International Journal of Engineering Research and General Science Volume 3, Issue 2, Part 2, March-April, 2015
ISSN 2091-2730

An Art of Generating Automatic Timetable Using Genetic Algorithm

Dipesh Mittal[1], Mohammed Sunasra[2], Hiral Doshi[3]

Bachelors of Engineering, Department of Information Technology, Atharva college of Engineering, University of Mumbai, India[1,2,3]

Email: dipeshpawanmittal@gmail.com[1]

Contact: +919920479186[1]

Abstract— Timetable creation is a very arduous and time consuming task. To create timetable it takes lots of patience and man hours. Time table is created for various purposes like to organize lectures in school and colleges, to create timing charts for train and flight schedule and many more. To create timetable it requires sufficient amount of time and man power. In the below paper we have tried to reduce these difficulties of generating timetable by using Genetics Algorithm. By the use of Genetic algorithm we are able to reduce the time required to generate timetable and generate a timetable which is more accurate, precise and free of human errors. The manual system of preparing time table in colleges with large number of students is very time consuming and usually ends up with various classes clashing either at same room or with same teachers having more than one class at a time. To overcome all these problems we propose to make an automated system. Presently this timetable is prepared manually, by manipulating those of earlier years, with the only aim of producing a feasible timetable.

Keywords— Genetic algorithm, timetable, constraints, chromosomes, mutation, crossover, fitness function.

INTRODUCTION

The class timetabling problem is a typical scheduling problem that appears to be a tedious job in every academic institute once or twice a year [3]. In earlier days, time table scheduling was done manually with a single person or some group involved in task of scheduling it manually, which takes a lot of effort and time. Planning timetables is one of the most complex and error-prone applications.

Timetabling is the task of creating a timetable while satisfying some constraints. There are basically two types of constraints, soft constraints and hard constraints. Soft constraints are those if we violate them in scheduling, the output is still valid, but hard constraints are those which if we violate them; the timetable is no longer valid [1].

The search space of a timetabling problem is too vast, many solutions exist in the search space and few of them are not feasible. Feasible solutions here mean those which do not violate hard constraints and as well try to satisfy soft constraints. We need to choose the most appropriate one from feasible solutions. Most appropriate ones here mean those which do not violate soft constraints to a greater extent [1]. Using Genetics Algorithm, a number of trade-off solutions, in terms of multiple objectives of the problem, could be obtained very easily. Moreover, each of the obtained solutions has been found much better than a manually prepared solution which is in use.

LITERATURE SURVEY

Genetic algorithms are general search and optimization algorithms inspired by processes and normally associated with natural world. Genetic algorithm mimics the process of natural selection and can be used as a technique for solving complex optimization problems which have large spaces [10]. They can be used as techniques for solving complex problems and for searching of large problem spaces. Unlike many heuristic schemes, which have only one optimal solution at any time, Genetic algorithms maintain many individual solutions in the form of population. Individuals (parents) are chosen from the population and are then mated to form a new individual (child). The child is further mutated to introduce diversity into the population [10]. Rather than starting from a single point within the search space, GA is initialized to the population of guesses. These are usually random and will be spread throughout the search space. A typical algorithm then uses three operators, selection, crossover and mutation, to direct the population toward convergence at global optimum. A GA, as shown in fig.1 requires a process of initializing, breeding, mutating, choosing and killing. It can be said that most methods called GAs have at least the following elements in common: Population of chromosomes, Selection according to fitness, Crossover to produce new offspring, and random mutation of new offspring.

ISSN 2091-2730

```
Create a population of creatures.
Evaluate the fitness of each creature.
While the population is not fit enough:
{
Kill all relatively unfit creatures.
While population size< max;
{
Select two population members.
Combine their genetic material to create a new creature.
Cause a few random mutations on the new creature.
Evaluate the new creature and place it in the population.
}}.
```

Fig 1: Top Level description of a GA [6]

A. GA Operators

1) Chromosome representation: Chromosome is a set of parameters which define a proposed solution to the problem that the genetic algorithm is trying to solve. The chromosome is often represented as a simple string. The fitness of a chromosome depends upon how well that chromosome solves the problem at hand.

2) Initial population: The first step in the functioning of a GA is the generation of an initial population. Each member of this population encodes a possible solution to a problem. After creating the initial population, each individual is evaluated and assigned a fitness value according to the fitness function. It has been recognized that if the initial population to the GA is good, then the algorithm has a better possibility of finding a good solution and that, if the initial supply of building blocks is not large enough or good enough, then it would be difficult for the algorithm to find a good solution.

3) Selection: This operator selects chromosomes in the population for reproduction. The fitter the chromosome, the more times it is likely to be selected to reproduce [11].

4) Crossover: In genetic algorithms, crossover is a genetic operator used to vary the programming of a chromosome or chromosomes from one generation to the next. It is analogous to reproduction and biological crossover, upon which genetic algorithms are based. Cross over is a process of taking more than one parent solutions and producing a child solution from them. There are methods for selection of the chromosomes. This operator randomly chooses a locus and exchanges the subsequences before and after that locus between two chromosomes to create two offspring. For example, the strings 10000100 and 11111111 could be crossed over after the third locus in each to produce the two offspring 10011111 and 11100100. The crossover operator roughly mimics biological recombination between two single−chromosome organisms [11].

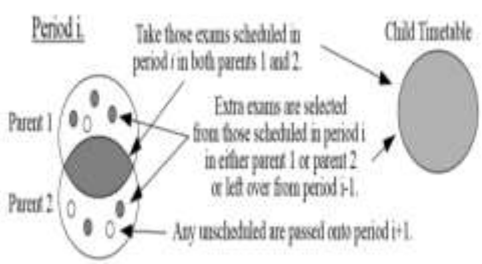

Fig 2: Crossover Operator [9]

5) Mutation: Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. It is analogous to biological mutation. Mutation alters one or more gene values in a chromosome from its initial state. In mutation, the solution may change entirely from the previous solution. Hence GA can come to better solution by using mutation. This operator randomly flips some of the bits in a chromosome. For example, the string 00000100 might be

mutated in its second position to yield 01000100. Mutation can occur at each bit position in a string with some probability, usually very small [11].

6) Fitness Function: The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent In particular, in the fields of genetic programming and genetic algorithms, each design solution is commonly represented as a string of numbers referred to as a chromosome. After each round of testing, or simulation, the idea is to delete the 'n' worst design solutions, and to breed 'n' new ones from the best design solutions. Each design solution, therefore, needs to be awarded a figure of merit, to indicate how close it came to meeting the overall specification, and this is generated by applying the fitness function to the test, or simulation, results obtained from that solution.

IMPLEMENTED APPROACH

In order to deal with timetabling issues we have implemented a system which would mechanically generate timetable for the institute. Course and lectures will be scheduled in accordance with all possible constraints and given inputs and thus a timetable will be generated.

There are few steps involved in the overall functioning of genetic algorithm which is stated below:

- Create population and then check the fitness function of each chromosomes of the population.
- If the chromosome is unfit for processing kill it and proceed further.
- Describe the population size that is the maximum number of chromosome to be there in initial population and then select two chromosomes from the same which will act as parent.
- From the parent chromosome a child chromosome is created and then again the fitness function of the child chromosome is checked if it is fit for the operation it is passed into the system or else killed.

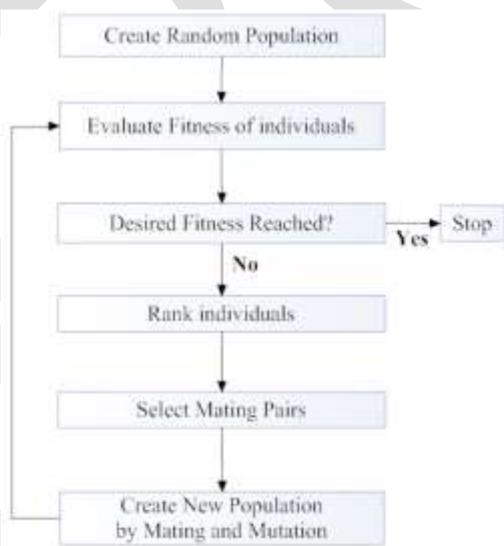

Fig 3: Flow graph of genetic algorithm

Structure of time table generator consists of input data, relation between the input data, system constraints and application of genetics algorithm.

A. Input Data

The input data contains:

1) *Professor:* Data describes the name of lecturers along with their identification number.
2) *Subject:* Data describes the name of courses in the current term.
3) *Room:* Data describes the room number and their capacity.
4) *Time intervals:* It indicates starting time along with duration of a lecture.

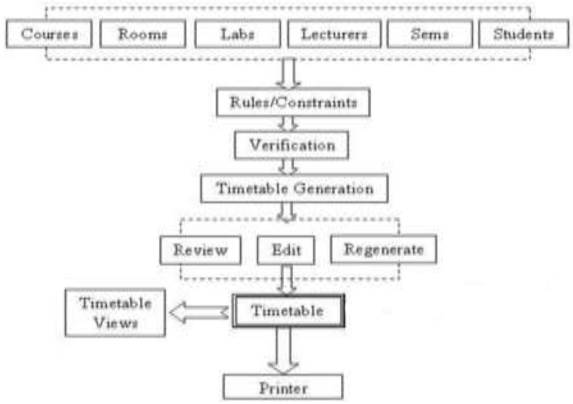

Fig 4: General view of Timetable Generator [4]

B. System Constraints

System constraints are divided into 2 categories:

1) Hard Constraints: The timetable is subjected to the following four types of hard constraints, which must be satisfied by a solution to be considered as a valid one:
   a. A student should have only one class at a Time.
   b. A Teacher should have only one class at a time.
   c. A room should be booked only for one class at a time.
   d. Some classes require classes to have particular equipment. For example, audio visual equipment, projectors etc.

2) Soft Constraints: These are the constraints that are of no great concern but are still taken into contemplation. They don't need to be satisfied but the solutions are generally considered to be good if they are satisfied.
   a. Courses must be eventually distributed.
   b. Students should not have any free time between two classes on a day.
   c. Scheduling of teachers should be well spread over the week.

3) Constraints Relations Pertaining to Timetable Generation:

The timetabling algorithm consists of set of different lists, such as Teachers List, Subjects List, Rooms List, Times Slots List, Semesters List, and Days List. The input will be taken from user.

a. Teacher-Subject Constraint

In this constraint, Teacher is assigned to various Subjects. A Teacher can be assigned to more than one subject.

Fig 5: Teacher Subject Mapping

b. Teacher –Time Slot Constraint

A Teacher may have some hours as favourable hours and some as forbidden hours for conducting lectures. Our algorithm will consider this constraint checking.

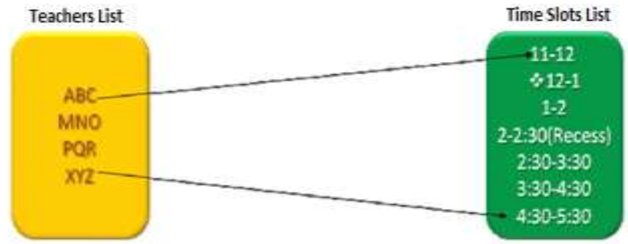

Fig 6: Teacher Timeslot Mapping

c. Teacher-Year Constraint

A Teacher can teach more than one year simultaneously. Algorithm must consider this association.

Fig 7: Teacher Year Mapping

When all data is entered in the all lists and when all constraints are mentioned by user. Then whole matrix is created in memory which helps algorithm to check constraints and perform allocation.

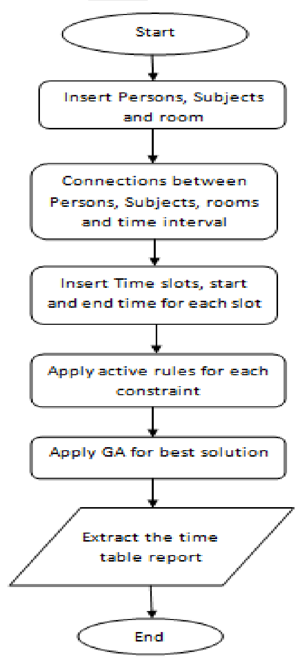

Fig 8: The structure of time table generator [7]

RESULT

Working with real data in real situation was our goal. So we used the data from the Department of Information Technology, Atharva College of Engineering, University of Mumbai. We've entered the raw data for a semester (courses being offered with their corresponding professors). Our implementation which is developed in C# used a .NET framework for representing chromosomes, generations and processing the evolution. We ran our tests on a dual core 2 GHz CPU with 2 GB of RAM.

Atharva College Of Engineering

Regular	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
9.00 To 10.00	Computer Networks Practicals |	Web Programming Practicals |	Applied Mathemactics-IV |Navita Agraval	Applied Mathemactics-IV |Navita Agraval	Applied Mathemactics-IV |Navita Agraval
10:00 To 11:00	Computer Networks Practicals |	Web Programming Practicals |	Applied Mathemactics-IV |Navita Agraval	Computer Networks |Suvarna Jadhav	Computer Networks |Suvarna Jadhav
11:00 To 12:00	Computer Networks |Suvarna Jadhav	Computer Networks |Suvarna Jadhav	Computer Organization&Architecture |Sumita Chandak	Computer Organization&Architecture |Sumita Chandak	Computer Organization&Architecture |Sumita Chandak
12:00 To 13:00	Computer Organization&Architecture |Sumita Chandak	Automata Theory |Mamata Ghelot	Automata Theory |Mamata Ghelot	Automata Theory |Mamata Ghelot	Web Programming |Poonam Joshi
13:00 To 14:00	LUNCH BREAK	LUNCH BREAK	LUNCH BREAK	LUNCH BREAK	LUNCH BREAK
14:00 To 15:00	Web Programming |Poonam Joshi	Web Programming |Poonam Joshi	Web Programming |Poonam Joshi	Information Theory& Coding |Reena Mahe	Information Theory& Coding |Reena Mahe
15:00 To 16:00	Information Theory& Coding |Reena Mahe	Information Theory& Coding |Reena Mahe	Applied Mathemactics-IV Tutorials |Yogini Bazaz	Automata Practicals |Snighdha W	Information Theory & Coding Practicals |Jayshree Jha
16:00 To 17:00	|	|	|	|	|

Fig 9: Result

FUTURE SCOPE

The future scope of the implemented system is listed below which can be applied to the system with further studies:

- The application can generate time tables of various departments of an institute simultaneously on single system.
- Online time tabling is also possible. But the administrator will have the control over generating timetables, to avoid the misuse of the given access to students as well as faculty.
- In online time table scheduling application, the details and qualifications of the professors and details of student identity and attendance can be viewed by clicking on particular subject or professor and classes respectively.
- The time table of all the classrooms containing respective lectures, professors and semesters in the entire day can be viewed. The vacancy of the classrooms and labs can also be shown.
- The time table for faculties and labs can also be generated independently.

ACKNOWLEDGMENT

We are thankful to Prof. Nileema Pathak, Prof. Sumita Chandak and Prof. Renuka Nagpure, Department of Information Technology, Atharva college of Engineering.

CONCLUSION

As discussed, an evolutionary algorithm, genetics algorithm for time tabling has been implemented. The intention of the algorithm to generate a time-table schedule automatically is satisfied. The algorithm incorporates a number of techniques, aimed to improve the efficiency of the search operation. By automating this process with the help of computer assistance timetable generator can save a lot of precious time of administrators who are involved in creating and managing various timetables of the institutes. Also the timetables generated are much more accurate, precise than the ones created manually. The project reduces time consumption and the pain in framing the timetable manually. The benefits of this approach are simplified design and reduced development time.

REFERENCES:

[1] M. Doulaty, M. R. Feizi Derakhshi, and M. Abdi, "Timetabling: A State-of-the-Art Evolutionary Approach", *International Journal of Machine Learning and Computing*, Vol. 3, No. 3, June 2013.

[2] Anirudha Nanda, Manisha P. Pai, and Abhijeet Gole, "An Algorithm to Automatically Generate Schedule for School Lectures Using a Heuristic Approach", *International Journal of Machine Learning and Computing*, Vol. 2, No. 4, August 2012

[3] Dilip Datta, Kalyanmoy Deb, Carlos M. Fonseca, "Solving Class Timetabling Problem of IIT Kanpur using Multi-Objective Evolutionary Algorithm". *KanGAL* 2005.

[4] Anuja Chowdhary, Priyanka Kakde, Shruti Dhoke, Sonali Ingle,Rupal Rushiya, Dinesh Gawande, 'Time table Generation System", *International Journal of Computer Science and Mobile Computing*, Vol.3 Issue.2, February- 2014.

[5] Mughda Kishor Patil, Rakhe Shruti Subodh, Prachi Ashok Pawar, Naveena Narendrasingh Turkar, "Web Application for Automatic Time Table Generation", *International Journal of current Engineering and Technology*, E-ISSN 2277-4106, P-ISSN 2347-5161.

[6] Sandeep Singh Rawat, Lakshmi Rajamani, "A Time table Prediction for Technical Educational System using Genetic Algorithm", *Journal of Theoretical and Applied Information Technology*, 2005-2010JATIT.

[7] Bharka Narang, Ambika Gupta, Rashmi Bansal, "Use of Active Rules and Genetic Algorithm to Generate the Automatic Time-Table", *International Journal of Advances in Engineering Sciences*, Vol.3 (3).

[8] Alberto Colorni, Marco Dorigo, Vittorio Maniezzo, "A Genetic Algorithm to Solve the Time Table Problem", *Computonal Optimization and Applications Journal*.

[9] Rupert Weare, Edmund Burke and Dave Elliman, "A Hybrid Algorithm for Highly Constrained Timetabling Problem", *Computer Science Technical Report* No. NOTTCS-TR-1995-8.

[10] D. Abramson, J.Abela,"A Parallel Genetic Algorithm for Solving the School Timetabling Problem", *15 Australian Computer Science Conference, Hobart*, Feb 1992.

[11] Dipesh Mittal, Hiral Doshi, Mohammed Sunasra, Renuka Nagpure "AutomaticTimetable Generation Using Genetic Algorithm" *International Journal of Advance Research in Computer and Communication Engineering*. Vol. 4, Issue 2, Feb 2015.

[12] Melanie Mitchell, "An Introduction To Genetic Algorithm", *A Bradford Book The MIT Press*, Fifth printing 1999.

[13] David A.Coley, "An Introduction To Genetic Algorithms For Scientists And Engineers",World *Scientific Publishing Co. Pvt. Ltd.*,1999