International Journal of Enhanced Research in Management & Computer Applications
ISSN: 2319-7471, Vol. 4 Issue 8, August-2015

Exam Time Table Scheduling using Genetic Algorithm

Manoj Kr. Mahto[1], Mr. Lokesh Kumar[2]

[1]M. Tech. Student, CSE Dept., RIEM, MDU, Rohtak, Haryana
[2]Asst Prof. & HOD, CSE Dept., RIEM, MDU, Rohtak, Haryana

ABSTRACT

Exam scheduling problems are NP-complete problems. This problem has been studied by numerous researchers due to its NP complete nature. The exam scheduling problem is a typical combinatorial optimization problem. The method of using the traditional and manual exam arrangement was time consuming and laborious. Therefore it is very need to arrange the exam with the help of modern computerized optimization techniques. Exam scheduling is a multi-constrained problem. The essential of the exam-arrangement problem is arranging the class, the courses and the classroom in a time. There should be no conflict in the arrangement and it should meet a number of conditions. The purpose is to ensure the arrangement according with the requirement that the exam class and the exam courses required about the time and the space, reaching the global optimum as far as possible. The exam timetabling problem requires the assignment of a given number of exams within a fixed amount of time. Exam time scheduling is a multi-constrained problem. The main aim is to produce the most optimized schedule for examinations under given set of constraints like Maximum seat capacity must be taken into account during exam period, Maximum number of days for exam completion should be scheduled, Days on which exam can't be scheduled (such as Saturday, Sunday), In a day for same student only one exam should be scheduled, Two exams must not be scheduled at the same time for same student.

Keywords: NP Complete, combinatorial optimization problem, multi-constrained problem.

1. INTRODUCTION

Exam scheduling problems are NP-complete problems. This problem has been studied by numerous researchers due to its NP complete nature. The exam scheduling problem is a typical combinatorial optimization problem. The method of using the traditional and manual exam arrangement was time consuming and laborious. Therefore it is very need to arrange the exam with the help of modern computerized optimization techniques. Exam scheduling is a multi-constrained problem. The essential of the exam-arrangement problem is arranging the class, the courses and the classroom in a time. There should be no conflict in the arrangement and it should meet a number of conditions. The purpose is to ensure the arrangement according with the requirement that the exam class and the exam courses required about the time and the space, reaching the global optimum as far as possible. The exam timetabling problem requires the assignment of a given number of exams within a fixed amount of time. Exam time scheduling is a multi-constrained problem. The main aim is to produce the most optimized schedule for examinations under given set of constraints like Maximum seat capacity must be taken into account during exam period, Maximum number of days for exam completion should be scheduled, Days on which exam can't be scheduled (such as Saturday, Sunday), In a day for same student only one exam should be scheduled, Two exams must not be scheduled at the same time for same student.

2. RELATED WORK

Several works have approached the timetabling problem. Le Xiao et al. developed optimization technique for exam timetable generation. High school timetables were studied and prepared. Priority was given to the high school subjects and lower grade exams. Exam rooms were arranged properly according to the number of students. Large rooms were preferred and timetables were generated based on the conditions mentioned and its optimization was done using improved ant colony algorithm.

Oliveira et al., in year 2000, presented a language for timetable representation. UniLang was considered as the standard input language for timetable. Fernandes, in 2002, used strong and weak constraints for class teacher timetabling. Strong constraints like violation of schedule of teacher in the classes at same time and weak constraint like preference of teachers for certain hours was studied and timetables were tested for quality. Merlot et al., in the year 2003 proposed hybrid approach for solving the exam timetabling using stimulated annealing and with hill climbing to obtain a better solution. In the same year, Ender Ozcan et al. developed a tool named FES . This was the first tool which accepts TTML input. FES used hierarchical hill climbing method for generating the timetable.

Further, a new algorithm, named Greedy algorithm was used. Greedy method was used to made initial timetable and then genetic algorithms was used to optimize it. Course timetable considered various conditions like each teacher was not allowed to deliver more than one lesson in a period. This method was achieved by greedy heuristic starting by course assignments in an empty timetable. When there were too many courses, than two courses were combined for the proper utilization of rooms.

José JoaquimMoreira1, in year 2008, presented another optimized solution method for automated exam timetable construction. Use of the matrix for the construction of timetables was proposed and the method used was genetic algorithms to generate the optimized timetable on which the constraints are imposed. Again the constraints considered were availability of classrooms and preference of teachers.

After getting matrix concept for timetable generation using GA, Oluwasefunmi T. Arogundadeet. Al, in year 2010, explored this concept of using genetic algorithms for using the university timetable. Various constraints were taken into account like both the timeslots and the days assigned for exams. Room capacity for rooms was also included in the datasets specification. The numerical results were studied and table was made for rooms available and courses assigned . Tony Wong et al.made use of genetic algorithm for getting the optimized timetable. This generator had the ability to generate several quality timetables for analysis and selection. The factors considered mainly were that there were no exam conflicts; no student was allowed to give consecutive exams. Free periods between exams were considered by them. Its algorithmic implementation was simple and did not involve any special heuristics.

3. EXAM TIME TABLE

The resolution of the exam timetables problem can be claimed by different areas, such as the School Administration, Artificial Intelligence, Mathematics or Operational Research. Probably, we must appeal the techniques of simulation imported from fields as diverse as physics or biology, to solve the problem. The purpose of the exam timetable is scheduler exams, according to pre-defined periods of time; minimizing losses teaching for the students, such as realize examinations on the same day or on consecutive days. But here, it considers each student individually, since the choice may depend only of the route of each school students.

The importance of the constraints, the quantity and quality of which are, stems directly from the attempt to organize the problem. In this sense, we go classify, previously the constraints. Classified as constraints of the first order, or rigid, those are not being met, and it makes the scheduling illegal, calling themselves 'impossible solutions'. Other constraints, which should obey, and which, if not met, do not make illegal the scheduling, considered being of second order constraints, or flexible. So, we called the 'impossible solutions' the scheduling, that check the constraints of the first order, Regardless of check, or not, the constraints of second order.

This division represents two moments in the resolution of the exam timetables problem. The first, consisting in the search for possible solutions, in the development of heuristics to ensure that the scheduling chosen corresponds to a possible solution. The second, consisting in finding the best solution. The first runs in the space of all scheduling - which includes possible and impossible solutions; the second follows, just in the space of possible solutions.

4. GENETIC ALGORITHM

Genetic Algorithms are among several types of optimization methods that use a stochastic approach to randomly search for good solutions to a specified problem, including Simulated Annealing, Hill Climbing and numerous variations. It gives better results than various traditional methods. GA belongs to the class of probabilistic algorithms. GA is more robust than existing various search methods. Genetic Algorithms are adaptive methods which may be used to solve search and optimization problems[3]. They are based on the genetic processes of biological evolution. GA is a powerful technique in optimization problem problems. GA is a search algorithm based on a simple idea from biology "Survival of the Fittest". They have been used for many different applications including scheduling, predicting the stock market and creating the art etc. Genetic Algorithms start with an initial random population, and subsequently allocate more trials to regions of the search space and found to have high fitness. GA can be combined with hill-climbing techniques to speed up the search process, but are able to 'jump' from local maxima because the elements that promote a good solution are being mixed up in subsequent iterations.

- Comparison of GA from other Search methods

Goldberg (1989) presents four fundamental characteristic that differ GA from other traditional optimization and search procedures . These characteristics are:

- GA works with encoding of the parameter set, not the parameters themselves: GA require natural parameter set to be coded as finite length of strings. Coding can be done in several ways.

- GA searches from a population of points, not from a single point: In many optimization methods, we move from single point to next in space using some transition rule to find the next point. This point to point method may locate false peaks in multimodal search spaces. But GA works from a rich database of points simultaneously, climbing many peaks in parallel. Thus probability of finding false peak reduces.

- GA uses objective function information, not derivatives or other auxiliary knowledge: Many search methods require auxiliary information to work properly. GA has no need for such information. To perform effective search in GA for better and better results, only objective function values are needed which are associated with individual strings.

- GA uses probabilistic transition rules, not deterministic rules: GA use probabilistic transition rules to guide their search. GA use random choice as a tool to guide search.

- **Brief Introduction to GA**

GA developed by John Holland's in 1960 and further studied by De Jong's, Goldberg, Davis, Koza, Mitchell, an many more, have been proposed as a general model of adaptive processes. John Holland's book "Adaptation in natural and artificial systems" as well as De Jong's book ,"Adaptation of the behavior of a class of genetic adaptive systems," are seen as the foundation of Genetic Algorithms. Genetic algorithms are stochastic algorithms which are based on natural phenomenon. A genetic algorithm is an optimization tool used to solve optimization problems. Optimization problems attempt to find the best solution for a given problem that has several parameters with associated constraints. Genetic Algorithms are based on the evolutionary ideas of natural selection and genetics. Genetic algorithms are a part of evolutionary computing, inspired by Darwin's theory of evolution -"Survival of the fittest" . Genetic Algorithms is an

optimization technique which takes large, potentially huge population and finds optimal solution. Genetic algorithms are based on the concept of biological evolution .

Flow Chart of basic Genetic Algorithm :

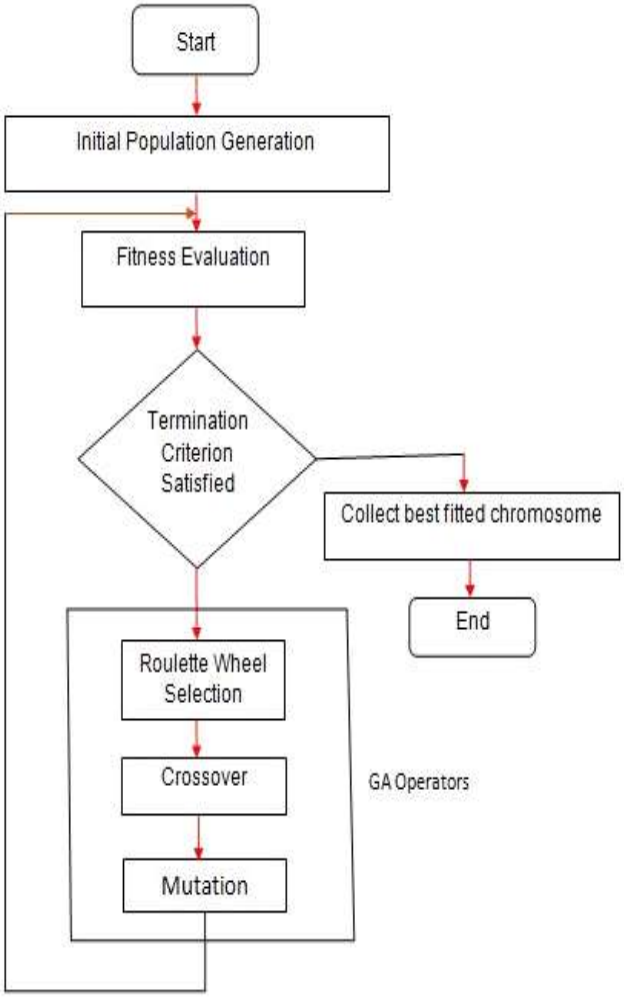

Figure 4: Flow chart of basic GA

Benefits of GA:

Some benefits of using GA are listed below:

- Concept is easy to understand
- Modular–separate from application (representation); building blocks can be used in hybrid applications.
- Supports multi-objective optimization
- Good for "noisy" environment
- Always results in an answer, which becomes better and better with time
- Can easily run in parallel
- The fitness function can be changed from iteration to iteration, which allows incorporating new data in the model if it becomes available

5. IMPLEMENTATION

The Exam Scheduling is implemented using the MATLAB which is very useful to be used with the evolutionary computing. MATLAB is efficient tool and is easy to implement.

Subroutines are programmed in MATLAB for

- Initial population generation
- Fitness value evaluation
- Roulette wheel selection operator
- Crossover operator
- Mutation operator

Subroutines are simulated for getting optimized schedule for exams. Comparison between the minimum fitted and the maximum fitted timetable will show the improved results after genetic action.

CONCLUSION

Having finishing the work, it has been evaluated that the Genetic Algorithms in Timetabling Problem in particular is interesting. It can effectively solve complex optimization problems. On the basis of various parameters and setting we have generated an optimized timetable with respect to given parameters with best selected on the basis of fitness value

REFERENCES

[1] Schaerf, et. al., "A survey of automated timetabling", Artificial intelligence review,1999, 13(2), 87-127.
[2] D.E Goldberg, "Genetic Algorithms in search, Optimization and machine learning", Addison-Wisley-1989.
[3] J. E. Smith, et. al., "Operator and parameter adaptation in genetic algorithms," Soft computing : a fusion of foundations, methodologies and applications, vol. 92, pp. 81–87, 1997.
[4] H. Holland," Adaptation in Natural and Artificial Systems" The University of Michigan Press, Ann Arbor, Michigan, 1975.
[5] M. Sipper. Machine Nature: The Coming Age of Bio-Inspired Computing. McGraw-Hill, New-York, first edition, 2002.
[6] Rakesh Kumar, "Study of selection and replacement on performance of genetic algorithms", IJRIM,Vol2, Issue 2 , February 2012, ISSN 2231-4334.
[7] Tomasz DominikGwiazda , " Genetic algorithm reference: Crossover for single objective numerical optimization problems"' volume-1, 2006.
[8] Omar Al Jadaan,et. al., "Improved Selection Operator For GA", Journal of Theoretical and Applied Information Technology, 2005 - 2008 JATIT.
[9] N. Chaiyarataiia and A. M. S. Zalzala, "Recent Developments in Evolutionary and Genetic Algorithms: Theory and Applications", Genetic Algorithms in Engineering Systems: Innovations and Applications, 2-4 September , 1999.
[10] Le Xiao, et. al., "An improved ant colony algorithm and its application in exam arrangement", 978- 1-4244- 5874, 2010 IEEE.