ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

Genetic Algorithm With Meta- Heuristic Approach For Generating Timetable

Anam Shaikh[1], Tazeen Ansari[2], Faizan Baig[3], Anand Bali[4].

1. Dept. of Computer Engineering, M.H. Saboo Siddik College of Engineering, Mumbai, India.
2. Dept. of Computer Engineering, M.H. Saboo Siddik College of Engineering, Mumbai, India.
3. Dept. of Computer Engineering, M.H. Saboo Siddik College of Engineering, Mumbai, India.
4. Dept. of Computer Engineering, M.H. Saboo Siddik College of Engineering, Mumbai, India.

Manuscript Info

Manuscript History:

Received: 14 December 2015
Final Accepted: 26 January 2016
Published Online: February 2016

Key words:
Question Genetic Algorithm, Local Search, Heuristic Crossover, Natural Selection.

Corresponding Author

Anam Shaikh.

Abstract

Timetable creation is very burdening and time consuming task. Scheduling course timetables for a large array of courses is very complicated problem which often has to be solved by the center staff manually even though the result are not always satisfying. In this paper Genetic Algorithm along with heuristic approach is used to schedule timetable which includes various entities of university such as rooms, curriculum, teachers, time on which a local search is performed and it also ensures that fundamental constraints are not violated. Genetic Algorithm are heuristic search algorithm which is based on the ideas of natural selection. The algorithm repeatedly mutates a population of individual solutions. In each step, the algorithm calculates the fitness function of every individual and selects them randomly from the population to produce children for the next generation. Every successive generation drives towards an optimal solution.

Introduction:-

This paper elaborates the usage of Genetic Algorithms for finding optimized solutions to the problem of Timetabling. We are providing a detailed introduction to the topic of Genetic Algorithms- their history, methods and how to apply them to schedule optimal timetable. [4]

Almost all universities have problem concerning with scheduling. Many things have to be considered in order to generate schedule. One of them is availability of teachers, faculties, rooms, time slots etc. and their availability.

Since every university has its own course scheduling problem, the commercially available applications may not suit the need of every college. Hence the need is to develop a practical approach for building course timetabling system, which can be personalized to fit to any university timetabling problem. The university course timetabling problem asks us to find some time slots and rooms which satisfy the constraints imposed on offered.

The constraints are classified in hard constraints and soft constraints. Hard constraints are mandatory. They need to be fulfilled necessarily. Soft constraints can be violated if necessary. They do not need to be really satisfied but the solutions are considered optimized if large numbers of them are taken care.

Genetic algorithm:-

Genetic algorithms are methods of solving problems based upon an abstraction of the process of Natural Selection. They attempt to mimic nature by evolving solutions to problems rather than designing them. Genetic algorithms work by analogy with Natural Selection as follows. First, a population pool of chromosomes is maintained. The chromosomes are strings of symbols or numbers. There is good precedence for this since humans are defined in DNA using a four-symbol alphabet. The chromosomes are also called the genotype (the coding of the solution), as opposed to the phenotype (the solution itself). In the Genetic algorithm, a pool of chromosomes is maintained,

which are strings. These chromosomes must be evaluated for fitness. Poor solutions are purged and small changes are made to existing solutions and then allow "natural selection" to take its course, evolving the gene pool so that steadily better solutions are discovered.

The basic outline of a Genetic Algorithm is as follows: [3] Initialize pool randomly
For each generation {
Select good solutions to breed new population
Create new solutions from parents
Evaluate new solutions for fitness
Replace old population with new ones
}
The randomly assigned initial pool is presumably pretty poor. However, successive generations improve, for a number of reasons:
1) **Selection:** During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected[6]
2) **Mutation**: It allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution.[8]

```
for each gene in individual
{
 if(p(Random)< pm)
{
 gene = get random value from
 possible values list;
 }
 }
```

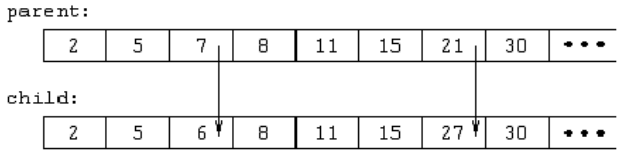

Figure 1: Mutation for Individual[2]

3) **Crossover**: It combines the genetic material from parents order to produce children, during breeding. Since only the good solutions are picked for breeding, during the selection procedure, the crossover operator mixes the genetic material, in order to produce children with even greater fitness.

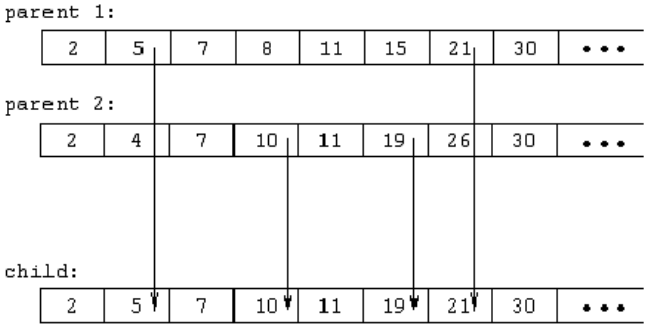

Figure 2: Crossover Individual[1]

For example, assume single point crossover at position 3 two binary chromosomes with values (000000, 111111) will produce (000111, 111000) as children. Moreover, there can be multiple point crossover.[3]
What we propose here, is an “automatic” way of selecting the best action to execute upon an event occurring? The action is selected by a genetic algorithm. For the moment conditions are supported by active rules when an event has

occurred the system can take several actions. For each of possible events, the system holds an ordered set of possible actions that can be taken when the event occurs. The first action is always selected, but a genetic algorithm running in parallel may dynamically change the order of the actions. [7]
Since the genetic algorithm controls the way the agents (constraints) respond to events, the reactive behavior of the agent is controlled by the genetic algorithm. But there can also be another "level" (the "rational" level) to control the agent, especially if a architecture is part of an agent built partially using another method and controlled partially by the constructs this method provides. Actions will be selected for execution using the traditional approach, but some others using the Genetic Algorithm approach. This rational part of the agent can also control several parameters of the Genetic Algorithm, restart it when needed, or schedule it to be run. This architecture can also be embedded in more complex systems. When an event/action language is necessary for the building of an agent type system, this method can be used for a subset of the events and the actions of the system. This simplifies the design and reduces testing and maintenance times when compared to a deterministic rule set with many conditions and checks. [5]

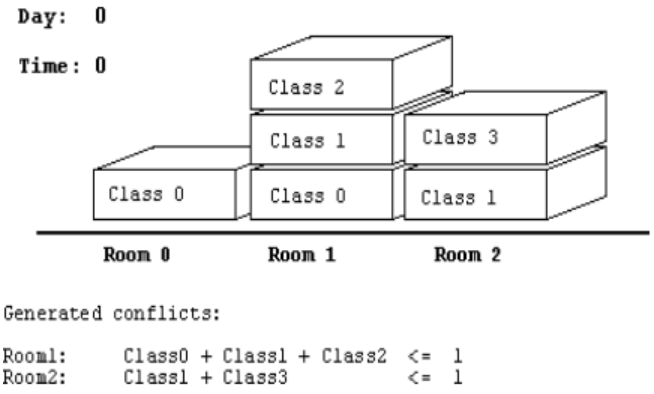

Figure 3: Generation of conflicts and bounds.[2]

I. PROPOSED SYSTEM

In order to schedule timetable, we are introducing a System, which would automatically generate feasible timetable for the institute. Teachers will be allotted to various courses in accordance with all possible constraints needed to be satisfied to generate appropriate timetable. These scheduling of timetable will be depended on two constraints i.e. Hard constraints and Soft constraints.

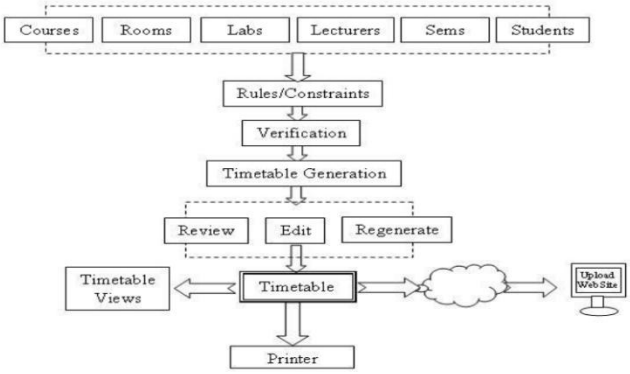

Figure 4: Timetable generation general view

Timetable generator Structure consists of various System Constraints, input data, relation between the input data, and various operations of genetics algorithm.

A. Input Data

The input data consist of:

1) Teachers: Data describes the various information about the lecturer such as name, identification number, related course.

2) Courses: Data describes the name of Subject in the current term.

3) Classroom: Data describes various rooms and their capacity.
4) Time slots: It indicates the total duration of the course along with the information of lecturer.
B. System Constraints
There are two types of System Constraints.
1) Hard Constraints:
Hard constraints are needed to be fully satisfied.
Following are the various hard constraints.
a. Classrooms must not be double booked.
b. Every class must be scheduled exactly once.
c. A classroom must be large enough to hold each class booked to it.
d. Lecturers must not be double booked.

2) Soft Constraints: These are constraints that are not that compulsory but still demanding. They not to be really satisfied but the solutions are considered good if large numbers of them are taken care.
Following Soft Constraints can be considered.

a. No consecutive lectures of the same lecturer in a class.
b. No consecutive lectures of the same teacher in a class
c. Courses must be evenly distributed
d. Same teacher must not have consecutive lectures unless specified

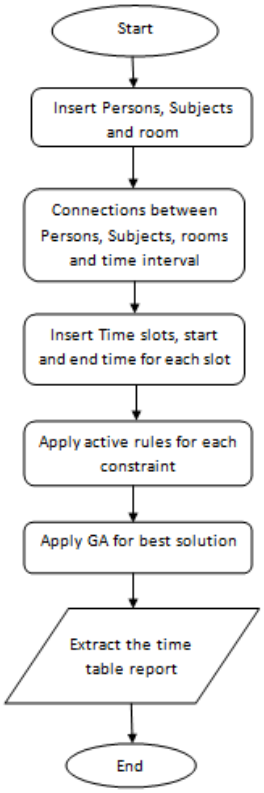

Figure 5: The Structure of timetable generator

Future scope:-

To generate timetable for the university with no human errors and which will be less time consuming along with high level of adaptability and precision. Moreover, improve the overall process of timetable scheduling with help of genetic algorithm along with the use of different technologies.
For Further work there is need to implement different types of genetic algorithms, such as heuristic approach to schedule the timetable. For example, the individual with overlapping populations such as steady state or incremental

Genetic Algorithm. In such cases, that the Genetic Algorithm could be used for generating the nodes at real time (once an initial good state has been attained) and not just training them. Plan to investigate other methods for finding the optimal rule set (for example, NN or other heuristic search methods like simulated annealing) and compare the results with manually generated results obtained by a statistical analysis of the network.

Conclusion:-

The Timetable Generation application will ease the process of timetable scheduling which may otherwise needed to be done manually by the center staffs possibly leading to constraints problem that are difficult to determine when time table is generated. The intention of algorithm is to generate the time table automatically is satisfied. To improve the efficiency search operation the algorithm incorporates lots of techniques. It also addresses the important hard constraint.

Acknowledgment:-

Our thanks to M.H. Saboo Siddik College of Engineering, Department of Computer Engineering, for giving us the initiative to do constructive work. We also thank anonymous reviewers for their constructive suggestions.

References:-

[1] J. J. Grefenstette, editor. Proceedings of the Second International Conference on Genetic Algorithms and their Applications. Practice and Theory of Automated Timetabling VI Proceedings of The 6thInternational Conference on the Practice and Theory of Auto

[2] Solving Timetable Scheduling Problem by Using Genetic Algorithms Branimir Sigl, Marin Golub, Vedran Mornar Faculty of Electrical Engineering and Computing, University of Zagreb Unska 3, 10000 Zagreb, Croatia

[3] Sanjay R. Sutar , Rajan S. Bichkar "University Timetabling based on Hard Constraints using Genetic Algorithm" Available : http://research.ijcaonline.org/volume42/number15/pxc3877964.pdf

[4] Alberto Colorni, Marco Dorigo "A Genetic Algorithm to solve the time table problem" Available : http://citeseerx.ist.psu.edu

[5] Om Prakash Shukla, Amit Bahekar, Jaya Vijayvergiya, "Effective Fault Diagnosis and Maintenance Optimization by Genetic Algorithm" Available : http://researchjournals.in/documents/published/2204.pdf

[6] Cite Seerx , Optimisation of Active Rule Agents using a Genetic Algorithm approach, Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.2599

[7] Leon Bambrick, "Lecture Timetabling Using Genetic Algorithms" Available : http://secretgeek.net/content/bambrilg.pdf

[8] Genetic Algorithms, Conclusion and Future WorkAvailable: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/tcw2/report.html