ISSN 2007-9737

A Comparative Analysis of Selection Schemes in the Artificial Bee Colony Algorithm

Ajit Kumar[1], Dharmender Kumar[2], S.K. Jarial[1]

[1] University of Science and Technology, Deenbandhu Chhotu Ram, Murthal, Sonepat, India

[2] University of Science and Technology, Guru Jambheshwar, Hisar, Haryana, India

{ajit.hisar, dharmindia24}@gmail.com, s.jarial@rediffmail.com

Abstract. The Artificial Bee Colony (ABC) algorithm is a popular swarm based algorithm inspired by the intelligent foraging behavior of honey bees. In the past, many swarm intelligence based techniques were introduced and proved their effective performance in solving various optimization problems. The exploitation of food sources is performed by onlooker bees in accordance with a proportional selection scheme that can be further modified to avoid such shortcomings as population diversity and premature convergence. In this paper, different selection schemes, namely, tournament selection, truncation selection, disruptive selection, linear dynamic scaling, linear ranking, sigma truncation, and exponential ranking have been used to analyze the performance of the ABC algorithm by testing on standard benchmark functions. From the simulation results, the schemes other than the standard ABC prove their efficient performance.

Keywords. Swarm based algorithm, artificial bee colony, optimization, selection scheme.

1 Introduction

A number of complex tasks are systematically performed by honey bees; a good example of such tasks is collection and processing of nectar [1]. The effectiveness and simplicity of the whole process is due to the decentralized decision making approach of honey bee colonies [2]. Such swarm intelligence features as autonomy, self-organizing, distributed functioning employed by a bee swarm provided inspiration to solve complex traffic, transportation problems [3, 4] and deterministic combinatorial problems in dynamic and uncertain environments [5, 6, 7]. Swarm intelligence algorithms based on the behavior of bees can be classified into two categories: the foraging behavior and the marriage behavior. Algorithms in the first category are inspired by searching for food sources and nest sites, while those of the second category are based on the marriage behavior [8]. One of the most important algorithms inspired by the foraging behavior of honey bee swarms is the Artificial Bee Colony (ABC). It was proposed by Karaboga and is used for solving various optimization problems [9, 10].

The remainder of the paper is organized as follows. Section 2 presents the original ABC algorithm and its selection scheme. Various selection schemes applied to the ABC are described in Section 3. The experimental results are presented and analyzed in Section 4. The paper is concluded in Section 5.

2 Artificial Bee Colony Algorithm

The ABC is a population based optimization algorithm which is iterative in nature. Basically, the ABC consists of cycles of four phases: the initialization phase, the employed bees phase, the onlooker bees phase, and the scout bees phase. The bees going to a food source already visited by them are the employed bees, while the bees looking for a food source are unemployed. The scout bees carry out search for new food sources, and the onlooker bees wait for the information from the employed bees for food sources. The information exchange among bees takes place through the waggle dance. There is one employed bee for every food source. An employed bee becomes scout when the position of a food source does not get improved through the predetermined number of attempts called "limit". In this way, the exploitation process is performed by the employed and onlooker bees, whereas the scouts perform exploration of the search space [10].

There are three control parameters used in the ABC algorithm: the number of employed or onlooker bees to represent the number of food sources (N), the value of limit, the maximum cycle number (MCN). The main steps of the ABC are as follows.

- Step 1. Generate the initial population of solutions x_i^j, i=1…N, j=1…D using (1) and evaluate the fitness using (2).
- Step 2. Generate new solutions for the employed bees using (3) and evaluate the fitness.
- Step 3. Apply the greedy selection process for the employed bees.
- Step 4. Calculate the probability values for the current solution using (4) so that the onlooker bee can choose one according to its value.
- Step 5. Assign the onlooker bees to the solutions according to the probability, generate new solutions using (3) and evaluate the fitness.
- Step 6. Apply the greedy selection process for the onlooker bees.
- Step 7. If there is a solution abandoned by the bees, stop its exploitation and replace it with a new solution produced by (1).
- Step 8. Memorize the best solution found so far.
- Step 9. Check the termination criteria. If not satisfied, go to Step 2, otherwise end.

$$x_i^j = x_{min}^j + rand(0,1)(x_{max}^j - x_{min}^j), \tag{1}$$

where x_i^j is a parameter for the ith employed bee on the jth dimension, x_{max}^j and x_{min}^j are the upper and lower bounds for x_i^j.

$$fit_i = \begin{cases} \dfrac{1}{1+f_i} & f_i \geq 0 \\ 1 + abs(f_i) & f_i < 0 \end{cases}, \tag{2}$$

where f_i is a specific objection function and fit_i is a fitness value.

$$v_{ij} = x_{ij} + \phi\left(x_{ij} - x_{kj}\right), \tag{3}$$

where i, k∈{1…N}, i ≠ k and j∈{1…D}, x_{ij} is the ith employed bee in the jth dimension, v_{ij} is a new solution for x_{ij}, x_{kj} is the neighbor of x_{ij}, ϕ is a random number in the range [-1,1] to control the production of neighbor solutions around x_{ij}.

$$p_i = \frac{fit_i}{\sum_{j=1}^{N} fit_j}, \tag{4}$$

where fit_i is the fitness value of the ith solution and p_i is the selection probability of the ith solution.

2.1 Selection Scheme in the Basic ABC

As explained above, food sources are chosen by the onlooker bees using a stochastic selection scheme in accordance with the probability value $\boldsymbol{p_i}$. The process employs three stages [11]:

(i) Calculate the fitness value using (2).
(ii) Calculate the probability value using (4).
(iii) Choose a food source according to the probability value based on the roulette wheel method.

However, the proportional selection scheme employed in the ABC has two shortcomings viz. reduction in population diversity and premature convergence. Thus, the ABC is not able to maintain the balance between exploration (diversification) and exploitation (intensification) of the search space and is considered as an inefficient algorithm.

3 Description of Selection Schemes

The selection scheme plays an important role in the ABC algorithm as it drives the search space in a proper direction. These schemes may be classified in two categories: proportionate selection and ordinal based selection. In the proportionate selection scheme, individuals are selected on the basis of their fitness values relative to the fitness of others, whereas in the ordinal based scheme, individuals are selected based on their rank in the population. The rank is determined in accordance with their fitness values. The schemes presented in this paper except the proportional selection in the basic ABC are covered in the ordinal based selection category. In this work, we performed experiments on the ABC using different selection schemes. The details of the schemes are given in what follows.

3.1 Tournament Selection

This selection scheme works by holding a tournament of N individuals chosen from the population, where N is taken as the tournament size [11, 12, 13, 14]. The fitness values of individuals are compared and some score (say, s) is assigned to the best one. The process is repeated till the best in the population achieves the highest score. The individuals are then selected according to the probability using the following equation:

$$\mathrm{P_i} = \frac{\mathrm{S_i}}{\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{S_i}}. \tag{5}$$

3.2 Truncation Selection

This selection scheme assigns equal selection probabilities to the μ best individuals selected in a population of size λ and is equivalent to (μ,λ)-selection used in evolution strategies [12, 15, 16]. The selection probabilities are given as

$$\mathrm{p_i} = \begin{cases} 1/\mu, 1 \le \mathrm{i} \le \mu \\ 0, \mu < i \le \lambda \end{cases}. \tag{6}$$

3.3 Disruptive Selection

This scheme introduces the concept of normalized-by-mean fitness function. The idea is to give more chances to better and worse solutions in comparison to moderate solutions so that the population diversity can be improved [11, 17, 18]. The selection probability is calculated as follows:

$$\mathrm{p_i} = \frac{\mathrm{fit_i}}{\sum_{\mathrm{j=1}}^{\mathrm{N}} \mathrm{fit_j}}, \tag{7}$$

where $\mathrm{fit_i}$ is the fitness value of the ith solution and P_i is the selection probability of the ith solution. The fitness function is given by

$$fit_i = \left|f_i - \bar{f}\,\right|, \tag{8}$$

where $\boldsymbol{f_i}$ is a specific objective function, $\bar{\boldsymbol{f}}$ is the average of the objective values for the individuals in the population.

3.4 Linear Dynamic Scaling

In order to improve the performance of the proportional selection, it is combined with a scaling technique called linear dynamic scaling [12]. The dynamic scaling is introduced to favor better individuals resulting in improved population fitness over generations. The selection probability is given by

$$P_i = \frac{f_i - c}{S_f - \lambda.c}, \tag{9}$$

where $S_f = \sum_{j=1}^{\lambda} f_j$, c > 0, and λ is the number of solutions in the population.

3.5 Linear Ranking

In this scheme, the ranks are assigned to the individuals based on their fitness values. The individual having the worst fitness is assigned rank 1 and the best fitness is assigned rank N. The method uses a linear function to calculate selection probabilities according to the rank of individuals [12, 16]:

$$p_i = \frac{1}{N}\left(\eta^- + (\eta^+ - \eta^-)\frac{i-1}{N-1}\right), i \,\epsilon\, \{1, \dots, N\}. \tag{10}$$

To satisfy the constraints, two conditions must be fulfilled:

$$\boldsymbol{\eta^+ = 2 - \eta^-} \text{ and } \boldsymbol{\eta^- \ge 0}.$$

3.6 Sigma Truncation

In order to improve the fitness of a population, low fitness individuals are discarded using the standard deviation of fitness values before scaling

Table 1. Results of algorithms (varying parameters)
[Colony size=100, Limit=100, Max Cycles=100, Runs=10]

		ABC			TABC			TRABC			DABC		
	D	10	50	100	10	50	100	10	50	100	10	50	100
f1	Mean	7.86E-04	18197.9	119349	1.26E-02	15293.3	101725	3.23E-04	11503.3	101015	0.0390	6139.17	49425.8
	SD	4.79E-04	5773.81	11340.1	1.13E-02	3129.45	11241.6	2.14E-04	2799	5199.34	0.0184	2877.18	14375.5
f2	Mean	77.67	2.37E+09	4.83E+10	308.28	2.41E+09	3.97E+10	61.64	1.09E+09	3.44E+10	722.16	1.14E+09	1.98E+10
	SD	41.85	1.62E+09	7.54E+09	278.05	8.10E+08	6.79E+09	50.20	7.45E+08	5.97E+09	381.39	9.01E+08	5.07E+09
f3	Mean	1.89	257.31	943.99	2.057	228.82	864.08	1.598	214.035	801.84	2.329	146.91	596.78
	SD	0.94	26.58	42.13	0.831	24.41	26.08	0.907	23.45	46.22	1.149	35.60	90.38
f4	Mean	0.1446	177.71	1068.53	0.1715	153.75	901.85	0.139	98.75	936.073	0.2242	65.108	575.22
	SD	0.0640	41.95	101.08	0.0605	41.73	125.67	0.070	32.8	101.12	0.0773	15.979	141.43
f5	Mean	0.7058	17.05	19.71	0.7835	15.60	19.185	0.229	15.7	19.14	0.2550	6.753	16.243
	SD	0.4060	1.00	0.121	0.5037	0.644	0.231	0.072	0.818	0.095	0.374	2.325	2.195
f6	Mean	-3881.45	-12678.8	-18256.9	-3948.01	-12809.1	-18606.5	-3892.84	-12632.1	-18548.1	-3988.83	-14476.8	-23211.3
	SD	93.39	341.11	779.43	84.84	462.49	897.98	147.18	436.095	635.19	72.67	767.98	1393.06
		LDABC			**LRABC**			**STABC**			**ERABC**		
	D	10	50	100	10	50	100	10	50	100	10	50	100
f1	Mean	1.08E-04	29135.9	146802	7.91E-02	27470.9	139846	7.55E-04	4907.19	51886.7	5.57E-03	11149	89782.1
	SD	1.07E-04	4280.13	7117.67	4.61E-02	5451.15	7891.2	3.32E-04	304.61	16109.3	3.79E-03	4065.45	7305.54
f2	Mean	689.50	6.06E+09	5.46E+10	779.85	6.11E+09	5.77E+10	108.27	2.68E+08	1.64E+10	174.54	1.53E+09	3.298E+10
	SD	285.78	2.27E+09	8.34E+09	660.82	1.68E+09	7.75E+09	71.12	1.3E+08	3.11E+09	77.083	9.34E+08	6.069E+09
f3	Mean	1.477	308.83	1014.3	3.85	302.34	1007.81	0.945	142.656	693.134	2.232	203.157	772.51
	SD	1.142	20.001	60.858	1.19	19.09	37.66	0.735	29.66	61.55	0.848	22.893	35.863
f4	Mean	0.0977	273.024	1255.24	0.238	252.194	1217.57	0.101	28.204	551.79	0.125	110.507	818.615
	SD	0.0378	30.143	119.355	0.087	45.30	92.28	0.046	13.77	134.22	0.058	20.925	111.986
f5	Mean	0.462	17.521	19.670	1.536	17.537	19.693	0.107	11.614	17.83	0.505	14.684	18.705
	SD	0.342	0.728	0.1256	0.436	0.439	0.146	0.070	2.258	0.686	0.399	0.837	0.461
f6	Mean	-3832.67	-12651.7	-19983.4	-3839.7	-11398.6	-16375.4	-3970.52	-13403.1	-20470.4	-3929.34	-13026.4	-20521.8
	SD	104.237	591.847	1193.19	91.427	558.706	1020.43	132.09	435.806	1356.8	41.297	533.769	1102.73

them. This scheme ensures the selection of good fitness individuals [19, 20]. The fitness values of individuals are calculated as

$$fit' = fit - (\overline{fit} - c\sigma), \tag{11}$$

where $\overline{fit}$ is the average fitness value of the population, σ is the standard deviation of the fitness values, c is a small constant having values from 1 to 3.

3.7 Exponential Ranking

In this scheme, ranks are assigned to the individuals similar to linear ranking. The difference lies in exponential weighing of ranked individuals to compute probabilities as follows [12, 16]:

$$p_i = \frac{c-1}{c^N - 1}\, c^{N-i}, i \in \{1, \dots, N\}, \tag{12}$$

where c<1, an indicative of the selection probability of the best individual.

4 Experimental Results and Discussions

4.1 Test Problems

Six benchmark functions were used for simulation to evaluate the performance of various selection

Table 2. Results of algorithms (varying maximum cycles)
[Colony Size=100, Limit=100, Parameters=100, Runs=10]

		ABC			TABC			TRABC			DABC		
	MCN	10	50	100	10	50	100	10	50	100	10	50	100
f1	Mean	250930	185124	126051	246890	172770	102611	247661	168817	104904	242465	128258	55980.9
	SD	12205.3	10471.2	11303.3	12918.1	11439.3	13032.3	11757.9	12539	10243.5	15606.1	13879.3	16367.7
f2	Mean	1.326E+11	8.38E+10	5.048E+10	1.267E+11	7.55E+10	4.098E+10	1.299E+11	7.54E+10	4.136E+10	1.169E+11	5.554E+10	2.509E+10
	SD	8.10E+09	1.21E+10	5.83E+09	1.66E+10	9.46E+09	4.86E+09	9.94E+09	5.687E+09	5.282E+09	1.426E+10	9.23E+09	8.797E+09
f3	Mean	1548.44	1216.53	913.72	1488.28	1163.74	859.263	1521.33	1112.25	840.877	1482.5	989.64	652.166
	SD	42.45	50.53	56.87	47.078	42.094	33.698	48.496	48.559	33.003	85.277	91.003	72.199
f4	Mean	2283.56	1575.77	1098.64	2200.89	1507.49	989.412	2211.95	1518.74	912.409	2159.39	1225.44	529.602
	SD	98.56	93.51	139.31	78.463	98.668	105.66	154.68	130.747	115.838	106.184	238.072	147.206
f5	Mean	20.823	20.341	19.766	20.773	20.077	19.249	20.819	20.079	19.256	20.609	19.444	14.618
	SD	0.0619	0.123	0.132	0.0654	0.123	0.182	0.064	0.162	0.167	0.198	0.321	1.694
f6	Mean	-7653.57	-14867.4	-18138.6	-6461.31	-13550.5	-18258.9	-6617.82	-14444.5	-18974.3	-7255.93	-17191	-23046.7
	SD	615.089	1269.98	501.176	671.23	676.86	709.486	762.275	727.245	730.611	621.579	1025.85	1469.03
		LDABC			LRABC			STABC			ERABC		
	MCN	10	50	100	10	50	100	10	50	100	10	50	100
f1	Mean	249375	197323	141879	249782	188344	132937	227712	120885	52807	240551	156249	89304.2
	SD	13949.8	11242.1	10079	11540.8	11004.1	10138.4	20660.9	23585.2	17806.5	10449.7	12120.8	8856.27
f2	Mean	1.28E+11	9.54E+10	6.22E+10	1.298E+11	9.042E+10	5.509E+10	1.206E+11	5.37E+10	1.789E+10	1.248E+11	6.903E+10	3.070E+10
	SD	7.52E+09	8.32E+09	4.21E+09	1.003E+10	8.421E+09	3.828E+09	9.359E+09	1.40E+10	7.726E+09	9.910E+09	8.229E+09	7.233E+09
f3	Mean	1530.99	1286.12	1007.75	1540.29	1265	972.159	1429.55	1011.67	637.973	1462.19	1051.77	768.871
	SD	36.66	33.04	49.335	40.013	31.583	40.345	55.543	91.026	67.286	42.485	62.256	46.358
f4	Mean	2273.92	1739.34	1280.77	2285.28	1711.15	1226.76	1993.09	1134.25	455.893	2184.82	1335.31	846.794
	SD	103.46	103.40	49.93	147.962	106.724	78.204	233.127	221.491	191.044	142.632	104.762	66.028
f5	Mean	20.525	20.347	19.682	20.799	20.299	19.694	20.699	19.659	17.892	20.712	19.93	18.717
	SD	0.049	0.078	0.130	0.097	0.111	0.108	0.088	0.361	0.564	0.057	0.145	0.273
f6	Mean	-8787.72	-15683.6	-20328.1	-6482.74	-11687.1	-16510.4	-8690.88	-15543.2	-20975.7	-7339.52	-15108.5	-20400.8
	SD	1043.64	1894.68	1685.1	953.017	579.232	394.528	1051.32	967.116	1121.47	708.848	679.761	1198.11

schemes in the ABC. These functions are the following ones:

i) Sphere function:

$$f_1(x) = \sum_{i=1}^{n} x_i^2,\ -100 \le x_i \le 100. \quad (13)$$

ii) Rosenbrock function:

$$f_2(x) = \sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2, \quad -100 \le x_i \le 100. \quad (14)$$

iii) Rastrigin function:

$$f_3(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10), \quad -5.12 \le x_i \le 5.12. \quad (15)$$

iv) Griewank function:

$$f_4(x) = \frac{1}{4000}\left(\sum_{i=1}^{n} x_i^2\right) - \left(\prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)\right) + 1, \quad -600 \le x_i \le 600. \quad (16)$$

v) Ackley function:

Table 3. Results of algorithms (varying colony size)
[Limit=100, Parameters=100, Max Cycles=100, Runs=10]

		ABC			TABC			TRABC			DABC		
	colony	10	50	100	10	50	100	10	50	100	10	50	100
f1	Mean	155803	120075	121949	153139	114532	104069	111492	111246	98360.8	128334	64734	61148
	SD	12251.9	16413.2	8622.49	15228	15009.8	9983.29	40111.8	11485.9	9452.93	18080.1	21111.3	18442.1
f2	Mean	7.396E+10	5.61E+10	4.611E+10	6.305E+10	4.362E+10	4.281E+10	5.765E+10	3.778E+10	3.501E+10	6.086E+10	2.674E+10	2.423E+10
	SD	1.595E+10	7.42E+09	1.229E+10	1.575E+10	1.107E+10	6.642E+09	2.378E+10	6.267E+09	7.195E+09	1.294E+10	8.313E+09	7.376E+09
f3	Mean	1096.59	977.255	932.744	984.253	887.64	847.167	967.669	846.695	821.185	905.62	636.894	595.537
	SD	45.81	62.85	47.755	46.754	53.319	52.681	77.5	47.25	58.120	73.099	87.505	86.610
f4	Mean	1408.62	1137.28	969.686	1318.7	1066.59	939.254	1132.43	925.896	892.631	1169.85	616.013	481.919
	SD	142.166	104.152	120.243	134.866	128.681	102.622	235.766	108.521	140.515	153.54	178.318	172.974
f5	Mean	20.029	19.783	19.622	19.721	19.269	19.160	19.688	19.325	19.229	19.604	17.439	14.885
	SD	0.119	0.207	0.152	0.218	0.294	0.215	0.615	0.250	0.223	0.348	1.018	2.381
f6	Mean	-15851.9	-18777.1	-18969	-15924.3	-17914.9	-18322	-16751.9	-18753.7	-19305.4	-17554.3	-20919.8	-22832.8
	SD	1732.79	1105.3	933.175	666.684	811.496	637.356	1872.09	568.458	955.061	1810.79	1160.48	1101.72
		LDABC			**LRABC**			**STABC**			**ERABC**		
	colony	10	50	100	10	50	100	10	50	100	10	50	100
f1	Mean	163777	148261	139431	163620	145658	140055	122765	77051.8	47392.9	159043	124064	87743.4
	SD	14833.9	12756.4	15247.9	12274.2	11780.2	11438	18427.8	14461.4	18164	11245.1	10493.2	8257.7
f2	Mean	8.805E+10	6.68E+10	6.065E+10	7.084E+10	5.91E+10	6.007E+10	5.26E+10	2.268E+10	1.234E+10	8.021E+10	4.966E+10	3.311E+10
	SD	1.155E+10	5.34E+09	9.48E+09	1.678e+10	6.547E+09	7.358E+09	1.39E+10	8.829E+09	8.756E+09	9.124E+09	8.241E+09	3.718E+09
f3	Mean	1117.99	1065.95	1026.66	1091.21	1024.36	986.227	865.64	667.954	600.535	1071.2	914.14	766.214
	SD	50.79	32.505	35.87	84.147	33.897	45.925	92.444	72.493	116.839	86.789	66.309	47.801
f4	Mean	1515.63	1382.41	1306.19	1575	1284.57	1221.79	1011.66	581.717	545.466	1491.3	1134.4	795.049
	SD	111.498	103.275	92.921	101.208	85.695	110.568	142.21	181.086	163.445	178.409	100.799	110.972
f5	Mean	20.088	19.751	19.77	20.021	19.780	19.812	19.47	18.684	17.72	20.117	19.425	18.75
	SD	0.102	0.216	0.114	0.154	0.138	0.078	0.343	0.440	1.034	0.129	0.203	0.285
f6	Mean	-16861.1	-19109.2	-19325.2	-15016.1	-16195.2	-16759	-17222.7	-19346.7	-20759	-14485	-17676.1	-20221
	SD	2151.08	959.05	836.13	1088.17	863.413	635.595	622.396	1316.32	1132.32	892.029	754.008	485.715

$$f_5(x) = 20 + e - 20e^{\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)} - e^{\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)}, \quad -32 \le x_i \le 32. \tag{17}$$

vi) Schwefel function:

$$f_6(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right), \quad -500 \le x_i \le 500. \tag{18}$$

4.2 Experimental Settings

The algorithms for various selection schemes are implemented using MATLAB R2012a on an Intel (R) Core (TM) i3 CPU 3.06 GHZ with 4 GB RAM. In the following tables, ABC represents the original proportional scheme. TABC means the tournament selection, TRABC represents the truncation selection, DABC is the disruptive selection, LDABC

Table 4. Results of algorithms (varying initialization range)
(FR: Full Range, LHR: Left Half Range, RHR: Right Half Range)
[Colony size=100, Limit=100, Parameters=100, Max Cycles=100, Runs=10]

		ABC			**TABC**			**TRABC**			**DABC**		
	Range	FR	LHR	RHR	FR	LHR	RHR	FR	LHR	RHR	FR	LHR	RHR
f1	Mean	121689	153620	150495	107252	142749	140865	103796	139065	135255	61719.5	84576.3	89298.2
	SD	10883.5	14247.6	12512.8	15826.4	9520.49	6115.17	12214.2	19030.2	8912.89	14714.7	13296.3	24164.7
f2	Mean	4.75E+10	5.79E+10	5.83E+10	3.81E+10	5.47E+10	5.79E+10	3.81E+10	5.39E+10	5.00E+10	2.39E+10	3.86E+10	3.12E+10
	SD	8.18E+09	1.02E+10	7.58E+09	4.587E+09	5.532E+09	5.079E+09	4.662E+09	9.373E+09	8.797E+09	1.001E+10	1.223E+10	9.103E+09
f3	Mean	909.938	1002.02	963.617	847.579	912.07	934.039	842.424	909.494	913.36	607.236	647.623	743.819
	SD	52.910	27.789	61.354	39.451	37.340	56.306	24.046	47.093	39.436	54.797	87.810	63.378
f4	Mean	1080.47	1399.01	1393.61	964.143	1217.68	1257.09	922.779	1171.9	1236.74	533.073	836.681	857.867
	SD	101.607	88.326	115.701	76.564	83.187	107.068	70.082	67.728	135.894	194.406	140.974	196.065
f5	Mean	19.692	20.022	20.028	19.076	19.626	19.645	19.090	19.686	19.646	14.543	17.628	18.204
	SD	0.122	0.0959	0.0952	0.242	0.154	0.164	0.233	0.214	0.146	2.885	1.326	0.537
f6	Mean	-18449.8	-18813.3	-20328.6	-18857.7	-15702.1	-21248.1	-18586.6	-20587.6	-20976.1	-24007.4	-20263.3	-24533.9
	SD	946.304	1376.26	1042.54	608.96	590.622	909.506	658.871	1208.27	846.723	1286.78	1160.12	945.887

		LDABC			**LRABC**			**STABC**			**ERABC**		
	Range	FR	LHR	RHR	FR	LHR	RHR	FR	LHR	RHR	FR	LHR	RHR
f1	Mean	146802	171035	173591	134539	166072	164595	56154.3	92882.7	85664	87489.8	120376	119350
	SD	7117.67	7075.02	11976.6	8120.69	13929.8	14076.1	12525.1	13795	17405.2	10384.3	16138.6	6182.65
f2	Mean	5.46E+10	7.67E+10	7.37E+10	5.680E+10	6.855E+10	7.297E+10	1.920E+10	2.137E+10	2.482E+10	3.271E+10	4.46E+10	4.241E+10
	SD	8.34E+09	4.93E+09	8.28E+09	6.75E+09	9.118E+09	6.504E+09	6.733E+09	9.248E+09	9.829E+09	4.734E+09	7.743E+09	4.052E+09
f3	Mean	1014.3	1088.44	1071.89	987.873	1061.34	1052.4	656.849	745.054	737.336	783.299	855.078	805.276
	SD	60.858	31.30	36.65	40.923	48.759	58.716	68.01	72.310	48.533	33.132	52.021	47.068
f4	Mean	1255.24	1552.89	1527.36	1179.54	1518.76	1543.07	480.133	863.346	883.199	785.88	1073.36	1043.12
	SD	119.355	60.67	68.09	93.621	91.467	113.547	175.216	118.049	147.164	67.001	74.112	127.377
f5	Mean	19.670	20.05	20.052	19.681	19.982	19.894	17.765	19.232	19.123	18.873	19.460	19.482
	SD	0.1256	0.077	0.108	0.120	0.069	0.16	0.725	0.185	0.428	0.280	0.149	0.119
f6	Mean	-19187.6	-26122.8	-19933.3	-15948.8	-13200.3	-19031.3	-20469.4	-19687.1	-23429.8	-20405.7	-17743.7	-22227.7
	SD	1174.77	1123.43	1162.49	655.972	708.532	887.43	958.378	1190.62	1566.3	723.357	761.21	847.707

is the linear dynamic scaling, LRABC means the linear ranking, STABC represents the sigma truncation, and ERABC is the exponential ranking scheme.

The experiments were performed on the six benchmark functions given above. In all the experiments, the limit was put to 100, and the values present the results of 10 runs (except Table 5 where runs=100). Alongside with comparing the mean values and standard deviations of the function values, the values of selection intensity, success rate, reproduction rate, and loss of diversity were also calculated.

4.3 Effect of Dimensions

We performed simulations on modified ABC algorithms to analyze the effect of varying dimensions of the problem. The colony size, maximum cycles, and limit were fixed as 100. The

Table 5. Results of algorithms
(SI: Selection Intensity, SR: Success Rate, RR: Reproduction Rate, Pd: Loss of Diversity)
[Colony size=100, Limit=100, Parameters=10, Max Cycles=100, Runs=100].

	ABC				**TABC**				**TRABC**				**DABC**			
	SI	SR	RR	Pd	SI	SR	RR	Pd	SI	SR	RR	Pd	SI	SR	RR	Pd
f1	0.058	100	1.062	0.989	0.005	100	1.062	0.989	0.029	100	1.142	0.988	0.018	100	1.066	0.989
f2	0.012	0	1.166	0.988	0.010	0	1.090	0.989	0.022	0	1.0	0.99	0.004	0	1.090	0.989
f3	0.009	81	1.052	0.989	0.008	38	1.066	0.989	0.008	92	1.0	0.99	0.014	27	1.0	0.99
f4	0.006	100	1.045	0.989	0.005	100	1.034	0.989	0.002	100	1.0	0.99	0.057	100	1.052	0.989
f5	0.015	100	1.041	0.989	0.005	100	1.052	0.989	0.005	100	1.052	0.989	0.011	100	1.0	0.99
f6	0.008	0	1.0	0.99	0.009	0	1.0	0.99	0.015	0	1.032	0.989	0.014	0	1.0	0.99
	LDABC				**LRABC**				**STABC**				**ERABC**			
	SI	SR	RR	Pd	SI	SR	RR	Pd	SI	SR	RR	Pd	SI	SR	RR	Pd
f1	0.012	100	1.166	0.988	0.014	100	1.0	0.99	0.002	100	1.0	0.99	0.013	100	1.0	0.99
f2	0.023	0	1.090	0.989	0.053	0	1.077	0.989	0.023	0	1.5	0.985	0.019	0	1.0	0.99
f3	0.029	74	1.043	0.989	0.018	11	1.045	0.989	0.001	100	1.0	0.99	0.007	19	1.066	0.989
f4	0.028	100	1.0	0.99	0.073	100	1.031	0.989	0.016	100	1.042	0.989	0.006	100	1.0	0.99
f5	0.024	100	1.0	0.99	0.032	100	1.052	0.989	0.004	100	1.0	0.99	0.018	100	1.0	0.99
f6	0.006	0	1.0	0.99	0.008	0	1.0	0.99	0.001	0	1.0	0.99	0.007	0	1.077	0.989

performance of all ABC algorithms deteriorated as the dimension of the problem was increased (10, 50, 100).

The results in Table 1 show that STABC generated better results for Rastrigin and Ackley functions followed by LDABC for Sphere and Griewank functions in less dimensions, i.e., 10. Again, STABC produced excellent results with an increase in dimensions up to 50. However, DABC had superior performance for 100 dimensions. From Fig. 1(a), we can see that the increase in dimensions makes the convergence of DABC method better for Sphere function and also for Rastrigin function as given in Fig. 1(b).

4.4 Effect of Cycles

We analyzed the performance of the ABC algorithms by varying the maximum number of cycles. The experiment was repeated for the six benchmark functions as given in Table 2.

The obtained values prove better results for the sigma truncation scheme on Sphere, Rosenbrock, Rastrigin, and Griewank functions. Figs. 2(a) and 2(b) prove better results of STABC on Rosenbrock function and of DABC on Ackley function. For a less number of cycles, i.e. 10, LDABC shows the best performance.

4.5 Effect of Colony Size

In the next experiment, we determined what size of population is suitable to generate better results. The experiment was conducted for all six test problems. Table 3 presents better results in case of STABC on Rosenbrock, Griewank functions, and in case of DABC on Rastrigin, Ackley, Schwefel functions for varying colony sizes.

For a small colony size of 10, the results of TRABC are good on Sphere function. The performance of DABC got improved with an increase in the colony size as given in Figs. 3(a) and 3(b).

4.6 Effect of Region Scaling

We also investigated the effect of initializing the solutions in various sub-regions of the search space. There was a possibility of variation in the performance of the algorithms during initialization in the left half and the right half of the search space. The results of the experiments using different selection schemes are reported in Table 4. The aim is to determine the sensitivity of the algorithms in finding global optima under varying initialization ranges. All the ABC algorithms were found to be less sensitive to initial solutions in finding global optima as shown in Figs. 4(a) and 4(b).

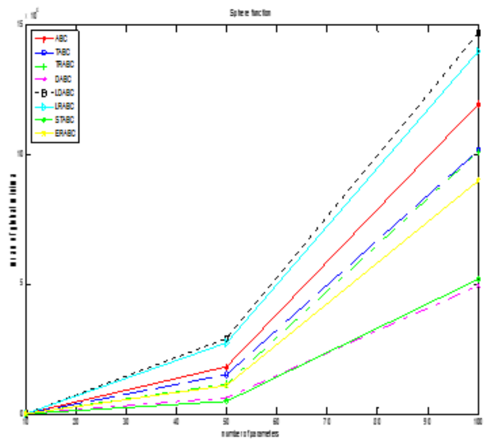

Fig. 1(a). Sphere

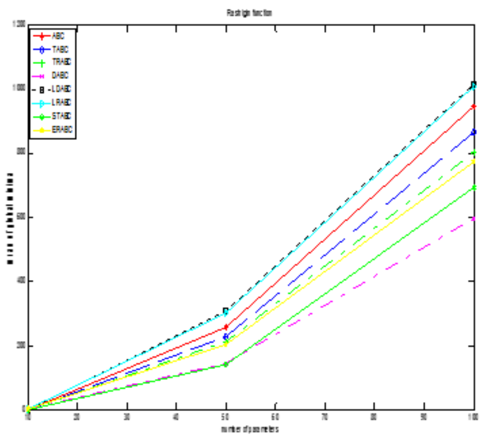

Fig. 1(b). Rastrigin

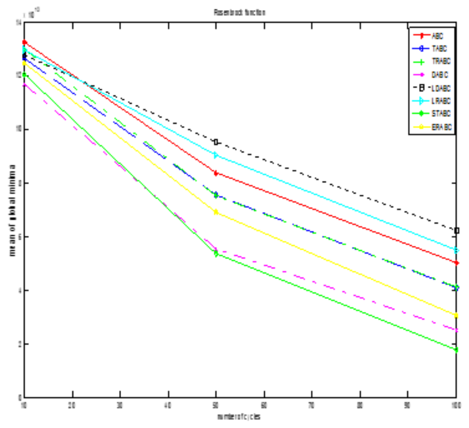

Fig. 2(a). Rosenbrock

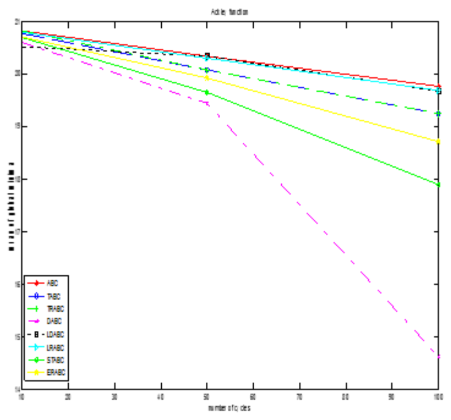

Fig. 2(b). Ackley

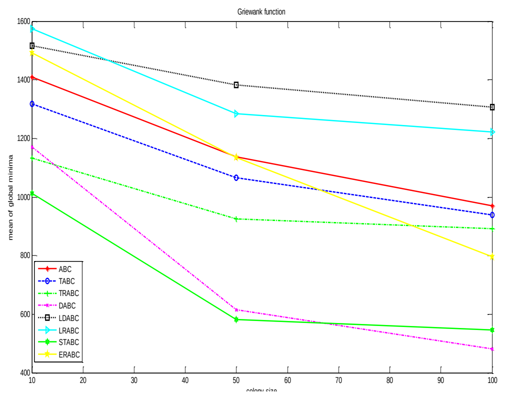

Fig. 3(a). Griewank

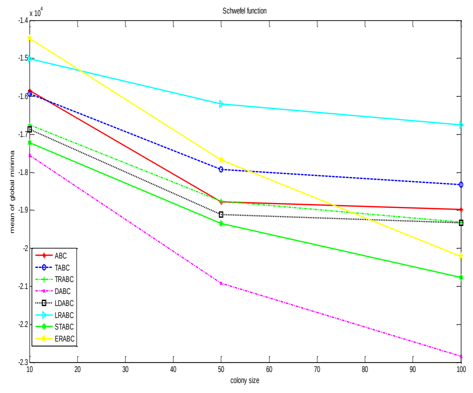

Fig. 3(b). Schwefel

4.7 Statistical Analysis

The proportional selection scheme used in the basic ABC lacks the driving force to attract better individuals which may result in premature convergence and a lack of population diversity. The tournament selection scheme randomly selects a number of N individuals and comparison is made based on their fitness values. The truncation selection scheme assigns equal selection probabilities to some selected best individuals in the population. The linear dynamic scaling scheme works by promoting better than average individuals at the cost of worse than average individuals. The linear ranking scheme is biased to favor the good fitness individuals in the population as the rank is assigned based on the fitness value. The exponential ranking scheme works in a similar manner to the linear ranking scheme except the use of the exponential function in computing selection probabilities.

From Figs. 1, 2, and 3, we can state that the DABC and STABC algorithms prove their effective performance in comparison to other algorithms. The disruptive selection scheme favors both high fitness and low fitness solutions and tends to maintain population diversity. Hence, this scheme improves the worse fitness solutions in concurrence with the high fitness solutions. In the case of STABC, the individuals having the fitness value less than c standard deviations of the average value are discarded, while a large portion of the population having the fitness values within c standard deviations of the average value are favored for selection.

Table 5 presents the analysis of the numerical results obtained with a slight change (i.e. 100 runs) in the experimental setting of subsection 4.2 using various selection schemes. Selection Intensity (SI) also called Selection Pressure measures the degree that drives the algorithm to improve the population fitness. It computes the difference between the population average fitness after and before selection. A high value of SI indicates high convergence rate, i.e. the algorithm is able to find optimal solutions early. Positive values of SI in Table 5 prove improvement in average fitness of the original ABC and the modified ABC algorithms due to selection for all test functions.

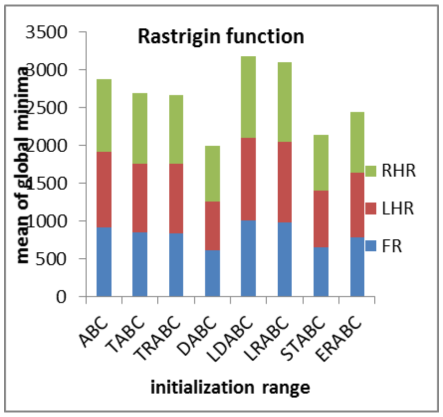

Fig. 4(a). Rastrigin

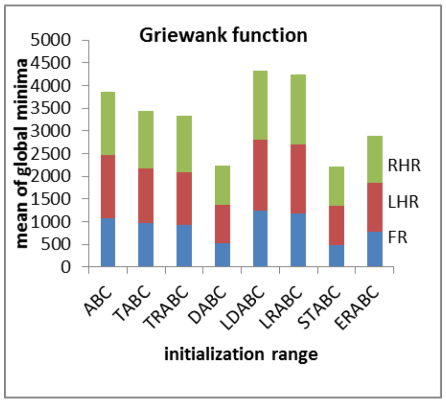

Fig. 4(b). Griewank

Success Rate (SR) shows that algorithm is able to obtain a desired function value (i.e. <2) using the given experimental settings. From the table, we can see that the success rate of the TRABC and STABC algorithms gets improved for Rastrigin function, whereas it is comparable to the original ABC for the remaining test functions.

Reproduction Rate (RR) is calculated to represent the ratio of the number of individuals with a certain fitness value after and before selection. A value of RR > 1 means better individuals are favored and bad individuals are discarded by a suitable selection scheme. Table 5 clearly shows that all selection schemes are able to replace bad individuals by better individuals.

Loss of Diversity (Pd) presents the ratio of the individuals of a population that are not selected during the selection stage. It means that Reproduction Rate and Loss of Diversity are related to each other. The value of Pd should be as low as possible, as a high value of Pd may increase the risk of premature convergence. The values in the table clearly confirm the results.

5 Conclusions and Future Work

In this paper, we compared the performance of the Artificial Bee Colony algorithm combined with different selection schemes on six numerical optimization functions. The simulations were performed by varying the values of different control parameters used in the ABC algorithm in addition to initialization ranges. On the basis of the results obtained, an analysis is made in terms of selection intensity, success rate, reproduction rate, and loss of diversity.

With an increase in the number of dimensions, it becomes difficult to find optimal solutions in all selection schemes. As the number of cycles increases, the algorithms explore and exploit efficiently the search space to provide proper convergence and population diversity. An increase in the colony size also provides an opportunity to find global optima values. The algorithms are also less sensitive to initialization ranges in obtaining optimal solutions.

Positive values of Selection Intensity in all schemes represent an increase in the population average fitness after selection. Success Rate is an indicative of obtaining a desired function value. All selection schemes favored good individuals by assigning the reproduction rate > 1. Similarly low values of loss of diversity support the avoidance of premature convergence. In general, the ABC algorithms combined with different selection schemes perform better on various parameters. In future work, the performance of the ABC can be improved by hybridizing it with a suitable selection scheme and an effective neighbor search technique.

References

1. **Camazine, S. & Sneyd, J. (1991).** A model of collective nectar source by honey bees: Self-organization through simple rules. *Journal of Theoretical Biology,* Vol. 149, No. 4, pp. 547–571.
2. **Seeley, T.D., Camazine, S., & Sneyd, J. (1991).** Collective decision-making in honey bees: how colonies choose among nectar sources. *Behav Ecol Sociobiol*, Vol. 28, pp. 277–290. DOI: 10.1007/BF00175101.
3. **Lucic, P. & Teodorovic, D. (2003).** Computing with bees: attacking complex transportation engineering problems. *International Journal on Artificial Intelligence Tools*, Vol. 12, No. 3, pp. 375–394. DOI: 10.1142/S0218213003001289.
4. **Teodorovic, D. (2003).** Transport modeling by multi-agent systems: a swarm intelligence approach. *Transport Plan Technol*, Vol. 26, No. 4, pp. 289–312. DOI: 10.1080/030810603200015 4593.
5. **Teodorovic, D. & Orco, M.D. (2005).** Bee colony optimization – a cooperative learning approach to complex transportation problems. *Proc. of 16 Mini-EURO conf. AI Transportation,* Poznan, Poland, pp. 51–60.
6. **Teodorovic, D., Lucic, P., Markovic, G., & Orco, M.D. (2006).** Bee colony optimization: principles and applications. *Proc. of 8 Seminar on Neural Network Applications in Electrical Engineering* (*NEUREL)*, Belgrade, Serbia & Montenegro, pp. 151–156.
7. **Karaboga, D., Gorkemli, B., Ozturk, C., & Karaboga, N. (2012).** A comprehensive survey: artificial bee colony (ABC) algorithm and applications. *Artificial Intelligence Review*, Vol. 42, No. 1, pp. 21–57.
8. **Bitam, S., Batouche, M., & Talbi, E. (2010).** A survey on bee colony algorithms. *Proc. of 24 IEEE Int'l Parallel and Distri Proces Sympos, NIDISC Workshop*, Atlanta, USA, pp. 1–8. DOI: 10.1109/IPDPSW.2010.5470701.
9. **Karaboga, D. (2005).** *An idea based on honey bee swarm for numerical optimization.* Technical Report–TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.
10. **Karaboga, D. & Ozturk, C. (2011).** A novel clustering approach: ABC algorithm. *Journal of Applied Soft Computing*, Vol. 11, pp. 652–657. DOI: 10.1016/j.asoc.2009.12.025.
11. **Bao, L. & Zeng, J. (2009).** Comparison and analysis of the selection mechanism in the artificial bee colony algorithm. *Proc. of Ninth Int'l Conf. Hybrid Intelligent Systems (HIS'09)*, Shenyang, China, pp. 411–416. DOI: 10.1109/HIS.2009.319.
12. **Back, T. (1994).** Selective pressure in evolutionary algorithms: a characterization of selection mechanisms. *Proc. of Conference on Evolutionary Computation, IEEE World Congress on*

Computational Intelligence (ICEC94), pp. 57–62. DOI: 10.1109/ICEC.1994.350042.

13. **Blickle, T. & Thiele, L. (1995).** A mathematical analysis of tournament selection. **L. Eshelman** (ed.) *Proc. of Sixth International Conf. Genetic Algorithms (ICGA95)*, Morgan Kaufmann, San Francisco, CA, pp. 9–16.
14. **Miller, B.L. & Goldberg, D.E. (1995).** Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, 9, pp. 193–212.
15. **Muhlenbein, H. & Voosen, D.S. (1993).** Predictive models for the breeder genetic algorithm. *Evolutionary Computation*, Vol. 1, No. 1, pp. 25–49.
16. **Blickle, T. & Thiele, L. (1995).** *A comparison of selection schemes used in genetic algorithm.* TIK-Report. Swiss Federal Institute of Technology, Computer Engineering and Communication Networks Lab, Switzerland.
17. **Kuo, T. & Hwang, S.Y. (1996).** A genetic algorithm with disruptive selection. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 26, No. 2, pp. 299–307. DOI: 10.1109/3477.485880.
18. **Kuo, T. & Hwang, S.Y. (1997).** Using disruptive selection to maintain diversity in genetic algorithms. *Applied Intelligence*, Vol. 7, pp. 257–267. DOI: 10.1023/A:1008276600101.
19. **Sivanandam, S.N. & Deepa, S.N. (2008).** *Introduction to Genetic Algorithms*. Springer Berlin Heidelberg, New York, 71 p.
20. **Srinivas, M. & Patnaik, L.M. (1994).** Genetic algorithms: A survey. *Computer*, Vol. 27, No. 6, pp. 17–26. DOI: 10.1109/2.294849.
21. **Miller, B.L. & Goldberg, D.E. (1996).** Genetic algorithms, selection schemes, and the varying effects of noise. *Evolutionary Computation*, Vol. 4, No. 2, pp. 113–131. DOI: 10.1162/evco.1996.4.2.113.

Ajit Kumar received the M.Tech. (Information Technology) from Guru Gobind Singh Indraprastha University, Delhi (India). He is pursuing the Ph.D. (Computer Science and Engg.) at Deenbandhu Chhotu Ram University of Science and Technology, Murthal (India). His research interests include artificial intelligence, data mining, and data warehousing.

Dharmender Kumar received his Ph.D. from Guru Jambheshwar University of Science and Technology, Hisar (India). He is Associate Professor in Computer Science and Engg. at Guru Jambheshwar University of Science and Technology, Hisar. He has to his credit a number of research papers in international journals and conferences. His research interests include data mining, data warehousing, swarm intelligence, and quality of service.

S.K. Jarial received his Ph.D. from Deenbandhu Chhotu Ram University of Science and Technology, Murthal (India). He is Associate Professor in Mechanical Engg. at Deenbandhu Chhotu Ram University of Science and Technology, Murthal (India). He has to his credit a number of research papers in international journals and conferences. His research interests include quality of service, data mining, software testing, and software engineering.

Article received on 13/04/2015; accepted on 16/12/2015.
Corresponding author is Ajit Kumar.