Schedule Shaper Using "Genetic Algorithm"

MINAL SONSARE[1], SHAMAL BAGDE[2], SAMEER BISEN[3], UJJWAL THAKRE[4], SHREYASH KHANZODE[5], PROF. SUPRIYA JAWLE[6] , PROF. GIRISH UMARATKAR[7]

[1, 2, 3, 4, 5] *Student, Department of Artificial Intelligence and Data Science, PCE, Nagpur, RTMN University*

[6, 7] *Lecturer, Department of Artificial Intelligence and Data Science, Priyadarshini college of Engineering, Nagpur, RTMN University.*

Abstract— For educators, creating time tables is a laborious task that requires a lot of manpower and time. It will be easier to generate time tables automatically if an automatic time table generator is available. Our project's proposed system will aid in its automatic generation and time-saving. The intricacy of establishing and overseeing timetable manually. We plan to employ resource scheduling, heuristic, and genetic algorithms in our project to lessen the challenges associated with creating timetables. Several strategies are incorporated into these algorithms with the goal of making the search process more efficient. The number of subjects, teachers, workload of a teacher, semester, and subject priority are just a few of the inputs that the system will receive. It will produce potential schedules for the working days of the week for teaching faculty based on these inputs. This will integrate by utilising all resources as efficiently as possible while taking into account the limitations.

I. INTRODUCTION

1.1 OVERVIEW: This project provides an example of how to use Genetic Algorithms (GAs) to solve the Timetable scheduling problem optimally. There are two objectives in this. First, to provide a detailed introduction to the topic of Genetic Algorithms, their method and their variations. The second objective to solve the Automated Timetable scheduling problem is the second goal. In this project, configuration file is given to genetic algorithm as an input and genetic algorithm resolves conflict and generate a timetable automatically.

1.2 BRIEF DESCRIPTION: A timetable scheduling using genetic algorithm which resolves all the conflicts and clashes in timetable efficiently. There is a set of valid results for every problem. This is said to form the solution space. In an optimization problem, the goal is to find results that maximize a set of criteria. A Genetic Algorithm is a method for efficiently searching the solution space. Timetable scheduling is complex optimization problem. Such problems where no efficient algorithm is known are ideal applications for genetic algorithms to be used to search a solution space. It's also critical to acknowledge that this kind of scheduling is an actual issue with immediate applications in timetabling.

II. SYSTEM ARCITECTURE

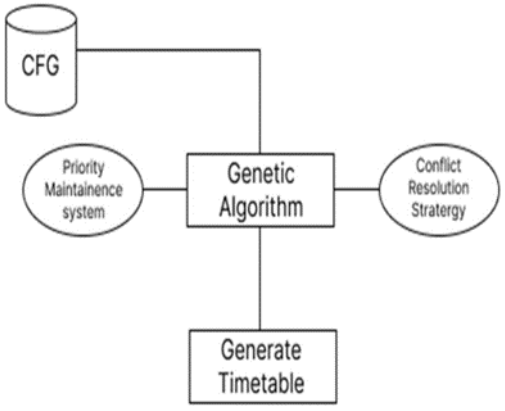

Fig. System Architecture

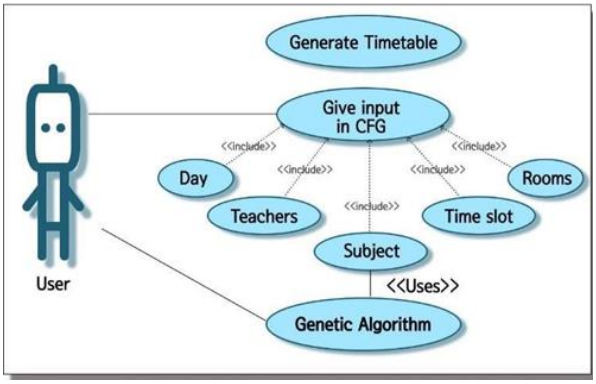

Fig.2 USECASE DIAGRAM

2.2 ALGORITHM:

The general genetic algorithm is:

population = randomly generated chromosomes; while (terminating condition is not reached) { run GA();}
// a single run of a genetic algorithm function Run GA() {
parents = getParents(); child = crossover(parents); child = mutate(child); population.add(child);
}

III. GENETIC ALGORITHM FLOWCHART

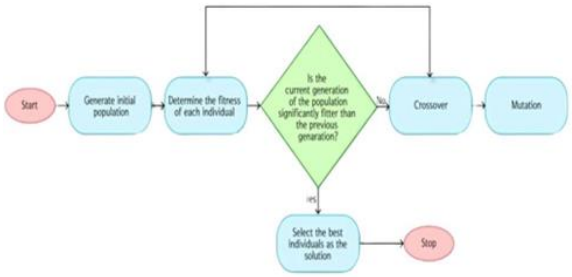

Fig .3 Genetic Algorithm Flowchart

IV. GA OPERATORS

4.1 CHROMOSOME REPRESENTATION:
A chromosome is a set of parameters that define a proposed solution to a genetic algorithm's problem, often represented as a simple string, and its fitness depends on its ability to solve the problem.

4.2 Initial population:
The first step in the functioning of a GA is the generation of an initial population. The GA algorithm uses an initial population of possible solutions, each evaluated and assigned a fitness value based on the fitness function. A good initial population increases the algorithm's chances of finding a suitable solution. 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.

4.3 SELECTION:
operator selects chromosomes in the population for reproduction. A chromosome is more likely to be selected for reproduction multiple times the more fit it is.

4.4 CROSSOVER:
Crossover is a genetic operator used in genetic algorithms to alter chromosome programming from one generation to the next, similar to reproduction and biological crossover. It involves taking multiple parent solutions and producing a child solution. There are methods for selection of the chromosomes. The operator randomly selects a locus and exchanges the subsequences before and after it between two chromosomes to produce two offspring.

4.5 MUTATION:
Mutation is a genetic operator utilized to maintain genetic diversity in a population of chromosomes from one generation to the next.. It is analogous to biological mutation. Mutation alters gene values in a chromosome, potentially leading to a better solution in GA by randomly flipping some bits in the chromosome, resulting in a completely different solution.

4.6 FITNESS FUNCTION:
The fitness function measures the quality of a genetic representation, which is problem-dependent and is commonly represented as a string of numbers called a chromosome in genetic programming and algorithms. The goal is to eliminate the 'n' worst design solutions from each testing or simulation round and create 'n' new ones by breeding the best design solutions. Each design solution must be given a merit figure to indicate its closeness to meeting the overall specification, which is determined by applying the fitness function to the test, or simulation, results obtained from that solution.

V. METHODOLOGY

A design methodology is a methodical process that uses a set of techniques and guiding principles to create a plan. We have followed these methodologies. The system's entire requirement, including the formulation of a schedule strategy, should be taken into consideration. It is necessary to create a database. Regarding all of the regulations implemented in order to preserve the records. Record all possible scenarios and their upcoming ones using flow-charts to effectively manage the situation. It is recommended to thoroughly test the scheme by executing all of the system's test cases.

- First, we should create the first-year class schedules by entering the information for each section's first-year schedule. Input the faculty, subjects, and section details into the database.
- Retrieve the first-year timetable timeslots and assign the timeslots to the respective faculty.
- Get the subjects assigned to faculty members in their first year and begin sorting through them in the schedule.
- The faculty who handles the theory classes will be allotted for tutorial and related lab.
- All faculties should get two hours of classes in the day session depending on the subject allotted.
- The workload allotted for professor, associate professor, and assistant professor have to be followed.

CONCLUSION

Genetic algorithm resolves the timetable scheduling efficiently. It resolve conflicts and automatically generate timetable. Genetic algorithm gives a optimal solution to problems. We can implement this time table scheduling using genetic algorithm application in our college also.

REFERENCES

[1] Bagul, M. R., Pushkar R. Patil, S. C., & Nagare, S. N. (October 2015). A Novel Approach for Automatic Timetable Generation . International Journal of Computer Applications , 6.

[2] Chowdhary, A., Priyanka Kakde, S. D., & Rupal Rushiya, D. G. (February 2015). TIMETABLE GENERATION SYSTEM

[3] International Journal of Computer Science and Mobile Computing , 6.

[4] Deshkar, M., Mayur kale, M. B., & Ghom, A. (MARCH 2016). TIME TABLE AT A CLICK. International research journal of engineering and technology, 4.

[5] Lahoti, Y., & Aaditya Punekar, H. P. (2015). Automated Timetable Generator . International Journal of Science and Research, 4.

[6] M, S., & Pranav Kiran Vaze, P. M. (MAY 2017). Automatic Time Table Generator. International Journal of Advanced Research in Computer Science and Software Engineering , 7(5), 8.

[7] Mittal, D., & Hiral Doshi, M. S. (February 2015). Automatic Timetable Generation using Genetic Algorithm . International Journal of Advanced Research in Computer and Communication Engineering, 4.

[8] Okunieff, P. (March 2006). Dynamic Timetable Generator. TRANS POilA*T¡ON RES EARC H BOARD. Patil,

[9] M. K., & Rakhe Shruti Subodh, P. A. (June 2014). Web Application for Automatic Time Table Generation . International Journal of Current Engineering and Technology , 3.

[10] Rathod, P. P., Kamlesh K. Lodhiya, M. P., Student CSE,

[11] p. S., & S. C. (2016). Automatic Timetable Generator. International Journal of Research In Science & Engineering , 8.

[12] S, D. C., A Kavya Reddy, N. A., & K Panimozhi, U. S. (April 2015). Automatic Timetable Generationm System. JETIR, 8.

[13] Anandaraj soundarya Raja Murugan,"University Timetabling Using Genetic Algorithm", Dalarna University, March 2009 .

[14] Branimir Sigl, Marin Golub, Vedran Mornar" Solving Timetable Scheduling Problem by Using Genetic Algorithms", Faculty of Electrical Engineering and Computing, University of Zagreb

[15] "Genetic algorithm", Wikipedia(2011), the free Encyclopaedia. http://en.wikipedia.org/wiki/Mainpage.

[16] Jiří Voráč, Ivo, Vondrák, Karel Vlček" SCHOOL TIMETABLE GENERATING USING GENETIC

[17] ALGORITHM" VSB – Technical University of Ostrava Czech Republic.

[18] Leon Bambrick,"Lecture Timetabling Using Genetic Algorithms", University of Queensland .

[19] M. Grobner, et al., "A standard framework for timetablingproblems ," Proc. International Conference on the Practice and Theory of Automated Timetabling 2002, LNCS 2740, pp.24-38, 2003.