greedy selection in order to decide what solution will be kept. The expression for evaluating probability that an onlooker bee goes to the i-th food source position $X_i = (x_{i1}, x_{i2},\dots,x_{iD})$, where $F(X_i)$ refers to the nectar amount of the food source located at X_i, is:

$$p_i = \frac{F(X_i)}{\sum_{k=1}^{SN} F(X_k)} \qquad (3)$$

Equations (2), (3) and the expression for initialization new food sources:

$$x_{ij} = l_i + \delta(u_i - l_i) \qquad (4)$$

where $j \in \{1,2,\dots,D\}$, l_i and u_i are the lower and upper bound of the parameter x_{ij} and δ is a random number in the range [0, 1), remained the same as in the version of the ABC proposed for unconstrained optimization problems.

SC-ABC algorithm has changed the initialization phase and the scout phase compared to the ABC. In the initialization phase only the first initialization of food sources is completely random. In other initialization phases the first new food source is the food source from the previous run of the algorithm which has the best fitness value. In other words, the runs of the SC-ABC algorithm are not completely independent. Therefore, exploitation of the good sources was increased. In order to increase the exploration the scout bee's phase was changed. In the scout phase the algorithm checks every possible solution. If the solution is not feasible, that food source is replaced with a new randomly produced solution.

The pseudo code of the SC-ABC algorithm is:

```
1. Initialize the population solutions $x_{ij}$, $i=1,2,...,SN$, $j=1,2,...,D$ by Eq. (4) for the
   first run. For every other run, if exists, $x_{1j}$, $j \in \{1,2,...,D\}$ is the best solution from previous
   run and $x_{ij}$, $i = 2,...,SN$, $j = 1,...,D$ are randomly produced solutions by Eq. (4)
2. Evaluate fitness value of the population
3. cycle = 1
4. repeat
5. Produce new solutions $v_{ij}$ for the employed bees by using Eq. (2) and evaluate them
6. Apply selection process based on Deb's method
7. Calculate the probability values $P_{ij}$ for the solutions $x_{ij}$ by Eq. (3)
8. Produce the new solutions $v_{ij}$ for the onlookers from the solutions $x_{ij}$ selected depending on
   $P_{ij}$ and evaluate their fitness value
9. Apply selection process based on Deb's method
10. Determine the abandoned feasible solution for the scout, if exists, and replace it with a new
    randomly produced solution $x_{ij}$ by Eq. (4)
11. Every infeasible solution replace with randomly produced solution $x_{ij}$ by Eq. (4)
12. Memorize the best solution achieved so far
13. cycle = cycle + 1
14. until cycle = MCN
```

The original ABC can be applied only to the continuous problems. However, the method can also be expanded to the discrete problems using discrete numbers. The state variables were treated in the SC-ABC as follows: for continuous variables, initial values were generated randomly between upper and lower bounds of the specification values. The value was also modified in the employed and onlooker bee's phases between the bounds. For discrete variables, they could be handled in Equations (2) and (4) with a small modification, i.e., as though they were continuous with nearest available discrete values then being chosen. In that way, both continuous and discrete numbers can be handled by the algorithm with no inconsistency.

4 Benchmark Problems

Proposed approach to Artificial Bee Colony Algorithm for constrained optimization problems (SC-ABC) was applied to three numerical examples, pressure vessel design optimization problem, tension/compression spring design optimization problem and speed reducer design optimization problem [1]. These non-linear engineering design problems have discrete and continuous variables. These problems represent optimization situations involving discrete and continuous variables that are similar to those encountered in everyday mechanical engineering design tasks.

4.1 Pressure Vessel design optimization problem

This example is to design a compressed air storage tank with a working pressure of 3000 psi and a minimum volume of 750 ft^3. A cylindrical vessel (Fig.1) is capped at both ends by hemispherical heads. Using rolled steel plate, the shell is made in two halves that are joined by the longitudinal welds to form a cylinder. The objective is to minimize the total cost, including the cost of the materials forming the welding. The design variables x_1 - the spherical head thickness and x_2 - the shell thickness have to be integer multiples of 0.0625 inch which are the available thickness of rolled steel plates. The radius x_3 and the length of the shell x_4 are continuous variables.

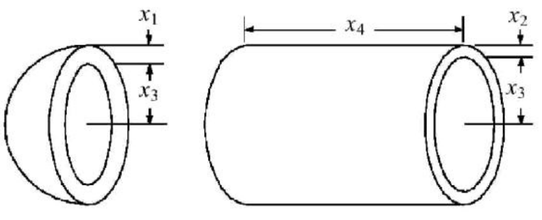

Fig.1: Pressure Vessel design

The mathematical model of the problem is:

Minimize

$$f(X) = 0.6224x_1x_3x_4 + 19.84x_1^2x_3 + 1.7781x_2x_3^3 + 3.1661x_1^2x_4 \quad (5)$$

subject to:

$$c_1(x) = -x_1 + 0.0193x_3 \le 0$$
$$c_2(x) = -x_2 + 0.0954x_3 \le 0$$
$$c_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0$$
$$c_4(x) = x_4 - 240 \le 0$$

where the bounds are: $1 \times 0.0625 \le x_1 \le 99 \times 0.0625$, $1 \times 0.0625 \le x_2 \le 99 \times 0.0625$ and $10 \le x_3, x_4 \le 200$

Best solution is $f(x^*) = 6059.714335$, where $x^* = (0.8125, 0.4375, 42.098446, 176.636596)$.

4.2 Tension/compression spring design optimization problem

This problem minimizes the weight of a tension/compression spring, subject to constraints of minimum deflection, shear stress, surge frequency, and limits on outside diameter and on design variables. There are three continuous variables: the wire diameter x_1, the mean coil diameter x_2, and the number of active coils x_3. The schematic of a pressure vessel is shown in Fig.2.

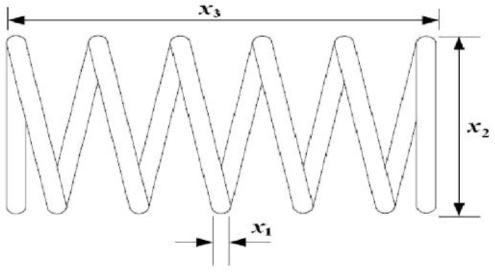

Fig.2: Tension/Compression Spring

The mathematical model of the problem is:

Minimize

$$f(X) = (x_3 + 2)x_1^2 x_2 \tag{6}$$

subject to:

$$c_1(x) = 1 - \frac{x_2^3 x_3}{7178x_1^4} \le 0$$

$$c_2(x) = \frac{4x_2^2 - x_1x_2}{12566x_1^3x^2 - x_1^4} + \frac{1}{5108x_1} - 1 \le 0$$

$$c_3(x) = 1 - \frac{140.45x_1}{x_2^2 x_3} \le 0$$

$$c_4(x) = \frac{x_1 + x_2}{1.5} - 1 \le 0$$

where the bounds are: $0.05 \le x_1 \le 2.0$, $0.25 \le x_2 \le 1.3$ and $2.0 \le x_3 \le 1.3$

Best solution is $f(x^*) = 0.012665$, where $x^* = (0.051690, 0.356750, 11.287126)$.

4.3 Speed Reducer design optimization problem

The design of the speed reducer shown in Fig.3, is considered with the face width x_1, module of teeth x_2, number of teeth on pinion x_3, length of the first shaft between bearings x_4, length of the second shaft between bearings x_5, diameter of the first shaft x_6, and diameter of the first shaft x_7. All variables are continuous except x_3 that is integer. The weight of the speed reducer is to be minimized subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shaft.

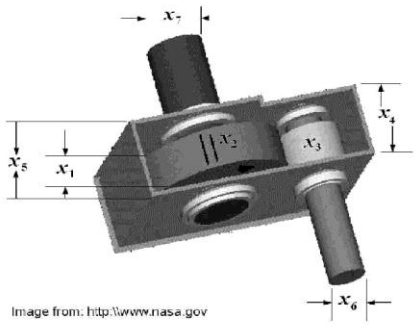

Fig.3: Speed Reducer

The mathematical model of the problem is:

Minimize

$$\begin{aligned} f(X) &= 0.7854x_1x_2^2(3.3333x_3^2 \\ &+ 14.9334x_3 - 43.0934) \\ &- 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^2 + x_7^2) \\ &+ 0.78054(x_4x_6^2 + x_5x_7^2) \end{aligned} \tag{7}$$

subject to:

$$c_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \le 0$$

$$c_2(x) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0$$

$$c_3(x) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0$$

$$c_4(x) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0$$

$$c_5(x) = \frac{1.0}{110x_6^3}\sqrt{\left(\frac{750.0x_4}{x_2x_3}\right)^2 + 16.9\times10^6} - 1 \le 0$$

$$c_6(x) = \frac{1.0}{85x_7^3}\sqrt{\left(\frac{750.0x_5}{x_2x_3}\right)^2 + 157.5\times10^6} - 1 \le 0$$

$$c_7(x) = \frac{x_2x_3}{40} - 1 \le 0$$

$$c_8(x) = \frac{5x_2}{x_1} - 1 \le 0$$

$$c_9(x) = \frac{x_1}{12x_2} - 1 \le 0$$

$$c_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0$$

$$c_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0$$

where the bounds are: $2.6 \le x_1 \le 3.6$, $0.7 \le x_2 \le 0.8$, $17 \le x_3 \le 28$, $7.3 \le x_4 \le 8.3$, $7.8 \le x_5 \le 8.3$, $2.9 \le x_6 \le 3.9$, and $5.0 \le x_7 \le 5.5$

Best solution is $f(x^*) = 2996.348165$, where $x^* = (3.5, 0.7, 17, 7.3, 7.8, 3.350214, 5.286683)$.

5 Parameter settings, results and discussion

Control parameters of the ABC algorithm are: swarm size, limit, number of employed bees, number of onlookers, number of scouts and maximum number of cycles [4]. In these experiments, the colony size was taken 40 and the maximum number of cycles was taken 4000. So, the total objective function evaluation number is 240000. Each experiment was repeated 60 runs. The percentages of onlooker bees and employed bees were 50% of the colony and the number of scout bees was changeable, as it was described in previous section. The value of "limit" is equal $SN(2D+1)$ where SN is the number of possible solutions and D is the dimension of the problem. The performance of the algorithm was considered in terms of the best and average optimum values, and the best solutions were recorded. Our approach to ABC algorithm has been implemented in Java programming language and run on a Pentium Core2Duo, 1.40-GHz personal computer with 2 GB RAM memory.

Parameters adopted for SC-ABC algorithm are given in Table 1.

Control parameters for SC-ABC algorithm	
swarm size	40
limit	*SN*(2D+1)*
number of onlookers	50% of the swarm
number of onlookers	50% of the swarm
number of scouts	changeable

Table 1. Control parameters adopted for SC-ABC algorithm

Tables 2, 3 and 4 show the solution vectors of the best solution reached by our approach to ABC algorithm and the values of the constrains for each of the problems tested.

	Best solution
x_1	0.812500
x_2	0.437500
x_3	42.098187
x_4	176.640750
$c_1(x)$	-4.988451
$c_2(x)$	-0.035883
$c_3(x)$	-5.297613
$c_4(x)$	-63.359250
f(x)	6059.768058

Table 2. ABC solution vector for pressure vessel design optimization problem

	Best solution
x_1	0.051871
x_2	0.361108
x_3	11.036860
$c_1(x)$	-1.634E-
$c_2(x)$	-4.383E-
$c_3(x)$	-4.06213
$c_4(x)$	-0.72468
f(x)	0.01266

Table 3. ABC solution vector for tension / compression spring design optimization problem

	Best solution
x_1	3.500000
x_2	0.700000
x_3	17
x_4	7.300000
x_5	7.800000
x_6	3.350215
x_7	5.286683
$c_1(x)$	-0.073915
$c_2(x)$	-0.197996
$c_3(x)$	-0.499172
$c_4(x)$	-0.90147
$c_5(x)$	-2.220E-16
$c_6(x)$	-3.331E-16
$c_7(x)$	-0.702500
$c_8(x)$	0.000000
$c_9(x)$	-0.583333
$c_{10}(x)$	-0.051326
$c_{11}(x)$	-0.010852
f(x)	2996.348165

Table 4. ABC solution vector for speed reducer design optimization problem

From Tables 2, 3 and 4 can be concluded that the SC-ABC reached for the first two tested problems almost the best known values, and for the third tested problem the best known value. It is important to mention that for the first and second tested problems the program in the most of executions found solution at value between 6060 and 6061 and at value 0.01269, respectively. For the problem Speed reducer, SC-ABC reached the best known value in every run of the program execution.

Our results were compared to the results reached by Simple Constrained Particle Swarm optimization algorithm (SiC-PSO) [2]. Tables 6 and 7 show best, average fitness values and standard deviation for each of the problems tested.

Prob.	Optimal	SC-ABC	SiC-PSO
Ex. 1	6059.714335	6059.768058	6059.714335
Ex. 2	0.012665	0.012667	0.012665
Ex. 3	NA	2996.348165	2996.348165

Table 6. Best results obtained by SC-ABC and SiC-PSO

	Average		St. Dev.	
Prob.	SC-ABC	SiC-PSO	SC-ABC	SiC-PSO
Ex. 1	6060.2097	6092.0498	0.0069	12.1725
Ex. 2	0.0127	0.0131	2.4 E-07	4.1 E-04
Ex. 3	2996.3482	2996.3482	0.0000	0.0000

Table 7. Average and Standard Deviations for the results obtained

The results from Table 6 and 7 show that the average values reached by SC-ABC, for each problem tested, are better than the average values reached by SiC-PSO. But the SiC-PSO reached the best known values for each problem tested. It can be seen that the SC-ABC can converge very quickly towards the global optimum. To have better results the SC-ABC algorithm needs to be modified in some way to avoid the algorithm to trap at some local attractors.

6 Conclusion

In this paper, we present an improved ABC algorithm for constrained problems (SC-ABC). The SC-ABC was tested on three constrained optimization problems which contain discrete and continuous variables. The algorithm showed a good performance. We compared our results to the results reached by Simple Constrained Particle Swarm optimization algorithm (SiC-PSO) which showed a very good performance when it was applied to the same problems. Although our algorithm did not obtain the optimal values for each tested problem, the average values reached by SC-ABC are better. We can conclude that the SC-ABC can quickly search toward the global optimum and can be a promising alternative for solving this sort of problems due to its simplicity and reliability. As part of our future work, we are interested to perform a more detailed statistical analysis of the performance of our proposed approach and to improve the new algorithm's ability to escape the local attractors.

Acknowledgment: This research is supported by Project 144007, Ministry of Science, Republic of Serbia.

References:

[1] L. C. Cagnina, S. C. Esquive: Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer, Informatica, No.32, 2008, pp. 319-326.

[2] O. Yeniay: A comparative study on optimization methods for the constrained nonlinear programming problems, Mathematical Problems in Engineering Hindawi Publishing Corporation, 2005, pp. 165-173.

[3] A. Baykasoglu, L. Özbakır, P. Tapkan: Artificial Bee Colony Algorithm and Its Application to Generalized Assignment Problem, Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, Book edited by:Felix T. S. Chan and Manoj Kumar Tiwari, pp. 532, December 2007, Itech Education and Publishing, Vienna, Austria.

[4] D. Karaboga: An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[5] D. Karaboga, B. Basturk: Artificial Bee Colony Optimization (ABC) Algorithm for Solving Constrained Optimization Problems, IFSA 2007, LNAI 4529, Springer-Verlag, Berlin, Heidelberg, pp. 789-798

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

[7] D.E. Goldberg, K. Deb: A comparison of selection schemes used in genetic algorithms Foundations of Genetic Algorithms, edited by G. J. E. Rawlins, pp. 69-93, 1991.