# Auto-Scheduler for School Timetable Using Bacterial Foraging Optimization Algorithm

Dr A Kunthavai[1], R Ranjithaa[2]
*[1,2]Department of Computer Science and Engineering, Coimbatore Institute of Technology, Coimbatore, Tamilnadu, India*

**Abstract- Timetable scheduling is a common process in all the educational institutions. Manual timetable may lead to imperfections which consumes more time and man power to schedule. Timetable scheduling problem is a NP hard problem. Staffs face more difficulties in scheduling the timetable such as considering the hard constraints, soft constraints, staff preferred timing etc., In most of the existing system, the algorithm used is Genetic Algorithm(GA). Only less number of constraints, less choice for students and staffs, few genetic operators were taken into consideration while using the Genetic Algorithm. Also it avoids selection process which results in elimination of best individual from the population. In order to reduce the difficulties faced by applying genetic algorithm, Bacterial Foraging Optimization Algorithm (BFOA) is used in combination with Genetic Algorithm (GA) with user specified constraints.**
**Keywords – timetable scheduling,constraints,BFOA.**

## I. INTRODUCTION

Auto-Scheduler is used to generate the timetable in which the timetable consists of the timing from start of the class to end of the class, location of the class etc., The constraints involved and the definition of an optimum timetable can vary between institutes but can be classified broadly into a set of hard constraints and soft constraints. Hard constraints are the physical constraints which must be satisfied (in any case) like, a) no student or teacher can be present at the same time in two different classes and b) for each class, there must be sufficient seats to accommodate all students. Soft constraints are the ones which can be considered as desirable but not necessarily essential like, a) a room should not be left unallocated for more than twenty percent of the time per day, and b) not more than one class of a subject per day.

Approaches based on Genetic Algorithms (GA) and hybrids have been devised and proved effective for solving the problem of time-table generation. Sigl et al[4]. have modeled the timetable problem as a special case of 3-D cutting problem but the proposed algorithm is unable to resolve all the conflicts in large instances of problem. A generic model of the time-table problem using basic Genetic Algorithm by Malek Rahoual et al. have proposed an approach based on hybrid GA and Tabu Search for solving the time-table problem[6]. The problem with GA based approaches is that they show a very fast initial convergence, followed by progressive slower improvements. If the fitness of all the models is similar, the convergence may be slow. Also, sometimes, good models are better than the rest of the population and result in the premature convergence to local minima.

An optimization approach based on the search and optimal foraging behavior of Escherichia coli (E. Coli) bacteria has been proposed in [8]. Bacterial foraging optimization algorithm (BFOA) has been applied in optimal watermarking[9] , RFID network scheduling [10]. Also BFOA faces premature convergence to local minima if the number of chemotactic steps chosen is too short or the number of reproduction steps are not sufficient.

In this paper, an approach based on bacterial foraging algorithm and genetic algorithm is used to solve the school timetabling problem. In this work, timetable without any clashes is considered as healthy bacteria. Here Genetic Algorithm is used to generate healthy bacteria.

The rest of the paper is organized as follows. Proposed algorithms are explained in section II. Experimental results are presented in section III. Concluding remarks are given in section IV.

## II. PROPOSED ALGORITHM

### *2.1 Bacterial foraging optimization algorithm [11]*

The key idea of BFOA is mimicking the chemotactic movement of virtual bacteria in the problem search space. The process of foraging includes

- Chemotaxis
- Swarming
- Reproduction
- Elimination and dispersal

http://dx.doi.org/10.21172/ijiet.101.06

In BFOA, the chemotaxis of the bacteria is like a biased random walk where the bacteria try to search for places with better nutrient gradient alternating between "swim" and "tumble". Swarming stage is basically the cell to cell signaling stage. The reproduction eliminates the weaker individuals and a fitter bacterium splits into two bacteria. Use of elimination dispersal mechanism is to avoid falling into premature convergence.

*a)* Chemotaxis:

This process simulates the movement of an E.coli cell through swimming and tumbling via flagella. Biologically an E.coli bacterium can move in two different ways. It can swim for a period of time in the same direction or it may tumble and alternate between these two modes of operation for the entire lifetime. The movement of bacterium is given in Eq. 1

$\Theta^{i}(j+1,k,l) = \theta^{i}(j,k,l) + C(i)\,\emptyset(j)$ (1)

*b)* Swarming:

The process by which intricate and stable spatio-temporal patterns (swarms) are formed in semisolid nutrient medium is called swarming. A group of E.coli cells arrange themselves in a traveling ring by moving up the nutrient gradient when placed amidst a semisolid matrix with a single nutrient chemo-effecter.[11]

*c)* Reproduction:

The least healthy bacteria eventually die when each of the healthier bacteria asexually split into two bacteria, which are then placed in the same location. This keeps the swarm size constant.

*d)* Elimination and Dispersal:

Gradual or sudden changes in the local environment where a bacterium population lives may occur due to various reasons. Events can occur such that all the bacteria in a region are killed or a group is dispersed into a new part of the environment.

The process involved in Bacterial Foraging Optimization Algorithm is described as follows.

Bacterial Foraging Optimization Algorithm for Auto-Scheduler

1. The initial set of solutions or generated timetable are referred to an initial population of bacterium.
2. The each structure of chromosome is represented as complete timetable.
3. After the generation of set of timetables, they are sorted based on the health or with minimum clashes and then the healthy bacteria are crossed to obtain the much better healthy bacterium.
4. If the generated bacterium is not as healthy as other bacteria, it will not be selected for the optimal solution then the next set of timetable is taken for crossover and mutation (reproduction event).
5. The timetable with maximum clashes is removed (elimination event).
6. The timetable with minimum clashes or without clashes is split into two which are placed in the same location to maintain the healthy bacterium population constant (dispersal event).

The pseudo code of the BFOA is shown in the fig.1

1. Initialize the bacterial foraging parameters
2. Start elimination-dispersal loop: l = l + 1.
3. Start reproduction loop: j = j + 1
4. Start chemotaxis loop: k = k + 1.
5. continue chemotaxis, if the life of the bacteria is not over otherwise go to step 3.
6. Reproduction: From the set of healthy bacteria clone the healthy ones and the bacteria which have higher number of clashes will die.
7. If j< number of reproduction steps, go to Step 3, otherwise, go to Step 8.
8. Elimination-Dispersal: The bacterium is eliminated or simply dispersed to a random location in the optimization domain
9. if l<number of elimination-dispersal events, then go to step 2; otherwise end

Fig 1 Pseudo code for Bacterial Foraging Optimization Algorithm

### *2.2 Genetic Algorithm*

The Genetic algorithms are adaptive systems inspired by natural evolution. They can be used as techniques for solving complex problems and for searching of large problem spaces. The power of genetic algorithms and other similar techniques (simulated annealing, evolutionary strategies) lies in the fact that they are capable to find global optimum in multi-modal spaces. Genetic algorithms work with a set of potential solutions, which is called the population. Each solution item (individual) is measured by a fitness function. Fitness in biological sense is a quality value which is a measure of the reproductive efficiency of chromosomes. The process of evolution is maintained by selection, crossover and mutation. In terms of genetic algorithm those processes are called genetic operators.

Crossover: The crossover operator is used to combine two bacterium in chemotactic step to get a better bacteria. From the list of generated timetables (bacteria), healthy bacteria are taken and the crossover is applied to get the better timetable.

Mutation: Mutation is applied for the offspring that is generated after crossover so that the number of clashes in the offspring will get lesser. So that the offspring is placed in the global optimal solution list. It helps to avoid getting trapped at local optima.

## III. EXPERIMENT AND RESULT

To verify the performance of the proposed system, it is implemented in C#, ASP.NET and set of input values is given dynamically. After getting the input values, values are passed as parameters to the algorithm in which it has to satisfy all the specified user constraints and produce the timetable. The input values include staff details, subject details, class details, classroom details, timeslot details etc., After generating the timetable, each timetable for the particular standard is verified for the clashes which are cleared using the genetic operators such as mutation and crossover. Since the project is for the schools, the generated timetable is used throughout the year.

Different types of timetable can be generated based on the timeslots mentioned. Some of the international schools have different time for different classes. The number of hours differ from 5 to 8 in different schools. Based on this different timeslots, different timetables are generated.

Following figures shows the results of the project.

| Column 0 | Column 1 | Column 2 | Column 3 | Column 4 | Column 5 | Column 6 |
|---|---|---|---|---|---|---|
| Monday | N01 | S03 | N01 | E02 | M02 | C01 |
| Tuesday | N01 | N01 | N01 | M02 | E02 | M02 |
| Wednesday | S03 | T03 | S03 | C01 | E02 | C01 |
| Thursday | S03 | N01 | T03 | C01 | C01 | C01 |
| Friday | S03 | S03 | S03 | C01 | E02 | M02 |
| **Column 0** | **Column 1** | **Column 2** | **Column 3** | **Column 4** | **Column 5** | **Column 6** |
| Monday | S03 | S03 | E01 | N02 | N02 | M03 |
| Tuesday | S03 | S03 | E01 | M03 | N02 | C02 |
| Wednesday | E01 | T01 | T01 | N02 | C02 | C02 |
| Thursday | T01 | E01 | E01 | C02 | C02 | N02 |
| Friday | T01 | E01 | S03 | C01 | C01 | N02 |
| **Column 0** | **Column 1** | **Column 2** | **Column 3** | **Column 4** | **Column 5** | **Column 6** |
| Monday | N01 | T03 | N01 | C01 | M02 | M02 |
| Tuesday | T03 | T03 | T03 | C01 | C02 | C01 |
| Wednesday | T03 | N01 | T03 | S01 | C02 | C02 |
| Thursday | T03 | E01 | E01 | C02 | S01 | S01 |
| Friday | E01 | E01 | E01 | E01 | S01 | M02 |

Fig 1. Timetable for 6 hours

| Column 0 | Column 1 | Column 2 | Column 3 | Column 4 | Column 5 | Column 6 | Column 7 | Column 8 |
|---|---|---|---|---|---|---|---|---|
| Monday | M01 | M01 | E01 | N01 | T02 | T02 | S01 | S01 |
| Tuesday | M01 | M01 | N01 | N01 | S01 | T02 | C01 | S01 |
| Wednesday | N01 | E01 | M01 | E01 | C02 | C01 | C02 | T02 |
| Thursday | M01 | N01 | E01 | N01 | T02 | C01 | S01 | C02 |
| Friday | M01 | E01 | E01 | E01 | C01 | T02 | T02 | S01 |
| **Column 0** | **Column 1** | **Column 2** | **Column 3** | **Column 4** | **Column 5** | **Column 6** | **Column 7** | **Column 8** |
| Monday | S03 | S03 | N03 | T03 | E03 | C01 | M02 | E03 |
| Tuesday | T03 | T03 | N03 | S03 | C02 | M02 | E03 | E03 |
| Wednesday | S03 | S03 | N03 | T03 | C01 | M02 | E03 | C01 |
| Thursday | S03 | N03 | N03 | N03 | C02 | E03 | C02 | C02 |
| Friday | N03 | T03 | T03 | S03 | T03 | E03 | M02 | M02 |
| **Column 0** | **Column 1** | **Column 2** | **Column 3** | **Column 4** | **Column 5** | **Column 6** | **Column 7** | **Column 8** |
| Monday | N01 | N01 | M01 | N01 | C02 | C02 | C02 | S01 |
| Tuesday | M01 | T01 | M01 | M01 | C02 | C02 | C02 | E03 |
| Wednesday | N01 | T01 | M01 | M01 | S01 | C01 | E03 | E03 |
| Thursday | T01 | M01 | T01 | T01 | T01 | E03 | E03 | E03 |
| Friday | T01 | N01 | N01 | N01 | E03 | S01 | S01 | S01 |

Fig 2 Timetable For 8 Hours

http://dx.doi.org/10.21172/ijiet.101.06

## IV.CONCLUSION

Bacterial foraging optimization algorithms and Genetic algorithms offer a great mechanism for solving combinatorial problems. This mechanism is utilized for solving the highly constrained and intractable problem of generating time tables for academic institutions. The adaption of the features of BFOA helps reduce the time taken to converge to a solution, is suitable to achieve global optimum and also helps avoid the problem of premature convergence. The use of GA in our approach helps in selecting the optimum step size and direction of chemotaxis.

## V. REFERENCE


[1] Om Prakash Verma , Rohan Garg and Vikram Singh Bisht., Optimal Time-Table Generation by hybridized Bacterial Foraging and Genetic Algorithms, Proceedings of International Conference on Communication Systems and Network Technologies 2012,pp.919-923

[2] Even, S., Itai A., Shamir, A., On the complexity of time table and multicommodity flow problems, Proceedings of the 16th Annual Symposium on Foundations of Computer Science, October 1975, pp.184-193,.

[3] Erben, W., Song, P.Y., A hybrid grouping genetic algorithm for examination timetabling. In: E.K. Burke and M. Trick, Editors, PATAT '04 Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling, Springer-Verlag, 2004, pp. 487–490.

[4] Sigl, B., Golub, M., Mornar V., "Solving Time-table Scheduling Problem by Using Genetic Algorithms", Proceedings of 25th IEEE Int. Conference on Information Technology Interfaces, Cavtat/Dubrovnik, Croatia, 2003, pp. 519-524.

[5] Colorni, A., Dorigo, M., Maniezzo, V., Genetic Algorithms and highly constrained probelms: The Time Table Case, Proceedings of the First International Workshop on Parallel Problem Solving from Nature, Dortmund, Germany, Lecture Notes in Computer Science 496, SpringerVerlag, pp. 55–59.

[6] Rahoual, M., Saad, R., Burke, E. K., Rudová, H., Solving Timetabling Problems by Hybridizing Genetic Algorithms and Tabu Search, (Eds.): PATAT 2006, pp. 467–472. ISBN 80-210-3726-1.

[7] Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, New York, 1999,pp. 99-105.

[8] Gazi, V., Passino, K.M., Stability analysis of social foraging swarms, IEEE Transactions on Systems Man and Cybernetics Part B – Cybernetics 34 (1) 2004, pp. 539–557.

[9] Huang, H. C., Chen, Y. H., Abraham, A., Optimized watermarking using swarm-based bacterial foraging, Journal of Information Hiding and Multimedia Signal Processing, vol. 1, no. 1, pp. 51-58, 2010.

[10] Liu, W., Chen Hexin, Chen Hanning, Chen M., RFID Network Scheduling Using an Adaptive Bacterial Foraging Algorithm, Journal of Computational Information Systems 7:4 ,2011, pp. 1238–1245.

[11] Senthilkumar M,"Bacterial Foraging Optimization Algorithm" Ph.D thesis, 2012.