Fitness based Position Update in Artificial Bee Colony Algorithm

Ashutosh Kumar ,Sandeep Kumar,Kiran dhayal
Faculty of Engineering and Technology,
Jagannath University
Jaipur, India

Dr. Kumar Swetank
Jai Prakash University,Chapra
Saran, Bihar, India

***Abstract*—Artificial Bee Colony (ABC) optimization algorithm is swarm intelligence based nature inspired algorithm which has been proved a competitive algorithm with some popular nature-inspired algorithms. However, it is found that the ABC algorithm prefers exploration at the cost of the exploitation. ABC algorithm sometimes shows early convergence and stagnation. Therefore, in this paper a self adaptive fitness based position update strategy is presented in which the perturbation in the solution is based on fitness of the solution. The proposed strategy is self-adaptive in nature and therefore no manual parameter setting is required. The proposed strategy named as fitness based position update in ABC (FPABC) algorithm. FPABC applied on 16 well-known benchmark functions and proves its superiority over other variants of ABC algorithm.**

Keywords—Swarm intelligence, Self adaptive mutation, Engineering optimization problems, Artificial Bee Colony, Nature Inspired Algorithms.

I. INTRODUCTION

Artificial bee colony (ABC) optimization algorithm introduced by D.Karaboga [5] is a recent popular swarm intelligence based algorithm. This algorithm is inspired by the behavior of honey bees when seeking a good quality food source. It falls into category of Nature Inspired Algorithms (NIA) that is inspired by some natural phenomenon or extra ordinary behavior of intelligent insects. NIAs includes stochastic algorithms, Evolutionary algorithms, Physical algorithms, Probabilistic algorithms, Swarm algorithms, Immune algorithms and Neural algorithm based on their source of inspiration. Similar to other population based optimization algorithm, ABC consists of a population of possible solutions. The possible solutions are food sources of honey bees. The fitness is determined in terms of the quality (nectar amount) of the food source. ABC is relatively a simple, fast and population based stochastic search strategy in the area of nature inspired algorithms. There are two fundamental contradictory processes which drive the swarm to update in ABC: the adaptation process, which empowers exploring different fields of the search space, and the selection process, which ensures the exploitation of the previous experience. However, it has been shown that the ABC may occasionally stop proceeding toward the global optimum even though the population has not converged to a local optimum [6]. It can be observed that the solution search equation of ABC algorithm is good at exploration but poor at exploitation [12]. Therefore, to maintain the proper balance between exploration and exploitation behavior of ABC, it is highly desirable to develop a strategy in which better solutions exploit the search space in close proximity while less fit solutions explore the search space. Therefore, this paper proposed a self adaptive step size strategy to update a solution. In the proposed strategy, a solution takes small step sizes in position updating process if its fitness is high i.e. it searches the solution in its vicinity whereas a solution takes large step sizes if its fitness is low, hence explore the search space. The proposed strategy is used for finding the global optima of a uni-modal and/or multimodel functions by adaptively modifying the step sizes in updating process of the candidate solution in the search space within which the optima is known to exist. In the proposed strategy, ABC algorithm's parameter 'limit' is modified according to the fitness of the solution i.e. self adaptively. Now, there is separate 'limit' for every solution according to their fitness. The value of 'limit' is high for highly fitted solutions, while for less fit solutions, it is low. Hence, a better solution has more chances to update itself in comparison to the less fit solutions. Further, to improve the diversity of the algorithm, number of scout bees is increased. The proposed strategy is compared with original ABC and Modified Artificial Bee Colony (MABC) [1].

Rest of the paper is organized as follows: Basic ABC is explained in section 2. In section 3, fitness based position update in ABC is proposed and explained. In Section 4, performance of the proposed strategy is analyzed. Finally, in section 5, paper is concluded.

II. ARTIFICIAL BEE COLONY (ABC) ALGORITHM

In ABC algorithm, honey bees are classified into three classes namely employed bees, onlooker bees and scout bees. The number of employed bees is equal to the onlooker bees. The employed bees are the bees which search the food source and gather the information about the quality of the food source. Onlooker bees stay in the hive and search the food sources on the basis of the information gathered by the employed bees. The scout bee, searches new food sources randomly in places of the rejected foods sources. Similar to the other population-based algorithms, ABC solution search process is an iterative process. After, initialization of the ABC parameters and swarm, it requires the repetitive iterations of the three phases namely employed bee phase, onlooker bee phase and scout bee phase [5]. Each of the steps is described here as follows:

A. Initialization of the swarm

The parameters for the ABC are the number of food sources, the number of trials after which a food source is assumed to be deserted and the termination criteria. In the basic ABC, the number of food sources is equal to the employed bees or onlooker bees. Initially, a uniformly distributed initial

swarm of SN food sources where each food source x_i(i = 1, 2, ..., SN) is a D-dimensional vector, generated. Here D is the number of variables in the optimization problem and x_i represent the i^{th} food source in the swarm. Each food source is generated as follows:

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

Here x_{minj} and x_{maxj} are bounds of x_i in j^{th} direction and rand[0, 1] is a uniformly distributed random number in the range [0, 1].

B. Employed bee phase

In the employed bee phase, modification in the current solution (food source) is done by employed bees according to the information of individual experience and the new solution fitness value. If the fitness value of the new solution is greater than that of the old solution, then the bee updates her position to the new solution and old one is discarded. The position update equation for ith candidate in this phase is

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

here k ∈ {1, 2, ..., SN} and j ∈ {1, 2, ...,D} are randomly chosen indices. k must be different from i. ϕ_{ij} is a random number between [-1, 1].

C. Onlooker bees phase

In this phase, the new fitness information (nectar) of the new solutions (food sources) and their position information are shared by all the employed bees with the onlooker bees in the hive. Onlooker bees analyze the available information and select a solution with a probability $prob_i$ related to its fitness, which can be calculated using following expression (there may be some other but must be a function of fitness):

$$prob_i(G) = \frac{0.9 \times fitness_i}{\max fit} + 0.1 \quad (3)$$

Here fit_i is the fitness value of the i^{th} solution and maxfit is the maximum fitness of the solutions.

As in the case of employed bee, it produces a modification on the position in its memory and checks the fitness of the new solution. If the fitness is higher than the previous one, the bee remembers the new location and forgets the old one.

D. Scout bees phase

A food source is considered to be abandoned, if its position is not getting updated during a predetermined number of cycles. In this phase, the bee whose food source has been abandoned becomes scout bee and the abandoned food source is replaced by a randomly chosen food source within the search space. In ABC, pre-decided number of cycles is a pivotal control parameter which is called limit for abandonment. Assume that the abandoned source is x_i. The scout bee replaces this food source by a randomly chosen food source which is generated as follows:

$$x_{ij} = x_{\min j} + rand[0,1](x_{\max j} - x_{\min j}), \quad (4)$$

Where j taken from {1,2,….D}, x_{minj} and x_{maxj} are bounds of x_i in j^{th} direction.

Main steps of the ABC algorithm

Based on the above details, it is clear that the ABC search process contains three important control parameters: The number of food sources SN (equal to number of onlooker or employed bees), the maximum number of iterations and the value of limit. The pseudo-code of the ABC is outlined in Algorithm 1 [6].

Algorithm 1 Artificial Bee Colony Algorithm:

Initialize the parameters;

While Termination criteria is not satisfied do

Employed bee phase for generating new food sources;

Onlooker bees phase for updating the food sources depending on their nectar amounts;

Scout bee phase for discovering the new food sources in place of discarded food sources;

Keep the best food source in memory;

end while

Output the best solution found so far.

III. FITNESS BASED POSITION UPDATE IN ARTIFICIAL BEE COLONY (FPABC)

Exploration and exploitation are the two important characteristics of the population-based optimization algorithms such as GA [4], PSO [8], DE [10], BFO [9] and so on. In these optimization algorithms, the exploration represents the ability to discover the global optimum by investigating the various unknown regions in the solution search space. Some researchers tired to balances between these activities by applying different methods like HJABC [13], MeABC [14], RMABC [15], A novel hybrid crossover based ABC [16], Enhanced ABC [17], Balanced ABC [18], Dynamic Swarm ABC [19], Levy flight ABC [20], Modified ABC [21]. HJABC incorporate Hooke-Jeeves method in ABC. MeABC applied memetic search phase for better balance between diversification and intensification. RMABC introduce two new parameters in MeABC algorithm. Dynamic swarm ABC incorporated dynamic swarm mechanism (DSM) and assume that good solution has good neighbor and low fitness solution can explore search space in better way. Levy flight ABC tunes the levy flight parameters in order to balance diversification and intensification and also enhance convergence rate. Modified ABC suggested new mechanism for fitness calculation and probability calculation.

While the exploitation represents the ability to find better solutions by implementing the knowledge of the previously attained good solutions. In behavior, the exploration and exploitation contradict with each other, however both abilities should be well balanced to achieve better optimization performance. Dervis Karaboga and Bahriye Akay [6] tested

different variants of ABC for global optimization and found that the ABC shows poor performance and remains inefficient in exploring the search space. J. C. Bansal, H. Sharma and S. S. Jadon [22] outlined some intrinsic pitfalls with most of the population based stochastic algorithm is the early convergence or stagnation. ABC also shows these drawbacks. The location of solution updates using equ. 2 in ABC. After some iterations, usually all possible solutions work within a very small neighbourhood. Here the difference X_{ij}-X_{kj} becomes very small and so the improvement in the position becomes negligible. This phenomenon is known as the stagnation or premature convergence if the global optimal solution is not present in this small neighborhood. Any population based algorithm is regarded as an efficient algorithm if the convergence speed is high and able to explore the maximum area of the search space. In other words, if a population based algorithm is capable of balancing between diversification and intensification of the search space, then the algorithm is regarded an efficient algorithm.

In ABC, any potential solution updates itself using the information provided by a randomly selected potential solution within the current swarm. In this process, a step size which is a linear combination of a random number $\phi_{ij} \in [-1, 1]$; current solution and a randomly selected solution are used. Now the quality of the updated solution highly depends upon this step size. If the step size is too large, which may occur if the difference of current solution and randomly selected solution is large with high absolute value of ϕ_{ij}, then updated solution can surpass the true solution and if this step size is too small then the convergence rate of ABC may significantly decrease. A proper balance of this step size can balance the exploration and exploitation capability of the ABC simultaneously. But, since this step size consists of random component so the balance cannot be done manually. Therefore, to balance the exploration and exploitation, we modified the solution update strategy according to the fitness of the solution. In the basic ABC, the food sources are updated, as shown in equ. 2. In order to improve the exploitation, take advantage of the information of the global best solution to guide the search of candidate solutions, the solution search equation described by equ. 2 is modified as follows:

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) + (2.0 - prob_i) \times (x_{bestj} - x_{ij})$$

Algorithm 2 Solution update in Employed bee phase:

Input: solution x_i, $prob_i$ and j ∈ (1,D);

for j ∈ {1 to D} do

if U∈ (0, 1) > $prob_i$

then

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) + (2.0 - prob_i) \times (x_{bestj} - x_{ij})$$

else

$v_{ij} = x_{ij}$;

end if

end for

To enhance the exploitation capability of ABC, fitness based self adaptive mutation mechanism is introduced in the basic ABC and shown in Algorithm 2. In the proposed strategy, the perturbation in the solution is based on the fitness of the solution. It is clear from Algorithm 2 that the number of update in the dimensions of the i^{th} solution is depend on $prob_i$ and which is a function of fitness (refer equation 3).

The strategy is based on the concept that the perturbation will be high for low fit solutions as for that the value of $prob_i$ will be low while the perturbation in high fit solutions will be low due to high value of $prob_i$. It is assumed that the global optima should be near about to the better fit solutions and if perturbation of better solutions will be high then there may be chance of skipping true solutions due to large step size. Therefore, the step sizes which are proportionally related to the perturbations in the solutions are less for good solutions and are high for worst solutions which are responsible for the exploration. Therefore in the proposed strategy, the better solutions exploit the search space while low fit solutions explore the search area.

Here, $prob_i$ is a function of fitness and calculated as shown in equation (3). In Algorithm 2, it is clear that for a solution if value of $prob_i$ is high and that is the case of high fitness solution then for that solution the step size will be small. Therefore, it is obvious that there is more chance for the high fitness solution to move in its neighborhood compare to the low fitness solution and hence, a better solution could exploit the search area in its vicinity. In other words, we can say that solutions exploit or explore the search area based on probability which is function of fitness. Hence with help of modified step size it is able to maintain balance between diversification and intensification of search space and escape the situation of stagnation and early convergence. Experimental show that FPABC proves it superiority to solve considered problem in less efforts and with less number of function evaluations.

TABLE I. TEST PROBLEMS

Test Problem	Objective Function	Search Range	Optimum Value	D	Acceptable Error
Beale function	$f_1(x) = (1.5 - x_1(1-x_2))^2 + (2.25 - x_1(1-x_2^2))^2 + (2.625 - x_1(1-x_2^3))^2$	[-4.5, 4.5]	*f*(3. 0.5) = 0	2	1.0*E*-05
Colville function	$f_2(x) = 100(x_2 - x_1^2)^2 + (1-x_1)^2 + 90(x_4 - x_3^2)^2 + (1-x_3)^2 + 10.1[(x_2-1)^2 + (x_4-1)^2] + 19.8(x_2-1)(x_4-1)$	[-10, 10]	*f*(1) = 0	4	1.0*E*-05
Kowalik function	$f_3(x) = \sum_{i=1}^{11}(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4})^2$	[-5, 5]	*f*(0.1928, 0.1908, 0.1231, 0.1357) = 3.07*E*-04	4	1.0*E*-05
Shifted Rosenbrock	$f_4(x) = \sum_{i=1}^{D-1}(100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2 + f_{bias}, z = x - o + 1, x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	[-100, 100]	*f(o)=f*$_{bias}$*=390*	10	1.0*E*-01
Shifted Sphere	$f_5(x) = \sum_{i=1}^{D} z_i^2 + f_{bias}, z = x - o, x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	[-100, 100]	*f(o)=f*$_{bias}$*=-450*	10	1.0*E*-05
Shifted Rastrigin	$f_6(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f_{bias}, z = (x - o), x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	[-5, 5]	*f(o)=f*$_{bias}$*=-330*	10	1.0*E*-02
Shifted Schwefel	$f_7(x) = \sum_{i=1}^{D}(\sum_{j=1}^{i} z_j)^2 + f_{bias}, z = (x - o), x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	[-100, 100]	*f(o)=f*$_{bias}$*=-450*	10	1.0*E*-05
Shifted Griewank	$f_8(x) = \sum_{i=1}^{D}\frac{z_i^2}{4000} - \prod_{i=1}^{D}\cos(\frac{z_i}{\sqrt{i}}) + 1 + f_{bias}, z = (x - o), x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	[-600, 600]	*f(o)=f*$_{bias}$*=-180*	10	1.0*E*-05
Shifted Ackley	$f_9(x) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)) + 20 + e + f_{bias}, z = (x - o), x = (x_1, x_2, ...x_D), o = (o_1, o_2, ...o_D)$	[-32,32]	*f(o)=f*$_{bias}$*=-140*	10	1.0*E*-05
Goldstein-Price	$f_{10}(x) = (1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)) \times (30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2))$	[-2, 2]	*f(0, -1)=3*	2	1.0*E*-14
Easom's function	$f_{11}(x) = -\cos x_1 \cos x_2 e^{(-(x_1-\pi)^2 - (x_2-\pi)^2)}$	[-10, 10]	*f*(π, π) = -1	2	1.0*E*-13
Dekkkers and Aarts	$f_{12}(x) = 10^5 x_1^2 + x_2^2 - (x_1^2 + x_2^2)^2 + 10^{-5}(x_1^2 + x_2^2)^4$	[-20, 20]	*f(0,15)=f(0, -15)= -24777*	2	5.0*E*-01
McCormick	$f_{13}(x) = \sin(x_1 + x_2) + (x_1 - x_2)^2 - \frac{3}{2}x_1 + \frac{5}{2}x_2 + 1$	$-1.5 \le x_1 \le 4, -3 \le x_2 \le 3,$	*f(-0.547, -1.547) =-1.9133*	30	1.0*E*-04
Meyer and Roth Problem	$f_{14}(x) = \sum_{i=1}^{5}(\frac{x_1 x_3 t_i}{1 + x_1 t_i + x_2 v_i} - y_i)^2$	[-10, 10]	*f(3.13, 15.16,0.78)* = *0.4E*-04	3	1.0*E*-03
Shubert	$f_{15}(x) = -\sum_{i=1}^{5} i\cos((i+1)x_1 + 1)\sum_{i=1}^{5} i\cos((i+1)x_2 + 1)$	[-10, 10]	*f(7.0835, 4.8580) = -186.7309*	2	1.0*E-05*
Sinusoidal	$f_{16}(x) = -\left[A\prod_{i=1}^{5}\sin(x_i - z) + \prod_{i=1}^{n}\sin(B(x_i - z))\right]$ $A = 2.5, B = 5, z = 30$	[-10, 10]	*f(90+z)=-(A+1)*	10	1.00*E*-02

D-Dimension

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Test problems under consideration

In order to analyze the performance of FPABC, 16 unbiased optimization problems (solutions does not exists on axis, diagonal or origin) (f1 to f16) are selected (listed in Table I). Considered problems are of different characteristics (in terms of dimension, uni-model/multi-model, separable/nonseprable).

B. Experimental setting

To prove the efficiency of FPABC, it is compared with original ABC and recent variant of ABC named Modified ABC (MABC) [1]. To test FPABC, ABC, and MABC over considered problems, following experimental setting is adopted:

- Colony size NP = 50 [2, 3],
- ϕij= rand[−1, 1],
- Number of food sources SN = NP/2,
- limit = D × SN [7, 1],
- The termination criteria: maximum number of function evaluations (which is set to be 200000) is reached or the acceptable error (mentioned in Table I) has been achieved,
- The number of simulations per run =100,
- Parameter settings for the algorithms ABC and MABC are similar to their original research papers.

C. Results Comparison

Numerical results with experimental setting of above subsection are given in Table 2. In Table 2, standard deviation (SD), mean error (ME), average function evaluations (AFE), and success rate (SR) are reported. Table 2 shows that most of the time FPABC outperforms in terms of reliability, efficiency and accuracy as compare to the basic ABC, and MABC.

FPABC, ABC, and MABC are compared through SR, ME and AFE in Table 2. First SR is compared for all these algorithms and if it is not possible to distinguish the algorithms based on SR then comparison is made on the basis of AFE. ME is used for comparison if it is not possible on the basis of SR and AFE both. Outcome of this comparison is summarized in Table 3. In Table 3, '+' indicates that the FPABC is better than the considered algorithms and '-' indicates that the algorithm is not better or the difference is very small. The last row of Table 3, establishes the superiority of FPABC over ABC and MABC.

For the purpose of comparison in terms of consolidated performance, boxplot analyses have been carried out for all the considered algorithms. The empirical distribution of data is efficiently represented graphically by the boxplot analysis tool [11]. The boxplots for ABC, MABC and FPABC are shown in Figure 1. It is clear from this figure that FPABC is better than the considered algorithms as interquartile range and median are comparatively low.

TABLE II. COMPARISON OF THE RESULTS OF TEST PROBLEMS

Test Function	Algorithm	SD	ME	AFE	SR	Test Function	Algorithm	SD	ME	AFE	SR
f1	ABC	1.66E-06	8.64E-06	16520.09	100	f9	ABC	1.80E-06	7.90E-06	16767	100
	MABC	2.68E-06	5.47E-06	10350.53	100		MABC	9.96E-07	8.93E-06	14189.06	100
	FPABC	3.05E-06	5.03E-06	9314.71	100		FPABC	1.37E-06	8.31E-06	9366	100
f2	ABC	1.03E-01	1.67E-01	199254.48	1	f10	ABC	5.16E-06	1.04E-06	109879.46	62
	MABC	8.26E-03	1.25E-02	147787.15	52		MABC	4.11E-15	4.73E-15	14228.59	100
	FPABC	1.71E-02	1.95E-02	151300.35	46		FPABC	4.37E-15	4.87E-15	3956.05	100
f3	ABC	7.33E-05	1.76E-04	180578.91	18	f11	ABC	4.44E-05	1.60E-05	181447.91	17
	MABC	8.05E-05	2.02E-04	187320.13	13		MABC	1.45E-03	6.64E-04	199872.75	1
	FPABC	2.15E-05	8.68E-05	90834.53	97		FPABC	2.79E-14	4.02E-14	46909.7	100
f4	ABC	1.05E+00	6.36E-01	176098.02	23	f12	ABC	5.33E-03	4.91E-01	1460.56	100
	MABC	9.19E-01	6.99E-01	180961.73	23		MABC	5.74E-03	4.91E-01	2370.5	100
	FPABC	1.60E-02	8.45E-02	99219.48	99		FPABC	5.40E-03	4.90E-01	792	100
f5	ABC	2.42E-06	7.16E-06	9013.5	100	f13	ABC	6.67E-06	8.92E-05	1166.5	100
	MABC	1.61E-06	8.23E-06	8702	100		MABC	6.15E-06	8.95E-05	1702.28	100
	FPABC	2.08E-06	6.83E-06	5585.5	100		FPABC	6.45E-06	8.79E-05	622	100
f6	ABC	1.21E+01	8.91E+01	200011.71	0	f14	ABC	2.89E-06	1.94E-03	24476.88	100
	MABC	1.15E+01	8.00E+01	200015.14	0		MABC	2.79E-06	1.95E-03	9019.7	100
	FPABC	9.24E+00	8.56E+01	200006.8	0		FPABC	2.74E-06	1.95E-03	5127.73	100
f7	ABC	3.54E+03	1.11E+04	200029.02	0	f15	ABC	5.34E-06	4.86E-06	4752.21	100
	MABC	2.76E+03	1.03E+04	200015.92	0		MABC	5.60E-06	4.83E-06	33268.91	100
	FPABC	3.00E+03	1.08E+04	200016.04	0		FPABC	5.72E-06	5.07E-06	2550.57	100
f8	ABC	2.21E-03	6.95E-04	61650.9	90	f16	ABC	1.83E-03	7.77E-03	54159.26	99
	MABC	2.21E-03	6.24E-04	85853.52	92		MABC	1.03E-01	6.44E-01	200035.08	0
	FPABC	7.35E-04	7.88E-05	38328.96	99		FPABC	2.09E-03	7.87E-03	49230.85	100

SD-Standard Deviation, ME-Mean Error, AFE-Average function Evaluation, SR-Success Rate

V. CONCLUSION

In this paper, to improve the exploitation in ABC, a fitness based position update strategy is presented and incorporated with ABC. The so obtained modified ABC is named as fitness based mutation in ABC (FPABC). It is shown that, in the proposed strategy, better solutions exploits the search space in their neighborhood while less fit solutions explore the search area based on the fitness. Further, the proposed algorithm is compared to the recent variants of ABC, namely, MABC and with the help of experiments over test problems, it is shown

that the FPABC outperforms to the considered algorithms in terms of reliability, efficiency and accuracy.

TABLE III. SUMMARY OF TABLE II OUTCOMES

Function	FPABC vs ABC	FPABC vs MABC
f1	+	+
f2	+	-
f3	+	+
f4	+	+
f5	+	+
f6	=	=
f7	=	=
f8	+	+
f9	+	+
f10	+	+
f11	+	+
f12	+	+
f13	+	+
f14	+	+
f15	+	+
f16	+	+
Total number of + sign	14	13

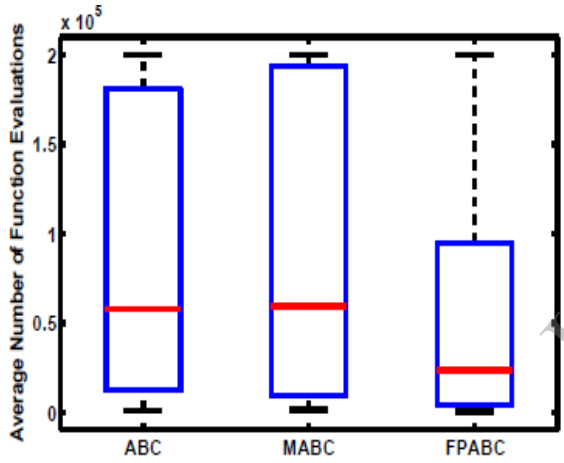

Fig. 1. Boxplots graph for average number of function evaluation

REFERENCES

[1] B. Akay and D. Karaboga, "A Modified artificial bee colony algorithm for real-parameter optimization," Information Sciences, doi:10.1016/j.ins.2010.07.015, 2010.

[2] K. Diwold, A. Aderhold, A. Scheidler, and M. Middendorf, "Performance evaluation of artificial bee colony optimization and new selection schemes," Memetic Computing, pages 1–14, 2011.

[3] M. El-Abd, "Performance assessment of foraging algorithms vs. evolutionary algorithms," Information Sciences, 182(1):243–263, 2011.

[4] D.E. Goldberg. Genetic algorithms in search, optimization, and machine learning. Addison-wesley, 1989.

[5] D. Karaboga, "An Idea based on honey bee swarm for numerical optimization," Techn. Rep. TR06, Erciyes Univ. Press, Erciyes, 2005.

[6] D. Karaboga and B. Akay, "A Comparative study of artificial bee colony algorithm," Applied Mathematics and Computation, 214(1):108–132, 2009.

[7] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," Foundations of Fuzzy Logic and Soft Computing, pages 789–798, 2007.

[8] J. Kennedy and R. Eberhart, "Particle swarm optimization," In Neural Networks, 1995. Proceedings, IEEE International Conference on, volume 4, pages 1942–1948. IEEE, 1995.

[9] K.M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," Control Systems Magazine, IEEE, 22(3):52–67, 2002.

[10] R. Storn and K. Price, "Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces," Journal of Global Optimization, 11:341–359, 1997.

[11] D.F. Williamson, R.A. Parker, and J.S. Kendrick, "The box plot: a simple visual method to interpret data," Annals of internal medicine, 110(11):916, 1989.

[12] G. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," Applied Mathematics and Computation, 217(7):3166–3173, 2010.

[13] F Kang, J Li, Z Ma, H Li, "Artificial bee colony algorithm with local search for numerical optimization." Journal of Software 6.3 (2011): 490-497.

[14] J.C. Bansal, H. Sharma, K.V. Arya and A. Nagar, "Memetic search in artificial bee colony algorithm." Soft Computing (2013): 1-18.

[15] S. Kumar, V. K. Sharma, R. Kumari, "Randomized memetic artificial bee colony algorithm," International Journal of Emerging Trends and Technology in Computer Science (IJETTCS).In Press.

[16] S. Kumar, V.K. Sharma, and R. Kumari. "A Novel Hybrid Crossover based Artificial Bee Colony Algorithm for Optimization Problem." International Journal of Computer Applications 82(8), 2013.

[17] S. Pandey and S. Kumar, "Enhanced Artificial Bee Colony Algorithm and It's Application to Travelling Salesman Problem." HCTL Open International Journal of Technology Innovations and Research, Volume 2, 2013:137-146.

[18] J. C. Bansal, H. Sharma, A. Nagar, and K. V. Arya. "Balanced artificial bee colony algorithm." International Journal of Artificial Intelligence and Soft Computing 3, no. 3 (2013): 222-243.

[19] H. Sharma, J. C. Bansal, K. V. Arya, and K. Deep. "Dynamic Swarm Artificial Bee Colony Algorithm." International Journal of Applied Evolutionary Computation (IJAEC) 3, no. 4 (2012): 19-33.

[20] H. Sharma, J. C. Bansal, and K. V. Arya. "Opposition based lévy flight artificial bee colony." Memetic Computing (2012): 1-15.

[21] V. K. Sharma and S. Kumar, "Modified artificial bee colony algorithm for optimization problems," unpublished.

[22] J. C. Bansal, H. Sharma and S. S. Jadon, "Artificial bee colony algorithm: a survey." International Journal of Advanced Intelligence Paradigms 5.1 (2013): 123-159.