CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

Timetable scheduling is an NP-hard problem, meaning it involves a high level of computational complexity, and finding an exact solution is often infeasible within a reasonable time for large instances. Therefore, the goal is typically to find an optimal or near-optimal solution within a complex set of variables and constraints. As a result, extensive research has been conducted on applying optimization techniques to address this challenge.

This literature review explores various existing approaches to automated timetable generation, with a particular focus on Genetic Algorithms. It has gained popularity due to their effectiveness in solving NP-hard optimization problems like timetabling. They operate on the principles of natural selection and evolution, using mechanisms such as selection, crossover, and mutation to iteratively improve solutions. Their ability to efficiently explore large and complex search spaces makes them well-suited for generating feasible and optimized timetables under multiple constraints.

2.2 Literature Survey

Han et al. [1] proposed a hybrid approach called POGA-DP, combining Genetic Algorithms with Dynamic Programming (DP) to solve the University Course Scheduling Problem (UCSP), particularly for complex joint-course timetables. The GA component optimizes time slot assignments using a swap-based mutation with a repair mechanism, while DP allocates classrooms to minimize seat wastage and improve utilization. The method achieved significant improvements in scheduling quality (up to 46.99%) and reduced classroom usage by 29.27% compared to standard GA. Tested on data from Beijing Forestry University, the approach outperformed GA, Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Producer–Scrounger Method (PSM) across multiple metrics, including fitness and occupancy rate. However, the model lacks real-time adaptability, parameter tuning, cross-institutional testing, and practical integration into university systems, limiting its deployment readiness.

Paramatmuni et al. [2] proposed a web-based timetable generation system using Genetic Algorithms, optimized for scalability and real-time adaptability. The system employs roulette wheel and tournament selection, single-point crossover, and mutation, with a weighted fitness

function evaluating hard and soft constraint violations. It achieved 95% conflict resolution and 90% resource utilization on real-world university data. The system is implemented using Django for the backend and HTML/CSS for the frontend. Despite its efficiency, the model requires manual weight tuning, offers limited support for collaborative user input, and shows reduced performance under extreme resource constraints.

Mahlous et al. [3] introduced a Genetic Algorithm for student timetabling that prioritizes individual preferences such as free days, preferred class times, and grouping with friends. The algorithm uses a binary chromosome representation to encode student-class assignments and applies customized multi-point crossover, mutation, and repair operators. A simulated annealing-based function enhances soft constraint satisfaction while preserving feasibility. The model also integrates parallel processing, fitness caching, and adaptive mutation rates to improve execution time and solution quality. However, it is limited to optimizing student allocations for fixed schedules, lacks a user-facing interface, and has not been deployed in real-time institutional settings.

Alnowaini et al. [4] propose a genetic algorithm-based approach to solve the university course timetabling problem using dynamic chromosomes, which adapt to varying course loads across departments and semesters. The system encodes scheduling variables as tuples of course, time, and room, and uses roulette wheel selection, two-point crossover, and optimized mutation to generate near-optimal schedules, achieving up to 93% conflict-free timetables in under five minutes. While the algorithm effectively satisfies both hard and soft constraints, it suffers from performance issues when scaling beyond 82 courses. However, the need for greater scalability, full automation, and dynamic real-time scheduling in future work.

Gore et al. [5] proposed an automated timetable scheduling system using a Genetic Algorithm to address the limitations of manual timetable generation in educational institutions. The model generates schedules by encoding each class with course, instructor, and room data, using crossover and mutation operators to evolve conflict-free solutions. A fitness function penalizes overlaps and lecture overloads, and the final timetables are exported in Excel format for user access. The system includes GUI modules for faculty and course input, and generates individualized timetables. While being effective at basic conflict resolution, the system does not account for advanced constraints such as faculty preferences, load balancing, or real-time adaptability. Additionally, it lacks scalability testing on large datasets and offers no support for multi-objective optimization or collaborative scheduling environments.

Ghiridhar et al. [6] proposed a Genetic Algorithm-based system for generating conflict-free timetables tailored to the regulations of A P J Abdul Kalam Technological University. Their approach introduced a conflict operator within each chromosome to dynamically detect and resolve scheduling clashes during evolution. The system addressed both hard constraints, such as subject-hour limits and lab continuity, and soft constraints, like balanced faculty workloads. Mutation was performed through a conflict-aware local descent strategy, outperforming traditional random mutation methods. Implemented as a web application using React and Flask, the model successfully generated valid timetables for six real student batches. However, the system lacked real-time adaptability, multi-objective optimization, and scalability testing across larger institutional datasets, limiting its broader applicability.

Assi et al. [7] proposed a hybrid approach that integrates Genetic Algorithms with Graph Colouring techniques to address the University Timetabling Problem (UTP), modelling courses as graph nodes and conflicts as edges to avoid timeslot overlaps. Their implementation represents each timetable as an array of genes, where each gene contains a course's instructor, room, and timeslot, and uses penalty-based fitness evaluation to quantify violations of hard and soft constraints. Genetic operators, including roulette wheel selection, constraint-aware crossover, and low-probability mutation are complemented by a repair function that resolves infeasible solutions post-crossover. Experimental results demonstrate a 70% reduction in penalties over 150 generations, validating the method's effectiveness in constraint satisfaction. However, the approach is limited by a simplified mutation strategy, narrow crossover scope (only one conflicting pair per generation), and testing on synthetic datasets. Furthermore, the system lacks support for real-world institutional policies, dynamic re-scheduling, and user-defined preferences.

Febrita et al. [8] proposed a modified Genetic Algorithm enhanced with fuzzy time windows to optimize high school timetable scheduling. The approach categorizes subjects by cognitive demand and uses fuzzy logic to assign more mentally intensive subjects to earlier time slots. Chromosomes represent weekly schedules with 210 genes, and a fuzzy-guided mutation prioritizes replacing low-satisfaction time slots. A 2D crossover and a repair mechanism ensure constraint satisfaction and diversity. Despite demonstrating improved fitness scores and convergence compared to standard GA, the system lacks subject frequency control per day, real-time adaptability, multi-department scalability, and a deployable user interface, limiting its application in broader institutional settings.

Sampebatu et al. [9] present a timetable management system utilizing Genetic Algorithms to address the highly constrained, NP-hard nature of university course scheduling. Their system adopts a hybrid GA approach adapted from examination timetabling techniques at the University of Nottingham employing selection and crossover to generate near-optimal timetables based on user-defined constraints such as course units, hall capacities, and lecturer availability. Implemented in Delphi for its performance efficiency, the system includes a random population generator and a crossover module to iteratively improve timetable feasibility. While the method achieved high average fitness scores (up to 0.986), it is limited to 2-unit courses, lacks formal soft constraint optimization, and does not support dynamic rescheduling or real-world institutional benchmarking.

Abdelhalim et al. [10] introduced a Utilization-based Genetic Algorithm (UGA) to solve the university course timetabling problem with a focus on optimizing space utilization alongside satisfying scheduling constraints. Their approach employed a two-dimensional chromosome representation, mapping events to room-timeslot pairs, and generated the initial population using heuristic methods, the authors proposed a utilization-based crossover operator aimed at reallocating underutilized events and a targeted mutation strategy employing local search to optimize room occupancy. A composite fitness function weighted towards occupancy rates, frequency rates, and minimizing scheduling gaps was used to evaluate solutions. The system was tested on real data from Alexandria University also, against two benchmark datasets from the International Timetabling Competition (ITC 2007). The Results demonstrated significant improvements in space utilization, reduced scheduling hours, and better overall timetable quality compared to manual scheduling. However, the study identified gaps in dynamic real-time adaptability and slight computational overhead for medium-sized problems.

Mittal et al. [11] addressed the timetable scheduling problem by proposing the use of Genetic Algorithms to automate and optimise the scheduling process. The authors applied Genetic Algorithms where the algorithm initialises the population of guesses, then three operators are applied-selection, crossover and mutation to create an optimal timetable. The system was tested with a real data within the author’s institution. The results demonstrated significant improvements in both efficiency and accuracy compare to manual scheduling. Although the study succeeded in creating a more efficient alternative to manual scheduling, it lacked mechanisms for dynamic adaptability and deeper real-world constraint handling.

Paresh et al. [12] proposed a Genetic Algorithm-based solution to address inefficiencies in manual academic timetable generation by introducing a novel chromosome structure. Each chromosome encoded multiple genes containing course, faculty, frequency, group, and room data, allowing a more granular and adaptable representation of timetable data. The system incorporated GA operations including one-point crossover and standard mutation, with the fitness function evaluating constraint satisfaction. The model effectively handled both hard constraints and soft constraints, thus successfully generating timetables for Information Technology and Computer Engineering departments. Despite these strengths, the system was limited in scope, lacking scalability across multiple departments or large institutions. Furthermore, faculty preferences and dynamic constraints such as elective course selection or live updates were not addressed.

Othman et al. [13] presented a Genetic Algorithm approach for university course timetabling, ensuring all hard constraints were satisfied while minimizing soft constraint violations. Each chromosome encoded subject, section, instructor, time, and room data. The method employed rank-based selection, single-point crossover, and random mutation, with exclusiveness to retain the best solutions across generations. A fitness function was designed to normalize and weigh both hard and soft constraint violations, covering instructor time preferences, room assignments, course prerequisites, and overload control. Tested on real data from the University of Jordan, the algorithm achieved zero hard constraint violations and significantly reduced instructor overload and scheduling conflicts. However, the model lacked real-time adaptability, sensitivity to constraint weights, and user interaction capabilities, limiting its flexibility and scalability in dynamic environments.

Sapru et al. [14] proposed a Genetic Algorithm based solution for university timetable scheduling, introducing a novel Guided Mutation operator that selectively accepts only beneficial mutations to improve convergence and constraint satisfaction. Chromosomes represent weekly timetables for multiple student groups, encoded in a binary structure that captures room, faculty, subject, and slot activity. A rank-based selection strategy is employed to maintain diversity and avoid premature convergence, while a single-point crossover enhances genetic variation. The guided mutation approach consistently outperformed conventional mutation, leading to faster generation of feasible timetables (fitness = 1.0) across varied test cases. However, the study focused solely on hard constraints, omitting critical soft constraints like instructor preferences or scheduling gaps. Additionally, scalability to large datasets and practical deployment aspects were not addressed.

Colorni et al. [15] proposed a Genetic Algorithm approach to solve the highly constrained school timetabling problem. Their method used a two-dimensional matrix representation of teachers and timeslots, with a hierarchical objective function focusing first on feasibility, second on management rules, and finally on individual teacher preferences. This method uses customized genetic operators, like a row-based crossover and order-k mutation, along with a genetic repair (filtering) process to correct infeasibilities. The integration of local search significantly enhanced timetable quality, achieving a substantial reduction in cost compared to manual scheduling and simulated annealing. However, the approach faced scalability challenges due to the computational complexity of the repair process and lacked mechanisms for dynamic timetable adjustments.

2.3 Summary of Literature Survey

Several studies on automated timetable generation, emphasizing methodologies such as Genetic Algorithms and hybrid approaches, are summarized in Table 2.1, which provides an integrated overview of these significant works. Each paper is outlined by its core method, main advantages, and future work. The comparison highlights innovation trends and uncovers common limitations, helping to identify gaps and guide future development in timetable scheduling systems.

Table 2.1: Observations of Literature Survey

Title	Author	Methodology	Advantages	Future Work / Limitations
Gradual Optimization of University Course Scheduling Problem Using Genetic Algorithm and Dynamic Programming	Xu Han, Dian Wang	Hybrid approach which is a combination of Dynamic Programming and Genetic algorithm.	Improved quality, reduced room use and outperformed PSO, ACO, PSM.	No cross-institution testing, lacks integration into university systems
Smart Timetable Generation using Genetic Algorithm.	Sahith Siddharth Paramatmuni, Dumpala	Web-based GA with roulette/tournament selection, weighted	Fast scalable and provides conflict resolution.	Manual weight tuning, limited collaborative input, reduced

	Yashwanth Reddy, Elakurthi Sai Spoorthi, Akhil Dharani, K. Venkatesh Sharma	fitness function; implemented in Django; outputs Excel files.		performance under extreme constraints.
Student timetabling genetic algorithm accounting for student preferences.	Ahmed Redha Mahlous, Houssam Mahlous	GA with binary chromosome, simulated annealing based function and adaptive mutation.	Solves student preference satisfaction with adaptive mutation.	No user interface, not deployed in real-time, fixed schedules only.
Genetic algorithm for solving university course timetabling problem using dynamic chromosomes.	Ghazi Alnowaini, Amjad Abdullah Aljomai	Genetic Algorithm using dynamic chromosomes for course-time-room tuples, roulette wheel selection; two-point crossover, optimized mutation.	93% conflict-free, handles varying loads	Poor performance beyond 82 courses, lacks real-time and automation.
Institute Timetable Scheduler.	Bhaven Gore, Disha Shirdhankar, Giriraj Belanekar	GA encoding course, instructor, room and GUI for input, fitness function penalizes clashes/overloads, output in Excel.	GUI interface, conflict-free timetables, Excel sheet export.	No soft constraints, Lacks scalability.
Timetable Generation Using Genetic Algorithm	Ghiridhar S, Sachin A, Edwin M.T,	Genetic Algorithm with conflict operator; conflict-	Custom web app, solved real institution	No multi-objective

For Batches Under Apj Abdul Kalam Technological University	Unnikrishnan K.N	aware local descent mutation; implemented via React + Flask web app.	cases, effective conflict resolution.	optimization, lacks scalability
Genetic algorithm analysis using the graph coloring method for solving the university timetable problem.	Maram Assi, Bahia Halawi, Ramzi A Haraty	Hybrid GA with Graph Colouring and genes encode instructor, room and timeslot.	70% weight reduction and effective infeasibility repair.	Simplified mutation, narrow crossover scope, no real-world testing or user preferences
Modified genetic algorithm for high school time-table scheduling with fuzzy time window.	Ruth Ema Febrita, Wayan Firdaus Mahmudy	Modified GA with fuzzy time windows, cognitive load-based slot assignment, fuzzy-guided mutation.	Better convergence & fitness; prioritizes cognitive load.	Lacks interface and multi-dept scalability.
Timetable Management Using Genetic Algorithms.	Limbran Sampebatu, Aries Kamolan	Hybrid GA adapted from exam timetabling and random population generator	High average fitness, handles hall capacities, lecturer availability.	Supports only 2-unit courses, no soft constraint optimization;
A utilization-based genetic algorithm for solving the university timetabling problem (uga).	Esraa A. Abdelhalim, Ghada A. El Khayat	Utilization-based genetic algorithm: focused on optimizing space utilization alongside satisfying scheduling constraints.	Improved space utilization, reduced hours, real data is being used.	Lacks real-time adaptability.

Automatic timetable generation using genetic algorithm.	Dipesh Mittal, Hiral Doshi, Mohammed Sunasra, Renuka Nagpure.	A Genetic Algorithm was implemented in C# using selection, crossover, and mutation to generate conflict-free timetables based on real departmental data.	Improved efficiency & accuracy vs. manual scheduling.	Lacks dynamic adaptability & deeper real-world constraint handling.
Solving time-table scheduling problem by novel chromosome representation using Genetic algorithm.	Paresh M Chauhan, Kashyap B Parmar, Mahendra B Mendapara	GA with novel chromosome (course, faculty, group, room info) one-point crossover and standard mutation, I/O is managed using XML/HTML interface.	Granular data representation, handles both hard & soft constraints.	Lacks faculty preferences, dynamic constraints, and scalability.
A novel genetic algorithm technique for solving university course timetabling problems.	Othman M. K. Alsmadi, S. Za'er, Dia I. Abu-Al-Nadi, Alia Algsoon	Genetic algorithm which involves rank based selection, with a generation limit of the fitness ≥ 0.99.	Zero hard constraint violations, reduced overload.	No real-time adaptability, lacks sensitivity to constraint weights
Time table scheduling using genetic algorithms employing guided mutation.	Vinayak Sapru, Kaushik Reddy, B Sivaselvan	GA with binary chromosome, Guided Mutation, rank-based selection and single-point crossover.	Avoids getting stuck on bad solutions too early, creates complete timetables.	Ignores soft constraints, not scalable.

A genetic algorithm to solve the timetable problem.	Alberto Colorni, Marco Dorigo, Vittorio Maniezzo	Genetic Algorithm with customized genetic operators like row-based crossover, order-k mutation, and a filter.	Reduced time vs. manual & simulated annealing.	Scalability issues, no dynamic adjustment can be made. Lacks faculty preferences.

2.4 Comparison with Existing Systems

Automated timetable scheduling has been the subject of extensive research, with numerous approaches leveraging various optimization algorithms. Traditional manual scheduling methods are prone to inefficiencies, leading to scheduling conflicts, workload imbalances, and increased effort. To address these issues, several advanced computational techniques have been proposed, primarily focusing on Genetic Algorithms due to their ability to efficiently generate optimized timetables under complex constraints.

Existing systems employing GA-based approaches demonstrate improvements in scheduling efficiency. For instance, some models utilize utilization-based genetic algorithms, optimizing space usage while satisfying hard constraints. Others integrate hybrid methods, combining Genetic Algorithms with Dynamic Programming or Fuzzy Logic to refine results further. While these approaches successfully minimize violations of scheduling constraints, they often lack real-time adaptability and user interactivity. Many of these solutions focus on pre-semester timetable generation, requiring manual adjustments for dynamic changes.

2.5 Proposed System

The Web-Based Automatic Timetable Scheduler is designed to address the complexities and inefficiencies associated with manual scheduling in educational institutions. The system automates the process using Genetic Algorithms and heuristic optimization techniques to generate conflict-free and optimized timetables while satisfying predefined constraints. The primary objective is to ensure an efficient, scalable, and user-friendly solution that accommodates multiple departments, diverse faculty preferences, and dynamic scheduling adjustments.

The system will leverage constraint satisfaction principles, categorizing conditions into hard constraints (mandatory requirements, such as preventing timetable clashes and ensuring faculty availability) and soft constraints (preferences like distributing lectures evenly and minimizing gaps). Unlike existing scheduling solutions, which often require manual intervention for modifications, the system incorporates real-time adaptability, allowing users to make necessary changes dynamically.

2.6 Objectives

The primary objective of the Web-Based Automatic Timetable Scheduler is to streamline the scheduling process for colleges by automating timetable generation through optimization techniques. The system aims to eliminate inefficiencies and human errors in manual scheduling while ensuring scalability and adaptability to diverse academic environments.

The key objectives of the proposed system include:

- To automate the timetable creation process, eliminating the inefficiencies and errors associated with manual scheduling.
- To apply a Genetic Algorithm that optimizes the allocation of courses, and teachers, while satisfying hard constraints and minimizing violations of soft constraints.
- To develop a scalable and efficient scheduling model capable of handling multiple batches, courses, and faculty simultaneously.
- To provide a user-friendly interface that allows users to easily input data, modify schedules, and view the generated timetable.
- To ensure reliability and accuracy in timetable generation by incorporating validation checks and conflict detection mechanisms.