International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Timetable Generator Using Genetic Algorithm and Constraint Satisfaction Problem

Bincy Manuel[1], Vishnu Mohan C[2]

[1]Scholar, Department of Computer Science, Sacred Heart College, Thevara, India
[2]Assistant Professor Department of Computer Science, Sacred Heart College, Thevara, India

Abstract
Timetable generation in academic institutions is time consuming, complex and error-prone task due to various constrains involving faculty availability, time restrictions and scheduling conflicts. Manual scheduling often leads to inefficiencies and human errors. The system proposes an Automatic timetable generator using Optimization algorithms to address complexities. Optimization algorithms find the best solution from all possible solutions under the given constrains. It combinations several possible timetable combinations and identifies the most suitable one while satisfying essential conditions. The model is built using a hybrid CSP-GA framework that combines Constraint Programming for efficient local repair and feasibility assurance and Genetic Algorithms for global exploration of the solution space. The system models hard constraints such as Faculty availability, Room Capacity, Elimination of overlaps and exclusion of fixed break times from scheduling etc. In addition, it also considers soft constraints such as teacher preferences, minimizing idle time between classes, and distributing lectures evenly throughout the week. The system thereby focuses on benefiting the academic institutions by providing a flexible, intelligent framework capable of generating efficient timetables of varying size and complexities.

Keywords: Timetable Generation, Optimization Algorithms, Genetic Algorithm, Predefined constraints

1. INTRODUCTION

Timetable generation is a major task in various academic institutions. However, creation of timetable manually is time consuming and error prone, especially when considering numerous constraints like faculty availability, limited classroom resources, overlapping sessions, and institutional policies. As the size of the institution grows manual scheduling becomes more complex. The proposed system focuses on overcoming all the challenges faced in manual timetable generation and helps in generating more efficient and accurate time tables by saving valuable time. The proposed model is built using a hybrid GA-CSP framework that combines Genetic Algorithms for global exploration of the solution space and Constraint Programming for efficient local repair and feasibility assurance. The system helps to generate both department-wise class schedule and examination schedule. It works by considering various hard and soft constraints. Where hard constraints are those that are non-negotiable like the faculty availability, Room capacity and facility compatibility, Elimination of scheduling overlaps, Fixed breaks and institutional blackout periods. While the soft constraints like Faculty time/day preferences, Minimization of idle periods between classes, Balanced distribution of workload across the week etc. are negotiable. Optimization algorithms act as a powerful ally, shouldering a huge combinatorial burden. They offer a systematic way to approach this problem by searching through numerous possible arrangements and

selecting the most suitable one based on these predefined rules/constraints. The purpose of the system is to create balanced, practical, and conflict-free schedules by intelligently navigating the search space of all feasible combinations. By reducing manual work load and increasing scheduling accuracy, the proposed system provides academic institutions with a robust and flexible tool for effective timetable management.

2. Related Research Work

Saini et al. [1] discusses about the need for automated timetable generation for resolving the conflicts occurring in various academic institutions. The proposed solution involves a timetabling algorithm that uses a JSON file as input and generates a content-based timetable. The system collects data on courses, rooms, labs, lectures, semesters, and students and apply various hard and soft constrains and generate timetable based on these constrains. The paper discusses about key features of modern timetable generators Constraint based scheduling, Flexible user customization, Integration with existing systems, Efficient handling of edge cases and conflicting requirements. Thus, the paper minimizes the conflicts, balances teacher workload, and maximise resource utilization.

Kumar et al. [2] proposes Particle Swarm Optimization (PSO) algorithm for time table generation. PSO is a population-based metaheuristic inspired by the social be haviour of bird flocking and fish schooling. It has a fast con vergence and fewer parameter setting. In context of timetable generation, each particle in the swarm represents a solution and these particles iteratively adjust their positions based on both individual and collective experience. The authors define a mathematical model that incorporates both hard constraints like number of teachers or classroom double-booking, lab sessions must be consecutive etc. and soft constrains teacher preferences for time slots or rooms, avoidance of back-to back sessions etc. The faculty preferences were collected and the constrains are encoded into a linear mathematical model. The paper uses the inertia weight value concept to balance the global search ability and local search ability and thereby boost the capability to locate the optimal solution and convergence rate. In conclusion the paper shows that PSO in timetabling shows a high potential for scalability, adaptability, and satisfaction optimization.

A comparative analysis of Genetic Algorithm (GA) and particle swarm optimization (PSO) was done by Alghamdi et al. [3] for evaluating the suitability for solving university course timetabling problem (UCTP). GA was used due to their high degree of computational parallelization and enhanced computational performance. The GA first considers the hard constrains and then for soft constrains. Repair functions are used to correct infeasible solutions post-mutation. The paper shows how PSO is enhanced using inertia weight, velocity clamping, and hybridization with other heuristics. It also shows the difficulty in handling constrained discrete spaces. The paper also reviewed the approaches like Simulated Annealing, Tabu search, Ant Colony Optimization, Hyper-heuristic. It shows the various advantages and disadvantages of each of these algorithms.

In the paper by Adrianto et al. [4], the author compares Particle Swarm Optimization (PSO) and Genetic Algorithm (GA). The author outlines the evolution of timetabling solutions, from graph-based techniques to sophisticated AI models. Sequential methods such as graph colouring, and cluster methods that reduce complexity through grouping, were found to be effective for small-scale problems but lacked scalability. Genetic Algorithms are used with operations such as crossover, mutation, and inversion to evolve a population of timetable solutions. PSO is computationally simpler, does not rely on evolutionary operators, and tends to converge faster. Results indicated that PSO consistently outperformed GA in minimizing the total penalty score associated with constraint violations, particularly from the 500th iteration onward. PSO

achieved similar or better results with fewer iterations, making it a more efficient choice for timetable generation. The study confirms that while both GA and PSO are viable for academic timetabling, PSO provides superior performance in terms of convergence speed and quality of solutions in large, complex environments.

A hybrid model by combining Bee colony Optimization (BCO) and Firefly Algorithm (FA) termed as BCFA was proposed by Sahoo et al. [5] for finding the optimal solutions of course time table. BCO emphasizing on foraging behaviour of honey bees for finding nectars or food sources. Which helps in generating the optimal number of test cases. The firefly algorithm which is derived and motivated by flashing or mating behaviour of fireflies. The BCFA was created by combining Bee Colony Optimization Algorithm with the approach used in firefly Algorithm. A fitness function guides the optimization process, calculated using a combination of teacher subject preferences and the probability of allocation, which is derived from the number of subjects a teacher is willing to teach. This fitness-driven method ensures solutions are not only conflict-free that satisfies hard constraints but also aligned with teacher preferences which is soft constraints. The best solution achieved a fitness value of 45.1667 by iteration 120. Graphical comparisons confirmed BCFA's dominance across iteration counts. In conclusion BCFA achieves optimized results more efficiently, minimizes computational overhead, and enhances the quality of generated timetables.

Constraint programming and rapid software development techniques have also been used to address multi-variable scheduling efficiently. In the paper by Firkeet al. [6], the author proposes an architecture that consists of three main components: The admin, The lecturer and the student. The admins are in charge of creating and adjusting timetables, ensuring that they are well organized and meet the needs of the institution. Additionally, the admins play a crucial role in maintaining the quality of timetables by receiving feedback from lecturers. This feedback loop allows for continuous improvements and the prompt resolution of any issues. Lecturers have their own set of roles within the system. They can access their timetables, which is vital for keeping track of their schedules. By combining constraint-based logic, web technologies, and user-driven design, the system addresses longstanding inefficiencies and sets a strong foundation for future development.

Cheema et al. [7] examines the impact of optimization algorithms on real-world applications by considering the effectiveness and limitations. It discusses about the emerging trends such as hybrid approaches and metaheuristic methods that offer promising directions for overcoming current challenges. It also discusses about various algorithms like Tabu search, Evolutionary computation algorithm, Bee Colony optimization-based algorithm, Ant Colony Optimization algorithm, Genetic Algorithm, Nelder-Mead Algorithm, Particle swarm Optimization algorithm. The paper explains about the various steps involved in these algorithms.

Mahajan et al. [8] introduces Timely Trigger, an innovative smart timetable generator that integrates push notifications, live tracking, and cloud-based communication to deliver a dynamic, real-time scheduling experience. The Push notifications was used to remind faculty of their upcoming classes and to notify students of any changes to the timetable. Live tracking can be used by students to track the location of their instructors and to estimate the time it will take for their instructors to arrive in class. The architecture consists of 3 layers Data Layer, Processing layer and the presentation layer. The data layer means the input layer where we input the faculty details, classroom details etc. Then comes the processing layer, The processing layer contains the algorithms and heuristics that are used to generate timetables. Then the presentation layer where the generated time table Is being shown by interfaces. The platform supports

real-time updates, error detection highlighting scheduling conflicts, customizable design templates, and even future integration with IoT-based floor tracking for administrative analytics. It combines automation with smart features that enhance efficiency, transparency and user satisfaction.

In the research paper by Latpate et al. [9] the author introduces an AI based timetable generator using ReactJS and Firebase, aiming to improve efficiency, accuracy, and user satisfaction in scheduling systems. It explores the intersection of frontend development and backend optimization, combining an intuitive user interface with algorithm driven logic to streamline operations in colleges. The paper also discusses the importance of constraint programming where scheduling problems are modelled using sets of variables and constraints that must be satisfied. The authors also highlight potential enhancements, including mobile application development, AI based conflict resolution, natural language processing, and gamification elements to boost user engagement. The AI based timetable generator using React provides a fresh take on timetable automation by combining modern web technologies with intelligent scheduling logic.

Timetable generation is identified as a complex task influenced by numerous constraints, such as the availability of faculty, classroom resources, and scheduling preferences. To tackle this issue, Puttaswamy et al. [10] propose an automatic system utilizing various optimization algorithms, including Genetic Algorithms, Heuristic methods, and Constraint Logic Programming, aimed at efficiently creating feasible timetables. The research highlights both hard and soft constraints impacting the scheduling process, stressing the importance of a systematic approach to achieve an optimal timetable that aligns with institutional requirements. Furthermore, the study reviews existing literature and methodologies for timetabling, providing insight into the effectiveness of local search procedures and constraint programming. Overall, the proposed system aims to streamline the timetable generation process, ultimately reducing the manual effort and time typically expended in academic settings.

A hybrid Constraint Satisfaction Algorithm (HCSA), based on rule-based systems, heuristic optimization techniques, and AI-driven models was introduced by Goyal et al. [11] In the paper the author address about the issues faced by the existing system and develop a robust and scalable automatic time table generator that can overcome the flaws faced by the existing systems. HCSA incorporates Real Time Adapt ability through the use of reinforcement learning. The AI driven component enables the system to dynamically alter the schedules in response to real time feedback and to predict potential conflicts, ensuring the timetable remains optimized even amidst unforeseen events. The Constraint Handling mechanism distinguishes between hard constraints and soft constraints, utilizing genetic algorithm for efficient and comprehensive conflict resolution. The validation process ensures the final timetable adheres to all specified constraints contributing to a high conflict resolution rate. A rule-based logic is applied initially to establish a foundational schedule that satisfies hard constrains. This is refined using the genetic algorithm through mutation, crossover etc., AI can be used to predict the conflicts and proposes changes dynamically. This approach strategically combines the strengths of different methodologies while mitigating their individual weaknesses.

Uma et al. [12] proposes a system using Genetic algorithm to resolve manual timetable generation problems. The inputs such as grade-wise subjects, teachers and workload are taken and generate a possible timetable. The system also generates a timetable for each teacher individually by using the data from the class timetable. This paper is built upon more sophisticated algorithmic approach foundations rather than simple rule-based systems. The N-queen algorithm was used to prevent the conflicts. The genetic algorithm refines potential timetables by selecting the fittest solution through processes like selection, crossover, and mutation, until an optimal solution emerges. It takes the inputs like grade-wise subjects and

teacher availability, processing them through the combined power of the N Queen and Genetic Algorithms thereby the system promises a faster and more efficient way to generate timetables.

Preethi et al. [13] focuses on preparing the timetable for university departments and scheduling examinations for each semester. The paper proposes a system using PHP, MySQL, HTML, and CSS. PHP as the programming language, MySQL for the database. The system was integrated with Oracle for advanced data management. The Constraint Satisfaction problem (CSP) was used to organize and address various hard and soft constrains effectively with a feasible solution. Which was then followed by the Ant Colony Optimization (ACO) for getting an optimal solution. ACO optimizes the timetable by mimicking ant foraging behaviour to find the shortest, most efficient paths in this case, the best scheduling combinations. Integrating CSP and ACO ensured feasibility and optimizes the timetable generation. The authors propose future improvements, including integration with machine learning for adaptive optimization, mobile-friendly interfaces, real-time updates making the system even more flexible and collaborative.

Hybrid approaches can be used to tackle the problem of time table generation by combining various algorithms for better efficiency. In the paper by Kusumawardani et al. [14], the author combines Greedy algorithms and Simulated Annealing (SA). The system was tested over two new real-world datasets and the performance was evaluated. Rather than working in the typical solution space as in traditional meta-heuristics, the hyper-heuristics framework searches in the space of low-level heuristics, making the system more generalizable and less dependent on problem-specific parameter tuning. The timetabling is defined as a problem that have four parameters the time slot, resource in the form of a classroom, the number of meetings and constrains. The timetabling is carried out by allocating time slots and resource on a number of meetings that meet the constrains. The greedy algorithm is used initially to generate initial feasible solution, then the SA algorithm is used to optimize the initial solution generated from the previous stage. The exam with highest number of conflicts is assigned to timeslots and rooms first followed by the second exam in the order and so on until all exams are assigned. In conclusion, the study demonstrates that a Greedy-Simulated Annealing Hyper-heuristics algorithm is highly effective in automating and optimizing exam timetables.

Nwufoh et al. [15] focuses on developing a hard constraint satisfying problem algorithm for course timetable scheduling. The constraint satisfaction problem works by the non-negotiable hard constraints that must be satisfied for a timetable to be feasible. Hard constraints like Lecture Scheduling, Room Occupancy, Conflict Avoidance etc are rigidly enforced while soft constraints like Lecture Room Capacity, Room Stability etc are optimized as much as possible within the algorithm's logic. The authors ensure that the iterative, rules-based approach ensures the generation of a conflict-free timetable. The authors evaluated the system's performance using Halstead Complexity Metrics, which confirmed its efficiency, maintainability, and low response time. The system was developed using Java for the user interface and MySQL as the database backend. The authors also suggest hybridizing CSP with fuzzy logic for further optimization.

An AI-driven solution named Adaptive Scheduler was developed by Saw et al. [16] for addressing the long-standing complexities in academic timetable generation. The author proposes a hybrid system that combines Genetic Algorithm (GA) with AI-based optimization techniques to automate and improve the timetable creation process for educational institutions. An adaptive mechanism allows the timetable to adjust in real-time to accommodate unforeseen changes, such as faculty absences or room issues. The project also focuses on ensuring fairness by balancing faculty workloads while respecting their preferences

and institutional policies. The authors highlight the systems adaptability, automation capability, and user friendliness as key strengths. The system is built with a Flask Python backend for optimization logic and a JavaScript/HTML frontend for user interaction.

Based on the research by Kaldoke et al. [17] the authors focus on reducing the errors, save time and effort, and promote a paperless environment for the task of manual time table generation by creating an intuitive interface that facilitates seamless integration. The system uses Genetic algorithms to optimize timetables by encoding schedules as chromosomes and evolving solutions through selection, crossover, and mutation. The timetable feasibility is maintained by various hard constraints and soft constraints. The hard constraints include assigning teachers to more than one class at a time, A single classroom cannot host multiple lectures simultaneously, Teachers can only handle two or three lectures per day etc. The soft constraints include Breaks should be placed between sessions, Schedules should consider lunch hours for better student and faculty experience etc. The GA processes inputs to produce conflict-free schedules, validated through iterative fit ness evaluations. Over iterations, the best-performing solutions are selected, crossed, and mutated to form new populations, gradually converging to an optimal or near-optimal timetable.

Rashmi et al. [18] categorizes the time table generation into two: Course Timetabling and Examination Timetabling. This task was done by a hybrid algorithm that fuses GA's local optimization strength with PSO's global search efficiency. Hard constraints and soft constraints are embedded into the fitness function to ensure feasible solutions. The system used QT and PyQt5 for the graphical user interface and SQLite as the backend database. The hybrid algorithm dynamically adjusts particle velocities and evolves solutions to minimize conflicts and optimize resource allocation. In conclusion the authors state that by using a hybrid approach the system shows improved stability and faster convergence compared to standalone GA or PSO. To solve the issues in manual time tabling Khokale1 et al. [19] proposes a system by using machine learning and artificial intelligence techniques like Decision Trees, K-Means Clustering, and Random Forest algorithms. The decision tree algorithm was used to allocate time slots based on predefined constraints. It ensures that scheduling rules are followed while reducing overlaps and mismatches. The K-Means clustering is applied to group subjects, faculties and students based on similar patterns, which allows for optimized allocation of resources like faculty time and classrooms. The Random Forest algorithm Enhances prediction accuracy by evaluating multiple allocation possibilities and selecting the best-fitting timetable. Various inputs like the classroom details, faculty details etc were collected and processed through a Genetic Algorithm based scheduling engine, which uses an N-dimensional array structure to model and solve the timetabling problem. By combining machine learning and optimization algorithms, the system produces a structured, fair, and conflict-free timetables.

Kudale et al. [20] combines GA with heuristic conflict resolution. The proposed system can dynamically modify timetables to account for real-time changes, such as instructor absences, late arrivals or early departures. It ensures a balanced distribution of teaching hours by establishing maximum and minimum teaching hours per day, week or month for faculty. When faculty members submit leave requests, the system verifies the availability of substitute teachers, preventing conflicts. It also prioritizes time slots for faculty members within the same department to reduce commutes between classes and to improve overall productivity. The system's design framework starts with user registration and login followed by inputting essential data like departments, divisions, rooms, timings, subjects, faculty details etc. This data is then processed, and the system iteratively checks if all constraints are met generating a schedule only when they are. In conclusion the proposed system significantly reduces administrative workload while

enhancing institutional productivity.

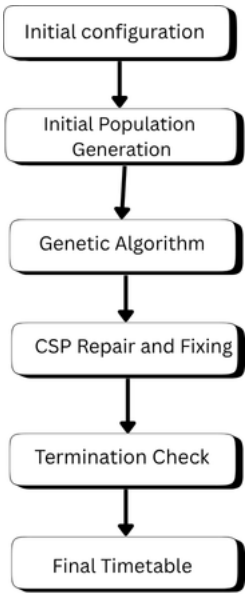
3. Methodology

The proposed system is an automated Timetable generator that helps in scheduling classes and examinations department wise. The system works by a hybrid approach by using Genetic algorithm and Constraint Satisfaction Problem by taking some hard and soft constraints into consideration. The system aims at helping educational institutions in overcoming various challenges caused by manual timetable generation. The proposed model is a combination of Genetic algorithm (GA) and Constraint Satisfaction Problem (CSP). Genetic algorithm provides global optimization while CSP helps in maintaining feasibility. The methodology showing how the work was carried out, is depicted in Figure 1, The methodology consists of the following steps.

3.1 Initial configuration

Within the stage the inputs required for the timetable is collected. Various inputs like Number of classes in each department, number of working days, number of periods per each day, faculty details, subject details, room availability, faculty preference etc are collected. The study was carried out considering various hard and soft constraints. Hard constraints are those that must be strictly followed and soft constraints are those that can be violated if a solution does not exist.

Figure 1: Methodology of proposed system



The various hard constraints defined are:

- Faculty conflicts: No faculties should be assigned in two different classes on the same day or time slot.
- Lab pairing consistency: The two designated lab slots in a day for a class must contain the same lab.
- Room conflict: No room can have two classes at the same time.
- Rooms must have required faculties.

The various soft constraints enforced are:

- Subject frequency balancing to target distribution: Subject occurrences are balanced based on remaining periods and credit-weighted distribution.
- Priority to higher-credit courses during conflict resolution and lab consistency: In conflicts at the same

slot, the class teaching the higher-credit course is kept; others are reassigned. In lab slot mismatches, the higher-credit course is chosen if neither is clearly a lab.

- Prefer certain time slots for certain courses

3.2 Initial population generation/ CSP repair

Generate a random schedule for all classes taking random subjects and random lab hours. Apply the initial CSP repair to ensure basic feasibility. In CSP there will be a set of variables and each variable has a domain where each variable is assigned to various constraints. Solving CSP means a value will be assigned to a variable so that all the constraints are satisfied, often using techniques like repair, backtracking, and heuristics to find a valid or high-quality solution. The model uses CSP to repair missing subjects in the random timetable generated and fix obvious faculty conflicts and ensure lab session consistency.

3.3 Genetic Algorithm Loop

Genetic Algorithm (GA) is a search and optimization technique that evolves a population of candidate solutions by mimicking natural selection—repeatedly selecting fitter solutions and generating new ones through crossover and mutation until it emerges a good solution. It maintains a population, evaluates each individual with a fitness function, selects parents, applies crossover to recombine them, mutates offspring to maintain diversity, and carries forward the next generation, often preserving the very best via elitism. GA can explore broadly and improve solutions iteratively. GA helps in generating an initial population of department timetables and scores each one with a fitness. Each generation preserves the top performers ensuring the best schedules are never degraded by crossover or mutation. In crossover, Children are built by two-point crossover at a relatively high-rate splicing segments from both parent class timetables to mix good patterns across schedules. The Mutations swap or replace only non-lab slots and occasionally replace a subject, which maintains feasibility while injecting diversity and exploration. If progress stalls, the mutation rate is temporarily increased to escape local optima, balancing exploration and exploitation.

3.4 CSP repair and fixing

After crossover/mutation, each child is passed through a CSP repairer that fixes hard violations (e.g., faculty conflicts, lab pairing) and nudges soft targets (subject frequencies), so evolution focuses on improving already-feasible or near feasible schedules. It iteratively fixes violations, checking constraints and repairing until critical issues are resolved or a limit is reached

3.5 Termination Check

Check whether maximum generations have reached, the fitness stagnation has detected and whether all the critical constraints have satisfied. Else the process continues until an optimal solution have reached.

3.6 Final Timetable

The final department level optimized timetable will be dis played along with the fitness score and constraints violations. The detailed workflow is depicted in Figure 2

4. Results and Discussions

A hybrid Genetic Algorithm (GA) and Constraint Satisfaction Problem (CSP) repair approach was employed to generate high quality, feasible timetables for a department. The GA provided global exploration through elitist tournament selection, two-point crossover, and adaptive, domain aware mutation restricted to non-lab slots, while a CSP repairer enforced hard rules after each evolutionary step. Over generations, fitness improvements reflected reductions in critical violations, with adaptive mutation mitigating stagnation. This integration yielded timetables to satisfy strict feasibility while aligning with

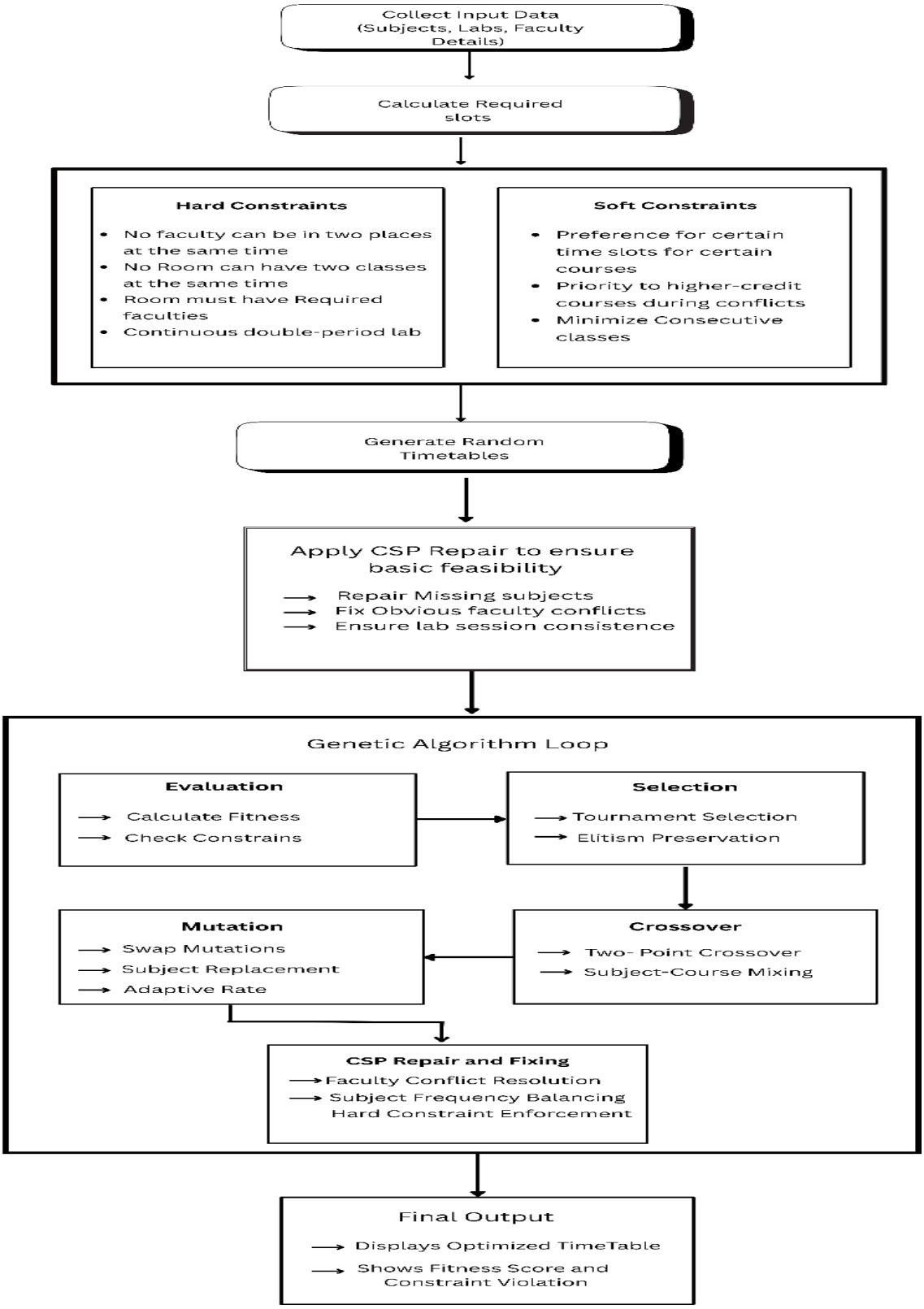
pedagogical priorities, demonstrating the effectiveness of coupling GA's exploratory search with CSP's targeted repairs for complex scheduling.

Genetic Algorithm (GA) Parameters: A population size of 100 was used. where the population size shows the Number of candidate solutions (timetables) in each generation. The GA runs over 300 iterations before stopping. A mutation rate of 0.2 was used. The mutation rate helped in modifying a gene i.e., to swap a subject slot. Crossover rate of 0.7 was used for combining two parents to create offsprings. CSP Repair Parameters: CSP Parameters enforce hard constraints (labs, faculty conflicts) before optimizing soft ones. Based on Credit of subjects the model ensures subjects are scheduled enough times. Faculty conflicts (-1000 penalty) are prioritized over subject balance (-10). Evaluation Metric- The hybrid Genetic Algorithm (GA) and Constraint Satisfaction Problem (CSP) timetabling model uses a combination of hard constraint satisfaction metrics and soft constraints optimization metrics.

Hard Constraint Metrics: The constraints that must be satisfied. These ensures the timetable is feasible. Violations should be minimized to zero. The model showed Zero faculty conflicts across all departments and 100% lab consistency with all lab sessions occupying consecutive time slots as required.

Soft Constraint Metrics: The soft constraints measure the timetable quality. Subject distribution imbalance was reduced to a mean deviation of 0.8 slots per subject. Preferred slot allocation reached 78% compliance for high-priority faculty requests.

Figure 2: Workflow of Timetable Generation Model



5. Empirical Modelling

The proposed system is formulated as a constraint optimization problem which mainly focuses on assigning courses to different time slots in respective rooms by considering various constraints. There are

mainly four entities: Courses (C), Faculties(F), Rooms(R) and Time slots(T). Each course has a faculty associated with different rooms in a particular time slot. Let the possible assignment A be; A: C-> F x R x T

The proposed model is implemented using Genetic Algorithm and Constraint Satisfaction Problem. The GA explores the vast solution space using various mechanisms like se lection, crossover, and mutation. The fitness function was introduced to quantify feasibility based on penalizing violations of hard constraints and rewarding the occurrence of soft constraints. But due to the complexity of constraint interaction GA sometimes struggles to maintain feasibility. In-order to resolve the problem CSP is introduced. The CSP is integrated into the proposed model in two different points: During initial population generation and to repair the offspring produced as a results of GA.

Parameter tuning was a critical phase in empirical modelling. Population size (40-100 individuals), mutation rate (0.1-0.2), and crossover rate (0.7-0.9) were optimized through parameter sweeping. The proposed system was able to produce high quality solutions under different scheduling scenarios, which indicates good generalization ability.

6. Conclusion

The paper shows how optimization algorithms can be brought to solve the problems faced in the task of timetable generation in academic institutions. Through the development of a hybrid model the system was able to attain an accurate result. The usage of Genetic Algorithm helped in Global Exploration and CSP for target feasibility repair. It delivers a system that is not only theoretically proved but also can be practically implemented. By considering both hard and soft constraints, the system ensures that the essential requirements are met by enhancing overall schedule quality. By this the system not only improves efficiency but also contributes to better resource management and increased user satisfaction. The approach reduces the burden on administrative staff and minimizes human errors making the scheduling process smoother and more dependable. The thesis also covered the existing approaches that were used for the task of timetable generation. The work can be extended in future for building a model with more advanced techniques and to improve the accuracy. Finally, the implementation of optimization-based timetable generation marks a meaningful step toward smarter, adaptive and reliable academic scheduling systems that can grow with institutional needs.

7. Acknowledgement

I would like to thank the God Almighty for the benevolent blessings without which this work is impossible. I wish to express my sincere thanks to, our Principal, Head of the Department, Sacred Heart College for the kind support during the phase of my project. I would like to express my sincere gratitude to my guide Mr. C. Vishnu Mohan, Assistant Professor, Department of Computer Science, Sacred Heart College for his valuable guidance, supervision and encouragement throughout the period of study. His prudent academic advice and deep knowledge in research have played an exceptionally vital role in this research work. I wish to express my esteem gratitude and respect for my parents for their continuous encouragement, valuable advice, prayers, concern. Finally, I thank all those, who have helped me directly and indirectly in the successful completion for my work.

References

1. Saini, S., Buhadia, M., Udaywa, P., Rani, A. (2024, April), “Optimizing academic schedules: A

timetable generator approach", International Journal of Innovative Research in Technology, 10(11).
2. Kumar, A., Singh, K., Sharma, N. (2013), "Automated timetable generator using particle swarm optimization", International Journal of Recent Innovation Trends in Computing and Communication, 1(9).
3. Alghamdi, H., Isubait, T. A., Lhakami, H. A., Baz, A. (2020), "A review of optimization algorithms for university timetable scheduling", Engineering, Technology Applied Science Research, 10(6).
4. Adrianto, D. (2014), "Comparison using particle swarm optimization and genetic algorithm for timetable scheduling", Journal of Computer Science, 10(2).
5. Sahoo, R. K., Ojha, D., Mohapatra, D. P., Patra, M. R. (2017), "Automatic generation and optimization of course timetable using a hybrid approach", Journal of Theoretical and Applied Information Technology, 95(1).
6. Firke, R., Bhabad, P., Gangarde, O., Magar, A., Tawlare, A. (2023), "Automatic timetable generation system", International Journal of Creative Research Thoughts, 11.
7. Kumar, S., Cheema, S. S. (2024), "A survey on optimization algorithms: Challenges and future opportunities", International Journal of Advanced Research in Computer Science, 15.
8. Mahajan, D., Malakar, G., Lath, R., More, T. (2024), "Timely trigger: A smart timetable generator", International Journal of Novel Research and Development, 9(4).
9. Latpate, A., Sayyed, N., Bargal, C., Sawant, A., Choudhari, J. S. (2024), "AI based automatic timetable generator using React", International Journal of Creative Research Thoughts, 12(4).
10. Puttaswamy, A., Khan, H. M. A. A., C. S. V., P. A. (2018, May), "A study on automatic timetable generator", International Journal of Innovative Research and Growth.
11. Goyal, P., Khadelwal, V., Dubey, V., Kishor, I. (2025), "Automatic time table generator: Revolutionizing scheduling efficiency", International Journal of Innovative Research in Technology, 11(11).
12. Uma, P., Sharvesh, P. S., Pradeep, M., Sathishkumar, P., Senthilnathan, R. (2023, April), "Automatic timetable generation", International Journal of Research in Applied Science and Engineering Technology, 11(4).
13. Preethi, P. G., Pratippa, T., Janani, S., "Automated university timetable generation system using PHP" Journal of Information Technology and Digital World.
14. Kusumawardani, D., Muklason, A., Supoyo, V. A. (2019), "Examination timetabling automation and optimization using greedy-simulated annealing hyper-heuristic algorithm, "In Proceedings of the 12th International Conference on Information Communication Technology and Systems.
15. Nwufoh, C. V., Achimugu, P. (2021), "A hard constraints satisfaction problem algorithm for university course time tabling", In Proceedings of the 1st International Conference on Data Science and Engineering, Nigeria.
16. Saw, S. K., Lawrence, S., Karunakara, S., Komal, N. H., Kumar, V. G. (2025), "Adaptive scheduler: AI optimization of academic timetable", International Journal of Scientific Development and Research.
17. Kaldoke, R., Matkar, G., Bhalerao, G., Tawlare, A. (2024, May), "Automated timetable optimization: A machine learning-based adaptive technique", International Journal of Creative Research Thoughts, 12(5).
18. Rashmi, K. R., Abhishek, A. M. B. (2021, June), "Automated university timetable generation using prediction algorithm", International Research Journal of Engineering and Technology, 8(6).

19. Khokale, S. R., Jadhav, A., Chavan, R., Wani, S., Iwanate, P. (2025, March), "A survey paper on timetable generator using AI methods", International Research Journal of Advanced Engineering Hub, 3(3).
20. Kudale, V., Jangam, P., Khair, C., Jadhav, R., Mote, A., Chadchankar, A. (2025, April), "From concept to code: Implementing timetable generation", International Journal of Ingenious Research Invention and Development, 4(2).