e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

REVIEW ON DUAL-METHOD TIMETABLE GENERATOR USING AI AND GENETIC ALGORITHM

Nikita J. Aghicha*1, Prof. Sana Shaikh*2, Aditya R. Deshmane*3, Krishna Y. Suralkar*4, Atharva R. Koshti*5

*1,3,4,5Student, Department Of Information Technology, GS Moze College Of Engineering Balewadi, Pune, Maharashtra, India.

*2Professor, Department Of Information Technology, GS Moze College Of Engineering Balewadi, India.

DOI: https://www.doi.org/10.56726/IRJMETS64111

ABSTRACT

This project aims to develop a robust timetable generator that combines artificial intelligence (AI) techniques and genetic algorithms (GA) to simplify and optimize academic scheduling. Leveraging AI language models, the system autonomously gathers university-specific data and rule-based constraints, ensuring each timetable meets institutional requirements. The GA then optimizes the AI-generated schedule by refining it through iterative processes, producing a final timetable that minimizes conflicts and maximizes resource allocation. This dual approach provides academic institutions with an efficient, scalable solution that requires minimal manual intervention.

Keywords: Timetable Generator, Artificial Intelligence, Genetic Algorithms, Dual Approach, Academic.

I. INTRODUCTION

In educational institutions, scheduling timetables is a complex, resource-intensive task that demands careful consideration of constraints such as lecturer availability, classroom capacity, and student distribution across courses. Current manual or semi-automated scheduling methods often struggle to handle these complexities, leading to inefficiencies and errors. This project addresses these challenges through a dual-method approach that combines the strengths of AI and genetic algorithms, providing a streamlined solution that not only generates but also optimizes academic schedules. By automating rule adherence and conflict reduction, this approach reduces the administrative workload and enhances scheduling accuracy.

This dual-method approach—combining AI for initial rule-based generation and GAs for iterative refinement—leverages the unique strengths of both approaches. AI ensures timetables are generated with institution-specific accuracy, while GAs provide the scalability and flexibility needed to optimize timetables over successive iterations. This approach represents a cutting-edge solution to timetabling that offers both structural integrity and adaptability to complex scheduling environments.

II. METHODOLOGY

Algorithm Used:

The system design includes two primary components:

1. AI-Driven Data Collection and Initial Timetable Generation:

The first stage involves collecting and interpreting institution-specific timetabling rules using AI-driven language models, such as Google Gemini. When a user selects their institution, the model automatically scrapes the web for relevant information, including the number of subjects, required theory and practical sessions, and any specific scheduling guidelines. This data is then combined with user-provided information, such as subjects, teacher availability, and lecture/practical ratios, to generate an initial timetable. This AI-driven component ensures the schedule aligns with university regulations, while minimizing errors in data entry and interpretation.

2. Genetic Algorithm for Timetable Optimization:

The initial timetable generated by the AI model is then optimized using a genetic algorithm, which is well-suited for handling multi-constraint optimization challenges like timetabling. The GA begins with a population of potential timetable configurations, each evaluated for fitness based on adherence to scheduling constraints. Constraints include faculty availability, room capacities, balanced distribution of lectures, and avoidance of overlapping classes. Over successive generations, the GA applies selection, crossover, and mutation operators to iteratively improve the timetable, selecting configurations that best meet the constraints. This process

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

continues until an optimal or near-optimal timetable is produced, minimizing conflicts and maximizing resource utilization.

The Genetic Algorithm (GA) in this project is a search and optimization technique inspired by the principles of natural selection. It aims to generate an optimal or near-optimal timetable by evolving a population of possible schedules over multiple generations.

Here's a breakdown of how the GA is applied to the timetable generation project:

1. Initial Population Generation

- The algorithm begins by creating an initial population of random schedules. Each individual (schedule) in this population represents a unique arrangement of classes, time slots, rooms, and instructors.

2. Fitness Function

- A fitness function evaluates each schedule based on how well it meets predefined constraints, such as avoiding class overlaps, adhering to faculty availability, and respecting room capacities.
- The fitness score indicates how "fit" or feasible a particular schedule is; schedules with fewer conflicts and better resource utilization have higher scores.

3. Selection

- Based on their fitness scores, the best-performing schedules are selected as parents for generating the next generation.
- Higher-scoring schedules have a greater chance of being selected, promoting schedules that meet constraints more effectively.

4. Crossover

- Pairs of selected schedules undergo a crossover operation, where parts of each parent's schedule are combined to create a new "child" schedule.
- This process helps combine the best aspects of different schedules, such as time slot allocations from one parent and room assignments from the other.

5. Mutation

- To introduce diversity, random mutations are applied to some schedules. A mutation may involve changing a class's time slot, reassigning a room, or adjusting instructor assignments.
- This prevents the population from stagnating in local optima and allows exploration of new potential solutions.

6. Iteration and Termination

- The algorithm iteratively repeats selection, crossover, and mutation across multiple generations.
- Each generation theoretically produces fitter schedules, converging toward an optimal or near-optimal timetable.
- The algorithm stops when it reaches a termination condition, such as achieving a maximum number of generations or producing a schedule with no conflicts.

Example Constraints Considered by GA in Timetabling

- No overlapping classes for students or instructors.
- Room capacities must accommodate class sizes.
- Time preferences or unavailability for faculty.
- Balanced schedules to avoid back-to-back classes for instructors or student groups.

III. MODELING AND ANALYSIS

Software Requirements:

- Python 3.8+ for backend development and AI integration
- HTML/CSS/JavaScript for frontend development
- MySQL or SQLite for data management and storage of generated timetables

Libraries and Tools:

- NumPy, Pandas, Flask/Django for backend development
- An AI model integration such as Google Gemini for web scraping and prompt-based timetable generation
- A genetic algorithm library or custom implementation for the optimization component.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

Hardware:

The system requires a server capable of efficiently running both the AI-driven and GA components to handle real-time adjustments and generate timetables without delay.

Technologies Used:

The backend is developed in Python, while the frontend relies on HTML, CSS, and JavaScript. The LLM integration automates the initial data collection and prompt-based timetable generation, while the genetic algorithm optimizes the timetable by iterating through possible configurations to reduce conflicts and enhance resource allocation.

IV. CONCLUSION

The Timetable Generator effectively combines AI-driven models with genetic algorithms (GAs) to address the complex, multi-constraint problem of academic scheduling. Traditional methods often require significant manual input and struggle to adapt to changing institutional needs, leading to inefficiencies and scheduling conflicts. By integrating AI and GA, this project creates a robust solution that automates and optimizes the process with minimal human intervention.

1. **AI-Driven Scheduling:**
   - The AI component automates rule-based scheduling by autonomously gathering institution-specific regulations and constraints, such as required lectures, practicals, and adherence to specific time allotments.
   - This automation significantly reduces the initial workload on administrators and faculty by minimizing manual data entry and ensuring compliance with university rules right from the start.
2. **Genetic Algorithm Optimization:**
   - After the initial timetable is generated, the genetic algorithm component iteratively refines it, handling complex constraints like faculty availability, room capacity, and lecture distributions.
   - The GA's iterative nature allows it to identify optimal configurations by continually selecting, crossing over, and mutating timetable variations to minimize scheduling conflicts and maximize the efficient use of resources, such as classrooms and faculty time.
3. **Efficiency and Scalability:**
   - Together, the AI and GA provide a solution that can efficiently handle both small-scale and large-scale scheduling challenges. The system's scalability makes it suitable for academic institutions with various complexities and constraints.
   - This dual-method approach proves that a combined AI-GA model can deliver not only accuracy and rule-compliance but also adaptability, reducing the administrative burden associated with timetabling.

V. REFERENCES

[1] Schaerf, A. (1999). A survey of automated timetabling. Artificial Intelligence Review, 13(2), 87-127.
[2] Burke, E. K., & Petrovic, S. (2002). Recent research directions in automated timetabling. European Journal of Operational Research, 140(2), 266-280.
[3] Kastner, M., & Igbaria, M. (2006). FET: Open source timetabling solution for educational institutions. Software Review Journal.
[4] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.
[5] Kingston, J. H. (2010). Timetable construction: The algorithms and complexity perspective. Springer Lecture Notes in Computer Science.
[6] Felipe de la Rosa-Rivera, et al. (2021). Algorithm selection for solving educational timetabling problems. Expert Systems With Applications, 174, 114694. https://doi.org/10.1016/j.eswa.2021.114694
[7] Sermeno, J. P., & Secugal, K. A. (2021). Class scheduling framework using design patterns. 2021 2nd Int. Conf. on Innovative Tech. Convergence (CITC). IEEE
[8] Psarra, E., & Apostolou, D. (2023). Genetic algorithms for university timetabling. 2023 14th Int. Conf. on Info, Intelligence, Systems & Apps. IEEE.