Next Generation Sequencing Using Ant Colony Optimization Algorithm and Binary Particle Swarm Optimizers

Pradipta Deb, Moitree Basu

Tata Consultancy Services Limited, India

deb.pradipta@gmail.com, sfurti.basu@gmail.com

Abstract

In this paper, An Ant Colony Optimization Algorithm along with Binary Particle Swarm optimization function has been proposed for Next Generation Sequencing. The Ant Colony Optimization Algorithm is a novel optimization algorithm inspired by a particular intelligent behavior of ants whereas Binary Particle Swarm optimizers are inspired by the behavior of birds. A new method of Ant Colony Optimization Algorithm for determining food source in neighborhood is proposed taken into consideration the discreteness of Next Generation Sequencing Problem. Experimental results shows that the new approach is more robust and obtains result of better quality.

Keywords: DNA Squencing; Multi-objective Optimization; Ant Colony Optimization Algorithm.

I. Introduction

1994 is considered as a revolutionary year in history of Bio-Medical Science. In this year Dr. Adleman L for the very first time proposed DNA as a calculation medium. After that milestone of DNA computing came when molecular biology was applied to solve Hamiltonian problem [1]. An essential tool for analyzing biological sequences is

Next Generation Sequencing (NGS). NGS is a NP-complete problem [2] and it is one the most important and challenging tasks in computational biology because of time complexity of NGS which grows exponentially with the size of the problem. Mainly four kinds of algorithm is used for NGS. They are Iterative algorithm, Exact matching, Inexact matching, Algorithm based on graph theory and Evolutionary algorithms.

NGS is a multi-objective optimization problem which has many constraints for DNA Sequencing [3] like Melt Chain Temperature, Free Energy, Hairpin Constraint, H-Measure Constraints, Hamming Constraints, Similarity Constraints, GC content constrains etc. We have used selected no of constraints for evaluation which every DNA sequence will satisfy and then applied Ant Colony Optimization algorithm (ACO) along with Binary Particle Swarm Optimizer functions (BOF) to get the best possible result for NGS.

This paper is organized as follows. In section 2, the DNA sequence design problem is defined. Section 3 gives an overview of ACO and BOF for Next Generation DNA sequence design. In Section 4, the sequence generated are shown and compared with those of other existing sequences with some benchmarks. In Section 5, conclusion is drawn.

II. DNA Sequence Design

According to the computational complexity, Next Generation Sequence design problem is NP-Hard and a multi-objective optimization problem. DNA Sequence hybridization and base-pairing complement is most important factor for good sequences. So it important that sequences are duplex with each other and no two sequence is complement to each other. In our paper, BOF functions are mainly used to select the best from the sequence set generated by ACO.

A. *Analysis of DNA Encoding Design*

There are many objective functions and constraints used to obtain a set of good DNA sequences. But, since some conditions overlap with others, the criterions should be selected cautiously. All of these objective functions and constraints can be classified into three categories [5]:

- **Mis-hybridization Prevention:**

  Mis-hybridization readily diminishes the reliability and efficiency of sequences generated in DNA computing. So, it is very much important to prevent mis-hybridization to perform NGS based computation successfully. Below mentioned criterion forces the set of sequences to form the duplexes between a given DNA sequence and its complement. The main constraints in preventing mis-hybridization are mentioned below:

  - **Hamming**
  - **H-measure**
  - **Similarity**

- **Hindering undesired secondary structures:**

  Undesired DNA complexes are also produced by the formation of undesired secondary structures. Stopping undesired secondary structure generation enables NGS computing to be more reliable and efficient. This aspect contains three constraints, which are:

  - **Self-Complementary**
  - **Hairpin**
  - **Continuity**

- **Keep uniform chemical characteristics:**

  Generally, it is preferable that each DNA sequence used in DNA computing behaves uniformly in fundamental chemical reactions. There are two constraints in this aspect:

- **GC-content**
- **Tm**

B. Sequence Design Criteria

- **Hamming Constraint:**

  The Hamming distance between xi and xj should be not less than a threshold value, namely H (xi ,xj) ≥ d .The evaluation function of Hamming constraint is defined as follows:

  $$fHamming(i) = \min \{ H (xi ,xj) \} \quad \forall \ 1 \leq j \leq m, j \neq 1$$

  Where fHamming(i) represent Hamming evaluation function of the i-th I individual.

- **Continuity Constraint:**

  If the same bases occur continuously in a sequence, the sequence might show unexpected structure.

  $$fCon(i) = - \sum (j - 1) Nj^{(i)} \qquad \forall \ 1 \leq j \leq n$$

  Where Nj(i) is the quantity of j-times that the same letter continuously presents in the designated sequence i.

- **GC Constraint:**

  $$fGC(i) = | GC^{(i)} - GC^{(i)} \text{defined}|$$

  Where $GC^{(i)}$ is the percentage of letter G and C in sequence xi; $GC^{(i)}$ defined indicate the content of specified content of GC and the general selection is 50%.

III. Ant Colony Optimization (ACO) Algorithm

Ant Colony Optimization (ACO), is a paradigm for designing meta-heuristic algorithms for combinatorial optimization problems, is inspired by the ability of ants to find the shortest path between their nest and a source of food. Marco Dorigo first introduced ACO and applied it to the Traveling Salesman Problem (TSP) [6]. After that it has been applied to the quadratic assignment problem [7], the vehicle routing problem [8], and RNA secondary structure prediction [9] [10] etc.

A. Principle of ACO Algorithm

The main idea behind ACO algorithm came from the observations of process of ant colony searching for food in nature. In real ant colonies, a pheromone, which is an odorous substance, is used as an indirect communication medium. When a source of food is found, ants lay some pheromone to mark the path. The quantity of the laid pheromone depends upon the distance, quantity and quality of the food source. While an isolated ant that moves at random detects a laid pheromone, it is very likely that it will decide to follow its path. This ant will itself lay a certain amount of pheromone, and hence enforce the pheromone trail of that specific path. Accordingly, the path that has been used by more ants will be more attractive to follow. In other words, the probability with which an ant chooses a path increases with the number of ants that previously chose that path. This process is hence characterized by a positive feedback loop.

B. Mathematical Model of ACO Algorithm:

In every generation each of m ants constructs one solution. For the selection of a start time the ant uses heuristic information as well as pheromone information. The heuristic information, denoted by η_{js}, and the pheromone information denoted by τjs, are indicators of how good it seems to put start time S as start time of activity j. The heuristic value is generated by some problem dependent heuristic whereas the pheromone information gathered from former ants that have found good solutions. The start time is chosen according to the probability distribution over the set of eligible start time S determined by evaluation according to:

$$p_{js} = \frac{[\tau_{js}]^{\alpha}[\eta_{js}]^{\beta}}{\sum_{t\epsilon S}[\tau_{js}]^{\alpha}[\eta_{js}]^{\beta}}$$

Where α and β are constants that determine the relative influence of the pheromone values and the heuristic values on the decision of the ant.

The best solution found so far and the best solution found in the current generation are then used to update the pheromone matrix. But before that some of the old pheromone is evaporated on all the edges where parameter ρ determines the evaporation rate. The reason for this is that old pheromone should not have a too strong influence on the future. Then, allowing each ant to deposit pheromone on the elements that belong to its tour. This is an elitist strategy that leads ants to search near the best found solution

$$\tau_{js}(t+1) = (1-\rho).\,\tau_{js}(t) + \sum_{k=1}^{m} \Delta\tau_{js}^{k}(t) \qquad \forall(j,s)$$

On the elements which are not chosen by the ants, the associated pheromone strength will decrease exponentially with the number of iterations. $\Delta\tau^{k}{}_{js}(t)$ is the amount of pheromone ant k deposits on the elements; it is defined as:

$$\Delta\tau_{js}^{k}(t) = \begin{cases} Q/L^{k}, & if\ (j,s) is\ used\ by\ ant\ k \\ \quad 0, & otherwise \end{cases}$$

Where Lk (t) is the fluctuation rate (will be described in the following section) of the k-th ant’s tour. The better solution ant’s tour is, the more pheromone is received by elements belonging to the tour. The algorithm runs until some stopping criterion is met, e.g. a certain number of generations nc have been done or the average quality of the solutions found by the ants of one generation has not changed for several generations (stagnation of best result).

C. Principle of Binary Particle Swarm Optimizers (BOF):

A canonical Particle Swarm Optimization model requires only three algebraic operators, “modifying a velocity”, “combining three velocities” and “applying a velocity to a position”. Moreover, for binary optimization, it is possible to define a toolbox of specific ones, but most of the optimizers are extremely efficient only on some kind of problems, but just reasonably efficient for other problems.

In binary optimization, it is very easy to design some algorithms that are extremely good on some benchmarks (and extremely bad on some others). It means we have to be very careful when we choose a test function set. For our article, we have chosen the below defined 2 optimizers:

- **Goldberg's order-3:**

The fitness f of a bit-string is the sum of the result of separately applying the following function to consecutive groups of three components each:

$$f(x) = \begin{cases} 0.9 \; if \;\; |y| = 0 \\ 0.6 \; if \;\; |y| = 1 \\ 0.3 \; if \;\; |y| = 2 \\ 1.0 \; if \;\; |y| = 3 \end{cases}$$

For example, if the string is x = 010110101, the total value is f1 (010) + f1 (110) + f1 (101) = 0.9 + 0.3 + 0.3 = 1.5. If the string size is D, the maximum value is obviously D/3, for the string 1111...111. In practice, we will then use as fitness the value D/3 − f so that the problem is now to find the minimum 0.

- **Bipolar order-6:**

The fitness is the sum of the result of applying the following function to consecutive groups of six components each:

$$f(x) = \begin{cases} 1.0\ if\ \ |y| = 0\ or\ 6 \\ 0.0\ if\ \ |y| = 1\ or\ 5 \\ 0.4\ if\ \ |y| = 2\ or\ 6 \\ 0.8\ if\ \ |y| = 3 \end{cases}$$

So the solutions are all combinations of sequences 6x1 and 6x0. In particular, 1111...111 and 0000...000 are solutions. The maximum value is D/6.

IV. Algorithm Design

The main idea of optimization problem is to select appropriate amount of individuals with high fitness value. To achieve that we have applied fitness value to each individuals and continue running the evaluation until the whole process converges or max no of iterations is reached.

According to the mathematical model of ACO shown in section 3.2, each short sequence is considered as a node in our ACO algorithm. “m” ants are randomly placed at “n” nodes and each ant move towards all nodes in accordance with probability and edge value between two nodes(Overlap between two sequences is considered as edge value between those two nodes). Combining the DNA short sequences according to overlap, which corresponds to the nodes which ants have passed, with constraints conditions, the optimal results are obtained. After providing these ACO optimized result into BOF function as input, a single optimal result (Most optimum sequence) is obtained.

The basic steps of ACO algorithm as follows:

Alg. 1 Pseudo-code of Algorithm

```
• initialize all edges to (small) initial pheromone level τ0;
• place each ant on a randomly chosen node;
• for k := 1 to m do
      o initialize candidate list k to the c closest node of k
• end;
• for itr := 1 to itr_max do
```

```
        o  for k := 1 to number of ants do
              ▪  until (tour T(k,t) for ant k is complete) do
                    •  if there is at least one unvisited node in candidate list k
                          o  choose the next node among the candidates by applying
                             the probabilistic transition rule;
                    •  else
                          o  choose the next node as the next node still to be visited;
                    •  endif
                    •  Store the sequence of nodes passed by all ants as a probable
                       solution to a tabu_list.
                    •  perform local trail update;
                    •  Calculate fitness value of the all solutions in tabu_list
                       according to the formula, and save the all fitness value at
                       present
                    •  'x' solutions which have higher fitness value as local optimal
                       solution will be stored.
              ▪  end
              ▪  for every edge on the current solution do
                    •  apply global trail update;
              ▪  end
        o  end;
  •  end;
```

Output: List of x solutions (Sequences) having same fitness value according to ACO.

The basic steps of BOF optimization as follows:

Alg. 1 Pseudo-code of BOF optimization

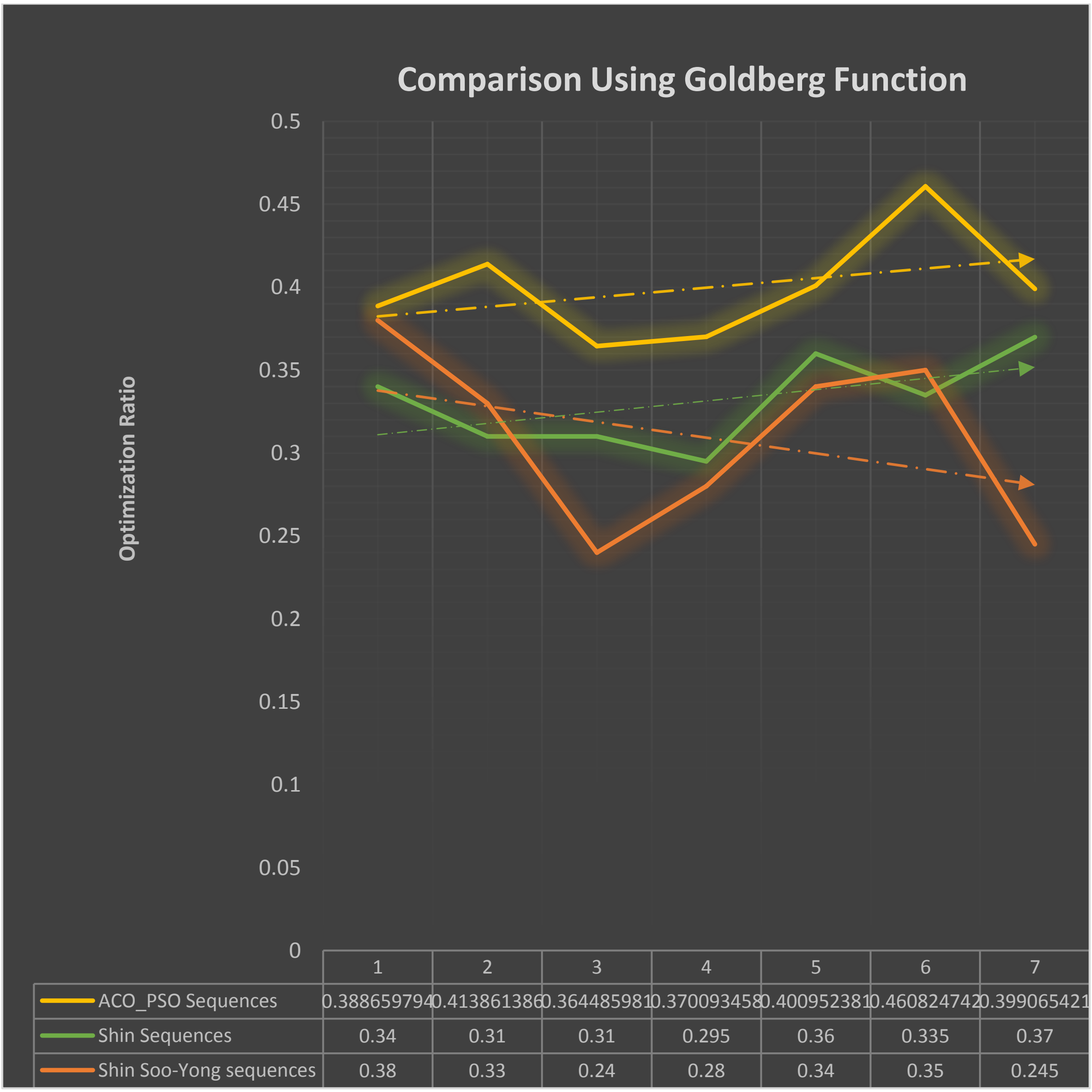
```
  •  Initialize optimization function (Goldberg or Bipolar-six)
  •  For each sequence i= 1 to n do
        o  Evaluate each sequence fitness according to optimization function
        o  If (current fitness > optimal fitness) then
              ▪  Store current result as optimal fitness result
        o  endif
  •  end
```

Output: the fittest resulting sequence.

V. Experimental Results and Analysis

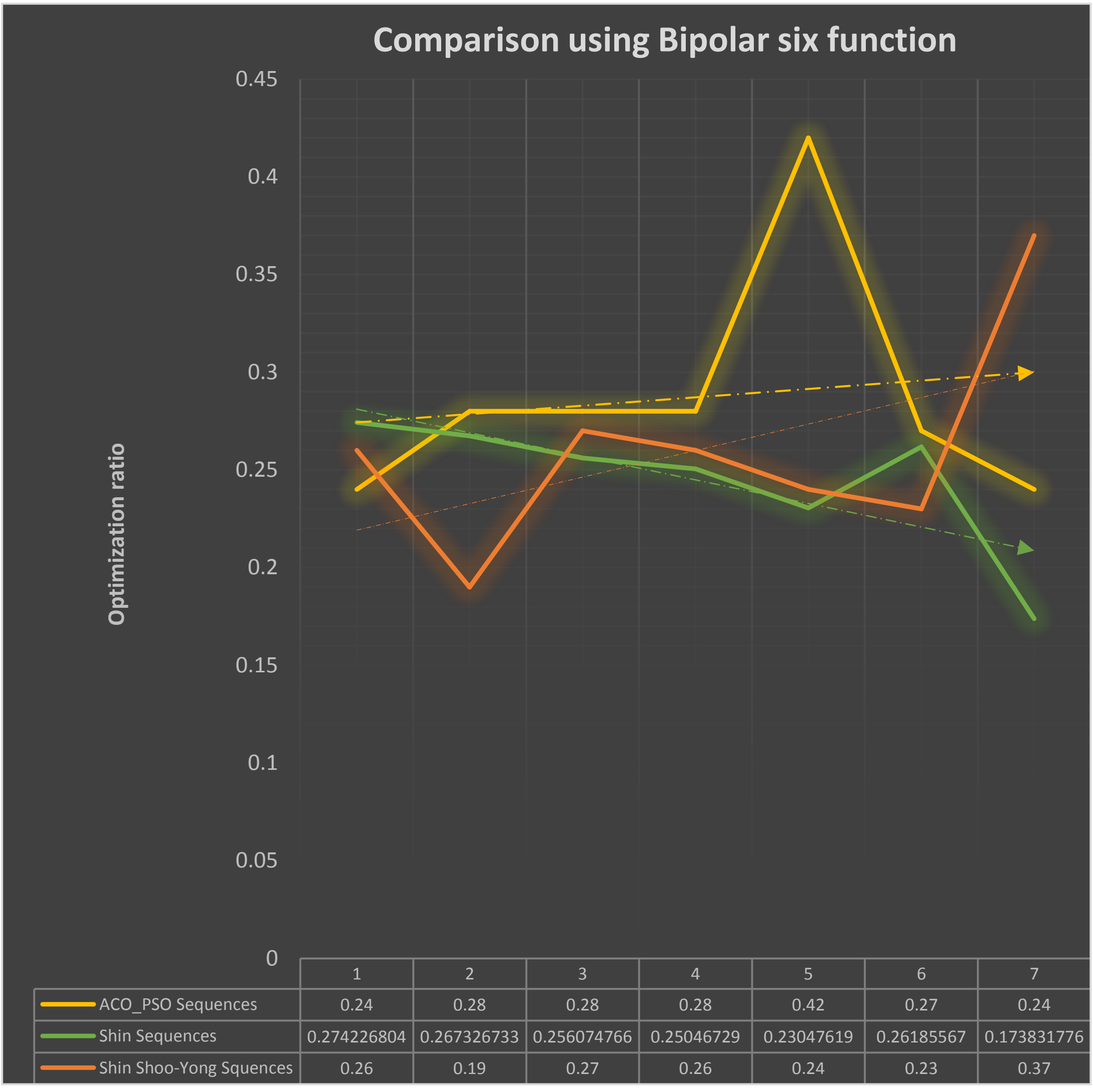
We used the above mentioned ACO algorithm to design the DNA sequence set. To evaluate the performance of this algorithm, we analyzed it by contrasting sequence set in literature

[11] and [12]. The comparison results between our sequence, Shin Soo-Yong sequences in literature [11] and Shin sequences in literature [12] using Goldberg function (as optimization reference) is shown in Fig. 1 and using bipolar six function (as optimization reference) is shown in Fig. 2. As, length of sequences, which are in comparison are not same, so in order to have a comparative result, we have taken optimization ratio (Optimization Value/ Sequence Length) as our value of comparison.



	1	2	3	4	5	6	7
ACO_PSO Sequences	0.388659794	0.413861386	0.364485981	0.370093458	0.400952381	0.460824742	0.399065421
Shin Sequences	0.34	0.31	0.31	0.295	0.36	0.335	0.37
Shin Soo-Yong sequences	0.38	0.33	0.24	0.28	0.34	0.35	0.245

Fig.1 Comparison of sequences using Goldberg optimization function



	1	2	3	4	5	6	7
ACO_PSO Sequences	0.24	0.28	0.28	0.28	0.42	0.27	0.24
Shin Sequences	0.274226804	0.267326733	0.256074766	0.25046729	0.23047619	0.26185567	0.173831776
Shin Shoo-Yong Squences	0.26	0.19	0.27	0.26	0.24	0.23	0.37

Fig.2 Comparison of sequences using Bipolar Six optimization function

It is clearly visible from the average trend lines (broken arrow lines) that our sequences produce a higher optimization ratio in average than the other sequences. Hence, sequences generated by ACO_PSO algorithm are superior to the Shin Soo-Yong sequences in literature [7] and the Shin sequences in literature [5]. So the results illustrate the ACO algorithm, which meet the design requirements, is feasible and effective.

VI. Conclusion

In this article, we considered the variety of constrained conditions of DNA encoding, and abstract it as multi-objective optimization problem, then make application of ant colony optimization algorithm to realize the optimization of DNA encoding sequence. At last, good DNA sequences are generated. Compared with the DNA sequence which produced under identical standards by earlier experiments, the results prove the feasibility and effectiveness of this method and also reflect that the Ant Colony Optimization algorithm have certain advantages in solving multi-objective optimization problem at the same time.

References

[1] L.M. Adleman, “Molecular computation of solutions to combinatorial problem,” Science, vol. 66, no. 1, 1994, pp. 1021-1024.

[2] S. Xu, Q. Zhang, “Optimization of DNA Coding Based on GA/PSO Algorithm,” Computer Engineering, vol. 34, no. 1, 2008, pp. 1.

[3] W. Liu, “Research on the Encoding Problem and Algorithms of DNA Computing,” Huazhong University of Science&Technology, Wuhan, China, 2003

[4] Z. Yin, C. Ye, M. Wen, “Research on DNA Encoding Based on Cultural Particle Swarm Optimization Algorithm,” Computer Engineering, vol. 37, no. 3, 2011, pp. 2.

[5] R. Deaton, R. C. Murphy, M. Garzon, et al. “Good Encodings for DNA-based Solutions to Combinatorial Problems,” Proceedings of the 2nd DIMACS Workshop on DNA-Based Computers. [S.1]: IEEE Press, 1966.

[6] M. Dorigo, “Optimization, learning and natural algorithms. Dipartimento di Elettronica,” Politechico di Milano, Italy, 1992.

[7] V. Maniezzo, A. Colorni, and M. Dorigo, “The ant system applied to the quadratic assignment problem,” Universite Libre de Bruxelles, Belgium, Tech. Rep. IRIDIA, 1994, pp. 94-28.

[8] B. Bullnheimer, R.F. Hartl, C. Strauss, “An improved Ant System algorithm for the Vehicle Routing Problem,” Annals of Operations Research, vol. 89, no. 0, Jan 1999, pp. 319–328.

[9] N. McMellan, “RNA Secondary Structure Prediction usingAnt Colony Optimisation,” School of Informatics, University of Edinburgh, 2006.

[10] T. Basuki Kurniawan, N. Khafifah Khalid, Z. Ibrahim, “An Ant Colony System for DNA Sequence design based on thermodynamics. Advances in Computer Science and Technology,” 2008, pp 1-6.

[11] S. Y. Shin, D. Kim, I. H. Lee, et al. “Evolutionary Sequence Generation for Reliable DNA Computing,” Proc of Congress on Evolutionary Computatation. [S.1]: IEEE Computer Society Press, 2002.

[12] S. Y. Shin, I. H.Lee, D. Kim, “Multiobjective Evolutionary Optimization of DNA Sequences for Reliable DNA Computing,” IEEE Transactions on Evolutionary Computation, vol. 9, no. 2, 2005, pp.143-158.

[13] Z. Yin, C. M. Ye and M. Wen, “Research on DNA encoding design constrain by minimal free energy,” Computer Engineering and Application, vol. 46, no. 12, 2010, pp. 25-28.

[14] F. Ducatelle and J. Levine, “Ant Colony Optimisation for Bin Packing and Cutting Stock Problems,” presented at UK Workshop on Computational Intelligence (UKCI-01), Edinburgh, 2001.

[15] H. Duan, “Ant Colony Algorithms:Theory and Applications,” Science Press, Beijing, 2005.

Pradipta Deb is currently working in Tata Consultancy Services Limited as Assistant System Engineer. He received his B.Tech. Degree in Dept. of Information Technology from Netaji Subhash Engineering College in 2012. He will be starting his PhD degree in US in fall 2014. His current research interests include Cloud Computing, Algorithms, Computational Biology and Bioinformatics etc. He possesses interest in pencil drawing, playing musical instruments.

Moitree Basu is currently working in Tata Consultancy Services Limited as Assistant System Engineer. She received her B.Tech. Degree in Dept. of Information Technology from Netaji Subhash Engineering College in 2012. She will be starting her PhD degree in US in fall 2014. Her current research interests include Cloud Computing, Algorithms, Computational Biology and Bioinformatics etc. She possesses interest in painting, listening to music.