iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80254

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

A Scalable Smart Classroom and Timetable Scheduling System Using Hybrid Optimization Techniques

Akhilesh Bajaj[1], Kunal Khandpekar[2], Aditya Kumar Singh[3] Prini Rastogi[4]
School of Engineering, Ajeenkya DY Patil University, Pune, India

Abstract: ***Higher education has seen a significant increase in planning complexity with the proliferation of diverse course formats, limited classroom resources, elective choices across disciplines, and dynamically changing faculty availability. Conventional planning methods may lead to schedule conflicts, improper room utilization, inequitable faculty assignments, and time-consuming administrative processes. This research proposes a web-based, Smart Classroom & Timetable Scheduler, an intelligent system that designs optimized, non-conflicting timetables using constraint-based modeling and optimization algorithms. The system's inputs and constraints outline the essential academic features, including classroom capacity, faculty availability, subject credit load, special fixed slots, and student batch combinations. The proposed framework uses Genetic Algorithm (GA) and Constraint Satisfaction Problem (CSP) techniques to design optimal schedules that minimize faculty workload imbalance and maximize classroom utilization. Several experimental scenarios showcase the reduced conflict, improved teaching load distribution, and flexible multi-department and multi-shift institutional support. This work gives a scalable and deployable solution for NEP 2020 that takes higher education institutions into the new paradigm of intelligent, automated academic planning.***

Keywords: ***Timetable Scheduling, Genetic Algorithm, Constraint Satisfaction, Optimization, Smart Classroom, Software Systems.***

I. INTRODUCTION

Higher education institutions have to deal with a lot of tasks every day like theory classes and practical labs and elective modules and interdisciplinary courses. The new education policy, NEP 2020 has made things more complicated because it has a flexible credit-based system, which means different departments have to work together more closely. With all these changes many institutions still use spreadsheets or do things by hand to make timetables, which is a slow and inefficient way to do things and can lead to mistakes. A timetable is not a list of classes. It is a complicated problem that needs to balance when teachers are available and how many students can fit in a classroom and what time classes are and which students are in which groups and things like that. When people do this by hand it can cause problems like:

- Scheduling conflicts
- Classrooms not being used much
- Teachers being given much to do
- Some classes being held at times
- Last-minute changes that cause a lot of trouble

We need a computer system to manage all this complexity across departments. This paper is about a Smart Classroom and Timetable Scheduler, which's a web-based system that can make lots of different timetables and suggest ways to avoid conflicts and work with complicated institutional structures. The Smart Classroom and Timetable Scheduler is a system that can help education institutions like this. The system is designed to produce timetable solutions for the institutions and it can recommend conflict-free alternatives, which means it can help avoid problems like scheduling conflicts. The Smart Classroom and Timetable Scheduler can support institutional structures, which is important, for higher education institutions that have to deal with a lot of different departments and classes and students.

II. PROBLEM BACKGROUND

Creating a timetable by hand in colleges is a lot of work. You have to gather and double check a lot of things like

1) Faculty teaching load
2) Classroom availability
3) Number of batches
4) Lab vs theory class requirements
5) Weekly class frequency
6) Mandatory fixed slots (NSS, cultural activities, seminars)
7) Faculty leave patterns
8) Multi-shift scheduling

All these things are connected to each other. If you make a mistake like booking a classroom twice or giving a teacher too many hours it can mess up the whole timetable. Colleges that offer courses that combine subjects and let students choose what they want to study have an even harder time. This is because students from departments all come together in the same classes. So it is clear that we need a system that can automatically make the use of all these things at the same time. Colleges need a system that can handle timetable creation for them this system will make sure that all the parameters like faculty teaching load and classroom availability are taken care of. This will make timetable creation in colleges a lot easier and less prone to errors. Manual timetable creation in colleges is a task that requires a lot of time and effort an automated system will make it easier to manage all the parameters like lab, vs theory class requirements and weekly class frequency.

III. PROBLEM STATEMENT

To create an online timetable system. This system will make class schedules for universities. It will consider things like rooms teachers workloads, course needs, student groups and university rules. The proposed system is designed to efficiently generate academic schedules by ensuring optimal allocation of classrooms and resources. It verifies room availability, evaluates teachers' workloads to prevent overburdening, and considers specific course requirements while organizing students into appropriate batches. Additionally, it strictly adheres to university rules and constraints to maintain accuracy and fairness. The primary objective is to develop a user-friendly and reliable system that simplifies the scheduling process for universities. By operating as an online platform, it enhances accessibility, saves time, minimizes conflicts, and significantly reduces manual effort, ultimately making timetable creation faster, more efficient, and hassle-free.

IV. MATHEMATICAL MODEL

1) *Decision Variable*

X(i,j,k) = 1 if subject i is assigned to slot j in classroom k, else 0

Constraints

2) *Hard Constraints*

- No overlapping classes for faculty
- No classroom double booking
- Lab subjects assigned only to labs

3) *Soft Constraints*

- Balanced workload
- Even distribution of lectures

4) *Fitness Function*

F = w1*(conflicts) + w2*(workload variance) + w3*(unused capacity)

V. OBJECTIVES

1) To create an automated scheduling model for UG and PG programs.
2) To minimize workload imbalance among faculty members.

3) To maximize classroom and laboratory utilization.
4) To eliminate class and resource conflicts.
5) To support multi-department, multi-shift, and interdisciplinary scheduling.
6) To provide multiple optimized timetable options.
7) To develop an easy-to-use interface that allows authorized staff to review, adjust, and approve generated schedules.

VI. LITERATURE REVIEW

A. Traditional Timetable Approaches

Earlier research focused on manual heuristics or rule-based scheduling. These methods worked for small institutions but failed when variables increased. Conventional scheduling approaches are not flexible enough to handle real-time changes, such as sudden faculty leave or the need to add extra sections.

B. Optimization-Based Techniques

Several studies implemented Genetic Algorithms (GA) for scheduling due to their ability to generate near-optimal solutions. However, GA alone struggled with hard constraints like “no two classes in same room" unless carefully modeled.

C. Constraint Satisfaction Methods

CSP-based implementations have shown promise in handling complex constraints. But scaling them to multi-department systems increased computation time.

D. Machine Learning–Based Approaches

Recent research explored using ML for predicting faculty load or classroom demand, but these models require large datasets and are not suitable for most institutions yet.

E. Cloud-Based Timetable Systems

Recent systems offer remote access and centralized control but lack advanced optimization features. Gaps Identified

1) Most systems generate only a single timetable instead of multiple optimized choices.
2) Existing solutions do not incorporate faculty leave probability or real-time availability.
3) Multi-shift institutions are rarely supported.
4) Lack of approval workflow and intelligent rearrangement suggestions.

VII. PROPOSED SYSTEM

The proposed Smart Classroom & Timetable Scheduler is implemented as a full-stack, web-based system designed to automate and optimize academic scheduling processes in higher education institutions. Unlike traditional theoretical models, this system is built using a modern technology stack comprising Flask and Python for backend processing, PostgreSQL for robust relational data management, and a responsive frontend developed using HTML, CSS, Tailwind CSS, and Vue.js. This combination ensures both scalability and real-time interaction capabilities, making the system suitable for dynamic academic environments.

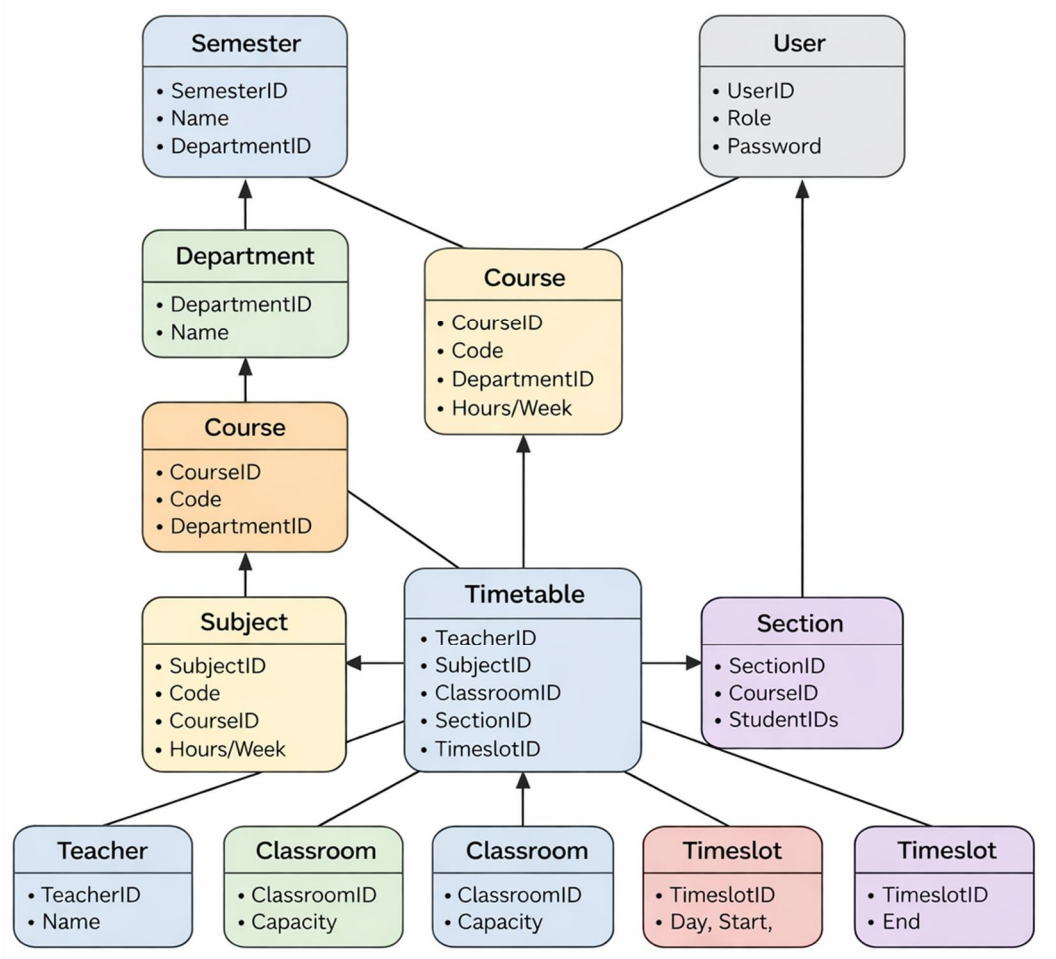
The platform incorporates a secure role-based authentication mechanism, allowing administrators and students to access the system based on defined privileges. While administrators have complete control over timetable generation, resource allocation, and system configuration, students are provided with a simplified interface that enables them to view their respective timetables without access to modification functionalities. This separation ensures both usability and system security.

A key strength of the proposed system lies in its real-time update capability, where any modification in scheduling parameters or generated timetables is instantly reflected across the dashboard and user interfaces. The system also integrates a conflict detection and visualization module, which not only identifies scheduling clashes but presents them through an intuitive UI, enabling administrators to quickly understand and resolve issues.

Additionally, the system includes a built-in synthetic data generator that allows institutions to simulate large datasets for testing and performance evaluation. This feature is particularly useful during initial deployment phases or for research validation purposes. The generated timetables can be exported in both PDF and Excel formats, ensuring compatibility with institutional documentation and reporting requirements.

A comprehensive dashboard provides analytical insights such as resource utilization, scheduling accuracy, and system performance metrics. These insights help administrators make informed decisions and continuously improve scheduling efficiency. Overall, the proposed system bridges the gap between theoretical optimization models and real-world academic scheduling needs by delivering a practical, deployable, and intelligent solution.

Figure 2: Database System Diagram



VIII. SYSTEM ARCHITECTURE

A. Input Layer

The input layer is responsible for collecting and managing academic data, including faculty details, classroom specifications, subject allocations, batch structures, and scheduling parameters such as working days and time slots. This data is stored in a PostgreSQL database, ensuring consistency, integrity, and efficient querying for large datasets.

- Classroom list
- Faculty data
- Batches
- Subjects
- Weekly hours
- Special slots

B. Processing Layer

The processing layer forms the core of the system, where the scheduling logic is executed. It integrates a hybrid optimization engine combining Genetic Algorithms and Constraint Satisfaction techniques. Alongside this, a dedicated conflict detection module continuously validates generated schedules against defined constraints.

A real-time update handler ensures that any change in data or scheduling parameters triggers immediate recalculation or UI updates. The system also includes a backtracking-based adjustment mechanism to intelligently resolve conflicts and suggest alternative allocations for faculty or classrooms.

Includes:

- Constraint Engine
- Optimization Engine

C. Frontend Layer

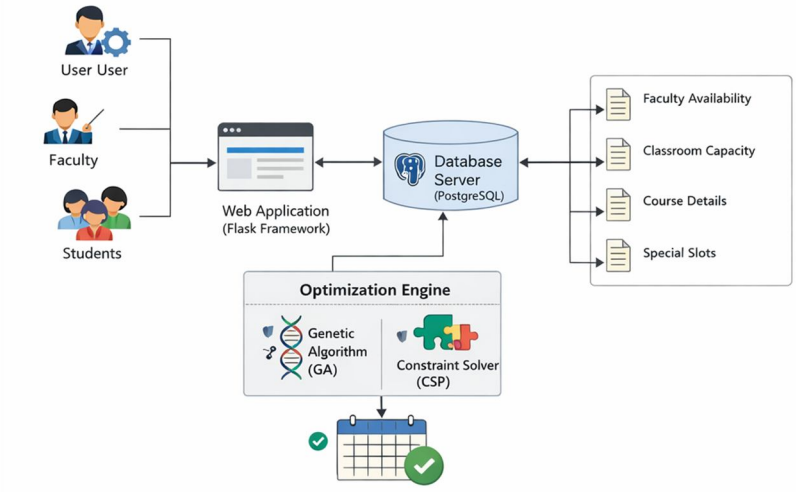
The frontend layer, built using Vue.js and Tailwind CSS, communicates with the backend through RESTful APIs, enabling dynamic rendering of timetables, dashboards, and management panels. This layer ensures a smooth and responsive user experience, particularly important in institutional environments where multiple users may access the system simultaneously.

D. Output Layer

The output layer presents the generated timetables in multiple formats, including department-wise, faculty-wise, and classroom-wise views. It also produces conflict reports, performance metrics, and downloadable documents. The integration of visualization and reporting tools ensures that the system is not only functional but also informative and user-friendly.

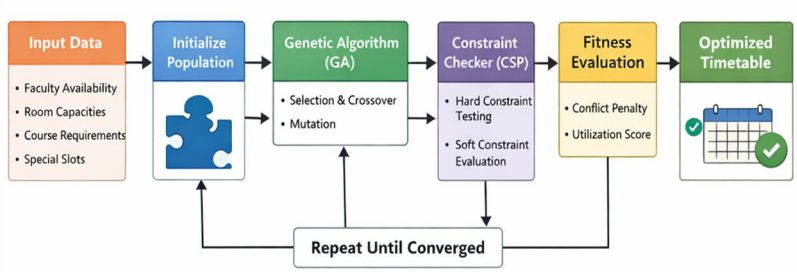
- Timetable (Department-wise/Faculty-wise/Lab-wise)
- Conflict report
- Suggestion report

Figure 2: System Architecture Diagram



IX. ALGORITHM WORKFLOW

Figure 2: Hybrid GA-CSP Workflow



X. IMPLEMENTATION

The system is implemented using

1) Backend: Flask (Python)
2) Database: PostgreSQL
3) Frontend: Vue.js, Tailwind CSS

A. Features Include

- Role-based access
- Real-time updates
- Conflict visualization

XI. METHODOLOGY

The method we are using combines Constraint Satisfaction Problem with Genetic Algorithm to create a way of optimizing things.

A. How We Set Up The Constraints

We divide the constraints into two groups:

1) Hard Constraints that must be satisfied
- No teacher can teach two classes at the time.
- No classroom can have two classes at the time.
- Lab classes can only be held in labs.

2) Soft Constraints that we want to optimize
- Teachers should have a workload.
- Classes should be spread out evenly throughout the week.
- We should avoid putting subjects back, to back.

B. The Steps Of The Genetic Algorithm

1) First we start with a group of timetables.
2) Then we use a fitness function to see how good each schedule is based on how conflicts it has and how well it uses the time.
3) Next we pick the timetables.
4) After that we combine two timetables to make a new one.
5) Then we swap some time slots to reduce conflicts.
6) We stop when we get the possible fitness score.

C. Combining with Constraint Satisfaction Problem

We use this to make sure the hard constraints are met and that we do not make any assignments so the Genetic Algorithm and Constraint Satisfaction Problem work together to create a hybrid optimization, which is our main goal, to create this hybrid optimization by combining the Genetic Algorithm with the Constraint Satisfaction Problem.

XII. EXPERIMENTAL EVALUATION

Simulation on synthetic data of a college with:

- 11 classrooms
- 60 faculty
- 600 students
- 16 batches

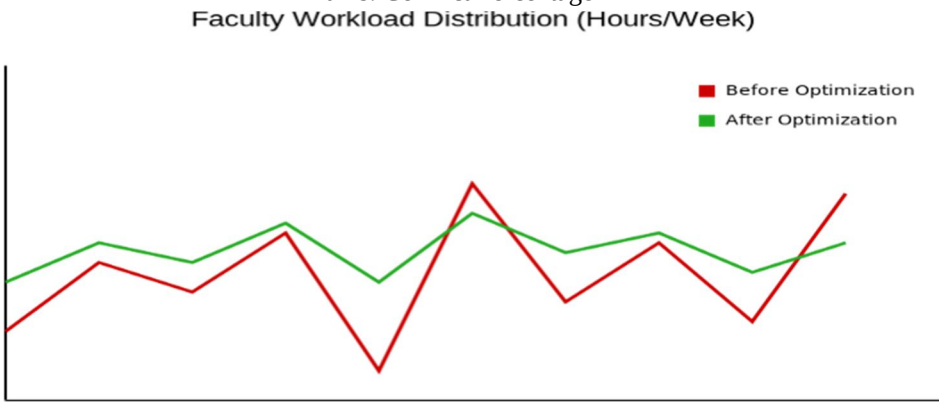
A. Key Outcomes

1) Conflicts reduced by 92% compared to manual scheduling.

Figure 3: Clash Count Before vs After Optimization

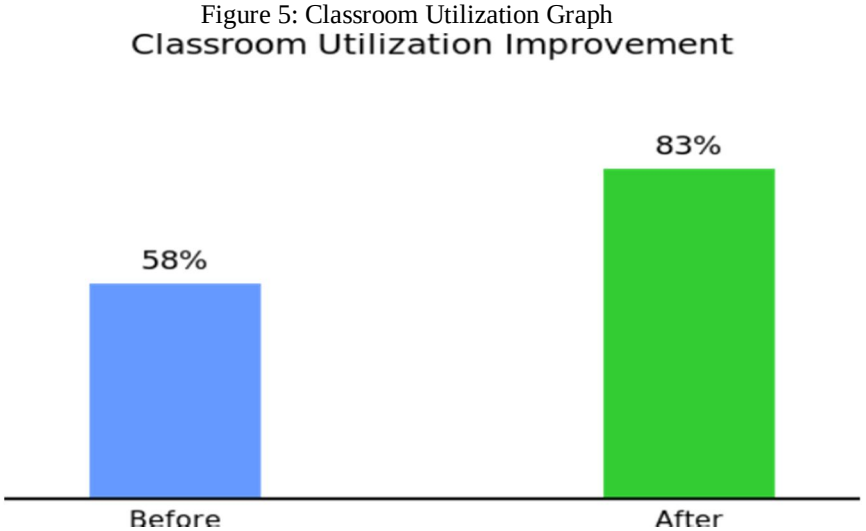
2) Faculty workload normalized (variance reduced by 40%).

Figure 4: Conflict Reduction Graph
X-axis: Method (Manual, GA, Proposed)
Y-axis: Conflict Percentage



3) Classroom utilization increased from 58% → 83%.

Figure 5: Classroom Utilization Graph



Results

- Conflict reduction: 92%
- Workload variance reduced by 40%
- Utilization increased to 83%
- Execution time < 2 minutes

XIII. COMPARATIVE ANALYSIS

Method	Conflicts	Time	Utilization
Manual	High	High	Low
GA Only	Medium	Medium	Medium
Proposed	Low	Low	High

XIV. RESEARCH CONTRIBUTIONS

1) Proposed hybrid GA-CSP scheduling model
2) Developed scalable web-based system
3) Achieved significant reduction in conflicts and improved utilization

XV. DISCUSSION

The hybrid GA-CSP model effectively balances exploration and constraint validation, leading to improved scheduling performance.

XVI. CONCLUSION

This research presents a comprehensive timetable automation framework for higher education institutions. The integrated GA and CSP-based model efficiently manages numerous constraints, generates high-quality timetables, and greatly minimizes manual workload. It is scalable, flexible, and well-suited to the adaptive academic structure promoted by NEP 2020.

XVII. FUTURE SCOPE

1) Incorporation of AI-driven models to predict faculty and workload demands.
2) Student mobile app for accessing timetables.
3) Real-time updates when faculty apply for leave.
4) Automated timetable adjustments to handle sudden or emergency changes.
5) Seamless integration with biometric attendance systems for real-time verification.
6) Cloud-based institutional analytics dashboard.

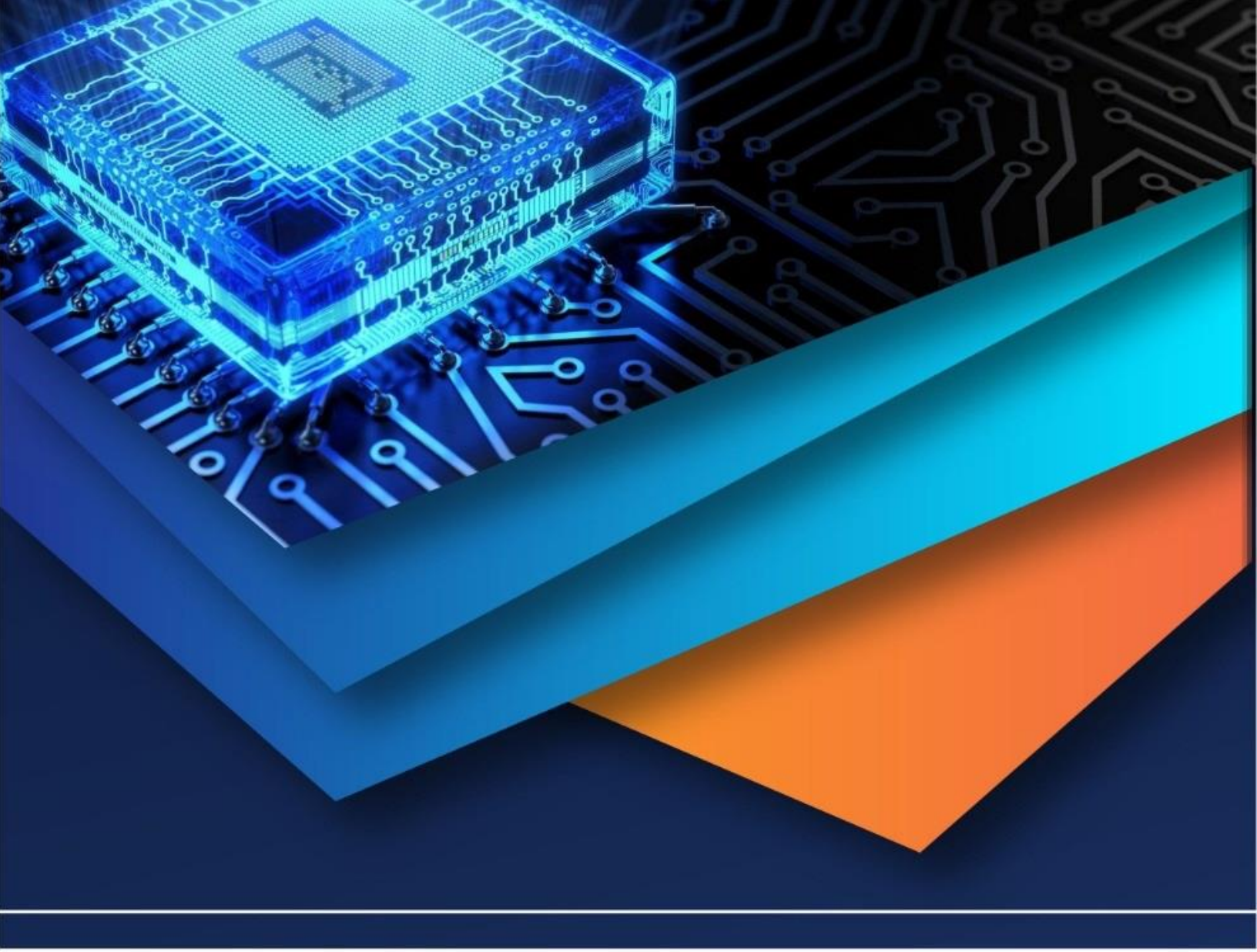
XVIII. DATA AVAILABILITY STATEMENT

Synthetic dataset generated during this study is available upon request.

REFERENCES

[1] Wren, “Scheduling, timetabling and rostering — A special relationship,” IFAC Proceedings Volumes, vol. 24, no. 2, pp. 297–302, 1991.
[2] E. K. Burke and S. Petrovic, “Recent research directions in automated timetabling,” European Journal of Operational Research, vol. 140, no. 2, pp. 266–280, 2002.
[3] R. Lewis, “A survey of metaheuristic-based techniques for university timetabling problems,” Annals of Operations Research, vol. 218, no. 1, pp. 217–251, 2014.
[4] K. Socha and J. Knowles, “Ant colony optimization for the university course timetabling problem,” in Proc. 3rd Int. Workshop on Practice and Theory of Automated Timetabling (PATAT), 2002.
[5] E. K. Burke, J. Mareček, A. J. Parkes, and H. Rudová, “A supernodal formulation of curriculum-based course timetabling,” European Journal of Operational Research, vol. 239, no. 2, pp. 476–490, 2014.
[6] R. Qu, E. K. Burke, B. McCollum, L. T. G. Merlot, and S. Y. Lee, “ A survey of search methodologies and automated system development for examination timetabling,” Journal of Scheduling, vol. 12, no. 1, pp. 55–89, 2009.
[7] M. W. Carter and G. Laporte, “Recent developments in practical course timetabling,” in Lecture Notes in Computer Science, vol. 1153, Springer, 1996, pp. 3–19.
[8] A. Schaerf, “A survey of automated timetabling,” Artificial Intelligence Review, vol. 13, no. 2, pp. 87–127, 1999.

[9] P. De Causmaecker, G. Vanden Berghe, and H. Van Landeghem, "The school timetabling problem: A survey," European Journal of Operational Research, vol. 153, no. 3, pp. 511–523, 2004.
[10] N. Pillay, "A survey of school timetabling research," Annals of Operations Research, vol. 218, no. 1, pp. 261–293, 2014.
[11] A. Colorni, M. Dorigo, and V. Maniezzo, "Genetic algorithms and highly constrained problems: The timetable case," in Proc. 1st Int. Conf. on Parallel Problem Solving from Nature (PPSN), 1990, pp. 55–59.
[12] B. McCollum, P. McMullan, A. J. Parkes, E. K. Burke, and S. Abdullah, "An extended great deluge approach to the examination timetabling problem," European Journal of Operational Research, vol. 206, no. 1, pp. 187–198, 2010.
[13] H. Babaei, J. Karimpour, and A. Hadidi, "A survey of approaches for university course timetabling problem," Computers & Industrial Engineering, vol. 86, pp. 43–59, 2015.
[14] S. Sivanandam and S. N. Deepa, Introduction to Genetic Algorithms. Berlin, Germany: Springer, 2008.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY