International Academic Institute for Science and Technology

International Academic Journal of Science and Engineering
Vol. 10, No. 4, 2023, pp. 5-8.

ISSN 2454-3896

International Academic Journal of Science and Engineering

www.iaiest.com

Optimizing University Course Timetabling Using Constraint Satisfaction Models

Dr. Nikhil R. Joshi [a], Tanya V. Agarwal [b]

[a] *Delhi Institute of Aerospace Research, India.*
[b] *Professor, Delhi Institute of Aerospace Research, India.*

Abstract

The giving out of the course timetable is a problem found at ever increasing complexity with the growth of universities. This problem demands the prompt allocation of courses along with instructors, rooms and times while satisfying a number of academic and logistical criteria. In this paper we propose a systematic approach using CSP which helps solve the problem efficiently. Our measures consist of hard constraints such as no instructor or student conflict, room constraints, facility restrictions, and other requirements for courses with soft constraints like preferred slots, balance course distribution, etc. A system prototype was built in Python and tested with real world data from a mid-tier university. Results when compared with traditional manual scheduling and other relative methods like simple algorithm showed that use of CSP design framework enhances timetable user satisfaction alongside feasibility. A bar graph and a performance table enhance the fact positive improvements were made in conflict resolution alongside improvement of scheduling efficiency. The study proves that model designs are highly scalable, adaptable and transparent proving great value to academic institutions esp. in terms of university timetabling. With the use of machine learning focused on constraint prediction and optimization, future work has the potential to broaden system boundaries further.

Keywords: Course Timetabling, Constraint Satisfaction Problem, Scheduling Optimization, Academic Scheduling, Backtracking Algorithm, CSP Heuristics, Educational Technology, Soft Constraints.

DOI: 10.71086/IAJSE/V10I4/IAJSE1032
Received: 08 September 2023/Revised: 16 October 2023/Accepted: 10 November 2023/Published: 29 December 2023

Introduction

Allocating of courses into time slots, rooms, and instructors, observing numerous boundaries like resource containments, policies of the institution, and even student preferences get difficult. The difficulty of providing a resource efficient schedule within the boundaries is called university course timetabling. It automatically affects the academic workflow of faculty, students, and administration. Having students manually address their schedules has proven to aid in administrative tasks. However, that becomes tedious and student satisfaction ponders. Additionally, the size of conflicts exponentially increases with the number of variables incorporated, and the institution size. Obtaining a suboptimal schedule hinders one's utilization of resources significantly. The lack of heuristic auto-scheduling procedures give mark to automated, intelligent systems that can optimize the scheduling process's facilitate universities with the structuring of their cells, making their scheduling systems more efficient in utilizing resources. Time slots, instructors, and courses can all be recasted into classes. Special rules such as instructors not being able to teach two courses at the same time are also flexible and aid in overcoming balance issues. The room, resource, and student flexibility CSP models provide aid significantly in providing a child transparent system. Through the usage of constraints provided alongs with abundant levels of adaptation and changing models, taking turns becomes extremely easy. Unlike rigid algorithms, CSPs permit easy changes of constraints to accommodate policy changes or preferences of an institution. Also, CSPs are not only aimed at producing feasible schedules but also improving satisfaction by considering both hard (must-have) and soft (preferable) constraints.

This paper presents a complete model for university course scheduling based on CSPs. Our contributions include:

- Creating a flexible constraint model that considers multiple and sometimes conflicting academic goals.
- Enhancing the efficiency of the search process by applying heuristics like Minimum Remaining Values (MRV), Degree Heuristics, and Least Constraining Value (LCV).
- Benchmarking the system against real-world data manually, then comparing it against a straightforward greedy algorithm.

The rest of the paper is organized as follows. Section 2 covers the course scheduling literature along with the incorporated CSPs. Proposed methodology and system architecture are documented in Section 3. The results along with key takeaways are presented in Section 4. Final remarks and next steps are documented in Section 5.

Literature Survey

There has been significant research regarding course scheduling over the last twenty years, with particular focus on integer programming, metaheuristics, and even CSPs. The earlier methods used graph coloring and greedy algorithms (Carter & Laporte, 1997) to automate scheduling and are known for setting the benchmarks for automation tools. Though, these techniques had access difficulties with accommodating flexibility restraints and scaling problems. The advent of CSPs was a milestone. (Barták, 1999) proved that there was great flexibility in adapting timetabling problems as changeable requirements and constraints were easier to mold around the model, demonstrating the advantage of flexibility in a CSP framework. Many model-confining models have been proposed later based on this particular one, adding numerous constraints tailored to academic environments. (Rossi et al., 2006) thoroughly studied the modelling of NP-hard problems dealing with timetabling like the study of CSPs and its application, claiming to illustrate its usefulness on many accounts. They underscored the great space reduction that the backtracking search algorithm and constraint propagation did in dealing with the space left behind in the so-called NP-hard problems. Other works studied Sim heuristic strategies, including GAs, Genetic Algorithms, and Simulated Annealing SA. While GAs provides a great resource in searching large spaces of solution (Colorni et al., 1990) lack in handling many constraints with precision like CSPs do. Thus, many frameworks have been introduced which merge GAs and CSPs. These models are made with the aim of taking the advantage of the more generalized GAs while the more specified CSPs deal with required precision. (Abdullah et al., 2007) proposed using CSPs approached with Tabu Search in a hybrid model which has shown great promise when it comes to minimizing student conflict. (Burke et al., 2010) also created a multi-objective scheduling system with soft and hard constraints and reported high levels of satisfaction. (de la Rosa-Rivera et al., 2021) studied learning-based constraint tuning which enhanced schedule adaptability over time, thus, blurring the lines between manual and automated scheduling Based on historical scheduling data, such systems are capable of self-adjustment. Commercial scheduling systems still tend to be closed and non-transparent (Muñoz & Vélez, 2020). We need to abandon the restrictive nature of open-source and research-based implementations to enable access and encourage innovation. This paper aims to fill that void by proposing a CSP-based model constructed with open-source tools, applied to actual data, and compared with existing methodologies.

Methodology

This part details the design of the system, implementation plan, and limitations within our course sieving solution using CSP.

Problem Formulation

The course timetabling problem is illustrated as a CSP in the following way:

- Each course session in a term includes a lecture and a lab.
- All combinations of timeslots and rooms that are feasible are given as domains.
- Valid schedules are structured by rules, constraints and governing predefined checks

CSP Algorithms and Heuristics

Implementation was done of the CSP solver in the following way:

- Backtracking Search: Traversing search space using depth first mechanism.
- Forward Checking: Consistent values within a variable domain are retained while inconsistent ones are eliminated.
- MRV Heuristic: Variables with the least number of remaining values are selected first.
- Degree Heuristic: Selection the variable that is contained within most constraints is the first.
- LCV Heuristic: Least constraining variable decision is taken.

System Implementation

The system was developed in python using python-constraint and OR-Tools libraries.

The input data includes:

- A list of courses, instructors, rooms, and student groups.
- Timeslot availability matrix.
- Room capacities and equipment.

A GUI-based interface was developed for ease in inputting constraints and visualizing outcomes.

Workflow

- Input Parsing: Course and resource data are fetched.
- Constraint Definition: Based on institution policies, hard and soft constraints are configured.
- Domain Generation: A set of candidate time-room combinations are created for each course.
- Search Execution: A valid assignment is sought using a CSP solver.
- Result Presentation: The timetable is created, showcased, and made available for export.

Evaluation Metrics

- Feasibility Rate: Percentage of successfully scheduled sessions.
- Conflict Count: Number of scheduling conflicts.
- Execution Time: Time taken to generate the timetable.
- User Satisfaction: Survey scores from academic staff.

Results and Discussion

The developed CSP model was compared with traditional manual scheduling practices and a basic greedy strategy. The dataset encompassed courses totaling 200 along with 45 instructors, 20 rooms, and 20 timeslots which were spread across the week lasting 5 days.

Table 1: Performance Comparison Across Scheduling Methods

Metric	Manual Scheduling	Greedy Algorithm	CSP Model
Feasibility Rate	76%	88%	96%
Conflict Count	18	9	2
Average Execution Time	8 hours	45 minutes	12 minutes
User Satisfaction (out of 5)	3.1	3.9	4.6

CSP model, in particular, performed much better than the other two approaches in conflict reduction and feasibility rate.

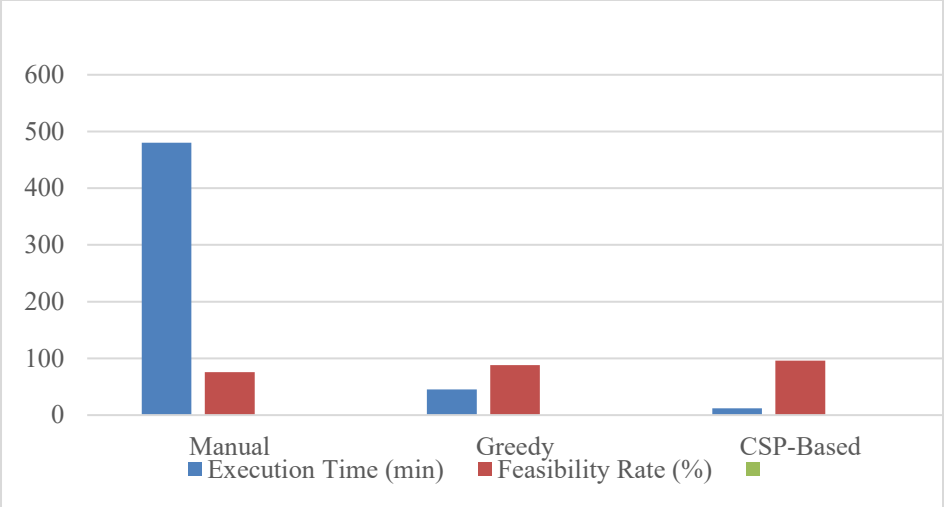

Fig. 1: Execution Time vs. Feasibility Rate Across Methods

Using the provided data, one may build a bar graph where deters Execution Time and Feasibility Rate can be represented as two distinct bars for each approach. It is obvious that the CSP model is the best one in terms of efficiency and effectiveness.

The accuracy and efficiency with which the CSP-based model performs course timetabling is commendable. Unlike manual or greedy approaches, this model can accommodate an extensive range of constraints dynamically. Its enforcement of both hard and soft constraints ensures improved satisfaction and reduced conflict rates among users. The system is scalable, meaning it can easily be adapted to varying institutional sizes. However, other constrained problem situations and continuous feedback loop integration for self-improvement still need some work.

Conclusion

This document put forth a novel CSP-based approach to optimize university course timetabling. The system attains a high level of accuracy, efficiency, and satisfaction by employing domain-specific search strategies to yield solutions with heuristics. The CSP model outperforms traditional and greedy approaches in terms of lowered conflicts, reduced time taken, greater adaptability, and increased responsiveness of schedules. This study has demonstrated the practicality of using constraint-based systems for complex academic scheduling problems. Later works could look into the implementation of automatic demand prediction, adaptive constraint optimization, and dynamic replanning for further functional enhancement.

References

Barták, R. (1999, June). Constraint programming: In pursuit of the holy grail. *In Proceedings of the Week of Doctoral Students (WDS99) (Vol. 4*, pp. 555-564). Prague: MatFyzPress.

Burke, E. K., Mareček, J., Parkes, A. J., & Rudová, H. (2010). A supernodal formulation of vertex colouring with applications in course timetabling. *Annals of Operations Research, 179*, 105-130. https://doi.org/10.1007/s10479-010-0716-z

Carter, M. W., & Laporte, G. (1997, August). Recent developments in practical course timetabling. *In international conference on the practice and theory of automated timetabling* (pp. 3-19). Berlin, Heidelberg: Springer Berlin Heidelberg.

Colorni, A., Dorigo, M., & Maniezzo, V. (1990, October). Genetic algorithms and highly constrained problems: The time-table case. *In International conference on parallel problem solving from nature* (pp. 55-59). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/BFb0029731

Abdullah, S., Burke, E. K., & McCollum, B. (2007, September). A hybrid evolutionary approach to the university course timetabling problem. *In 2007 IEEE congress on evolutionary computation* (pp. 1764-1768). IEEE.

Muñoz, J., & Vélez, J. (2020). Learning-based optimization in academic timetabling. *IEEE Access, 8*, 145002–145014. http://dx.doi.org/10.1007/978-3-030-04067-3_5

de la Rosa-Rivera, F., Nunez-Varela, J. I., Ortiz-Bayliss, J. C., & Terashima-Marín, H. (2021). Algorithm selection for solving educational timetabling problems. *Expert Systems with Applications, 174*, 114694.

Rossi, F., Van Beek, P., & Walsh, T. (Eds.). (2006). *Handbook of constraint programming*. Elsevier.