Enhancing the Local Exploration Capabilities of Artificial Bee Colony using Low Discrepancy Sobol Sequence

Tentu Monica[1,1], Anguluri Rajasekhar[2], Millie Pant[3], Ajith Abraham[4],

[1,1] Dept of Information Technology, SSJ Engineering College, Hyderabad, INDIA
[2]Dept of Electrical Engineering, National Institute of Technology, Warangal, INDIA
[3]Dept of Paper and Pulp Technology, Indian Insitute of Technology, Roorke, INDIA
[4]Director of Machine Intelligence Research Laboratoires (MIR LABS), USA
mounika.ssj@gmail.com, rajasekhar.anguluri@ieee.org, millifpt@iit.ernet.in, ajith.abraham@ieee.org

Abstract. In this paper we propose a mechanism for enhancing the performance of the Artificial Bee Colony Algorithm (ABCA) by making use low discrepancy Sobol sequence. The performance of the proposed Sobol sequence guided ABC (S-ABC) is analyzed over several benchmark functions and also compared to that of basic ABC. The empirical results show that the presence of low discrepancy sequence like that of Sobol, significantly improves the performance of the basic ABCA.

Keywords: artificial bees; quasi random sequences; continuous functions, global optimization

1 Introduction

With the growing complexities of real world application problems, researchers are concentrating on general purpose algorithms that can be applied to a wide and diverse range of problems efficiently. Some of the algorithms that have gained popularity in the last few decades because of their wide applicability and simple structure include Genetic Algorithms (GA) [1], Particle Swarm Optimization [2], Differential Evolution (DE) [3] etc. A latest addition to the family of such algorithms is Artificial Bee Colony (ABC) algorithm, which is a swarm intelligence algorithm based on the foraging behavior of honey bees.

ABC was proposed by Karaboga and Bastruk for optimizing various numerical functions [4, 5] in 2005. Due to its simplicity and robustness it has been successfully applied to various practical optimization problems like Clustering [6]; IIR filter design [7]; extraction of MESFET [8] and some of the modifications for improvising the performance are provided in [9, 10, 11, 12, 13];

In basic ABC, each virtual bee updates its trajectory towards the nectar, based on the information provided by onlooker bees via *waggle dance*. This is done by roulette wheel selection criterion. Finally, a '*scout*' is employed so that there is no scope of local optimum. In literature it had been proved through various instances that ABC provides promising solutions in less computational time when compared to traditional methods like GAs and other soft computing tools, but the quality of solution don't improve at the higher iteration count and remains stagnated causing premature convergence leading to an intermediate optimal solution. This drawback is mainly due to the lack of a proper path which drives the colony of bees to converge to a solution found (for a pre-determined termination criterion) which may not be a global solution. Thus a mechanism is needed to enhance the local exploration capabilities of ABC.

One of the best methods for maintaining the good trajectory for the population is introducing the concept of mutation in the optimization of algorithm (Mutation: an intelligent approach plagiarized from the Evolutionary Algorithms, which is mainly used for the perturbation of population from getting it stuck in a local optimum). Almost all the mutation operators defined in the past make use of random probability distribution (eg., Gausssian Mutation, Cauchy Mutation).

In this work we propose the use of a quasi-random, Sobol sequence to guide the movement of scout bees. Although, low discrepancy sequences like that of Sobol have been applied to various general purpose algorithms [14, 15], where it was observed that the use of these sequences improve the performance of algorithms quite appreciably. However, to the best of our knowledge, the effect of applying low discrepancy sequences to ABC is still unexploited.

The remaining orientation of the paper is as follows; Section 2 gives a brief overview of Quasi Random Sequences (QRS) and Sobol Sequence. Section 3 describes the basic structure of Artificial Bee Colony Algorithm. Section IV describes the proposed algorithm. Section V gives the experimental settings and numerical results of some standard unconstrained benchmark problems. Finally we provided some conclusions towards end.

2 Quasi Random Sequences (QRS)

QRS or low discrepancy sequences are less random than pseudo random number sequences but are more useful for computational methods, which depend on the production of random numbers. Some of these tasks involve approximation of integrals in higher dimension, simulation and global optimization. Some well-known QRS are Vander Corput, Sobol, Halton and Faure. QRS are said to be better than pseudo random sequences, because of their ability to cover the search space more evenly in comparison to pseudo random sequences.

2.1 Discrepancy of a Sequence.

Mathematically, discrepancy of a sequence is the measure of its uniformity, It is computed by comparing the actual number of sample points in a given volume of a multi-dimensional space with the number of sample points that should be there assuming a uniform distribution defined.

For a defined set of points $x^1, x^2, ..., x^N \in I^S$ and a subset of $G \subset I^S$, define a counting function $S_N(G)$ as the number of points $x^i \in \mathrm{G}$. For each $x = (x_1, x_2, ..., x_S) \in I^S$ let G_x be the rectangular **S** dimensional region, such that $G_x = [0, x_1) \text{ x } [0, x_2) \text{ x....x } [0, x_S)$, with volume $\mathrm{x}_1\mathrm{x}_2...\mathrm{x}_\mathrm{N}$. Then the discrepancy of points is given by $D_N^*(x^1, x^2, x^3.....x^N) = \mathrm{Sup}\,|\,\mathrm{S_N}(\mathrm{G_x}) - \mathrm{N}x_1x_2....x_S\,|\ \ x \in I^S$

2.2 Construction of low-Discrepancy of a Sequence.

The construction of the Sobol sequence [16] uses linear recurrence relations over the finite field, F2, where F2= {0, 1}. Let the binary expansion of the non-negative integer n be given by $n = n_1 2^0 + n_2 2^1 + ... + n_w 2^{w-1.}$. Then the n^{th} element of the j^{th} dimension of the Sobol sequence $X_n^{(j)}$ can be generated by:

$$X_n^{(j)} = n_1 v_1^{(j)} \oplus n_2 v_2^{(j)} .....n_w v_w^{(j)}$$

Where $v_1^{(j)}$ is a binary fraction called the i^{th} direction number in the j^{th} dimension. These direction numbers are generated by the following q-term recurrence relation.

$$v_i^{(j)} = a_1 v_{i-1}^{(j)} \oplus a_2 v_{i-2}^{(j)} \oplus ... \oplus a_q v_{i-q+1}^{(j)} \oplus v_{i-q}^{(j)} \oplus (v_{i-q}^{(j)} / 2^q)$$

We have $i > q$ and the bit a_i comes from the coefficients degree-q primitive polynomial over F2.

3 Artificial Bee Colony Algorithm (ABCA)

ABCA is a swarm intelligent optimization algorithm inspired by honey bee foraging behavior. It classifies the foraging artificial bees into three groups namely *employed bees, onlooker bees* and *scouts*. The first half colony consists of the *employed bees* and second half consists of *onlooker bees*. A bee that is currently searching for food or exploiting a food source is called an *employed bee*. A bee waiting in the hive for making decision to choose a food source is named as an *onlooker*. For every food source, there is only one employed bee and the employed bee of abandoned food source becomes scout. In ABC algorithm, each solution to the problem is considered as *food source* and represented by a D-dimensional real-valued vector, where the fitness of the solution corresponds to the *nectar amount* of associated food resource. ABCA is an iterative process like most of its contemporary search algorithms.

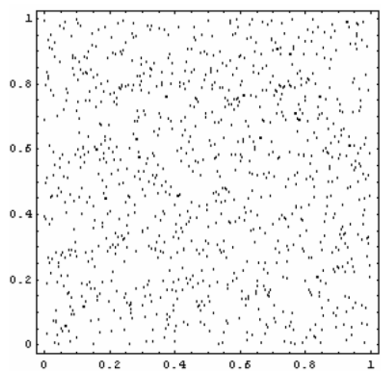

Fig 1(a)

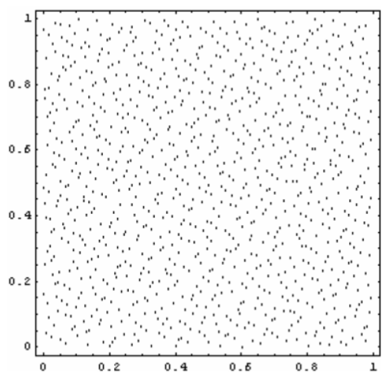

Fig 1 (b)

Fig 1(a) and Fig 1(b). Sample points of pseudo Random and Quasi Random Sequences respectively (reproduced from [14])

The algorithm starts by initializing all *employed bees* with randomly generated food sources (solutions). For each iteration every employed bee finds a food source in the neighborhood of its current food source and evaluates its nectar amount i.e., (*fitness*). In general the position of i_{th} food source is represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. After the information is shared by the employed bees, *onlooker bees* go to the region of food source at X_i based on the probability p_i defined as

$$p_i = \frac{fit_i}{\sum_{k=1}^{FS} fit_k} \tag{1}$$

Where FS is total number of food sources. Fitness value fit_i is calculated by using following equation.

$$fit_i = \frac{1}{1 + f(X_i)} \tag{2}$$

Here $f(X_i)$ is the objective function considered. The onlooker finds its food source in the region of X_i by using following equation

$$x_{new} = x_{ij} + r*(x_{ij} - x_{kj}) \quad (3)$$

If the obtained new fitness value is better than the fitness value achieved so far, than the bee moves to this new food source leaving this old one otherwise it retains its old food source. When all employed bees have completed this process, the information is shared with onlookers. Each of the onlookers selects food source according to probability given above. By this scheme good sources are well accommodated with onlookers. Each bee will search for a better food source for a certain number of cycles (*limit)*, and if the fitness value doesn't improve then that particular bee becomes scout bee. The food source is initialized to that *scout bee* randomly.

Pseudo code for ABC Algorithm

```
1. Initialization
2. Move the employed bees onto their food sources and evaluate their nectar amounts.
3. Place the onlookers depending upon the nectar amounts obtained by employed bees
4. Send the scouts for exploring new food sources.
5. Memorize the best food sources obtained so far.
6. If a termination criterion is not satisfied, go to step 2; otherwise stop the procedure and display the best food source obtained so far
```

4 Sobol Sequence Guided Artificial Bee Colony Algorithm (S-ABC)

The proposed S-ABC algorithm is an extension to the original ABCA by including the concept of Sobol sequence in it. The main point which differentiates between the basic ABCA and S-ABCA is that in ABCA, the scout is assigned a random food location (please note that scout bee is the one which does not show any improvement in fitness value for a certain number of cycles), while in S-ABCA the initialization of food source to the scout is done with the help of Sobol sequence. Now, instead of moving randomly, the scout moves systematically and is able to explore the search space more efficiently. This exploration helps ABC in detecting the food sources (indicating the fitness solutions) effectively.

Pseudo code for ABC Algorithm

```
Step1. Initialize the population of solution $x_{ij}, i = 1,2,...FS, j = 1,2,...D, trail_i = 0$ ; trail is the non-improvement number of the solution abandonment of food source via trial (limit).
Step2. Evaluate the population
Step3. Cycle=1
```

```
Step4.  REPEAT
        {----Produce new food source population for employed bee-----}
Step5.  For $i=1$ to $FS$ do
    i.    Produce a new food source $v_i$ for the employed bee of the food
          source $x_i$ by using (3)
    ii.   Apply a greedy selection process between $v_i$ and $x_i$ and then select
          the better one
    iii.  If solution $x_i$ doesn't improve $trail_i = trail_i + 1$, otherwise $trail_i = 0$;
        End for
Step6.  Calculate the probability values $p_i$ by (1) for the solutions using
    fitness values.
        {----Produce new food source population for onlooker bee-----}
        t=0;
        i=1;
Step7.  REPEAT
    If $rand < P_i$ then
    i.    Produce a new food source $v_i$ for the employed bee of the food
          source $x_i$ by using (3)
    ii.   Apply a greedy selection process between $v_i$ and $x_i$ then select the
          better one
    iii.  If solution $x_i$ doesn't improve $trail_i = trail_i + 1$, otherwise
          $trail_i = 0$;
    iv.   t=t+1
    End if
        UNTIL (t=FS)
      {--------Determine Scout---------}
Step8.  If $max(trail)>limit$ then
    i.    Replace $x_i$ with a new solution produced using quasi random
          sequence
          For d = 1 to dimension D
                  $temp_d$ = Sobolrand( )
          //Sobolrand() is a random number generated by Sobol sequence//
          End For
          $x_{ij} = temp_d$
        End If
    ii.   Memorize the best solution achieved so far
            Cycle=Cycle+1;
        UNTIL (Cycle=Maximum Cycle Number)
```

5 Standard Benchmarks and Results

In the present study we had taken 6 benchmark problems (Table 1). And all the test problems are scalable in nature and except for f_1 (sphere function), each function is multimodal in nature.
Each function is tested for dimension 10, 50, 100 and the size of bee colony is set to 20 for all the dimensions. The maximum number of generations is set as 500, 2000 and 3000 corresponding to the dimension 10, 50 and 100 respectively. Maximum of 25 best runs for each experimental setting are conducted and the average fitness of the best solutions throughout the run is recorded.

The numerical results for different benchmark problems are recorded in Tables 2-4 for dimensions 10, 50 and 100 respectively. From these Tables it is clearly evident that the proposed modification in the behavior of the scout bee, using Sobol sequence significantly improves the performance the basic ABCA.

Convergence graphs of all the functions are illustrated in Fig 2 – Fig 7, which further strengthen the fact that the proposed modification not only improves the solution quality but it also results in faster convergence.

Table 1. Description of Benchmark Functions [17]

Function	Range of Search	Optimum
Sphere	(-100, 100)	$f_1(0)=0$
Rosenbrock	(-100, 100)	$f_2(0)=0$
Rastrigin	(-5.12, 5.12)	$f_3(0)=0$
Griewank	(-600, 600)	$f_4(0)=0$
Ackley	(-32, 32)	$f_5(0)=0$
Schwefel	(-500, 500)	$f_6(0)=0$

Table 2. Comparison of ABC and S-ABC in terms of Error, Standard and fitness (Dim=10)

Fun	Fitness (Dim=10)		Error and (standard deviation)	
	ABC	S-ABC	ABC	S-ABC
f_1	6.96613E-016	**7.87847E-017**	2.77454E-016 (1.16624E-016)	**1.4081E-016 (6.51012E-017)**
f_2	6.86374E+00	**1.07752E+00**	1.51599E+001 (8.93638E+00)	**7.37742E+00 (2.27918E-001)**
f_3	2.84217E-014	**0**	8.12595E-008 (2.41958E-007)	**1.23609E-010 (3.90886E-010)**
f_4	3.82397E-010	**0**	1.13837E-002 (9.03497E-003)	**4.44089E-017 (5.73317E-017)**
f_5	8.53568E-010	**1.74791E-013**	7.08422E-009 (7.11042E-009)	**1.91331E-012 (2.17882E-012)**
f_6	1.59379E-003	**1.27275E-004**	8.35989E+001 (9.84111E+001)	**2.36906E+001 (4.99365E+001)**

Table 3. Comparison of ABC and S-ABC in terms of Error, Standard and fitness (Dim=50)

Fun	Fitness (Dim=50)		Error and (standard deviation)	
	ABC	S-ABC	ABC	S-ABC
f_1	3.0736E-014	**9.38831E-016**	5.66491E-012 (7.22051E-012)	**1.39918E-015 (3.41194E-016)**
f_2	4.65730E+001	**1.99201E+00**	4.71152E+001 (2.87453E+01)	**2.77842E+001 (2.79576E-001)**
f_3	1.12777E-003	**1.64845E-011**	1.16177E+00 (6.64121E-01)	**1.07070E-001 (3.12879E-001)**
f_4	3.03109E-011	**2.22044E-016**	4.16361E-003 (8.77758E-003)	**5.66214E-016 (2.47977E-016)**
f_5	1.15782E-006	**1.03473E-009**	5.8194E-006 (5.55291E-06)	**3.01690E-009 (1.38824E-009)**
f_6	1.97656E+003	**3.55300E+002**	1.43142E+003 (7.05152E+02)	**7.71971E+002 (2.84908E+002)**

Table 4. Comparison of ABC and S-ABC in terms of Error, Standard and fitness (Dim=100)

Fun	Fitness (Dim=100)		Error and (standard deviation)	
	ABC	S-ABC	ABC	S-ABC
f_1	4.7862E-009	**3.76130E-012**	7.59245E-008 (7.0909E-008)	**1.63246E-011 (1.04497E-011)**
f_2	5.4879E+001	**9.58502E+00**	1.0069E+002 (2.9026E+001)	**9.68073E+01 (3.83279E-001)**
f_3	6.28160E+00	**2.21558E-006**	9.79577E+00 (2.74344E+001)	**1.59070E+00 (1.34787E+001)**
f_4	1.00884E-007	**5.35132E-014**	1.24815E-003 (3.89934E-003)	**2.68802E-012 (2.47904E-012)**
f_5	1.38373E-004	**7.71599E-007**	7.95947E-004 (6.37673E-004)	**2.52222E-006 (1.66672E-006)**
f_6	2.38494E+003	**1.97656E+003**	3.97203E+003 (3.38619E+003)	**2.55821E+003 (3.81167E+002)**

5.1 Convergence of Standard benchmark Functions towards optimum

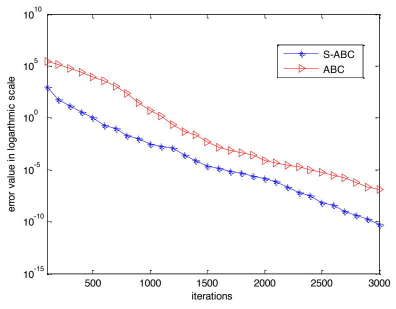

Fig. 2. Sphere function (Dim=100)

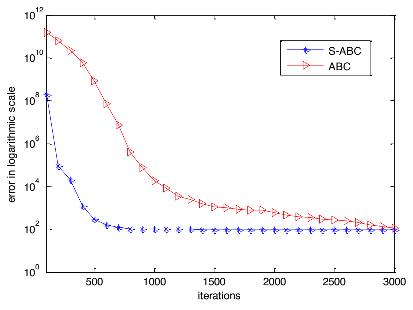

Fig. 3. Rosenbrock function (Dim=100)

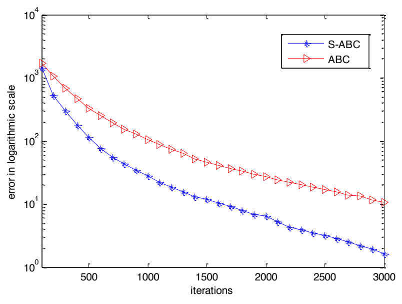

Fig. 4. Rastrigin Function (Dim=100)

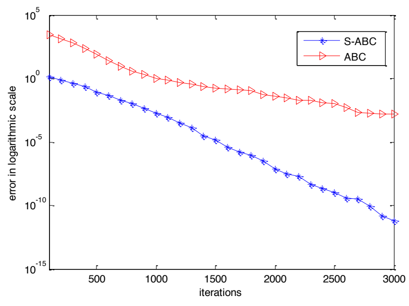

Fig. 5. Griewank function (Dim=100)

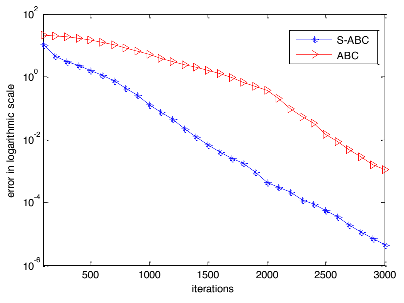

Fig. 6. Ackley Function (Dim=100)

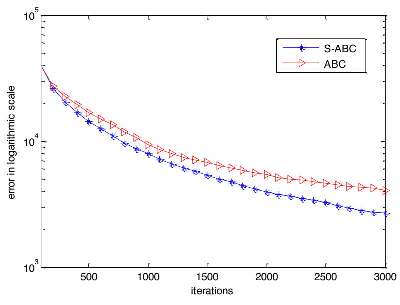

Fig. 7. Schwefel function (Dim=100)

6 Conclusions

In the present study we proposed the application of a low discrepancy Sobol sequence to guide the movement of scout bee in an ABCA. Use of a QRS or low discrepancy sequence to population based search algorithms like that of ABCA is particularly interesting because the working of these algorithms depend largely on random number numbers. Empirical analysis of the proposed S-ABC and its comparison with the basic ABCA indicates that the use of low discrepancy Sobol sequence improves the convergence rate besides maintaining the solution quality. The present work can be extended in several directions. In future, we plan to apply other low discrepancy sequences and compare it with the Sobol sequence. Also we are working on extending it for constrained optimization problems as well.

References

1. Goldberg, D.: Genetic Algorithms in Search Optimization and Machine Learning. Addison Wesley Publishing Company, Reading, Massachusetts, (1986),
2. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In Proc IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, 1942—1948 (1995).
3. Price, K., Storn R.: Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Technical Report, International computer Science Institute, Berkley, (1995).
4. Karaboga, D., Basturk,B..: A powerful and efficient Algorithm for Numerical Optimization: Artificial bee Colony (ABC) Algorithm. J. Global. Optim, Vol. 39, 459--472 (2007)
5. Karaboga, D., Basturk,B..: On the Performance of Artificial bee Colony (ABC) Algorithm. Applied Soft Computing, Vol. 8, 687--697 (2008)
6. Karaboga, D., Basturk,B..: A Novel Clustering Approach: Artificial bee Colony (ABC) Algorithm. Applied Soft Computing, Vol. 11, 652--657 (2011)
7. Karaboga, N. A new design method based on artificial bee colony algorithm for digital IIR filters. J. Franklin Inst. Vol 346, 328—348
8. Sabat, S., Udgata, S.K., Abraham, A.: Artificial bee Colony Algorithm for Small Signal Model Parameter Extraction of MESFET. Engineering Applications of Artificial Intelligence, Elsevier Science, (2010)
9. Zhu, G., Kwong, S.: Gbest-guided Artificial Bee Colony Algorithm for Numerical Function Optimization, 3166—3173
10. Drias, H., Souhila, S., Yahi, S.: Cooperative bees for solving the maximum weighted satisfiability problem. Computational Intelligence and Bio inspired Systems, LNCS, Vol. 3512, 318-325, (2005)
11. Sundarewsaran, K., Sreedevi, V.T.: Development of novel optimization procedure based on honey bee foraging behavior. In IEEE International Conference on Systems, Man and Cybernatics (2008)
12. Akay, B., Karaboga, D.: A modified Artificial Bee Colony Algorithm for Real-Parameter Optimization. Information Sciences, Elsevier (2010), doi: 10.1016/j.ins.2010.07.015
13. Abraham, A., Jatoth, R.K., Rajasekhar A.: Hybrid Differential Artificial Bee Colony Algorithm. Journal of computational and Theoretical Nano Science, USA, (2011) accepted.
14. Pant, M., Thangaraj R., Grosan, C., Abraham, A.: Improved Particle Swarm Optimization with Low-Discrepancy Sequences. Proc. IEEE Congress on Evolutionary Computation, (2008)
15. Nguyen, X.H., Nguyen, Q.Uy., Mckay, R.I. and Tuan, P.M.: Initializing PSO with Randomized Low-Discrepancy Sequences: The Comparative Results. Proc IEEE Congress on Evolutionary Algorithms, 1985—1992 (2007)

16. Chi, H.M., Beerli, P., Evans, D.W., and Mascagni, M.: On the Scrambled Sobol Sequence. Proc of Workshop on Paraleel Monte Carlo Algorithms for Diverse Application in a Distributed Setting, LNCS 3516, Springer Verlag, 775—7782, (1999)
17. Yao, X., Liu, Y., and Lin, G.: Evolutionary Programming Made Faster. Evol Compt., IEEE Trans., vol. 3, 82—102 (1999)