Automated Timetable Generator

Shweta Gajbhiye[1] Diksha Gondane[2] Nisha Shende[3]
[1]Assistant Professor [2,3]Final Year Student
[1,2,3]Department of Computer Science & Engineering
[1,2,3]Priyadarshini J.L College of Engineering - Nagpur, Maharashtra 440009, India

Abstract— Timetable creation is a very tedious and time consuming task; it takes lots of patience and man hours. Graph colouring technology helps to resolve this problem and introduces a practical timetabling algorithm capable of taking care of both hard and soft constraints effectively, used in an automated timetabling system for educational institutions. To resolve the complexity and to reduce the efforts of generating timetable manually, there are some other technologies like backtracking algorithm, Ant Colony Optimization, Bee Colony Optimization; Genetic Algorithm etc. are also available. Though these technologies provide the solution for timetable generation, these are slower and sometimes to provide the optimal solution there is a risk of losing the data when applied on large spaces. Graph colouring provides the exact solution for generating a conflicts-free timetable that too in optimal time.

Key words: Graph Colouring Algorithm, Timetable, Hard Constraints, Soft Constraints, Conflicts-Free, Optimal

I. INTRODUCTION

The Automated Timetable Generator System may be a sensible timetabling approach having capabilities and care of each laborious and soft constraints needed for getting ready timetable in faculties with sizable amount of scholars and restricted resources like class-rooms or labs. The automated time table scheduling process provides the simplest way to get a timetable for lecturers and student to look at their timetable once they're finalized, having individual login id and password for the admin. Some faculties usually takes the previous year's timetable to reduce the efforts for generating a time table and modify it but still it's a tedious job to include changes. It will be simple to use if any faculty is on leave or is not available on that particular time, he/she wants to engage his lecture with some other faculty who can engage their lecture and our project will be helpful to search those faculties for the same.

The Automated Timetable Generator is required as a result of timetable generation done manually by using pen and paper normally. As we know all Institutions or organizations have its own timetable, managing and maintaining these won't be troublesome. Workload consideration with this scheduling can create it a lot of complicated. When Timetable generation is being done, it should consider the maximum and minimum workload within a college. In these cases, timetable generation will become a lot of complicated. There are various technologies available today in order to generate Timetable like Ant Colony Optimization Algorithm, Backtracking Technique, Bee Colony Optimization, Graph Colouring Algorithm and Genetic Algorithm.

The methodology we have a tendency to use during this application is graph theory. The graph colouring algorithmic program may be a special case of graph labelling; it's an assignment of labels historically referred to as "Colours" to parts of a graph, subject to certain constraints. In its simplest type, it's some way of colouring the vertices of a graph such no two adjacent vertices share an equivalent colour. This is often referred to as a vertex colouring. Graph colouring is one in every of the foremost vital ideas in graph theory and is employed in several real time applications in computer science. Graph colouring employed in varied analysis areas of computer science such datamining, image segmentation, clustering, image capturing, networking etc. and vital applications such as Guarding an Art Gallery, Physical layout segmentation, Round-Robin Sports Scheduling, Aircraft scheduling, Biprocessor tasks, Frequency assignment, Final Exam Timetabling as a Grouping Problem, Map colouring and GSM mobile phone networks, and Student Time Table.

II. LITERATURE REVIEW

A. Backtracking Technique

Backtracking is a general algorithm for finding all (or some) solutions to some computational problems, notably constraint satisfaction problems that incrementally builds candidates to the solutions, and abandons each partial candidate c ("backtracks") as soon as it determines that c cannot possibly be completed to a valid solution.[2]

S.ALAGU LAKSHMI proposed "Timetable Automation Systems using Backtracking Technique." [2] They have generated the timetable for each class with condition of each subject should not appear at the same time for same year. They have checked each condition for over all subjects and finally generate timetable for separately. In their application small amount population have given a bad output as the amount of population can get smaller after the invalid chromosomes produced by the evolution get killed. In the other hand, large population gave better results but the system become slower as more chromosomes need to go through back tracking process. [2]

1) Strength:

- Simple to implement.
- Different states are stored into stack so that the data can be usable anytime.
- Easy to debug as every step got its own logical sequence.

2) Weakness:

- Multiple Function calls are expensive.
- The overall runtime of Backtracking algorithm is normally slow.
- To solve large problem sometime it need to take help of other techniques.

B. Ant Colony Optimization Algorithm

Ant Colony Optimization (ACO) is a metaheuristic approach for solving hard combinatorial optimization problems. It's an evolutionary method inspired from the foraging behaviour of real ants that enables them to find

shortest paths between a food source and their nest. This method differs from other evolutionary methods such as, Genetic Algorithms (GA) and Scatter Search (SS) by the fact that at each stage of the solution construction, it takes into account information resulting from the preceding iterations and the desirability of each element that can be added to the current solution Moreover, in an ACO algorithm, simple agents called artificial ants communicate indirectly to find good solutions for the optimization problem.[3]

Priyanka Gore proposed "Timetable Generation Using Ant Colony Optimization Algorithm." [3] This project reduced time consumption for the generation of timetable. The system described was successfully applied to the laboratory exercises timetabling problem at the institution. But it takes more time as it works on iteration and it is used to find the shortest path between source and destination which is time consuming process. [3]

Vatroslav Dino Matija proposed University Course Timetabling Using ACO: A Case Study on Laboratory Exercises. [8] This paper presents a study of applying ACO metaheuristic for determination of complex large scale timetabling problem at an institution. They gift associate degree innovative, memory efficient problem that is applicable for large problem instances. It makes the system manageable, which is extraordinarily vital for sensible timetabling applications. The approach represented here isn't restricted to LETP, since it shares several commonalities with different UCTP instances. Since their previous work on ant colony optimisation technique mainly focused on generated UCTP instances artificially, their work proves that ACO can be successful in solving real world timetabling problems. Some problem instance have a solution where all students are scheduled could not be found. Their Process of planning is usually an iterative process of querying the system for the most effective results, interpreting those results, and permitting the employees to form familiar choices. The performance of those approaches cannot be directly compared since completely different quality measures and optimised variables square measure utilized in these approaches. [8]

1) Strength:

- Positive feedback account for discovery of good solution.
- Distributed computation avoids premature conversion.
- Can be used in dynamic application

2) Weakness:

- Coding is somewhat complicated, not straightforward.
- Convergence is guaranteed, but time to convergence is uncertain.
- Theoretical analysis is difficult due to sequences of random decisions.

C. Bee Colony Optimization

Artificial bee colony optimization algorithm (ABC) is based on the intelligent for aging behaviour of honey bees and proposed by Karaboga in 2005. The Artificial Bee Colony (ABC) is a meta-heuristic algorithm which is population based stochastic method which is derived and motivated by the behaviour of honey bees. [4]

Deeptimanta Ojha proposed "Automated Timetable Generation using Bee Colony Optimization." [4] They have solved complex problems like course timetable problem. This work discusses Bee Colony Optimization (BCO) optimal solutions for designing bees which reduces the problem of computational solution and also is able to satisfy teachers and classes toward the schedule conflicts between the teacher's schedule, and the classroom schedules. For any the class schedules the timetable generation code was executed several times to get an optimal result using Bee Colony Algorithm. 10 different bees are used to produce 10 new source positions or candidate solutions. In each food source position or candidate solution is selected and position is memorized. [4]

Rajesh Kumar Sahoo proposed "AUTOMATIC GENERATION AND OPTIMIZATION OF COURSE TIMETABLE USING A HYBRID APPRAOCH."[7]. This paper planned a technique of candidate solutions for timetable generation and optimized by various EA like Firefly Algorithm (FA), Bee colony algorithm (BCA) and to get hybrid approach by combining FA and BCA collectively known as BCFA. Here total population of the candidate answer is divided into 2 elements. One part of the solution undergoes Bee Colony Optimization (BCO) and another part undergoes Firefly Algorithm (FA). A hybrid firefly algorithm (BCFA) gives higher result as compared to others algorithms. This work also finds the optimal solution to design a course time table. BCFA are designed on the basis of timeslots which reduces the time complexity of the problem. This proposed problem improves the teachers' satisfaction and class schedule of teachers. The hybrid BCFA produces the result in much less time and takes less iteration to achieve optimization. [7]

1) Strength:

- Ability to explore local solution.
- Easy to implement.
- The structure of the algorithm is favourable for parallel processing thus saving time.

2) Weakness:

- The possibility of losing relevant information on the behaviour of the function to be optimized.
- Slow down when used in sequential processing
- Slow to obtain accurate solution.

D. Genetic Algorithm

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. They can be used as techniques for solving complex problems and for searching of large problem spaces. [5]

Dipesh Mittal proposed "Automatic Timetable Generation using Genetic Algorithm."[5] This algorithm reduces time consumption and the pain in framing the timetable manually. The approaches of this algorithm are simplified design and reduced development time. This project is little time consuming and consist of Human errors along with low level of efficiency. Once a timetable is generated, it cannot modify. [5]

Spyros Kazarlis proposed "Solving University Timetabling Problems Using Advanced Genetic Algorithms."[6]

In this paper they present an advanced GA based method for solving university course timetabling problems. The GA method proposed uses an indirect representation featuring event allocation priorities, and invokes a "timetable builder" routine for constructing the complete timetable. It also uses a number of advanced local search operators, including the Micro-GA combinatorial hill-climbing operator, in order to avoid local optima, fulfil constraints and discover optimal solutions efficiently. It has been seen that evolutionary method does not manage to satisfy all hard constraints of the problem, it achieves a significantly better score in satisfying soft constraints and, therefore, its performance can be characterized as promising. GA's inability to satisfy all hard constraints may be attributed to the difficulty of the specific problem and to the limited resources it used during the experimental simulations. It seems like the man-made solution was the outcome of a focused effort to satisfy hard constraints, while soft constraints haven't enjoyed much attention. On the other hand the GA's solution is well developed concerning soft constraints but fails to be 100% valid. [6]

1) Strength:

- Inherently parallel; easily distributed.
- Chances of getting optimal solution are more.

2) Weakness:

- They cannot always find the exact solution but they always find best solution.
- GAs is very slow.

III. METHODOLOGY

A. Graph Colouring Algorithm

In graph theory, The graph colouring algorithmic program may be a special case of graph labelling; it's an assignment of labels historically referred to as "Colours" to parts of a graph, subject to certain constraints. In its simplest type, it's some way of colouring the vertices of a graph such no two adjacent vertices share an equivalent colour. This is often referred to as a vertex colouring. Graph colouring is one in every of the foremost vital ideas in graph theory and is employed in several real time applications in computer science. Graph colouring employed in varied analysis areas of computer science such datamining, image segmentation, clustering, image capturing, networking etc. and vital applications such as Guarding an Art Gallery, Physical layout segmentation, Round-Robin Sports Scheduling, Aircraft scheduling, Biprocessor tasks, Frequency assignment, Final Exam Timetabling as a Grouping Problem, Map colouring and GSM mobile phone networks, and Student Time Table.[1]

REFERENCES

[1] Dr. Cauvery N K, "Timetable Scheduling using Graph Coloring." Vol. 1 Issue 2 - 2011.
[2] S.ALAGU LAKSHMI "Timetable Automation Systems using Backtracking Technique." Vol. 2, Special Issue 15, March 2016.
[3] Priyanka Gore "Timetable Generation Using Ant Colony Optimization Algorithm." Vol. 5, Issue 3, March 2017.
[4] Deeptimanta Ojha, Rajesh Kumar Sahoo, Satyabrata Das. "Automated Timetable Generation using Bee Colony Optimization." Vol. 10 – No.9, May 2016.
[5] Dipesh Mittal, Hiral Doshi, Mohammed Sunasra, Renuka Nagpure. "Automatic Timetable Generation using Genetic Algorithm." Vol. 4, Issue 2, February 2015.
[6] Spyros Kazarlis, Vassilious and Pavlina Fragkou. "Solving University Timetabling Problems Using Advanced Genetic Algorithm."
[7] Rajesh Kumar Sahoo, Deeptimanta Ojha, Durga Prasad Mohapatra, Manas Ranjan Patra,"AUTOMATIC GENERATION AND OPTIMIZATION OF COURSE TIMETABLE USING A HYBRID APPRAOCH". Vol.95. No.1, 15 January 2017.
[8] Vatroslav Dino Matija, Goran Molnar, Marko _Cupi_c, Domagoj Jakobovi_c, and Bojana Dalbelo Ba_si_ "University Course Timetabling Using ACO: A Case Study on Laboratory Exercises".

