



International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

www.iasir.net

Enhanced Local Search in Artificial Bee Colony Algorithm

¹Sandeep Kumar, ²Dr. Vivek Kumar Sharma, ³Rajani Kumari

Faculty of Engineering and Technology

Jagannath University

Chaksu, Jaipur-303901

INDIA

Abstract: Artificial Bee Colony (ABC) algorithm is a Nature Inspired Algorithm (NIA) which based in intelligent food foraging behaviour of honey bee swarm. ABC outperformed over other NIAs and other local search heuristics when tested for benchmark functions as well as factual world problems but occasionally it shows premature convergence and stagnation due to lack of balance between exploration and exploitation. This paper establishes a local search mechanism that enhances exploration capability of ABC and avoids the dilemma of stagnation. With help of recently introduces local search strategy it tries to balance intensification and diversification of search space. The anticipated algorithm named as Enhanced local search in ABC (EnABC) and tested over eleven benchmark functions. Results are evidence for its dominance over other competitive algorithms.

Keywords: artificial bee colony algorithm; nature inspired algorithm; local search; memetic computing; swarm intelligence

I. Introduction

Artificial bee colony Optimization algorithm is one of the trendy swarm intelligence optimization algorithm proposed by D. Karaboga [1]. These algorithms are used to come across a set of values for the non-aligned variables that optimizes the value of one or more leaning variables. There are number of multivariable optimization problems with haphazardly elevated dimensionality which cannot be solved by exact search algorithms in evaluated time. So search algorithms capable of searching near-optimal or good solutions within up to standard computation time are very useful in factual life. In last a small number of years, the community of scientists and researchers has noticed the significance of a large number of nature-inspired metaheuristics and hybrids of these nature-inspired optimization algorithms. Metaheuristics are identified as a general algorithmic structure that can be employed to different optimization problems less number of modifications to adapt them for a pinpointed problem. Metaheuristics are deliberated to extend the capabilities of heuristics by combining one or more heuristic methods (referred to as procedures) using a higher-level methodologies (hence 'meta'). Metaheuristics are methods that provide direction to the search process. Hyper-heuristics are yet another augmentation that focuses on heuristics that redefine their parameters (either online or offline) to enhance the reliability of end result, or the effectiveness of the computation process. Hyperheuristics provide high-level strategies that may employ machine learning and adapt their search behavior by modifying the relevance of the sub-procedures or even which procedures are used (operating on the space of heuristics which in turn control within the problem domain) [2]. Algorithms from the meadows of Computational intelligence, biologically motivated intelligent computing, and Metaheuristics are applied to strenuous problems, to which more classical approaches may not be applicable. Michalewicz and Fogel point towards that these problems are difficult [3] as: they has large number of feasible solutions in the search space due to which they not able to exploit the best results; The problem is so roundabout, that just to facilitate any conclusion at all, one need to use such librated models of the problem that any result is essentially a waste of time; the evaluation function that describes the quality of any proposed solution is noisy or varies with time, thereby requiring not just a single solution but an entire series of solutions; the achievable solutions are so knowingly constrained that generating even on its own unobjectionable answer is very hard-hitting job, let alone searching for an optimum solution; the human being solving the problem is deficiently composed or assumes some psychological fencing that prevents them from identifying precise solution.

Nature inspired algorithms are stimulated by some natural discernible fact, can be classified as per their source of inspiration. Major categories of NIA are: Stochastic Algorithms, Evolutionary Algorithms, Swarm Algorithms, Physical Algorithms, Probabilistic Algorithms, Immune Algorithms and Neural Algorithms [2].

Stochastic Algorithms are algorithms that focus on the introduction of randomness into heuristic methods. Examples of stochastic algorithms are Random Search, Adaptive Random Search, Stochastic Hill Climbing, Guided Local Search, Iterated Local Search, Variable Neighborhood Search, Greedy Randomized Adaptive, Tabu Search, Reactive Tabu Search and Scatter Search [2].

Evolutionary Algorithms are inspired by encroachment by means of innate assortment. Examples of evolutionary algorithm are Genetic Algorithm, Genetic Programming, Learning Classifier System, Evolution Strategies, Differential Evolution, Evolutionary Programming, Grammatical Evolution, Gene Expression Programming, Non-dominated Sorting Genetic Algorithm, and Strength Pareto Evolutionary Algorithm [2].

Swarm Algorithms focus on strategies that exploit the properties of cooperative intelligence. Some examples of swarm intelligence are Particle Swarm Optimization, Bees Algorithm, Ant System, Ant Colony System and Bacterial Foraging Optimization Algorithm [2].

Physical Algorithms are inspired by corporal and societal systems like Simulated Annealing, Extremal Optimization, Harmony Search, Cultural Algorithm, and Memetic Algorithm [2].

Probabilistic Algorithms are strategies that focus on methods that build models and educated guess distributions in search domains. For example; Population-Based Incremental Learning, Univariate Marginal Distribution Algorithm, Compact Genetic Algorithm, Bayesian Algorithm, Cross-Entropy Method [2].

Immune Algorithms are encouraged by the adaptive immune system of vertebrates like Clonal Selection Algorithm, Negative Selection Algorithm, Artificial Immune Recognition System, Immune Network Algorithm and Dendritic Cell Algorithm [2].

Neural Algorithms are inspired by the plasticity and learning qualities of the human nervous system. Some well known neural algorithms are Perceptron, Back-propagation, Hopfield Network, Learning Vector Quantization and Self-Organizing Map [2].

Real-world optimization problems and sweeping statements thereof can be strained from the largest part of science, management, engineering, and information technology. Importantly, function optimization problems have had a long tradition in the fields of Artificial Intelligence in motivating basic research into new problem solving strategies, and for finding and verifying complete behavior against benchmark problem instances [1]. The enhanced local search strategy proposed in this paper is based on one of the youngest constituent of NIA family the Artificial Bee Colony (ABC) algorithm [1], which mimics the extra ordinary food foraging behavior of most intelligent creepy-crawly that is honey bees swarm. ABC is a swarm-based intelligent stochastic optimization algorithm that has shows considerable concert in solving continuous [4]-[6], combinatorial [7]-[10] and many more complex optimization problems. Stochastic optimization algorithms are those that use unpredictability to encourage non-deterministic characteristics, contrasted to entirely deterministic strategies. Most strategies from the fields of biologically motivated Computation, Computational Intelligence and Metaheuristics may be evaluated to belong the field of Stochastic Optimization [2].

The organization of rest of paper is as follow: In section 2, it initiate Artificial Bee Colony Algorithm in detailed manner, one of the most up-to-date swarm based system introduces by D. Karaboga [1]. Next section discusses some recent and important improvement and modifications in ABC. Section 4 introduces the proposed EnABC algorithm. Section 5 contains experimental setup and results followed by conclusion and references.

II. Artificial Bee Colony Algorithm

The ABC algorithm mimics the extra ordinary food foraging behavior of the honey bee insects with three groups of bees: employed bees, onlookers and scouts. A honey bee working to forage a food source (i.e. solution) previously visited by itself and searching only around its locality is called an employed bee. Employed bees accomplish waggle dance after coming back to the hive to barter the information of its food source to the rest of the colony. A bee waiting in close proximity to the dance floor to designate any of the employed bees to follow is known an onlooker. A bee randomly searching a search space for finding a food source is called a scout. For every food source, there is only one employed bee and a number of follower bees. The scout bee, after finding some food source better than some threshold value, what's more performs a waggle dance to share this information. In the ABC algorithm implementation, half of the colony consists of employed bees and the other half constitutes the onlookers. The number of food sources (i.e., solutions being exploited) is equal to the number of employed bees in the colony. The employed bee whose food source (i.e. solution) is exhausted (i.e. the solution has not been improved after several attempts) becomes a scout. The detailed algorithm is given below:

Step 1: Initialize population of N uniformly distributed individuals. Each individual x_{ij} is a food source and has D attributes. D is the dimensionality of the problem. x_{ij} represent i^{th} solution in j^{th} dimension. Where $j \in \{1, 2, \dots, D\}$

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

Step 2: Estimate the fitness of each individual solution using the following method,
if (solution value ≥ 0)

$$\text{then } fit_i = \frac{1}{(2 * \text{solution value} + 1)}$$

else

$$fit_i = (1 + fabs(\frac{1}{\text{solution value}}))$$

Step 3: Each employed bee, placed at a food source that is different from others, search in the proximity of its current position to find a better food source. For each employed bee, generate a new solution, v_i around its current position, x_i using the following formula.

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

Here, $k \in \{1, 2, \dots, N\}$ and $j \in \{1, 2, \dots, D\}$ are randomly chosen indices. N is number of employed bees. ϕ_{ij} is a uniform random number from $[-1, 1]$.

Step 4: Compute the fitness of both x_i and v_i . Apply greedy selection strategy to select better one of them.

Step 5: Calculate and normalize the probability values, P_i for each solution x_i using the following formula.

$$P_{ij} = \frac{fit_i}{\sum_{i=1}^{SN} fit_i}$$

Step 6: Assign each onlooker bee to a solution, x_i at random with probability proportional to P_i .

Step 7: Generate new food positions (i.e. solutions), v_i for each onlooker bee.

Step 8: Compute the fitness of each onlooker bee, x_i and the new solution, v_i . Apply greedy selection process to keep the fitter one and abandon other.

Step 9: If a particular solution x_i has not been improved over a predefined number of cycles, then select it for rejection. Replace the solution by placing a scout bee at a food source generated evenly at random within the search space using

$$x_{ij} = x_{min j} + rand[0,1](x_{max j} - x_{min j}) \text{ for } j = 1, 2, \dots, D$$

Step 10: Keep track of the best food sources (solution) found so far.

Step 11: Check termination criteria. If the best solution found is acceptable or reached the maximum iterations, stop and return the best solution found so far. Otherwise go back to step 2 and repeat again.

III. Modifications in Artificial Bee colony Algorithm

Usually real world dispense some complex optimization problems that cannot be easily treated by available mathematical optimization methods. If the user is not very aware about the exact solution of the problem in hand then intelligence emerged from social behavior of social colony members may be used to solve these kinds of problems. Honey bees are in the category of social insects. The foraging behavior of honey bees produces an intelligent social behavior, called as swarm intelligence. This swarm intelligence is simulated and an intelligent search algorithm namely, Artificial Bee Colony (ABC) algorithm is established by Karaboga in 2005 [1]. Since its inception, a lot of research has been carried out to make ABC more and more efficient and to apply ABC for different types of problems.

In order to get rid of the drawbacks of original ABC, researchers and scientists have modified ABC in many ways. The potentials where ABC can be improved are fine tuning of ABC control parameters SN, ϕ_{ij} and limit (maximum cycle number). Hybridization of ABC with other population based probabilistic or deterministic algorithms. New control parameters also introduced in different phases of ABC. D. Karaboga [1] has suggested that the value of ϕ_{ij} should be in the range of $[-1, 1]$. The value of limit (maximum cycle number) should be $SN \times D$, where, SN is the number of solutions and D is the dimension of the problem. Wei-feng Gao et al. [11] proposed an enhanced solution search equation in ABC, which is based on the fact that bee searches only around the best solution of the previous iteration to increase the exploitation. A. Banharnsakun et al. [12] introduced a new variant of ABC namely the best-so-far selection in artificial bee colony algorithm. To enhance the exploitation and exploration processes, they propose to make three major changes by introducing the best-so-far method, an adjustable search radius, and an objective-value-based comparison method in DE. J.C. Bansal et al. [83] proposed balanced ABC; they introduced a new control parameter, Cognitive Learning Factor and also modified range of ϕ in Artificial Bee Colony algorithm. Qingxian and Haijun proposed a modification in the initialization scheme by making the initial group symmetrical, and the Boltzmann selection mechanism was employed instead of roulette wheel selection for improving the convergence ability of the ABC algorithm [13].

In order to maximize the exploitation capacity of the onlooker stage, Tsai et al. introduced the Newtonian law of universal gravitation in the onlooker phase of the basic ABC algorithm in which onlookers are selected based on a roulette wheel (Interactive ABC, IABC) [14]. Baykasoglu et al. incorporated the ABC algorithm with shift neighborhood searches and greedy randomized adaptive search heuristic and applied it to the generalized assignment problem [15]. Furthermore, modified versions of the Artificial Bee Colony algorithm are introduced and applied for efficiently solving real-parameter optimization problems by Bahriye Akay and Dervis Karaboga [16]. To modify ABC behavior for constrained search space Mezura et al. [17] proposed four modifications related with the selection strategy, the scout bee operator, and the equality and boundary constraints. Instead of fitness proportional selection, tournament selection is performed to exploit employed bee food sources by onlooker bees. Second, they employed dynamic tolerance for equality constraints. In 2010, Zhu and Kwong [18] proposed an improved ABC algorithm called gbest-guided ABC (GABC) algorithm by incorporating the information of global best (gbest) solution into the solution search equation to improve the exploitation. GABC is inspired by PSO [19], which, in order to improve the exploitation capability, takes advantage of the information of the global best (gbest) solution to guide the search by candidate solutions. J.C. Bansal et al. [24] introduced

memetic search in ABC algorithm in order to balance exploitation and exploration. In 2010, Derelia and Das [20] proposed a hybrid bee(s) algorithm for solving container loading problems. In the proposed algorithm, a bee(s) algorithm is hybridized with the heuristic filling procedure for the solution of container loading problems. In 2010, Huang and Lin [21] proposed a new bee colony optimization algorithm with idle-time-based filtering scheme and its application for open shop-scheduling problems. They categorized the foraging behaviors of bees in two terms Forward Pass and Backward Pass. Forward Pass expresses the process of a forager bee leaving the bee hive and flying towards a food source while Backward Pass denotes the process of a forager bee returning to the bee hive and sharing the food source information with other forager bees (role change). In 2011, Nambiraj Suguna et al. [22] proposed an independent rough set approach hybrid with artificial bee colony algorithm for dimensionality reduction. In the proposed work, effects of the perturbation rate, the scaling factor (step size), and the limit are investigated on real-parameter optimization. ABC algorithm hybridized with genetic algorithm to balance exploration and exploitation of search space [25], [26]. In 2012, Bin Wu et al. [23] proposed improvement of Global swarm optimization (GSO) hybrid with ABC and PSO. They use neighborhood solution generation scheme of ABC and accept new solution only when it is better than previous one to improve GSO performance. A detailed discussion on the performance and modification in ABC algorithm is available in literature [27], [28].

IV. Enhanced Local Search in ABC(EnABC)

The proposed enhanced local search in ABC adds an additional step for local search inspired by MeABC [24]. The proposed algorithm introduces memetic search strategies as local search. The MeABC algorithm developed by J. C. Bansal et al [24] influenced by Golden Section Search (GSS) [31]. In MeABC only the good particle of the current swarm updates itself in its neighborhood. Original GSS method does not use any gradient information of the function to find the optima of a uni-modal continuous function. GSS processes the interval $[a = -1.2, b = 1.2]$ and initiates two intermediate points (f_1, f_2) with help of golden ratio ($\Psi = 0.618$):

$$f_1 = b - (b - a) \times \psi \text{ and } f_2 = a + (b - a) \times \psi$$

The proposed strategy use interval $[a=-2, b=2]$ for GSS process in order to increase searching capability. The detailed enhanced local search in ABC algorithm is given below:

Step 1: initialize the population of N uniformly distributed individuals. Each individual x_{ij} is a food source and has D attributes. D is the dimensionality of the problem. x_{ij} represent i^{th} solution in j^{th} dimension. Where $j \in \{1, 2, \dots, D\}$

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

Step 2: Estimate the fitness of each individual solution using the following method,
if (solution value ≥ 0)

$$\text{then } fit_i = \frac{1}{(2 * \text{solution value} + 1)}$$

else

$$fit_i = (1 + \text{fabs}(\frac{1}{\text{solution value}}))$$

Step 3: Each employed bee, placed at a food source that is different from others, search in the proximity of its current position to find a better food source. For each employed bee, generate a new solution, v_{ij} around its current position, x_{ij} using the following formula.

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

Here, $k \in \{1, 2, \dots, N\}$ and $j \in \{1, 2, \dots, D\}$ are randomly chosen indices. N is number of employed bees. ϕ_{ij} is a uniform random number from $[-1, 1]$.

Step 4: Compute the fitness of both x_{ij} and v_{ij} . Apply greedy selection strategy to select better one of them.

Step 5: Calculate and normalize the probability values, P_{ij} for each solution x_i using the following formula.

$$P_{ij} = \phi \frac{fit_i}{\text{Maximum_fitness}} + (1 - \phi) \frac{fit_i}{\sum_{i=1}^N fit_i}$$

Here $\phi = 0.9$.

Step 6: Assign each onlooker bee to a solution, x_i at random with probability proportional to P_{ij} .

Step 7: Generate new food positions, v_{ij} for each onlooker bee.

Step 8: Compute the fitness of each onlooker bee, x_{ij} and the new solution, v_{ij} . Apply greedy selection process to keep the fitter one and abandon other.

Step 9: Memetic search phase inspired by GSS.

Repeat while termination criteria meet

$$\text{Compute } f_1 = b - (b - a) \times \psi, \text{ and } f_2 = a + (b - a) \times \psi,$$

Calculate $f(f_1)$ and $f(f_2)$
 If $f(f_1) < f(f_2)$ then
 $b = f_2$ and the solution lies in the range $[a, b]$
 else
 $a = f_1$ and the solution lies in the range $[a, b]$
 End of if

End of while

Step 10: If a particular solution x_{ij} has not been improved over a predefined number of cycles, then select it for rejection. Replace the solution by placing a scout bee at a food source generated evenly at random within the search space using

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

for $j = 1, 2, \dots, D$

Step 11: Keep track of the best food sources (solution) found so far.

Step 12: Check extinction criteria. If the best solution found is adequate or reached the maximum iterations, stop and return the best solution found so far. Otherwise go back to step 2 and repeat all over again.

Table I Test Problems

Test Problem	Objective Function	Search Range	Optimum Value	D	Acceptable Error
Griewank	$f_1(x) = \frac{1}{4000} \left(\sum_{i=1}^D (x_i^2) \right) - \left(\prod_{i=1}^D \cos \left(\frac{x_i}{\sqrt{i}} \right) \right) + 1$	[-600, 600]	$f(0) = 0$	30	1.0E-05
Rastrigin	$f_2(x) = \sum_{i=1}^D [x_i^2 - 10 \cos(2\pi x_i) + 10]$	[-5.12, 5.12]	$f(0) = 0$	30	1.0E-05
Zakharov	$f_3(x) = \sum_{i=1}^D x_i^2 + \left(\sum_{i=1}^D \frac{i x_i}{2} \right)^2 + \left(\sum_{i=1}^D \frac{i x_i}{2} \right)^4$	[-5.12, 5.12]	$f(0) = 0$	30	1.0E-02
Salomon Problem	$f_4(x) = 1 - \cos(2\pi \sqrt{\sum_{i=1}^D x_i^2}) + 0.1 (\sqrt{\sum_{i=1}^D x_i^2})$	[-100, 100]	$f(0) = 0$	30	1.0E-01
Colville function	$f_5(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1)$	[-10, 10]	$f(1) = 0$	4	1.0E-05
Kowalik function	$f_6(x) = \sum_{i=1}^{11} \left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right)^2$	[-5, 5]	$f(0.1928, 0.1908, 0.1231, 0.1357) = 3.07E-04$	4	1.0E-05
Shifted Rosenbrock	$f_7(x) = \sum_{i=1}^{D-1} (100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2 + f_{bias})$ $z = x - o + 1, x = [x_1, x_2, \dots, x_D], o = [o_1, o_2, \dots, o_D]$	[-100, 100]	$f(o) = f_{bias} = 390$	10	1.0E-01
Six-hump camel back	$f_8(x) = (4 - 2.1x_1^2 + \frac{1}{3}x_1^4)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$	[-5, 5]	$f(-0.0898, 0.7126) = -1.0316$	2	1.0E-05
Easom's function	$f_9(x) = -\cos x_1 \cos x_2 e^{-(x_1 - \pi)^2 - (x_2 - \pi)^2}$	[-10, 10]	$f(\pi, \pi) = -1$	2	1.0E-13
McCormick	$f_{10}(x) = \sin(x_1 + x_2) + (x_1 - x_2)^2 - \frac{3}{2}x_1 + \frac{5}{2}x_2 + 1$	$-1.5 \leq x_1 \leq 4, -3 \leq x_2 \leq 3,$	$f(-0.547, 1.547) = -1.9133$	30	1.0E-04
Meyer and Roth Problem	$f_{11}(x) = \sum_{i=1}^5 \left(\frac{x_1 x_3 t_i}{1 + x_1 t_i + x_2 v_i} - y_i \right)^2$	[-10, 10]	$f(3.13, 15.16, 0.78) = 0.4E-04$	3	1.0E-03

V. Experimental Setting, Results and Discussion

A. Test problems under consideration

In order to analyze the performance of EnABC different global optimization problems (f_1 to f_{11}) are elected. These are continuous optimization problems and have different degrees of complexity and multimodality. Test problems are taken from [29], [30] with the associated offset values.

Table II Comparison of the Results of Test Problems

Test Function	Algorithm\Measure	SD	ME	AFE	SR
f_1	ABC	9.10E-03	4.61E-03	81988.8	61
	MeABC	1.41E-03	2.53E-04	39138.74	97
	EnABC	1.03E-03	1.56E-04	79913.78	98
f_2	ABC	1.43E+00	3.80E+00	99396.6	4
	MeABC	1.63E-06	8.55E-06	42565.6	100
	EnABC	2.74E-06	7.04E-06	41491.4	100
f_3	ABC	1.49E+01	9.92E+01	100020	0
	MeABC	7.87E-03	2.01E-02	99739.44	5
	EnABC	5.25E-04	9.57E-03	116096.6	100
f_4	ABC	2.30E-01	1.67E+00	100020.1	0
	MeABC	3.47E-02	9.18E-01	20766.84	100
	EnABC	3.10E-02	9.20E-01	26623.07	100
f_5	ABC	1.36E-01	1.73E-01	98300.08	2
	MeABC	2.65E-03	7.07E-03	35764.62	97
	EnABC	1.40E-02	1.38E-02	109138.1	65
f_6	ABC	8.04E-05	1.69E-04	86379.58	32
	MeABC	5.37E-05	9.90E-05	57026.2	86
	EnABC	6.40E-06	5.22E-06	28039.09	87
f_7	ABC	9.79E+00	6.47E+00	97354.3	4
	MeABC	1.71E+00	8.71E-01	74392.64	50
	EnABC	4.12E+00	2.24E+00	178179.3	19
f_8	ABC	1.06E-05	1.73E-06	78185.92	30
	MeABC	4.24E-15	4.53E-15	5830.82	100
	EnABC	1.41E-09	2.01E-10	95473.5	76
f_9	ABC	4.02E-03	4.67E-03	100045.5	0
	MeABC	1.39E-05	1.55E-05	54525.6	51
	EnABC	1.37E-05	1.68E-05	52285.8	52
f_{10}	ABC	2.46E-02	2.68E-02	100033	0
	MeABC	1.33E-05	9.13E-06	52908.5	74
	EnABC	7.65E-06	6.16E-06	80974.28	87
f_{11}	ABC	4.16E-02	3.68E-02	100032.8	0
	MeABC	1.20E-05	9.23E-05	39390.54	88
	EnABC	4.93E-05	1.16E-04	126948.7	52

B. Experimental setting

To prove the efficiency of EnABC, it is compared with original ABC and MeABC. To test Enhanced local search in ABC over considered problems, following experimental setting is adopted:

- Colony size $N = 60$,
- $\phi_{ij} = \text{rand}[-1, 1]$,

- Number of food sources $S_N = N/2$,
- limit = 1500,
- The stopping touchstone is either maximum number of function evaluations (which is set to be 200000) is reached or the acceptable error (outlined in Table II) has been achieved,
- The number of simulations/run = 100,
- Parameter settings for the original ABC algorithm and MeABC algorithm are same as EnABC.

C. Results Comparison

Mathematical results of EnABC with experimental setting as per subsection 5.B are outlined in Table II. Table II show the relationship of results based on standard deviation (SD), mean error (ME), average function evaluations (AFE) and success rate (SR) are reported. Table II shows that most of the time EnABC outperforms in terms of efficiency (with less number of function evaluations) and reliability as compare to other considered algorithms. The proposed algorithm all the time improves AFE and most of the time it also improve SD and ME. It is due to randomness introduced during fitness calculation and probability calculation. Table III contains summary of table II outcomes. In Table III, '+' indicates that the EnABC 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 III, establishes the superiority of EnABC over MeABC and ABC.

Table III Summary of table II outcome

Function	EnABC vs ABC	EnABC vs MeABC
f_1	+	+
f_2	+	+
f_3	+	+
f_4	+	-
f_5	+	-
f_6	+	+
f_7	+	-
f_8	+	-
f_9	+	+
f_{10}	+	+
f_{11}	+	-
Total Number of + sign	11	6

VI. CONCLUSION

This paper, established a new phase in original ABC based on modified GSS process. Newly introduced step added just after onlooker bee phase. Proposed algorithm also modifies search range of GSS process in order to balance intensification and diversification of local search space. Further, the modified strategy is applied to solve 11 well-known standard benchmark functions. With the help of experiments over test problems, it is shown that the insertion of the proposed local search strategy in the original ABC algorithm improves the steadfastness, efficiency and accuracy as compare to their original version. Table II and III show that the proposed EnABC is able to solve almost all the considered problems with fewer efforts.

REFERENCES

- [1] D. Karaboga, "An idea based on honey bee swarm for numerical optimization", Technical Report-TR06, Erciyes Univ., Computer Engg. Department, 2005.
- [2] Brownlee, *Clever algorithms: nature-inspired programming recipes*. Jason Brownlee, 2011.
- [3] Z. Michalewicz and D. B. Fogel. *How to Solve It: Modern Heuristics*. Springer, 2004.
- [4] D. Karaboga, B. Basturk, *Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems*, LNCS: Advances in Soft Computing: Foundations of Fuzzy Logic and Soft Computing, CA, Vol: 4529/2007, pp: 789-798, Springer- Verlag, 2007, IFSA 2007.
- [5] D. Karaboga, B. Basturk, "An artificial bee colony (ABC) algorithm for numeric function optimization", in *Proceedings of the IEEE Swarm Intelligence Symposium 2006*, Indianapolis, Indiana, USA, 12-14 May 2006.
- [6] X. S. Yang, "Engineering Optimizations via Nature-Inspired Virtual Bee Algo.", *IWINAC2005*, LNCS 3562, J. M. Yang and J.R. Alvarez (Eds.), Springer-Verlag, Berlin Heidelberg, pp. 317-323, 2005.
- [7] P. Lucic, D. Teodorovic, "Computing with Bees: Attacking Complex Transportation Engineering Problems", in *International Journal on Artificial Intelligence Tools*, vol. 12, no. 3, pp. 375-394, 2003.
- [8] C. S. Chong, M. Y. H. Low, A. I. Sivakumar, K. L. Gay, "A Bee Colony Optimization Algorithm to Job Shop Scheduling", *Proc. of the 37th Winter Simulation*, California, pp. 1954-1961, 2006.
- [9] G. Z. Markovic, D. Teodorovic, V. S. Acimovic-Raspopovic, "Routing and Wavelength Assignment in All-Optical Networks Based on the Bee Colony Optimization", *AI Communications*, v.20 no.4, p.273-285, December 2007.
- [10] N. Quijano, K. M. Passino, "Honey Bee Social Foraging Algorithms for Resource Allocation Theory and Application", *American Control Conference*, New York City, USA, 2007.

- [11] W. Gao and S. Liu. A modified artificial bee colony algorithm. *Computers & Operations Research*, 2011.
- [12] A. Banharnsakun, T. Achalakul, and B. Sirinaovakul. The best-so-far selection in artificial bee colony algorithm. *Applied Soft Computing*, 2010.
- [13] D. Haijun F. Qingxian. Bee colony algorithm for the function optimization. *Science Paper Online*, August 2008.
- [14] P.W. Tsai, J.S. Pan, B.Y. Liao, and S.C. Chu. Enhanced artificial bee colony optimization. *International Journal of Innovative Computing, Information and Control*, 5(12), 2009.
- [15] Baykasoglu, L. Ozbakir, and P. Tapkan. Artificial bee colony algorithm and its application to generalized assignment problem. *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization*, pages 113–144, 2007.
- [16] B. Akay and D. Karaboga. A modified artificial bee colony algorithm for real-parameter optimization. *Information Sciences*, 2010.
- [17] E. Mezura-Montes and O. Cetina-Domínguez. Empirical analysis of a modified artificial bee colony for constrained numerical optimization. *Applied Mathematics and Computation*, 2012.
- [18] G. Zhu and S. Kwong. Gbest-guided artificial bee colony algorithm for numerical function optimization. *Applied Mathematics and Computation*, 2010.
- [19] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proceedings IEEE int'l conf. on neural networks Vol. IV*, pages 1942–1948, 1995.
- [20] T. Derelia and G.S. Dasb. A hybrid'bee (s) algorithm'for solving container loading problems. *Applied Soft Computing*, 2010.
- [21] Y.M. Huang and J.C. Lin. A new bee colony optimization algorithm with idle-time-based filtering scheme for open shop-scheduling problems. *Expert Systems with Applications*, 2010.
- [22] N. Suguna and K.G. Thanushkodi. An independent rough set approach hybrid with artificial bee colony algorithm for dimensionality reduction. *American Journal of Applied Sciences*, 8, 2011.
- [23] B. Wu, C. Qian, W. Ni, and S. Fan. The improvement of glowworm swarm optimization for continuous optimization problems. *Expert Systems with Applications*, 2012.
- [24] J.C. Bansal, H. Sharma, K.V. Arya and A. Nagar, "Memetic search in artificial bee colony algorithm." *Soft Computing* (2013): 1-18.
- [25] 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, 2013.
- [26] 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.
- [27] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga. "A comprehensive survey: artificial bee colony (ABC) algorithm and applications." *Artificial Intelligence Review* (2012): 1-37.
- [28] 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.
- [29] M.M. Ali, C. Khompatraporn, and Z.B. Zabinsky. "A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems." *Journal of Global Optimization*, 31(4):635–672, 2005.
- [30] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, YP Chen, A. Auger, and S. Tiwari. "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization." In *CEC 2005*, 2005.
- [31] J. Kiefer, "Sequential minimax search for a maximum." In *Proc. Amer. Math. Soc*, volume 4, pages 502–506, 1953.
- [32] S. Rahnamayan, H.R. Tizhoosh, and M.M.A. Salama. Opposition-based differential evolution. *Evolutionary Computation, IEEE Transactions on*, 12(1):64–79, 2008.