Enhanced Constrained Artificial Bee Colony Algorithm for Optimization Problems

Soudeh Babaeizadeh and Rohanin Ahmad
Department of Mathematical Sciences, Universiti Teknologi Malaysia, Malaysia

Abstract: *Artificial Bee Colony (ABC) algorithm is a relatively new swarm intelligence algorithm that has attracted great deal of attention from researchers in recent years with the advantage of less control parameters and strong global optimization ability. However, there is still an insufficiency in ABC regarding its solution search equation, which is good at exploration but poor at exploitation. This drawback can be even more significant when constraints are also involved. To address this issue, an Enhanced Constrained ABC algorithm (EC-ABC) is proposed for Constrained Optimization Problems (COPs) where two new solution search equations are introduced for employed bee and onlooker bee phases respectively. In addition, both chaotic search method and opposition-based learning mechanism are employed to be used in population initialization in order to enhance the global convergence when producing initial population. This algorithm is tested on several benchmark functions where the numerical results demonstrate that the EC-ABC is competitive with state of the art constrained ABC algorithm.*

Keywords: *ABC, constrained optimization, swarm intelligence, search equation.*

Received November 11, 2014; accepted March 2, 2015

1. Introduction

Global optimization deals with optimization problems that might have more than one local minimum. Therefore, finding global minimum out of a set of local minima solutions in a certain feasible region can be challenging. While these problems can even be more challenging when constraints are also involved. In real-world, most of the problems in science and engineering are Constrained Optimization Problems (COPs).

In general COPs can be formulated as following Problem.

$$\begin{aligned} & min \;\; f(\boldsymbol{x}) \\ & s.t \;\;\; g_j(\boldsymbol{x}) \leq 0, \quad j = 1, 2, \ldots, m \\ & \qquad h_j(\boldsymbol{x}) = 0, \quad j = m+1, \ldots, l \end{aligned} \tag{1}$$

Where $x=[x_1, x_2,\ldots, x_n]\in R^n$ is an n-dimensional decision vector and each x_i is bounded by lower and upper bounds as $[x_{min}, x_{max}]$. The objective function $f(x)$ is defined on S and is an n- dimensional search space in R^n.

Optimization methods to solve COPs can be classified into two main categories: derivative-based methods and derivative-free methods.

There have always been many real world problems with non-differentiable constraints, and disjoint feasible domains. These difficulties can make it very challenging for derivative-based methods to find even a feasible solution, let alone an optimal solution.

Furthermore, if derivative-based methods can obtain solutions they are usually only locally optimal. Derivative-free methods in contrast utilize a population of individuals in a search domain. Moreover, they only use the evaluations of the objective function to direct the search. Therefore, they do not usually pose limitations related to derivative-based methods, and they do not easily fall into local optima.

Population based algorithms as significant branch of derivative-free methods capture much attention in recent years in solving COPs. The most prominent Evolutionary Algorithms (EAs) suggested in the literatures are Genetic Algorithm (GA) [17, 18], Particle Swarm Optimization (PSO) [19], Ant Colony Optimization (ACO) [15], Differential Evaluation (DE) [12, 20] and Artificial Bee Colony (ABC) algorithm [21].

Among these population-based algorithms ABC is an effective algorithm proposed for global optimization. Numerical performance demonstrated that ABC algorithm is competitive to that of other population-based algorithms with the advantage of employing fewer control parameters and the need for fewer function evaluations to arrive at an optimal solution [22, 24, 25]. Due to its simplicity and ease of implementation, ABC has captured much attention and has been employed to solve many numerical as well as practical optimization problems since its inception [2, 11, 16, 28, 31].

In general, most of the optimization algorithms have been initially introduced to address unconstrained optimization problems. Therefore, constraint handling techniques are employed to direct the search towards the feasible regions of the search space.

In recent years, a variety of constraints handling techniques have been developed. These methods were categorized into four groups by Koziel and Michalewicz [20]:

1. Methods based on penalty functions which penalize constraints to deal with constrained problem as an unconstrained one,
2. Methods based on reservation of feasible solutions by transforming infeasible solutions to feasible ones with some operators,
3. Methods that separate feasible and infeasible solutions,
4. Other hybrid methods.

ABC algorithm was originally introduced by Koziel and Michalewicz [20] to tackle unconstrained optimization problems. Later on, this method was extended by Karaboga and Bastruck [22] to solve COPs. In recent years there have been many efforts to develop a constrained ABC algorithm possessing balanced exploration and exploitation behavior. However, based on the No Free Lunch (NFL) theorem [36] none of the available algorithms is entirely efficient for every problem.

In this paper an Enhanced Constrained-ABC (EC-ABC) algorithm is proposed to solve COPs by employing two new search equations for employed bee and onlooker bee phases. Moreover, chaotic search mechanism and opposition-based learning method are applied to initialize population with the aim of preventing algorithm from getting stuck at local minima.

The rest of this paper is organized as follows. Section 2 describes the original ABC algorithm. Section 3 includes brief review on constrained ABC algorithm, while section 4 details proposed method. After that, in section 5 experimental results are carried out to test the performance of EC-ABC algorithm on solving COPs. Finally, some conclusions are drawn in section 6.

2. Artificial Bee Colony

ABC is a relatively new population-based algorithm developed by Karaboga [21] emulating the foraging behaviour and waggle dance of honey bee swarm.

Artificial bee colonies are classified into three groups, employed bees, onlooker bees and scout bees. Half of the colony includes employed bee and the other half consist of onlooker bees. In ABC, the position of food source denotes a possible solution to the optimization problem and the nectar amount of food source represents fitness value of the associated solution. The number of employed bees or the onlooker bees is equal to the number of Solutions (SN) in the population. Each solution x_i (i= 1, 2, ..., SN) is a d-dimensional vector and $x_i=\{x_{i1}, x_{i2}, ..., x_{id}\}$ represents the i^{th} solution in the population.

At initialization step, ABC generates a randomly distributed initial population of SN solutions using following Equation 2.

$$x_{ij} = x_{min,j} + rand(0,1)(x_{max,j} - x_{min,j}) \quad (2)$$

Where each solution x_i, i=1, 2, ..., SN is d-dimensional vector for j=1, 2, ..., d. In addition, $x_{min,j}$ and $x_{max,j}$ are the lower and upper bounds for the dimension j respectively. These food sources are randomly assigned to SN number of employed bees and their fitness are evaluated.

After initialization, the population of the solutions is subjected to repeat the search processes for employed bee, the onlooker bees and the scout bee phases. The process continues until the algorithm reaches the Maximum Cycle Number (MCN). In employed bee phase each employed bees produces a modification on the solution x_i where only one dimension of this solution is changed using Equation 3 and the rest keep the same as x_i.

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \quad (3)$$

Where $k\in\{1, 2, ..., SN\}$ and $j\in\{1, 2, ..., d\}$ are randomly chosen indexes and k has to be different from i. Φ_{ij} is a random number in the range $[-1,1]$. After v_i is obtained its fitness value is evaluated and a greedy selection mechanism is applied comparing x_i with v_i. If the fitness value of the new solution v_i is less than the current solution then, the solution is replaced with the x_i, otherwise the current solution remains.

After the employed bee phase, the solution information is transferred to the onlooker bee phase. In this phase a solution is chosen depending on the probability value p_i associated with that solution calculated using the following equation:

$$p(x_i) = \frac{fit(x_i)}{\sum_{j=1}^{SN} fit(x_j)} \quad (4)$$

The $fit(x_i)$ is defined as following Equation 5.

$$fit(x_i) = \begin{cases} (\frac{1}{1+f(x_i)}) & f(x_i) > 0 \\ 1+|f(x_i)| & f(x_i) < 0 \end{cases} \quad (5)$$

Where $f(x_i)$ is the objective value of solution x_i. Once the onlooker has selected solution x_i a modification is done on this solution similar with employed bee using Equation 3. Then, fitness values of generated solutions are evaluated and greedy selection mechanism is employed. If new solution has better fitness value than current solution, the new solution remains in the population and the old solution is removed.

In the scout bee phase, if solution x_i cannot be improved further through a predetermined number of cycles (*limit)*, then that solution is abandoned and replaced with a new solution generated randomly by using Equation 2.

According to the abovementioned description, ABC main procedure can be summarized in Algorithm 1.

Algorithm 1: Original ABC algorithm.

Initialize the population of solution
Evaluate the initial population
cycle=1
Repeat
Employed bee phase
Apply greedy selection process
Calculate the probability values for
Onlooker bee phase
Scout bee phase
Memorize the best solution achieved so far i= 1, 2, …, SN
cycle=cycle+1
until cycle=maximum cycle number

3. Constrained ABC

ABC algorithm has been originally suggested to deal with un-COPs [20]. This algorithm is then adapted to tackle COPs. The presence of various constraints and interferences between constraints makes COPs more difficult to tackle than unconstrained optimization problems. In this section we present the available constrained ABC algorithms in the literature.

ABC algorithm for the first time was adapted by Karaboga and Bastruck [22] to solve COPs. To cope with constraints, Deb's mechanism [14] is employed to be used instead of the greedy selection process due to its simplicity, computational cost and fine tuning requirement over other constraint handling methods. Because initialization with feasible solutions is very time consuming and in some situation impossible to generate a feasible solution randomly, the constrained ABC algorithm does not consider the initial population to be feasible. As an alternative Deb's rules are employed to direct the solutions to feasible region of search space. In addition, scout bee phase of the algorithm provides a diversity mechanism that allows new and probably infeasible individuals to be in the population. In this algorithm, artificial scouts are produced at a Scout Predetermined Period (SPP) of cycles for generating new solution randomly. The numerical performance of proposed ABC algorithm is evaluated and compared with the constrained PSO and DE algorithms and results show that ABC algorithm can be effectively applied for solving COPs.

Mezura-Montes *et al*. [29] presented Smart Flight-ABC (SF-ABC) algorithm to improve the performance of constrained ABC. In this algorithm to direct search towards the best-so-far solution, smart flight operator is applied in scout bee phase instead of uniform random search in ABC. Based on this method, if the best solution is infeasible, the trial solution has the chance to be located near the boundaries of the feasible region of search space. However, if the best solution is infeasible, the smart flight will generate a solution in promising region of search space. In addition to aforementioned improvement on ABC, the combination of two dynamic tolerances are also applied in SF-ABC as constrained handling mechanism, to transform the original CNOP into unconstrained optimization. The numerical results demonstrate the competitive performance of SF-ABC with original ABC.

Babaeizadeh and Rohanin [6] applied chaotic search mechanism to initialize population for constrained ABC where numerical results indicate that the proposed method is competitive with the ABC [22].

Another modification on ABC algorithm was introduced by Karaboga and Akay [26]. What makes this algorithm different from the original ABC [22] is related with the probability selection mechanism and parameter setting process. In this algorithm a new probability selection mechanism is presented to enhance diversity by allowing infeasible solutions in the population where infeasible solutions are introduced inversely proportional to their constraint violations and feasible solution defined based on their fitness values. In addition, in this algorithm appropriate value for each parameter is obtained. To recognize this algorithm throughout this paper the abbreviation Modified-ABC (M-ABC) is used to refer to this algorithm.

A modified constrained ABC algorithm (mcABC) was proposed in which chaotic mechanism as well as opposition based method was applied for population initialization to enhance the global convergence of algorithm. The numerical results have shown the effectiveness of the proposed method [6].

In mcABC algorithm, three new solution search equations are introduced respectively to employed bee, onlooker bee and scout bee phases. In addition, both chaotic search method and opposition-based learning mechanism are applied to initialize population in order to enhance the global convergence [7].

Multiple Onlooker bees-ABC (MO-ABC) was developed in [33] to improve constrained ABC [22]. The numerical performance demonstrates comparative results with original ABC.

M-ABC introduced four modifications related with the selection mechanism, the equality and boundary constraints, and scout bee operators to improve the behaviour of ABC in constrained search space. The numerical results show that M-ABC provides comparable results with respect to the algorithms under comparison [30].

A Genetically Inspired ABC algorithm (GI-ABC) was presented for COP. In this algorithm uniform crossover and mutation operators from GA are applied to scout bee phase to improve the performance of ABC algorithm [10].

An efficient constrained ABC (eABC) algorithm was suggested in [5] where two new solution search equations was introduced to be used for employed bee and onlooker bee phases to enhance the exploitation of algorithm.

Stanarevic *et al*. [34] introduced a M-ABC algorithm in a form of Smart Bee-ABC (SB-ABC) to solve constrained problems which applies its historical memories for the solution. The numerical experiments show efficiency of the method.

An improved constrained ABC (iABC) algorithm was suggested to address COPs. The modifications included a novel chaotic approach to generate initial population and two new search equations to enhance exploitation ability of the algorithm. In addition, a new fitness mechanism, along with an improved probability selection scheme was devised to exploit both feasible and informative infeasible solutions [9].

ABC-BA is a hybrid algorithm presented by Tsai [35] that integrates ABC and Bee Algorithm (BA). In this algorithm individuals can perform as an ABC individual in ABC sub-swarm or a BA individual in the BA sub-swarm. In addition, the population size of the ABC and BA sub-swarms change stochastically based on current best fitness values achieved by the sub-swarms. Experimental results demonstrate that ABC-BA outperforms ABC and BA algorithm.

Constrained ABC algorithm was also applied to solve many real-world engineering problems in recent years. Brajevic *et al.* [4] proposed a Constrained ABC (SC-ABC). This method was tested on several engineering benchmark problems which contain discrete and continuous variables. The numerical results were then compared with results obtained from Simple Constrained PSO algorithm (SiC-PSO) which show very good performance.

Akay and Karaboga [1] used ABC to solve large scale optimization problems as well as engineering design problems. The numerical results show that the performance of ABC algorithm is comparable to those of state of the art algorithms under consideration.

Upgraded ABC (UABC) algorithm for COPs was presented by Brajevic *et al.* [13] to improve modification rate parameter and applying modified scout bee phase of the ABC algorithm. This algorithm was tested on several engineering benchmark problems and the performance was compared with the performance of the Akay and Karaboga algorithm [1]. The numerical results show that the proposed algorithm produces better results.

For latest survey on constrained ABC please refer to [8].

4. Enhanced Constrained ABC

According to the literature in most of the constrained ABC algorithms the role of population initialization is ignored. However, in order to have a powerful algorithm the initial solutions must be diversified on almost all over the search space. This scheme helps to generate at least some points in the neighbourhood of global solution. In this paper we employed both chaotic mechanism and opposition-based learning method into population initialization to enhance diversity.

Among available chaotic method, logistic is selected to be used in initialization step which can be formulated as

$$c_{k+1}=4(1-c_k) \qquad (6)$$

Where c_k is the k^{th} chaotic number, $c\in(0, 1)$ and c_k cannot get numbers from set {0.0, 0.25, 0.75, 0.5, 1.0}. The initialization process based on chaotic search mechanism and opposition learning method is coded in Algorithm 2.

Algorithm 2: Initialization approach.

```
Consider the maximum number of chaotic iteration K=300,
the population size SN and the counter i=1, j=1
  for i=1to SN/2
    for j=1 to d
      Randomly initialize variables c_{0, j}∈(0, 1) and set
         iteration counter k=0
      for k=1 to K
        c_{k+1, j}=α(1-c_{kj})
      end
        x_{i, j}= x_{min, j}+c_{j,k}(x_{max, j}- x_{min, j})
    end
  end
  Set the individual counter i=1 and j=1
  for i= SN/2 to SN
    for j=1 to d
      op_{i, j}= x_{min, j}+ x_{max, j}- x_{min, j}
    end
end
```

After initialization the main loop consists of employed bees, onlooker bees and scout bees is subjected to repeat until the stopping criterion is met.

In this algorithm the new search equation is proposed for employed bee phase using Equation 7 to improve the exploitation behaviour of ABC.

$$v_{ij}=\begin{cases} x_{ij}+\gamma_{ij}(x_{bj}-x_{r1j}) \\ \quad +\mu_{ij}(x_{r1j}-x_{r2j}) & R_j<MR \\ \qquad x_{ij} & otherwise \end{cases} \qquad (7)$$

Where r_1 and r_2 are two different random integer indices selected from {1, 2, …, SN}. γ_{ij} is a random number between [-1,1] and μ_{ij} is uniform random number between [0,1]. R_{ij} is uniformly distributed random number and MR is control parameter in range [0, 1]. In addition, x_{ij} is the j^{th} dimension of best solution found so far. In Equation 6 the second and third terms enhance exploration capability.

After producing a new solution, EC-ABC algorithm makes a selection using Deb's mechanism [14] instead of using greedy selection in unconstrained ABC. Applying Deb's rules, the bee either memorizes the new solution by forgetting the current solution or keeps the current solution.

Deb's method uses a tournament selection mechanism where two solutions are compared at a time by applying following rules.

- Any feasible solution is preferred to any infeasible solution,
- Among two feasible solutions, the one having better objective function value is preferred,
- Among two infeasible solutions, the one having smaller constraint violation is preferred.

After completion of the search by all employed bees, they share the information of the solutions with the onlooker bees. In this probability selection mechanism [19] infeasible solutions are also allowed to participate in the colony. The probability values of feasible solutions are between 0.5 and 1 and for infeasible solution between 0 and 0.5.

The probability method is defined as Equation 8.

$$p_i=\begin{cases}0.5+\left(\frac{fit\,(x_i)}{\sum_{j=1}^{SN} fit\,(x_i)}\right)\times 0.5 & if J\,(x_i)=0\\ \left(1-\frac{J\,(x_i)}{\sum_{j=1}^{SN} J\,(x_i)}\right)\times 0.5 & if J\,(x_i)^1 0\end{cases} \quad (8)$$

Where $fit(x_i)$ is fitness value of solution x_i and $J(x_i)$ is the constraint violation of solution x_i.

Based on the probability selection mechanism, solutions are selected proportional to their fitness values if solutions are feasible and inversely proportional to their constraint violation values if solutions are infeasible.

After receiving fitness values information from employed bees, onlooker bee selects a solution based on their probability values. Then, onlooker bees produce modification on the position of the selected solution using Equation 9.

$$v_{ij}=\begin{cases}x_{ij}+\varphi_{ij}(x_{bj}-x_{r1j})+\Phi_{ij}(x_{bj}-x_{r2j}) & R_j<MR\\ x_{ij} & otherwise\end{cases} \quad (9)$$

Where r_1 and r_2 are two different random integer indices selected from{1, 2, …, SN}. φ_{ij} and Φ_{ij} are uniformly distributed random real number in the range [-1, 1].

As in the case of employed bees Deb's rules are employed to compare current solution with new solution. If the new solution produces better result it remains in population and the old solution is removed. The employed bee phase is coded in Algorithm 3.

In Equation 9, the first, term improves the exploration ability and the second and third terms, enhance the exploitation capability.

Algorithm 3: Employed bee phase of EC-ABC algorithm.

for i=1:SN
 for j=1:d
 Produce the new solution v_i for employed bee using Equation 7
 end for
 if no parameter is changed, change one random parameter of the solution using Equation 7
 Evaluate the quality of v_i
 Apply Deb's mechanism to select between v_i and x_i
 if solution v_i does not improve
 $trial_i$= $trial_i$+1, otherwise, $trial_i$=0
 end if

After distribution of all onlooker bees, if a solution can not improve further through predetermined number of cycles (limit) it is abandoned and replaced with a new solution discovered by scout bees. The onlooker bee phase is coded in Algorithm 4. In EC-ABC algorithm a smart flight scout bee is proposed to enhance the exploitation ability of algorithm.

Scout bee phase is defined as the following Equation 10.

$$v_{ij}=x_{ij}+k_{ij}(x_{kj}-x_{ij})-(1-k_{ij})(x_{bj}-x_{ij}) \quad (10)$$

Where k_{ij} is uniformly real number in [-1, 1] and x_{bj} is the j^{th} dimension of the best solution found so far.

5. Numerical Experiments and Comparisons

To evaluate and compare the performance of the proposed algorithms, 24 constrained benchmark functions from CEC 2006 [27] are applied. EC-ABC and other constrained ABC algorithms under comparisons are coded in MALAB environment. The value of each parameters used are given in Table 1.

Table 1. Parameters Setting.

Parameters	Symbols	Value
Solutions Number	SN	20
Maximum Cycle Number	MCN	6000
Modification Rate	MR	0.8
Population Size	PS	40
Limit	Limit	150
Scout Production Period	SPP	150
Epsilon	ε	0.001

Algorithm 4: Onlooker bee phase for EC-ABC algorithm.

t=0 ,i=1
repeat
if random<P_i then
 t=t+1
 for j=1:d
 Produce a new solution v_i for the onlooker bee of the solution x_i using Equation 9
 end for
 Apply the selection process between v_i and based on Deb's method
 If solution x_i does not improve
 $trial_i$= $trial_i$+1,
 otherwise, $trial_i$=0
end if
i=i+1
i=i mod(SN+1)
until t=SN

The numerical performance of proposed EC-ABC algorithm was compared against constrained ABC [23], MABC [26], M-ABC [30], SF-ABC [29] and MO-ABC [32] algorithms. Each algorithm are tested for 24 test function and after 30 independent runs of each algorithm the average solution is considered which as shown in Tables 2 and 3. The Problems g20, g21, g22 are not considered because no feasible solutions can be found for these problems by the algorithms.

Table 2. Function values obtained by ABC, MABC, M-ABC, SF-ABC, MO-ABC and EC- ABC.

Problem		ABC	MABC	M-ABC	SF-ABC	MO-ABC	EC-ABC
g01	Best	-15.00000	-1500000	-15.00000	-15.00000	-15.00000	-15.00000
	Mean	-15.00000	-1.500000	-15.00000	-14.16321	-15.00000	-15.00000
	Worst	-15.00000	-1.500000	-15.00000	-12.52510	-15.00000	-15.00000
	Std.dev	0.000000	0.000000	0.000000	0.923125	0.000000	0.000000
g02	Best	0.803567	0.803538	0.803614	-0.708944	-0.803610	-0.803618
	Mean	-0.791744	-0.792927	-0.799450	-0.471249	-0.793510	-0.802729
	Worst	-0.752924	-0.750302	-0.778176	-0.319535	-0.744582	-0.794662
	Std.dev	0.013292	0.011052	-0.006440	0.010832	0.016124	0.002675
g03	Best	-1.004657	-1.004817	-1.000000	-1.000000	-1.000000	-1.005001
	Mean	-1.000096	-1.001941	-1.000000	-1.000000	-1.000000	-1.004975
	Worst	-0.979651	-0.989160	-1.000000	-1.000000	-1.000000	-1.004923
	Std.dev	0.005979	0.000375	0.0000	0.0000	0.0000	0.000027
g04	Best	-30665.542	-30665.42	-30665.539	-30665.539	-30665.539	-30665.54
	Mean	-30665.542	-30665.42	-30665.539	-30665.539	-30665.539	-30665.54
	Worst	-30665.542	-30665.42	-30665.539	-30665.539	-30665.539	-30665.54
	Std.dev	0.0000000	0.0000000	0.00000	0.00000	0.00000	0.000000
g05	Best	5126.489	5127.099	5126.734	5126.506	5126.657	5126.527
	Mean	5177.239	5236.991	5178.178	5126.527	5162.506	5249.384
	Worst	5307.988	5802.318	5317.183	5126.859	5229.119	5824.530
	Std.dev	57.86021	156.0343	56.000	0.0793	47.8203	202.4735
g06	Best	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814
	Mean	-6961.814	-6961.814	-6961.814	-6961.813	-6961.813	-6961.814
	Worst	-6961.814	-6961.814	-6961.814	-6961.805	-6961.804	-6961.814
	Std.dev	0.0000000	0.0000000	0.000000	0.0002	0.0001	0.000000
g07	Best	24.46138	24.47032	24.312235	24.16453	24.32325	24.31428
	Mean	24.70718	24.68698	24.416402	24.65821	24.45653	24.38785
	Worst	25.16577	25.36005	24.794032	25.55140	24.92938	24.70564
	Std.dev	0.181394	0.178641	0.12723	0.326021	0.135023	0.08238
g08	Best	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.09582504
	Mean	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.09582504
	Worst	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.09582504
	Std.dev	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000000
g09	Best	680.6381	680.6371	680.6331	680.6325	680.6312	680.6318
	Mean	680.6506	680.6515	680.6474	680.6450	680.6350	680.6487
	Worst	680.6757	680.6760	680.6768	680.8584	680.6363	680.7362
	Std.dev	0.0080749	0.009610	0.054310	0.041251	0.004215	0.021534
g10	Best	7160.63125	7220.5540	7051.7752	7049.5166	7053.3204	7117.8753
	Mean	7364.94034	7347.8433	7233.8101	7116.8236	7167.8015	7447.8854
	Worst	7691.30330	7924.1286	7604.1290	7362.7406	7418.3340	8034.5068
	Std.dev	129.8405	134.14103	101.325	82.12450	83.00825	236.67822
g11	Best	0.7490003	0.7490001	0.7500000	0.7500000	0.7500000	0.7490000
	Mean	0.7490022	0.7490032	0.7500000	0.7500000	0.7500000	0.7499815
	Worst	0.7490101	0.7490140	0.7500000	0.7500000	0.7500000	0.7529169
	Std.dev	0.0000020	0.000003	0.000000	0.000000	0.000000	0.0011032
g12	Best	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
	Mean	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
	Worst	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
	Std.dev	0.0000000	0.000000	0.000000	0.000000	0.000000	0.0000000
g13	Best	0.5551238	0.4895965	0.0538901	0.0539860	0.4542041	0.1846375
	Mean	0.9497812	0.9576896	0.1577912	0.2638542	0.4560438	0.7331250
	Worst	1.4929540	1.4375342	0.4419785	1.000000	0.4891204	1.0000000
	Std.dev	0.1469151	0.1613582	0.0172430	0.2162045	0.0215840	0.2321268

Table 3. Function values obtained by ABC, MABC, M-ABC, SF-ABC, MO-ABC and EC-ABC.

Problem		ABC	MABC	M -ABC	SF-ABC	MO-ABC	EC-ABC
g14	Best	-45.11878	-45.32082	-47.64541	-46.66514	-46.450835	-46.06795
	Mean	-42.68215	-42.65421	-47.27156	-46.46824	-45.998013	-43.94812
	Worst	-40.60165	-40.05962	-46.53698	-43.87123	-45.316798	-41.59548
	Std.dev	1.171236	1.195831	0.245762	0.520124	0.257106	0.9756126
g15	Best	941.21911	951.43752	961.71521	961.71511	961.71512	954.23680
	Mean	958.84762	960.89221	961.71879	961.71553	961.88313	966.58805
	Worst	972.95780	970.68460	961.79125	961.72013	964.33983	978.00416
	Std.dev	7.512742	4.878944	0.014319	000.159	0.54267	7.6150353
g16	Best	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155
	Mean	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155	-1905155
	Worst	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155
	Std.dev	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
g17	Best	8886.685	8879.576	8866.5986	8927.598	8939.125	8860.562
	Mean	9053.597	9053.567	8987.4589	8928.865	8946.172	8982.975
	Worst	9249.174	9215.365	9165.2543	8938.617	8956.235	9249.269
	Std.dev	123.0898	122.6397	95.6532	3.12132	9.528253	109.1514
g18	Best	-0.8405680	-0.8593651	-0.866023	-0.866025	-0.865976	-0.8660236
	Mean	-0.6895726	-0.7107018	-0.795019	-0.740748	-0.767198	-0.8265948
	Worst	-0.6616021	-0.6613345	-0.672223	-0.501205	-0.670714	-0.6713430
	Std.dev	0.05082904	0.06776626	0.093789	0.1453562	0.0960035	0.07813725
g19	Best	36.774012	37.580864	33.254703	32.662712	33.7698315	32.9962520
	Mean	39.297845	39.834920	34.265623	33.107137	35.3147859	33.6537328
	Worst	42.701610	42.427351	35.736841	34.914012	37.3645831	35.5405499
	Std.dev	1.4571242	1.1743492	0.631240	0.51325	0.687514	0.5274753
g23	Best			-159.7542	-350.12614		-1071.627
	Mean	-	-	-35.28473	-121.37464	-	-327.1549
	Worst			109.1275	276.00379		149.2063
	Std.dev			82.7698	157.895		325.5414
g24	Best	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013
	Mean	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013
	Worst	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013
	Std.dev	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

The simulation results demonstrate that all algorithms under comparison obtained the same results for problems g06, g12, g16 and g24. The EC-ABC is superior to other algorithms in problems g02, g03, g04, g08, g11, g17, g18 and g23. The SF-ABC algorithm in problems g05, g10, g13, g15, g19 has good performance compare with other algorithms. However, MO-ABC is outperformed in problems g09, g14.

The numerical performance showed that EC-ABC provided comparable result with respect to other state of the art algorithms in comparison to solving COPs.

In order to, compare the convergence ability of EC-ABC with the other state of the art algorithms Figures 1, 2, 3, and 4 are presented, which clearly show that EC-ABC is able to converge faster than other algorithms. It confirms that the new search equations can accelerate the constrained ABC convergence.

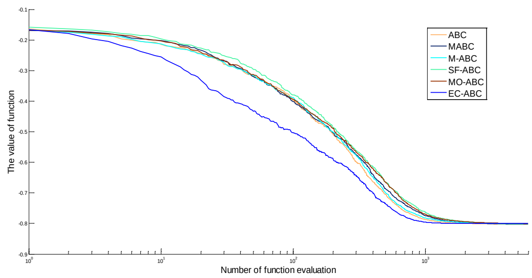

Figure 1. Iterations to convergence for problem g02.

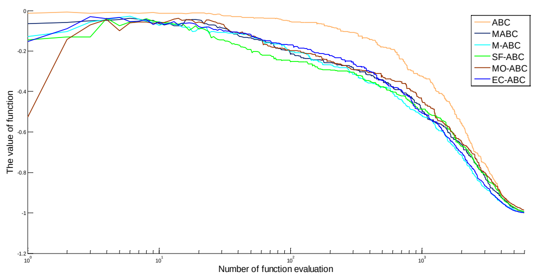

Figure 2. Iterations to convergence for problem g03.

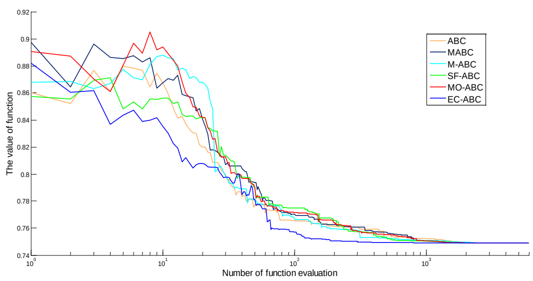

Figure 3. Iterations to convergence for problem g11.

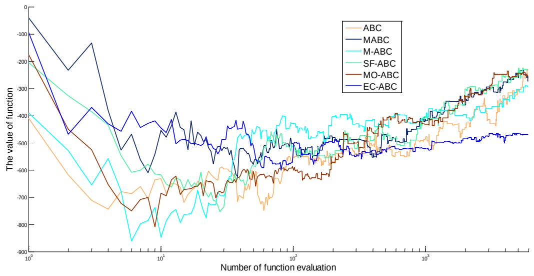

Figure 4. Iterations to convergence for problem g23.

6. Discussion

In this paper, we have introduced an enhanced constrained ABC called EC-ABC algorithm to solve COPs in which the initial population is generated using chaotic search method along with opposition-based learning method. In addition, two new search equations are proposed for employed bee and onlooker bee phases to enhance the global convergence of ABC algorithm. Smart flight method is also applied into scout bee phases to improve the exploitation behavior of algorithm. The experimental results were tested on 24 benchmark functions and show that EC-ABC is competitive with state of the art constrained ABC under comparison.

Acknowledgment

The authors would like to thank Universiti Teknologi Malaysia and Ministry of Education, Malaysia for the financial funding through grant RUG 08H47.

References

[1] Akay B. and Karaboga D., "Artificial Bee Colony Algorithm for Large-Scale Problems and Engineering Design Optimization," *Journal of Intelligent Manufacturing*, vol. 23, no. 4, pp. 1001-1014, 2012.

[2] Aydina D., Özyön S., Yaşar C., and Liao T., "Artificial Bee Colony Algorithm with Dynamic Population Size to Combined Economic and Emission Dispatch Problem," *Electrical Power and Energy Systems*, vol. 45, pp. 144-153, 2014.

[3] Brajevic I., Tuba M., and Subotic M., "Performance of the Improved Artificial Bee Colony Algorithm on Standard Engineering Constrained Problems," *International Journal of Mathematics and Computers in Simulation*, vol. 5, no. 2, pp. 135-143, 2011.

[4] Brajevic I., Tuba M., and Subotic M., "Improved Artificial Bee Colony Algorithm for Constrained Problems," *in Proceeding of the 11th World Scientific and Engineering Academy and Society International Conference On Nural Networks and 11th WSEAS International Conference On Evolutionary Computing and 11th WSEAS International Conference On Fuzzy Systems*, Iasi, pp. 185-190, 2010.

[5] Babaeizadeh S. and Ahmad R., "An Efficient Artificial Bee Colony Algorithm for Constrained Optimization Problems," *Journal of Engineering and Applied Sciences*, vol.9, no. 10-12, pp. 405-413, 2014.

[6] Babaeizadeh S. and Ahmad R., "A Modified Artificial Bee Colony Algorithm for Constrained Optimization Problems," *Journal of Convergence Information Technology*, vol. 9, no. 6, pp. 151-163, 2014.

[7] Babaeizadeh S. and Ahmad R., "Modified Artificial Bee Colony Algorithm with Chaotic Search Method for Constrained Optimization Problems," *Journal of Convergence Information Technology*, vol. 9, no. 6, pp. 151-163, 2014.

[8] Babaeizadeh S. and Ahmad R., "Performance Comparison of Constrained Artificial Bee Colony Algorithm," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 10, no. 5, pp. 537-546, 2015.

[9] Babaeizadeh S. and Ahmad R., "An Improved Artificial Bee Colony Algorithm for Constrained Optimization," *Research Journal of Applied Sciences*, *Engineering and Technology*, vol. 11, no. 1, pp. 14-22 2016.

[10] Bacanin N. and Tuba M., "Artificial Bee Colony (ABC) Algorithm for Constrained Optimization Improved with Genetic Operators," *Studies in Informatics and Control*, vol. 21, no. 2, pp. 137-146, 2012.

[11] Banitalebi A., Aziz M., Bahar A., and Aziz Z., "Enhanced Compact Rtificial Bee Colony," *Information Sciences*, vol. 298, pp. 491-511, 2015.

[12] Banitalebi A., Aziz M., and Aziz Z., "A Self-Adaptive Binary Differential Evolution Algorithm for Large Scale Binary Optimization Problems," *Information Sciences*, vol. 367-368, pp. 487-511, 2016.

[13] Brajevic I. and Tuba M., "An Upgraded Artificial Bee Colony (ABC) Algorithm for Constrained Optimization Problems," *Journal of Intelligent Manufacturing*, vol. 24, no. 4, pp. 729-740, 2013.

[14] Deb K., "An Efficient Constraint Handling Method for Genetic Algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2-4, pp. 311-338, 2000.

[15] Dorigo M. and Blum C., "Ant Colony Optimization Theory: A Survey," *Theoretical Computer Science*, vol. 344, no. 2, pp. 243-278, 2005.

[16] Gao W., Liu S., and Huang L., "Enhancing Artificial Bee Colony Algorithm Using More Information-Based Search Equations," *Information Sciences*, vol. 270, pp. 112-133, 2014.

[17] Holland J., *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, U Michigan Press, 1975.

[18] Javidi M. and Hosseinpourfard R., "Chaos Genetic Algorithm Instead Genetic Algorithm," *The International Arab Journal of Information Technology*, vol. 12, no. 2, pp. 163-186, 2015.

[19] Kennedy J., *Encyclopedia of Machine Learning*, Springer US, 2010.

[20] Koziel S. and Michalewicz Z., "Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization," *Evolutionary computation*, vol. 7, no. 1, pp. 19-44, 1999.

[21] Karaboga D., "An Idea Based on Honey Bee Swarm for Numerical Optimization," Technical Report-tr06 Erciyes university, 2005.

[22] Karaboga D. and Basturk B., "A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm," *Journal of Global Optimization* vol. 39, no. 3, pp. 459-471, 2007.

[23] Karaboga D. and Basturk B., *Foundations of Fuzzy Logic and Soft Computing*, Springer, 2007.

[24] Karaboga D. and Basturk B., "On The Performance of Artificial Bee Colony (ABC) Algorithm," *Applied soft computing*, vol. 8, no. 1, pp. 687-697, 2008.

[25] Karaboga D. and Akay B., "A Comparative Study of Artificial Bee Colony Algorithm," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108-132, 2009.

[26] Karaboga D. and Akay B., "A Modified Artificial Bee Colony (ABC) Algorithm for Constrained Optimization Problems," *Applied Soft Computing*, vol. 11, no. 3, pp. 3021-3031, 2011.

[27] Liang J., Runarsson T., Mezura-Montes E., Clerc M., Suganthan P., Coello Coello C., and Deb K., "Problem Definitions and Evaluation Criteria for the CEC 2006 Special Session on Constrained Real-Parameter Optimization," Technical Report Journal of Applied Mechanics, 2006.

[28] Li G., Peifeng N., and Xiao X., "Development and Investigation of Efficient Artificial Bee Colony Algorithm for Numerical Function Optimization," *Applied soft computing*, vol. 12, no. 1, pp. 320-332, 2012.

[29] Mezura-Montes E., Damián-Araoz M., and Cetina-Domingez O., "Smart Flight and Dynamic Tolerances in the Artificial Bee Colony for Constrained Optimization," *in Proceeding of IEEE Congress Evolutionary Computation*, Barcelona, pp. 1-8, 2010.

[30] Mezura-Montes E. and Cetina-Domínguez O., "Empirical Analysis of a Modified Artificial Bee Colony for Constrained Numerical Optimization," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 10943-10973, 2012.

[31] Mustaffa Z. and Yusof Y., "LSSVM Parameters Tuning with Enhanced Artificial Bee Colony," *The International Arab Journal of Information Technology*, vol. 11, no. 3, pp. 236-242, 2014.

[32] Subotic M., "Artificial Bee Colony Algorithm with Multiple Onlookers for Constrained Optimization Problems," *in Proceeding of the European Computing Conference*, Paris, pp. 251-256, 2011.

[33] Storn R. and Price K., "Differential Evolution-a Simple and Efficient Heuristic for global Optimization Over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341-359, 1997.

[34] Stanarevic N., Tuba M., and Bacanin N., "Modified Artificial Bee Colony Algorithm for Constrained Problems Optimization," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 5, no. 3, pp. 644-651, 2011.

[35] Tsai H., "Integrating the Artificial Bee Colony and Bees Algorithm to Face Constrained Optimization Problems," *Information Sciences*, vol. 258, pp. 80-93, 2014.

[36] Wolpert D. and Macready W., "No Free Lunch Theorems for Optimization," *IEEE Transactions On Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, 1997.

Soudeh Babaeizadeh received her master's degree from department of mathematical sciences, Universiti Teknologi Malaysia (UTM). She is currently a PhD candidate at the same university. Her current research interest include global optimization, derivative-free optimization and evolutionary algorithms.

Rohanin Ahmad received her master's degree from the Indiana State University in 1984, and her Ph.D. degree in the field of optimal control in 2005 from Universiti Teknologi Malaysia (UTM). She is currently Associate Professor at UTM where she also serves as head of Department of Mathematical Science at the same university. Her research interest include optimization, optimal control, and operation research.