Development of a Web Platform for Managing the Education Process and Booking Infrastructure

Emir Almazov * and Andrei Ermakov *

Posted Date: 29 April 2025

doi: 10.20944/preprints202504.2425.v1

Keywords: web platform; scheduling system; university schedule; academic scheduling; resource optimization; conflict resolution; booking infrastructure; automated scheduling; faculty management

 doi:10.20944/preprints202504.2425.v1

Article

Development of a Web Platform for Managing the Education Process and Booking Infrastructure

Emir Almazov and Andrei Ermakov

Ala-Too International University
* Correspondence: emazovdev@gmail.com

Abstract: This project is the development web platform for managing the education process and booking infrastructure. The motivation to develop this project is to provide improved function and better user experience in scheduling process. To maximize resource usage, minimize disputes, and improve academic experience, professors must schedule teaching activities efficiently. User able to use this project to overcome the problem they face such as generate schedule manually, importing data into software and overcome the constraints and conflict in the schedule. The system optimizes scheduling based on room limitations, instructor availability, and student preferences. In constructing a schedule, conflicts usually arise as to the availability of teachers, lectures and labs. The system automatically identifies errors, helping the administration avoid mistakes and preventing misunderstandings among students and professors. This software should save administrators time, improve academic operations, and improve staff and student academic experiences. Scalability and flexibility allow the system to be used in multiple faculties and incorporate new limits and requirements.

Keywords: web platform; scheduling system; university schedule; academic scheduling; resource optimization; conflict resolution; booking infrastructure; automated scheduling; faculty management

1. Introduction

In universities, scheduling is a vital component of planning education on time. In my academic years, however, I had to deal with actual inefficiencies and problems with handcrafting timetables. Administrators, teachers, and students generally have to deal with conflicts, limited resources, and several complex constraints that have to be handled in order to generate an implementable timetable. With increasing numbers of students, lectures, and teachers, the old-fashioned method of scheduling simply no longer works. Scheduling a timetable in a university is a complex task that involves lecturer availability and venue requirements. Given that there is a set of lecturers, a set of classes, a set of rooms, and hours of lectures in a week, creating a timetable manually is a very time-consuming and frustrating task for teachers. A web-based scheduling system must tackle these issues by producing conflict-free schedules that optimize university resources. The motivation behind carrying out this project is to optimize the usage of resources, eliminate conflicts in schedules, and improve the teaching and learning environment among students and teachers. To automate the process of making appointments, it is necessary for the software to have the facility of importing details from the database of the university, including details on students, lecturers, and courses. Manual entry of the details, taking into consideration the large number of subjects, lecturers, and rooms to be dealt with, is ineffective. Dealing with conflicts generated due to overlapping classes, missing professors, and lack of available rooms is perhaps one of the toughest problems of scheduling. Time-consuming manual scheduling is usually a tedious task and is typically full of mistakes, annoying everyone involved. In addition, an effective scheduling system must have clear display of timetable details, day, time, subject, and assigned lecturer. It must also be logically arranged with these details so that administrators and faculty can easily navigate through it and verify scheduling accuracy. A good sorting in the details table is important because it is used by lecturers and administrators of the software to check which

 doi:10.20944/preprints202504.2425.v1

classes will be lectured on which day. Another challenge in web-based scheduling is handling timetable constraints. Each hard constraint is an element of the timetable that is equally weighted and must be strictly followed. If the software is not able to overcome hard constraints, the timetable created will never work for the university. While soft constraints are less critical, addressing them improves the overall efficiency of the schedule and enhances the academic experience for students and faculty. Solving soft constraints can help transform a good timetable into a great timetable. This project proposes an automated system that is capable of recognizing and resolving scheduling conflicts automatically, significantly lightening the workload of academic administrators. Utilizing advanced filtering features and a user-friendly interface, the website will render scheduling easy while also being scalable and flexible. The proposed system is built on React.js as frontend with drag-and-drop to ensure it's easy to use. The backend is handled using Node.js, which ensures that the data communication and processing are smooth, and MongoDB is utilized in storing and processing all the scheduling information. The tech stack has been built to ensure a smoothened user experience.

2. Problem Statement

Creating an operational schedule that accommodates all courses, instructors, rooms, and student is a highly challenging task. It is about meeting several hard constraints (e.g., instructor schedules and room sizes) and soft constraints (e.g., preferable teaching times and avoiding long breaks between classes). In the majority of universities, scheduling is still manually or semi-manually executed, leading to frequent conflicts in lectures, inefficient use of space, and instructor and student discontentment. These inefficiencies negatively impact the learning experience, contribute to administrative burden, and reduce overall resource utilization. With these shortcomings, clearly what is needed is a system able to automate generation of timetables, output high-quality timetables, and easily develop to address changed academic requirements.

3. Objectives

The goal of this project is to design an effective and adaptable system for automatic timetabling generation taking into account a variety of constraints and preferences. The system being proposed attempts to maximize the use of available resources such as classrooms, teachers' timetables, and available time slots while ensuring the resulting timetables are conflict-free and feasible to implement in real life. In addition, the system should be scalable to take care of adjustments such as the addition of new courses or an increase in student enrollment. Another major goal is to provide a user-friendly interface, thus enabling the faculty administrators to interface with the system simply without requiring advanced technical knowledge. In achieving these goals, the platform will assist universities in saving time, cutting down on scheduling conflicts, and providing better and more efficient accommodations for staff and students.

4. Hypothesis

First we make the assumption that a system that strictly adheres to all "hard" constraints (instructor availability, room capacity) and then tries to minimize "soft" preferences (long breaks, abrupt changes in class sequence) will be able to generate schedules free of conflict in the vast majority of instances. This will ensure no classroom will be double-booked and no instructor will be in two classrooms at the same time by mistake.

Second, we consider the fact that the web application run on a light stack (React.js + Node.js + MongoDB) will be capable of building a complete schedule of hundreds of lessons within a short time frame—typically 10–30 seconds. This will make the system usable on a daily basis: the administrator will be able to get an actual schedule "on the fly," without spending hours on recalculation.

The third assumption pertains to the classroom booking module. We suppose that if properly integrated with the core schedule, it will manage the vast majority of requests for extracurricular events (seminars, conferences) flawlessly, booking already reserved time slots automatically and offering

 doi:10.20944/preprints202504.2425.v1

vacant ones. The success rate in at least 95% of cases without the administrator's interference is the ultimate goal.

Lastly, we expect the utilization of such a system to greatly enhance the satisfaction of the faculty members with the scheduling process itself. According to feedback, the time spent on routine administrative work is predicted to reduce at least five-fold, and the quality of the resulting schedule (readability, organization, adherence to personal preferences) will be ranked higher than with current manual or semi-manual techniques.

These assumptions collectively form the basis for ascertaining if the solution, as envisaged, has the ability to equal the quality and ease of use of heavier industrial-grade tools while remaining easy to deploy and manage.

5. Research Methodology

There will be a systematic approach that will be adopted to attain the research goals and develop an operational automatic timetable generation system for the institution. The section will describe the method that will be adopted in the various phases of the research, which include data collection, system design, implementation, and evaluation.

5.1. Problem Analysis and Requirements Gathering

The study will start with the thorough examination of the existing timetable generation process and the particular issues of the university. There are interviews with the deans, faculties, and students to be undertaken for the sake of determining the fundamental requirements, constraints, and preferences that need to be considered when creating the automatic timetable generation system.

5.2. System Design

Based on the elicited requirements and review of current solutions, the system architecture, algorithms, and data models for automatic timetable generation will be formulated. Appropriate algorithms will be chosen that have the potential to resolve optimally resource allocation optimization issues, conflict minimization, and fulfillment of the numerous constraints that were revealed during the problem analysis stage.

5.3. Data Collection and Integration

In order to successfully generate a timetable, the system will incorporate data on courses, instructor availability, student and instructor preferences, and other institutional data. Data collection is possible through questionnaires, interviews, or data mining from existing systems. Data collected will be processed and incorporated into the system to enable effective input in generating timetables.

5.4. Performance Analysis and Evaluation

Once the system is implemented and the data is integrated, its performance shall be evaluated. Schedules will be generated for various scenarios, and their results will be compared with pre-decided criteria and goals. The evaluation will cover aspects like schedule quality, resource utilization, and end-user satisfaction. The performance evaluation of the system will also be conducted, along with its responsiveness, scalability, and efficiency.

5.5. Iterative Improvement

Based on the outcomes of the assessment and participants' feedback, the system will be improved iteratively. This may include algorithmic optimization, interface improvement, and elimination of any identified limitations or weaknesses. The iterative improvement process will be continued until a satisfactory level of performance and user acceptability is achieved.

 doi:10.20944/preprints202504.2425.v1

6. Related Work

In recent years, web applications for appointment scheduling and booking have been integrated into corporate organizations and educational institutions. Among the solutions already available, there are some popular services providing planning and time management functionality.

6.1. Picktime

Picktime is an appointment scheduling and appointment management web service. The software provides flexible booking management tools, including calendar integration and the booking of various kinds of meetings. Picktime allows users to effectively organize time blocks, as well as provide an easy-to-use interface for participants and administrators. Picktime stands out from other websites because it prioritizes flexibility, allowing users to book appointments in various fields, including education, medicine, and business [9]

6.2. Sceduly

Sceduly is an online scheduling software that provides functionality for creating and managing events. Sceduly specializes in scheduling meetings and planning time for various uses. Sceduly offers a user-friendly interface with the feature to customize time slots and manage user availability. Compared to Picktime, Sceduly can be a more specialized software, specialized for use in academic and corporate settings. [10]

Existing solutions such as Picktime and Sceduly offer useful features for scheduling meetings and time management, but usually they lack the integration with educational platforms, such as classroom booking systems or student schedule management. In contrast to the described services, our service is designed specifically for the needs of education institutions, so not only can one manage the schedule properly, but also automatically generate it, preventing potential scheduling collisions and overlaps.

7. System Architecture

A web-based platform is proposed for the development of an Automatic Timetable Generation system for a faculty. The frontend of the system will be constructed using the React framework, while the backend will be developed using NodeJS with Express. The aforementioned architectural approach guarantees a design that is both scalable and modular, effectively segregating the presentation layer from the business logic and data processing layers. The proposed system is designed to adopt a client-server architecture, wherein the React-based frontend will establish communication with the Express-based backend via APIs.

8. Challenges Faced

First, we assume that a system that strictly satisfies all “hard” constraints (such as classroom capacity and teacher availability) and then seeks to minimize “soft” preferences (long breaks, unexpected changes in class order) will be able to generate schedules without conflicts in the vast majority of cases. This means that no classroom will be booked twice, and no teacher will be in two places at once by accident.

Second, we believe that a web tool implemented on a lightweight stack (React.js + Node.js + MongoDB) will be powerful enough to generate a full schedule of hundreds of classes in a short time—usually within 10–30 seconds. This makes the system suitable for everyday use: the administrator will be able to receive an updated schedule “on the fly,” without spending hours on recalculation.

The third assumption concerns the classroom booking module. We expect that if properly integrated with the main schedule, it will be able to process the vast majority of requests for extracurricular activities (seminars, conferences) without any hitches, automatically blocking already scheduled slots and opening access to free ones. Ideally, the success rate should exceed 95% of cases without manual intervention by the administrator.

 doi:10.20944/preprints202504.2425.v1

Finally, we hope that the implementation of such a system will significantly increase the satisfaction of faculty members with the scheduling process itself. Based on feedback, it is assumed that the time spent on routine administrative operations will decrease by at least five times, and the quality of the final schedule (understandability, structure, compliance with individual preferences) will be rated higher compared to existing manual or semi-manual methods.

Together, these assumptions form the basis for testing whether the proposed solution can compete in quality and convenience with heavier industrial tools, while remaining easy to deploy and manage.

9. Methods

The process of development followed an iterative engineering process. Initial requirements were derived from real-world academic use cases, with the priority on essential requirements such as scheduling, conflict resolution, and resource booking. After deciding on the system structure, development was conducted in phases to facilitate continuous feedback and revision. The timetabling begins with satisfying hard constraints. The classes are scheduled in available rooms and time slots according to teacher availability and room capacity. After satisfying all obligatory conditions, the system tries to satisfy soft constraints, e.g., minimizing idle times of students and fulfilling instructor preferences. If at any point a conflict is detected during the generation process, the system dynamically adjusts existing assignments to maintain feasibility, thus ensuring that no infeasible timetable is produced.

10. Technical Details of the Experiment

10.1. Complete Schedule Generation From Scratch

To examine the overall performance of the system, we generated a complete data set: 120 courses, 50 instructors, and 35 rooms. All the classes were to be assigned automatically to time slots and rooms by the system, without introducing any preliminary constraints on free windows or reservations. In this mode, the speed of the algorithm and the absence of hard conflicts were being evaluated. The average generation time was 12 seconds, and there were zero hard violations (double booking teachers or classrooms).

10.2. Incremental Addition of Classes

Often in the middle of the semester, new classes or practical classes are added to the schedule. For this test, we first generated a schedule for 80 courses, then gradually added another 10 courses and checked how long the recalculation takes and whether the accuracy is maintained. Even when adding the last blocks in 10 pairs, the recalculation took an average of 4-5 seconds, and the system successfully rebuilt the affected slots without creating new conflicts.

10.3. Introducing Artificial Conflicts and Resolving Them

To test the automatic detection and correction module of intersections, we manually assigned two classes to the same classroom and teacher for the same slot. The system immediately recognized the conflicts, recalculated the adjacent slots, and moved one of the pairs to the nearest free time slot. In 100% of cases, the algorithm found the correct solution, the requirements for classroom capacity and teacher availability were maintained.

10.4. Testing the Infrastructure Booking Module

In this scenario, the standard schedule was already generated, after which the classrooms were blocked for events (seminars, conferences). We checked how the system reacts to an attempt to book classrooms occupied according to the academic schedule. In 95% of cases, the booking was blocked correctly. In the remaining 5% — when the administrator manually removed the class before booking — the system immediately updated the availability and allowed the event to take place.

 doi:10.20944/preprints202504.2425.v1

10.5. Load Testing a Large Semester

To test scalability, we increased the data volume to 200 courses, 80 teachers, and 60 classrooms, adding restrictions on special groups of students (e.g., evening classes). In this mode, the overall generation time increased to 25–30 seconds, while the system withstood the load and still did not allow hard conflicts. The test showed that the algorithm is highly resilient to the growth of the problem size.

11. Analyzing the Performance of an Automatic Timetable Generation System

Analyzing the performance and efficiency of an automatic timetable generation system of an educational organization is one of the key points for analyzing its performance. This subsection describes various measures of performance analysis that allow us to analyze various aspects of the system. The performance analysis provides useful data for system optimization and improving its overall performance.

11.1. Schedule Quality Metrics

Measuring quality of a schedule involves how well it meets school requirements and constraints. One of the most critical indicators in measurement is conflict resolution, and it is measured based on the number of conflicts or inconsistencies in the schedule, e.g., when lectures or exams conflict in terms of timing. The less such conflicts exist, the higher the quality of the schedule. In order to calculate this parameter, you can use the Timefold Solver feature, which allows you to set the quality of the solution within a given time period. Another important characteristic is resource utilization. This is a metric that quantifies the efficiency of utilizing available resources, e.g., classrooms and teachers. Measuring resource utilization helps prevent underutilization and overutilization of resources. One of the important parameters is the allocation of the teaching burden between teachers. This measure attempts to allocate the teaching burden proportionally among teachers, taking into account such factors as the number of teaching hours, class gaps, and the duration of consecutive lessons. The algorithm should offer a balanced allocation of the burden, which translates to the effective utilization of teachers' time.

11.2. Computational Performance Metrics

The automated schedule generation system is also tested in terms of efficiency and scalability with a set of computational parameters. One of the significant parameters is the execution time, which refers to the time taken by the system in order to generate a schedule. The lower the execution time, the faster the schedule generation process, which is the desired outcome. System scalability examines its ability to maintain high performance as the size and complexity of the task increase. This measure indicates the effectiveness with which the system can process large volumes of data and generate schedules within reasonable time constraints. Also important is the memory usage measurement, which tells us how much memory the system requires to accomplish the schedule generation task. This measurement helps identify areas of wasteful memory use and system performance bottlenecks.

11.3. Surveys of User Satisfaction

Surveys of user satisfaction are a good method of gathering feedback about the usability of an automatic schedule generation system. Surveys may be employed for faculty, students, and administrators to gather feedback on the usability, functionality, and overall satisfaction of the system.

11.4. Comparison with Manual Schedule Generation

Comparative evaluation of an automated system and a manual schedule generation system helps in comparing the efficacy of the automated process. This may involve various parameters such as increased efficiency, lesser interpersonal conflict, and optimal resource utilization. Comparison among the two processes helps in bringing into focus the benefits and improvements that an automated solution provides.

 doi:10.20944/preprints202504.2425.v1

References

1. Diallo, F.P.; Tudose, C. Optimizing the Scheduling of Teaching Activities in a Faculty. Appl. Sci. 2024, 14, 9554.
2. Schaerf A., A Survey of Automated Timetabling. **1999**, Volume 13.
3. L. Zhang, D. Roy, P. Mundhenk and S. Chakraborty, Schedule Management Framework for Cloud-Based Future Automotive Software Systems, 2016 IEEE 22nd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), Daegu, Korea (South), 2016, pp. 12-21.
4. Bernd A. Knauer, Solution of a timetable problem, Computers & Operations Research, Volume 1, Issues 3–4, 1974.
5. Wankhede, S. Automatic College Timetable Generation, International Journal of Scientific Research in Computer Science, Engineering and Information Technology, **2019**.
6. Zhu, K.; Li, L.D.; Li, M. School Timetabling Optimisation Using Artificial Bee Colony Algorithm Based on a Virtual Searching Space Method. Mathematics 2022, 10, 73.
7. Feng, X.; Lee, Y.; Moon, I. An Integer Program and a Hybrid Genetic Algorithm for the University Timetabling Problem, Optimization Methods and Software, 2016, 32, 3, 625–649.
8. Ohashi, T.; Aghbari, Z.; Makinouchi, A. Hill-Climbing Algorithm for Efficient Color-Based Image Segmentation, IASTED International Conference on Signal Processing, Pattern Recognition, and Applications, 2003, 17-22.
9. Online Free Appointment Scheduling Software | Booking Software | Calendar Management System - Picktime (n.d.). https://www.picktime.com/
10. Web scheduling app - Sceduly (n.d.). https://sceduly.com/