

Analyzing Automated Timetable Scheduling Approaches for Higher Education Institutions

Yankit Puri

Department of Computer Science
& Engineering
B.Tech, Computer Science &
Engineering
Academic Year 2026

Dheeraj Puri

Department of Computer Science
& Engineering
B.Tech, Computer Science &
Engineering
Academic Year 2026

Ayush Kumar Jha

Department of Computer Science
& Engineering
B.Tech, Computer Science &
Engineering
Academic Year 2026

Utkarsh Kailash Mahale

Department of Computer Science
& Engineering
B.Tech, Computer Science &
Engineering
Academic Year 2026

Mr. Bhushan Mali,

Assistant Professor,
Central Website Coordinator,
Department of Computer Science and Engineering,
School of Computer Sciences and Engineering,
Sandip University, Nashik, MH, India

Abstract — Preparing a class timetable is one of the most difficult recurring tasks for any higher education institution. The person in charge must think about which faculty member is free at what time, which classroom or laboratory is available, how many hours each subject needs per week, and dozens of other rules, all at the same time. Under India's National Education Policy (NEP 2020), this problem has become even harder because students can now choose electives across departments, follow flexible multidisciplinary combinations, and move between UG and PG credit structures, which multiplies the number of constraints an institution must satisfy. Manual scheduling using spreadsheets cannot realistically keep up with this complexity, and it frequently leads to class clashes, underused classrooms, and an uneven workload across faculty. This survey reviews ten published research works on automated timetable scheduling, covering rule-based methods, genetic algorithms, constraint satisfaction, harmony search, and graph coloring. The strengths and limitations of each approach are compared, and the key research gaps still left open in the literature are identified — particularly around dynamic rescheduling, multi-department coordination, and integrated administrator approval workflows. Based on these gaps, the paper proposes a web-based Smart Timetable Management System that lets an authorized administrator enter data about classrooms, batches, subjects, and faculty, and then automatically generates multiple optimized timetable options, checks them for conflicts, suggests rearrangements when no perfect solution exists, and routes the final version through a review-and-approval workflow before publishing it on the college website.

Keywords — Smart Timetable Management; NEP 2020; Higher Education; Optimization Algorithms; Genetic Algorithm; Constraint Satisfaction Problem (CSP); Multidisciplinary Scheduling; Web-Based Application; Resource Allocation; Faculty Workload; Automation.

I. INTRODUCTION

Every higher education institution has to solve the same puzzle at the start of every semester: how to fit all classes, teachers, students, classrooms, and laboratories into a fixed number of hours in a week without any clashes. This problem, formally known as the timetabling problem, is classified as NP-hard [1], meaning the number of possible combinations grows explosively as more subjects, faculty, and rooms are added, making it impossible to check every option by hand.

In the Indian context, this problem has become significantly more complex since the introduction of the National Education Policy (NEP 2020). NEP 2020 encourages flexible, multidisciplinary learning, allowing students to pick electives outside their core department, mix UG and PG credit pathways, and follow individualized subject combinations. This means a single institution's timetable is no longer a simple grid of fixed classes for fixed batches — it must instead accommodate overlapping departmental requirements, limited infrastructure, varying faculty teaching loads, and student-level elective choices all at once.

Most colleges still rely on manual scheduling using spreadsheets or very basic tools. These tools cannot track real-time faculty availability, room capacity, or subject-combination rules, so the resulting timetables are usually the first workable option found rather than the best one. This leads to frequent class clashes, classrooms that sit empty for hours while others are overbooked, faculty members with uneven workloads, and general dissatisfaction among both students and staff.

Over the last two decades, researchers have proposed a wide range of algorithmic techniques to automate timetable generation. Early work used simple rule-based programs [5, 6], while later research explored more powerful methods borrowed from artificial intelligence and operations research, including genetic algorithms [1, 7, 9], constraint satisfaction and graph-based modeling [4], harmony search [2], and hybrid parallel search techniques that combine more than one method [8]. Most recently, adaptive genetic algorithms that tune their own parameters during the search have been used to speed up convergence and better handle flexible, NEP 2020-style scheduling constraints [10].

Despite this large body of research, a practical gap remains: most published solutions are algorithm-focused prototypes tested on limited datasets, not complete, ready-to-deploy web systems with login access, data management, an approval workflow, and export features that an institution's administration can actually use day to day.

This survey paper has four goals: (1) to explain, in simple language, the major timetable scheduling techniques proposed in the literature; (2) to compare these techniques so a reader without a deep algorithms background can understand their trade-offs; (3) to identify the research gaps that remain, especially in the context of NEP 2020; and (4) to propose a practical, web-based Smart Timetable Management System that brings together the best ideas from the literature into a usable tool. Section II reviews ten related works; Section III presents a comparative analysis; Section IV identifies research gaps; Section V describes the proposed methodology and architecture; Section VI discusses expected outcomes; and Section VII concludes the paper.

II. LITERATURE REVIEW

This section reviews ten research works related to timetable scheduling, spanning 2011 to 2022, so that both foundational

techniques and more recent adaptive methods are represented. Table I summarizes each paper's year, authors, method, key finding, and main limitation. Full inline citation numbers [1]–[10] correspond to the reference list at the end of this paper.

Genetic Algorithm (GA) based approaches remain the most widely studied technique [1, 5, 7, 9, 10]. GA works by treating each candidate timetable as a "chromosome" and gradually improving a population of such chromosomes through selection, crossover, and mutation, in the same way natural evolution improves a population over generations. Recent GA variants add adaptive parameter tuning [10] or hybridization with greedy/local search [8] to improve both speed and solution quality.

Constraint Satisfaction (CSP) and graph-based approaches [4] treat the timetable as a set of variables — classes, rooms, time slots — that must simultaneously satisfy a list of rules. These methods are excellent at guaranteeing that hard rules (e.g., a teacher cannot be in two rooms at once) are never broken, but can be slow for very large, real-world timetables.

Metaheuristic and hybrid search methods, including Harmony Search [2], are useful because they can escape a "stuck" or poor-quality solution and keep searching for something better, though they generally take longer to converge than a well-tuned genetic algorithm.

Among the reviewed works, reference [10] (2022) represents the most recent trend: an adaptive genetic algorithm that dynamically tunes its own mutation and crossover behavior instead of relying on a single fixed configuration. This adaptive-optimization direction is well aligned with the practical, high-flexibility scheduling needs of NEP 2020-style institutions, even though none of the ten reviewed works were built and evaluated as a complete, deployable web system — a gap this paper addresses directly in Section V.

TABLE I Summary of Ten Reviewed Research Works

No.	Year	Author(s)	Method	Key Finding	Limitation
1	2011	S. Yang & S. N. Jat	Genetic Algorithm (guided + local search)	GA combined with guided and local search improved solution quality for university course timetabling.	High computational cost for very large timetables.
2	2012	M. A. Al-Betar, A. T. Khader & M. Zaman	Hybrid Harmony Search	Harmony-search metaheuristic hybridized with local improvement gave strong, near-optimal timetables.	Parameter tuning (harmony memory size, pitch rate) is sensitive.
3	2014	N. Pillay	Survey / Review	Comprehensive review of school timetabling research, classifying methods and open problems.	Review only; no new algorithm or system proposed.
4	2017	R. Ganguli & S. Roy	Graph Coloring	Modeled timetabling as a graph-coloring problem to avoid clashes between	Struggles with soft constraints like faculty preference.

No.	Year	Author(s)	Method	Key Finding	Limitation
				overlapping classes.	
5	2020	P. Tanksali et al.	Genetic / Heuristic hybrid	Automated timetable generation system reducing manual administrative effort at a university.	Limited testing beyond a single institution's dataset.
6	2020	V. Amarnadh et al.	Rule-Based Automation	Built an automatic timetable generator to remove common human clash errors in colleges.	No optimization; only avoids hard clashes.
7	2020	S. Markal, S. Ghorpade & D. Chalke	Genetic Algorithm	GA-based timetable generator with a simple, user-friendly interface for faculty/room/subject input.	Optimization quality not benchmarked against other methods.
8	2021	A. Rezaeipannah, S. S. Matoori & G. Ahmadi	Hybrid Parallel GA + Local Search	Improved parallel GA with local search significantly improved timetable quality and runtime.	Requires multi-core / parallel computing infrastructure.
9	2021	M. K. Kakkar et al.	Evolutionary Algorithm	Generated class schedules using evolutionary computation with configurable constraint weighting.	Convergence speed drops as constraint count grows.
10	2022	S. Han & L. Xiao	Adaptive Genetic Algorithm	Improved adaptive GA dynamically adjusted mutation/crossover rates for faster convergence.	Tested mainly on simulated, not real institutional, data.

III. COMPARATIVE ANALYSIS OF TIMETABLE SCHEDULING APPROACHES

Table II compares the major scheduling approaches discussed in Section II across five practical dimensions: the quality of the timetable produced, how fast the method runs,

how well it scales to large institutions, its main advantage, and its main limitation. This comparison helps in choosing the right technique for an institution's size and available computing resources, rather than assuming one method is always best.

TABLE II Comparison of Timetable Scheduling Techniques

Approach	Optimization Quality	Execution Time	Scalability	Advantages	Limitations
Rule-Based	Low–Medium	Fast	Low	Simple to implement	Poor for complex constraints
Greedy Algorithm	Medium	Fast	Medium	Quick, feasible solutions	May miss the optimal solution
Genetic Algorithm	High	Medium	High	Produces near-optimal timetables	Requires careful parameter tuning
Constraint Satisfaction (CSP)	High	Medium	High	Handles many constraints reliably	Computationally expensive
Harmony Search / Graph Coloring	Medium–High	Medium	Medium	Good at avoiding hard clashes	Weaker on soft preferences
Simulated Annealing / ACO	High	Slow	Medium–High	Escapes poor local solutions	Longer convergence time
Hybrid AI/ML (GA+RL, adaptive GA)	Very High	Medium–Slow	High	Adaptive, learns institution patterns	Higher implementation complexity; needs data

As Table II shows, simple rule-based and greedy methods [5, 6] are fast and easy to build but scale poorly for NEP 2020-style multidisciplinary constraints. Genetic algorithm and CSP-based methods [1, 4, 7, 9] give high-quality results and reasonable scalability, which is why they dominate recent

literature. Hybrid and adaptive approaches [8, 10] offer the best overall quality and adaptability but need more development effort and carefully tuned parameters. This trade-off directly motivates the phased design used in Section V: the proposed system starts with a fast, practical rule-

based/greedy engine and is architected so a Genetic Algorithm or CSP module can be added later without a redesign.

IV. Research Gap Identification

Although the works reviewed in Section II show clear progress in automated timetable scheduling, several gaps remain when compared with what real institutions need under NEP 2020. Most published solutions [1, 5, 6, 7, 9, 10] focus narrowly on producing one feasible timetable for a fixed curriculum, without addressing the elective flexibility, dynamic rescheduling, and approval processes that real colleges require. Even the most recent adaptive approach [10] focuses mainly on the algorithm itself rather than the full administrative workflow around it. Based on the literature review and comparative analysis, the following research gaps have been identified:

- Limited support for dynamic timetable rescheduling — most systems generate one static timetable and struggle when a teacher goes on leave or a room becomes suddenly unavailable.
- Weak handling of NEP 2020-style multidisciplinary and elective scheduling, where students from different departments must share limited common time slots.
- Inadequate handling of multi-department and multi-shift scheduling, common in larger colleges running morning and evening batches.
- Lack of an integrated review-and-approval workflow that lets an administrator check and

formally approve a generated timetable before it is published.

- Insufficient optimization of classroom and laboratory utilization, leading to some rooms sitting empty while others are overbooked.
- Limited consideration of individual faculty leave patterns and workload balancing, which can cause dissatisfaction among staff.
- Few systems offer multiple alternative timetable options to choose from, or suggest rearrangements automatically when a perfect solution cannot be found.

These gaps show that the field needs to move from isolated scheduling algorithms toward complete, secure, and scalable web-based systems that combine automated generation with practical administrative features — review, approval, editing, and reporting — while explicitly supporting NEP 2020's multidisciplinary flexibility. This observation directly motivates the system proposed in Section V.

V. PROPOSED METHODOLOGY

Based on the research gaps identified in Section IV, this paper proposes a web-based Smart Timetable Management System designed to automate timetable generation for higher education institutions under NEP 2020, while respecting academic and resource constraints. The system treats timetable generation not as a one-time algorithmic exercise but as an ongoing administrative workflow — data entry, generation, validation, human review, approval, and publishing — with the scheduling algorithm as one replaceable component inside that workflow. Fig. 1 shows the overall system architecture.

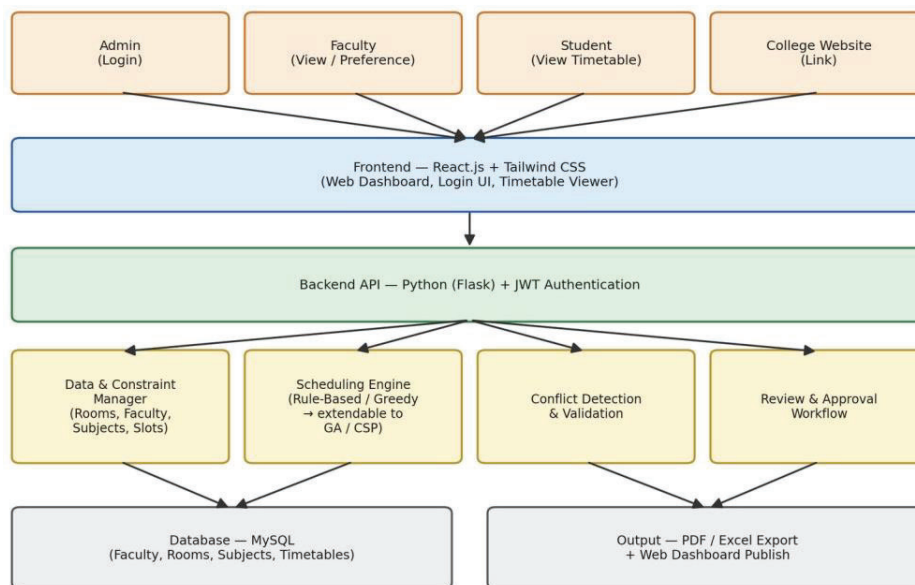


Fig. 1. Proposed system architecture of the Smart Timetable Management System

A. Methodology Steps

1. The administrator logs into the system using secure, password-protected authentication (JWT-based sessions).
2. Academic data is entered into the database: departments, semesters, faculty members, classrooms, laboratories, subjects, batches, and working hours.
3. Key scheduling parameters are defined, including number of classrooms, number of student batches, subjects per semester, maximum classes per day, classes required per subject per week, faculty available per subject, average faculty leave per month, and any fixed special-class slots.
4. The scheduling engine (starting with a Rule-Based/Greedy Algorithm, architected to later support a Genetic Algorithm or CSP module) automatically generates multiple optimized timetable options rather than just one.
5. Each generated option is automatically validated to detect clashes involving faculty, classrooms, laboratories, or student batches, including cross-department elective overlaps relevant to NEP 2020 scheduling.
6. Where no fully conflict-free option is available, the system suggests the smallest possible rearrangement (e.g., swapping two slots) instead of failing outright.

7. The administrator reviews the proposed options, edits if necessary, and formally approves the final timetable through a built-in approval workflow.

8. The approved timetable is published on the web dashboard (linkable to the college website) and can be exported as a PDF or Excel file.

B. System Components

- Frontend: React.js with Tailwind CSS — responsive dashboard for administrators, faculty, and students.
- Backend: Python (Flask) — REST APIs for data management, constraint handling, and timetable generation.
- Database: MySQL — stores departments, faculty, subjects, rooms, batches, and generated timetables.
- Authentication: JSON Web Tokens (JWT) — secure, role-based login (administrator, faculty, student).
- Scheduling Engine: Rule-Based/Greedy Algorithm for the first release, with the codebase structured so a Genetic Algorithm or CSP module can be added later without redesigning the system — matching the adaptive-optimization direction shown effective in [10].

- Output Layer: interactive web dashboard plus exportable PDF and Excel timetable reports, embeddable on the college website.

C. Why This Design Was Chosen

Section III showed that rule-based and greedy methods are fast and simple to deploy, while genetic algorithms and CSP methods give higher-quality results but need more development effort. Rather than forcing every institution to adopt a complex optimization method from day one, the proposed system starts with a practical, fast, easy-to-maintain rule-based engine that already solves the core pain points identified in Section IV — multiple options, dynamic editing, and an approval workflow. As an institution's needs grow, the same platform can plug in a Genetic Algorithm or CSP-based optimizer, following the adaptive-GA direction demonstrated in [10], without requiring users to switch to a completely new system.

VI. EXPECTED OUTCOMES

The proposed Smart Timetable Management System is expected to deliver several measurable benefits once deployed in a higher education institution:

- A significant reduction in the time administrators spend preparing and correcting timetables.
- Fewer scheduling conflicts (faculty double-bookings, room clashes, batch overlaps) due to automatic validation before publishing.
- Better utilization of classrooms and laboratories through week-wide usage tracking.
- A fairer distribution of teaching workload among faculty members, since limits and leave patterns are recorded and considered.
- Reliable support for NEP 2020 elective and multidisciplinary scheduling across departments.
- A clear, auditable review-and-approval trail, so administrators retain full control over the final published timetable.
- Multiple ready-to-compare timetable options instead of a single fixed result.
- A foundation that can be extended with more advanced optimization algorithms (GA/CSP), mobile access, and integration with other campus systems.

VII. CONCLUSION

Timetable scheduling remains one of the most complex administrative tasks faced by higher education institutions,

and NEP 2020's push toward multidisciplinary, flexible learning has only added to that complexity. This survey examined ten published research works spanning rule-based techniques, genetic algorithms, constraint satisfaction, harmony search, graph coloring, and recent adaptive AI-based systems. The comparative analysis in Section III showed that no single method is best for every institution — the right choice depends on the complexity of the constraints involved and the computational resources available.

The study also identified research gaps that persist across the literature: weak support for dynamic rescheduling, limited handling of multidisciplinary elective overlaps, poor multi-department coordination, and the frequent absence of a proper review-and-approval workflow. To address these gaps, this paper proposed a web-based Smart Timetable Management System that combines secure administration, automated conflict detection, multiple optimized timetable options, rearrangement suggestions, and a scalable, extensible scheduling engine.

Future work on this system may incorporate genetic algorithm or constraint-satisfaction modules as optional upgrades, following the adaptive machine-learning-guided approach shown effective in recent scheduling research [10], and add real-time rescheduling so the timetable can adjust automatically to sudden changes such as faculty leave. With these improvements, the proposed platform can grow from a simple automation tool into a genuinely intelligent academic planning assistant suitable for higher education institutions of any size.

REFERENCES

- [1] S. Yang and S. N. Jat, "Genetic algorithms with guided and local search strategies for university course timetabling," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 41, no. 1, pp. 93–106, Jan. 2011.
- [2] M. A. Al-Betar, A. T. Khader, and M. Zaman, "University course timetabling using a hybrid harmony search metaheuristic algorithm," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 42, no. 5, pp. 664–681, 2012.
- [3] N. Pillay, "A survey of school timetabling research," *Annals of Operations Research*, vol. 218, pp. 261–293, 2014.
- [4] R. Ganguli and S. Roy, "A study on course timetable scheduling using graph coloring approach," *Int. J. Computer Applications in Mathematics*, vol. 12, no. 2, pp. 469–485, 2017.
- [5] P. Tanksali, I. Dhond, S. Pednekar, V. Singbal, and S. Sivaraman, "Automated timetable generation," *Int. J. Scientific Research in Computer Science, Engineering and Information Technology*, vol. 6, no. 3, 2020.

- [6] V. Amarnadh, B. Hemanthi, G. Sravanthi, and S. S. Sanketh, "Automatic timetable generator," *Int. J. Research*, vol. 7, no. 5, 2020.
- [7] S. Markal, S. Ghorpade, and D. Chalke, "Timetable generator," *IOSR J. Computer Engineering*, vol. 22, no. 2, pp. 29–33, Mar.–Apr. 2020.
- [8] A. Rezaeipannah, S. S. Matoori, and G. Ahmadi, "A hybrid algorithm for the university course timetabling problem using the improved parallel genetic algorithm and local search," *Applied Intelligence*, vol. 51, pp. 467–492, 2021.
- [9] M. K. Kakkar, J. Singla, N. Garg, G. Gupta, P. Srivastava, and A. Kumar, "Class schedule generation using evolutionary algorithms," 2021.
- [10] S. Han and L. Xiao, "An improved adaptive genetic algorithm," *SHS Web of Conferences*, vol. 140, art. 01044, 2022.