International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 2 Issue: 11 3480 – 3483

Genetic Algorithm to Generate the Automatic Time-Table – An Over View

Asif Ansari[#1], Prof Sachin Bojewar[#2]

[#1]PG Scholar, Alamuri Ratnamala Institute of Engineering & Technology, Mumbai, India
[#2]Associate Professor, Vidyalankar Institute of Engineering & Technology, Mumbai, India
[#1]*ansariasif23@gmail.com*, [#2]*sachin.bojewar@vit.edu.in*

***Abstract*--**In this paper we glance through the various approaches used by the researchers to develop an automatic timetable using Genetic algorithms. The optimized genetic algorithm can be used with the heuristic approach to design and develop the timetable of an institute. At stake during the process of development, the stakeholders are the professors and the students. The efficient utilization of the infrastructure is the main aim of the authors. The crossover, mutation and the fitness function is to be calculated for the implementation. In genetic algorithm every individual are characterized by a fitness function. After analysis if there is higher fitness then it means better solution and then after based on their fitness, parents are selected to reproduce offspring for a new generation where fitter individuals have more chance to reproduce. The objective of the work is to create a model used to generate the acceptable schedule using probabilistic operators.

Keywords*—Rule-Based agents, Genetic Algorithm, fitness function, Timetable Generator, Heuristic approach.*

*****

I. INTRODUCTION

Planning timetable is one of the most complex and error prone application. There are still serious problems like generation of high cost time table are occurring while scheduling and these problems are repeating frequently. [6] Therefore there is a great requirement for an application distributing the course evenly and without collisions. The aim is here to develop a simple, easily understandable, efficient and portable application which could automatically generate good quality time table with in a second. [10] The outline of this paper is as follows: Active rules are described for the knowledge of intelligent agents (i.e. Constraints), GAs are described and their use in optimizing rule based agent is proposed, methods are apply to the problem of optimizing some results of this application are presented and finally, some conclusion and possible direction for future research are presented. A lecture timetable problem is concerned with finding the exact time allocation within limited time period of number of events (courses-lectures) and assigning to them number of resources (teachers, students and Lecture Halls) while satisfying some constraints. The constraints are classified into Hard Constraints and Soft constraints. Hard constraints are those that must be adhered to, while soft Constraints can be violated if necessary [2,3].

The advantage of GA is that they can explore the solution space in multiple directions at once [4].

Therefore, if one path turns out to be a dead end, they can easily eliminate it and continue work on more promising avenues, giving them a greater chance each run of finding the optimal solution

II. STRUCTURE OF THE AUTOMATED TIME TABLE GENERATOR

The structure of time table generator consist Input Date Module, relation between the input data module, time interval, time slots module, applying active rules and GA module then extract the reports.

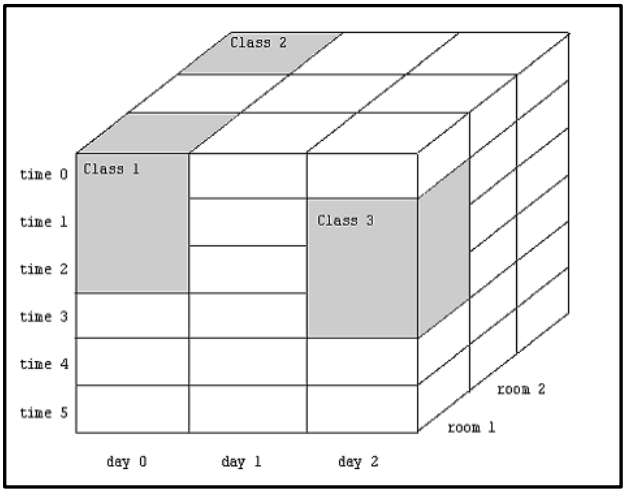

Figure 1: Time Table presented as 3 D Structure [1]

III. GENETIC ALGORITHM [5]

Genetic algorithms are methods of solving problems based upon an abstraction of the process of Natural Selection.

 ISSN: 2321-8169
3480 – 3483

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: [8]
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[12]
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.[14]

for each gene inindividual{
if($p_{(Random)} < p_m$){
gene = get random value from
possible values list;
}
}

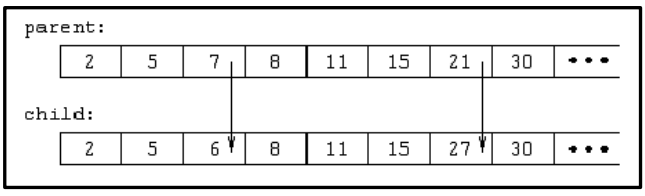

Figure 2: Mutation for Indiviual[1]

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.

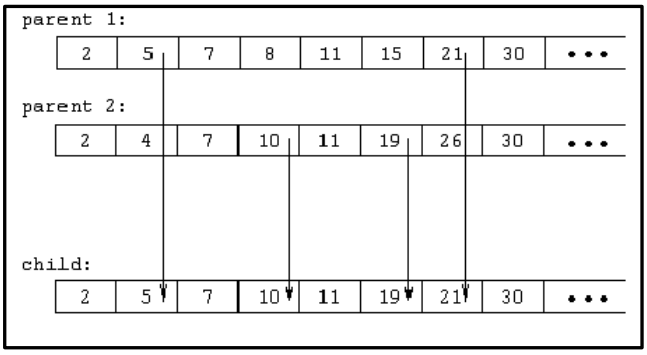

Figure 3: Crossover Indiviual[1]

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.[8]
What 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. [10]

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 GA approach. This rational part of the agent can also control several parameters of the GA, 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

 ISSN: 2321-8169
3480 – 3483

and maintenance times when compared to a deterministic ruleset with many conditions and checks. [5]

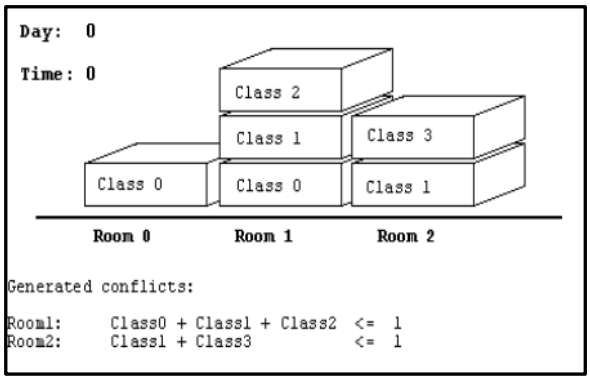

Figure 4: Generation of conflicts and bounds [1]

IV. CONCLUSION

The GA in timetabling framework has been shown to be successful on several real problems .It has been shown that the genetic algorithm perform better in finding areas of interest even in a complex, real-world scene. [13] This paper described how set of active rules can beused to express the knowledge of intelligent and how a genetic algorithm can be used to dynamically prioritize rules in the face of dynamically evolving environments. One could argue that the genetic algorithm can finda local optimum and then stop. This is always a danger with a genetic algorithm, but again it depends on the search space. In this time table generation approach, there are many good solutions and the genetic algorithm will find one of them.

In extreme cases where there is only one good solution the genetic algorithm may fail, but again it can be restarted by the Active Rules with many chances to find a better solution. [8] One could also argue that this architecture is not powerful enough since it does not work based on an event/action language. However there is nothing to prevent this architecture from being a subset of a rich and powerful event/action language. In such a case it can be used to pick the rule to be fired when there are no other criteria available for rule selection. In other cases it may be better to let the genetic algorithm pick the rule to be fired, instead of having many conditions which will complicate the active rule set and consequently increase design, test and maintenance times. The benefits of this approach are simplified design and reduced development and maintenance times of rule-based agents in the face of dynamically evolving environments. [11]

V. FUTURE SCOPE

For Further work there is need to explore different types of genetic algorithms, like heuristic approach to develop the application. for example ones with overlapping populations such as steady state or incremental GAs. In such cases, a small replacement percentage, so that the GA could be used for driving the nodes at real time (once an initial good state has been reached) and not just training them For plan to investigate other methods for finding the optimum rule set (for example, neural networks or other heuristic search methods like simulated annealing) and to formally compare the results with theoretical results obtained by a statistical analysis of the network. [11]

REFERENCES

[1]. 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

[2]. J. J. Grefenstette, editor. Proceedings of the First 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 Automa.

[3]. 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.

[4]. N. R. Jennings.Coordination Techniques for Distributed Artificial Intelligence. University of London Mile End Rd.London E1 4NS UK, 1995.

[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]. Leon Bambrick Supervisor Dr B Lovell “Lecture Timetabling Using Genetic Algorithms” Available : http://secretgeek.net/content/bambrilg.pdf

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

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

[9]. Eng. Ahmed Hamdi Abu ABSA, Dr. Sana'a Wafa Al-Sayegh, ” Elearning Timetable Generator Using Genetic Algorithms” Available : https://uqu.edu.sa/files2/tiny_mce/plugins /filemanager/files/30/papers/f18 9.pdf

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

[11]. Evaggelos Nonas, Alexandra Poulovassilis, ”optimisation of active rule agents using a genetic algorithm approach pdf”Available : http://ebookbrowse.com/optimisation-of-active-rule-agentsusing-a-genetic-algorithm-approach-pdf-d381872402

[12]. 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

[13]. Wikipedia, Selection (genetic algorithm), Available: http://en.wikipedia.org/wiki/Genetic_algorithm

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

[15]. Wikipedia,Mutation (genetic algorithm) Available: http://en.wikipedia.org /wiki/Mutation_%28genetic_algorithm%29