ENHANCED KNOWLEDGE IN SCIENCES AND TECHNOLOGY
e-ISSN: 2773-6385

EKST

Vol. 6 No. 1 (2026) 324-332
https://publisher.uthm.edu.my/periodicals/index.php/ekst

Graph Colouring Technique for Timetable Scheduling in UTHM

Arina Hanim Yahya[1], Wan Nor Munirah Ariffin[1*]

[1] *Department of Mathematics and Statistics, Faculty of Applied Sciences and Technology, UTHM Kampus Cawangan Pagoh, Hab Pendidikan Tinggi Pagoh, KM 1, Jalan Panchor, 84600 Pagoh, Muar, Johor, MALAYSIA.*

*Corresponding Author: normunirah@uthm.edu.my
DOI: https://doi.org/10.30880/ekst.2026.06.01.039

Article Info

Received: 2 January 2026
Accepted: 20 January 2026
Available online: 9 July 2026

Keywords

Graph Colouring, Timetable Scheduling, GUI, Conflict-Free

Abstract

Timetable scheduling is a complex combinatorial optimization problem that requires assigning courses, instructors, classrooms, and time slots while satisfying hard and soft constraints. Traditional methods often struggle in solving scheduling conflicts making it inefficient for students and instructors. This study proposes an automated timetable scheduling for university using the graph colouring technique. The main objective of this study is to create a conflict-free academic timetable for university scheduling using custom graphical user interface (GUI). This study implements graph colouring technique where each course is modelled as a vertex and edges represent conflicts. The Welsh-Powell algorithm ensures a valid colouring that no two conflicting vertices share the same colour. The problems were converted into a conflict graph to represent all conflicts, and the colouring process used to generate conflict-free timetable. A GUI was created to load dataset, interpret data, and evaluate the scheduling outcomes. The performance of the generated timetable is evaluated through the penalty score that measure soft constraint violations, slot utilization efficiency and engagement days. Experimental results demonstrates that the proposed approach successfully eliminates hard constraints, reduces soft constraints and produces a structured academic timetable. In this study, the graph colouring technique has effectively proven in producing conflict-free academic timetables, offering a practical tool for departmental timetabling. The current practice of this study is focusing on medium-sized dataset; therefore, the GUI performance must be tested before applying for larger dataset.

1. Introduction

Timetable scheduling is a complex process in high institutions, involving the assignment of courses, instructors, classrooms, and time slots, while ensuring the satisfaction of various constraints [1]. An effective academic timetable ensures the smooth operation of educational activities, optimizes resource utilization, and minimizes possible conflicts that may affect students and instructors [2]. Universities around the world, the complexity of timetable scheduling increases significantly with the growth in the number of programs, courses, and shared resources [3]. As a result, timetable construction is widely recognized as a combinatorial optimization problem that become increasingly difficult to manage using manual or semi-manual adjustment approaches.

In academic scheduling, timetable construction is governed by both hard and soft constraints [4]. Hard constraints must be strictly satisfied to avoid clashes, such as a student or instructor being assigned to multiple classes in the same time slot [5]. In contrast, soft constraints relate to preferences and quality considerations,

This is an open access article under the CC BY-NC-SA 4.0 license.

including time distribution, engagement days, and workload balance [6]. Semi-automated scheduling often struggles to manage these constraints effectively, particularly for large datasets, leading to inefficient schedules and increasing administrative workload [7]. Consequently, various computational techniques, including metaheuristic approaches such as genetic algorithm, greedy algorithm, simulated annealing, ant colony optimization, and graph colouring algorithm have been explored [8][9][10][11]. Among these methods, graph colouring has gained considerable attention due to its simplicity, interpretability, and effectiveness in resolving conflict-based scheduling problems [12].

In this context of academic timetabling, graph colouring has been widely applied to model courses as vertices and conflicts, such as shared instructors or overlapping students as edges [13][14]. A proper colouring of the conflict graph guarantees that no conflicting courses are scheduled simultaneously. However, real-world scheduling problems often involve dense conflict structures and multiple constraints, making the efficiency of the colouring strategy and its practical implementation critical [15]. Therefore, the usability of the scheduling system plays a crucial role in practical adoption. The integration of the algorithm into a GUI enhances user interaction, reduces human error, and improves transparency in the scheduling process by enabling users to visualize conflict graphs and generate timetables efficiently.

Motivated by these considerations, this study mainly focuses on applying the Welsh-Powell algorithm to generate an automated academic timetable and integrates the algorithm into a GUI system. The objectives are to create a conflict-free academic timetable using a GUI and to analyse the efficiency of the generated timetable using a penalty score that reflects soft constraint satisfaction and scheduling performance. The results demonstrate that the proposed graph colouring technique, supported by an interactive GUI, provides a practical and efficient solution for academic timetable scheduling in higher education institutions.

2. Methodology

2.1 Data Collection and Problem Formulation

Fig. 1 illustrates the overall workflow of the proposed methodology, from data preparation and conflict graph construction to timetable development and performance evaluation.

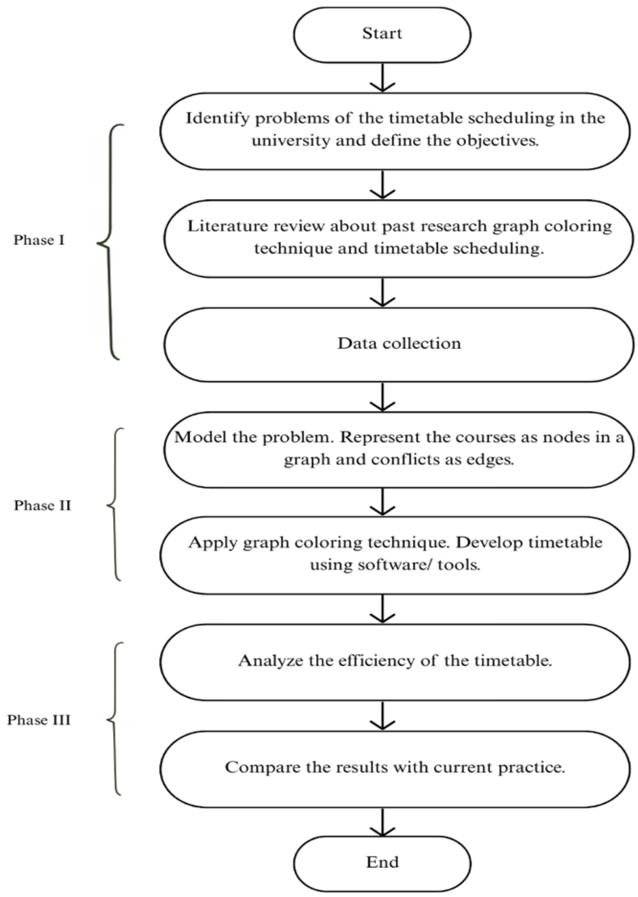


Fig. 1 Workflow of the Methodology

The dataset used in this study consists of course scheduling information collected from a higher education institution, which involves course identifiers, academic programs, year of study, instructor assignments, and course categories. The information was organized and structured into a mathematical graph to determine potential conflicts arising between student groups and instructors. The resulting conflict graph becomes the fundamental basis for applying the graph colouring technique to determine valid time slot allocations while effectively minimizing scheduling conflicts and reducing overlaps in student academic timetables.

2.2 Conflict Graph Modelling

To represent the academic timetabling problem in a structured and systematic manner, a conflict graph is constructed based on the collected data to represent scheduling constraints. Consider the proposed algorithm as follows,

$$c(v_i) = \min\{k \in \mathbb{N} \mid c(v_j) \neq k, \forall v_j \in Adj(v_i)\} \tag{1}$$

where $c(v_i)$ is the colour or time slot assigned to vertex, $Adj(v_i)$ is the set of vertices, (v_i) represents a course in the conflict graph, and $\mathbb{N}$ is the set of natural numbers used. This conflict graph serves as the foundation for applying the graph colouring technique.

2.3 Welsh-Powell Algorithm

The Welsh-Powell algorithm is a greedy graph colouring that assigns colours to vertices in descending order of vertex degree [1]. The algorithm was proposed due to its efficiency and suitability for medium-sized dataset scheduling problems. In this study, the colours represent time slots, and a valid colouring ensures that no two conflicting courses share the same time slots. After constructing the conflict graph, the algorithm will be implemented.

Thus, the calculation procedure of the Welsh-Powell algorithm is as follows [12].

Step 1 Identify the vertex with highest degree in the conflict graph.

Step 2 Sorts the vertices by descending order and assigns first colour to the vertex with the highest degree.

Step 3 The algorithm assigns the same colour to vertex that does not share an edge with any vertex already assigned that colour.

Step 4 After all feasible assignments for the current colour are completed, a new colour is assigned to the next uncoloured vertex. This step is repeated until all vertices are assigned a colour.

Step 5 Once all vertices have been coloured, courses assigned the same colour are allocated to the same time slot.

Step 6 The computation of timetable quality is performed using a penalty score that measures soft constraint violations.

2.4 Penalty Score Evaluation

To evaluate the generated timetable quality, a total penalty score is computed. It aims to calculate the penalty and how well the generated timetable meets the soft constraints, minimizes time slots, and produces a balanced academic timetable. The evaluation equation, as follows [1],

$$TP = \frac{P1+P2+P3}{|TL| \times X(G)} \tag{2}$$

where P1 is the number of soft constraints, P2 is the number of slot utilisation, P3 is the number of engagement days, TL is the number of instructors, and X(G) is the chromatic number of the graph needed for the total generated time slots. In this study, soft constraints include uneven distribution of classes across days, inefficient use of available time slots, and excessive instructor engagement days. All penalty components are normalized to ensure comparability across timetables. A lower penalty score indicates a more balanced and efficient timetable.

os

2.5 Graphical User Interface Implementation

A graphical user interface (GUI) was developed using Python language to serve as an integrated platform that supports multiple core functionalities including data loading, conflict graph visualization, timetable generation, and systematic interpretation of scheduling results. The GUI is designed to enhance usability by allowing users to filter timetables by academic program and student group, providing a practical scheduling tool for academic planners. Furthermore, the generated timetables are displayed directly through the GUI interface to reflect actual system output, ensuring transparency, ease of analysis, and consistency between the computational results and visual presentation.

3. Results and Discussion

The data collected was transformed into CSV format, which will be served as the primary input for the GUI model. This format enables the GUI to seamlessly load, interpret, and evaluate the dataset. Table 1 summarizes the dataset structure used in this study, including program, academic year, instructor identifier and course type. BWA, BWQ and BWS represent three different academic programs offered by the faculty. This can be seen in the table below.

Table 1 Course and Instructor Dataset

Course Code	Program	Year	Instructor ID	Category
10503	BWA	1	A(1312)	Core
10603	BWA	1	A(1318)	Core
10803	BWA	1	A(1255)	Core
21703	BWA	1	A(1112)	Core
11102	BWA	1	A(1261)	Core
20803	BWA	2	A(2621)	Core
21703	BWA	2	A(1112)	Core
21003	BWA	2	Q(1815)	Core
32803	BWA	2	A(1321)	Elective
22803	BWA	2	A(1925)	Elective
21102	BWA	2	A(8113)	Core
10102	BWA	2	A(2621)	Core
30803	BWA	3	A(6126)	Core
20403	BWA	3	Q(1182)	Core
10202	BWA	3	A(8113)	Core
32702	BWA	3	A(1991)	Core
31003	BWA	3	A(1921)	Elective
31503	BWA	3	A(1925)	Elective
32903	BWA	3	A(1151)	Elective
10503	BWQ	1	Q(141518)	Core
11203	BWQ	1	A(1681)	Core
11303	BWQ	1	Q(19121)	Core
11403	BWQ	1	Q(11113)	Core
21803	BWQ	2	Q(81941)	Core
23403	BWQ	2	Q(13118)	Core
23503	BWQ	2	Q(19121)	Core
23903	BWQ	2	Q(1815)	Core
23603	BWQ	2	Q(1182)	Core
23803	BWQ	2	Q(141518)	Core
10102	BWQ	2	Q(1921)	Core

10102	BWQ	3	Q(11113)	Core
44703	BWQ	3	Q(1921)	Core
32503	BWQ	3	Q(13118)	Core
32403	BWQ	3	Q(19121)	Core
44303	BWQ	3	Q(1321)	Elective
31603	BWQ	3	Q(1182)	Elective
43202	BWQ	3	Q(141518)	Elective
10303	BWS	1	S(1919)	Core
10503	BWS	1	S(23114)	Core
10603	BWS	1	S(1215)	Core
10703	BWS	1	A(1815)	Core
21303	BWS	2	S(1919)	Core
21403	BWS	2	S(1215)	Core
21503	BWS	2	S(10113)	Core
21603	BWS	2	Q(12613)	Core
21703	BWS	2	S(23114)	Core
10202	BWS	2	A(1991)	Core

The dataset was then transformed into a graph structure to facilitate the application of graph coloring technique for timetable scheduling. In the graph, each vertex represents a course code, and each edge will be present between two vertices, that border on each other. Two primary conflicts were identified, which are instructor conflicts and student group conflicts. From Fig. 2, the Welsh-Powell algorithm will be implemented to systematically assign different colours to adjacent vertices and produce the conflict-free academic timetables that satisfy the defined scheduling constraints.

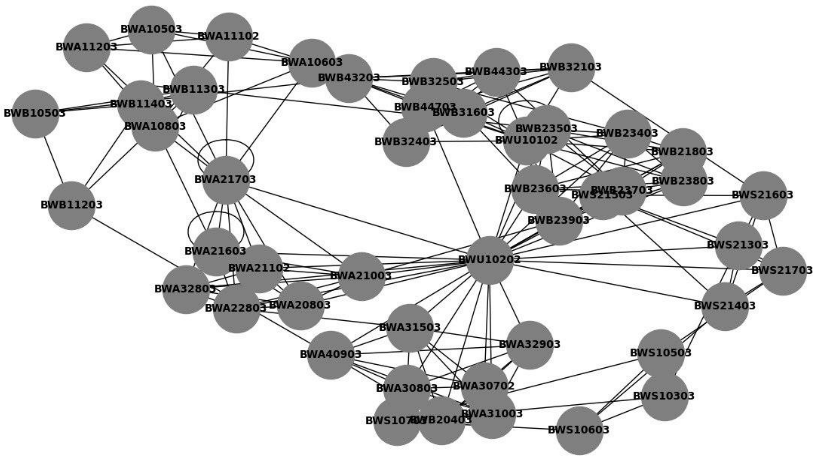


Fig. 2 Conflict Graph Model

Next, after implementing the Welsh-Powell algorithm, the conflict graph in Fig. 3 is successfully coloured with 9 solid colours, therefore the chromatic number is 9. In this context, chromatic number X(G) is the minimum number of colours for non-conflicting time slots scheduling in the given courses. Reaching this optimal value ensures the institutional schedule is as condensed and efficient as possible while strictly adhering to all hard constraints regarding instructor availability and student group requirements.

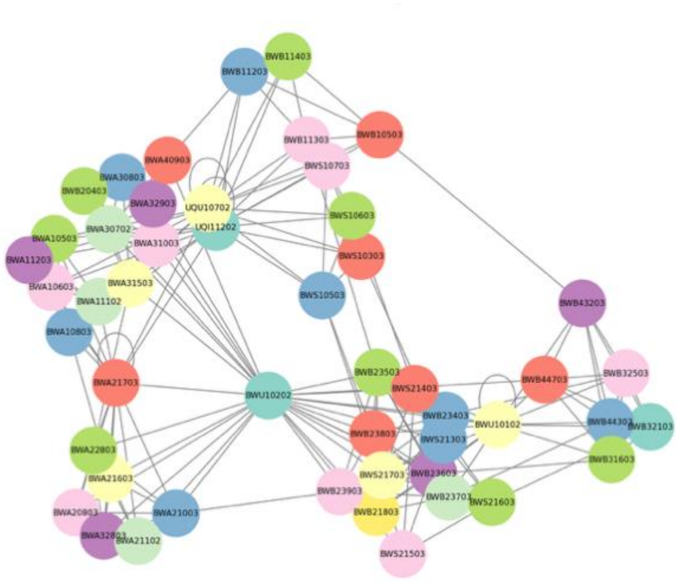


Fig. 3 The Fully Colored Conflict Graph

3.1 GUI Results

The GUI successfully generates a structured and conflict-free academic timetable. The results presented below are the generated academic timetables for all programs, incorporating detailed information such as course codes, instructor identifiers, and assigned classroom for each scheduled session. In total, the scheduling process involves 53 courses, 32 instructors, 8 student groups, and 9 available classrooms. All hard constraints as well as the relevant soft constraints are satisfied properly.

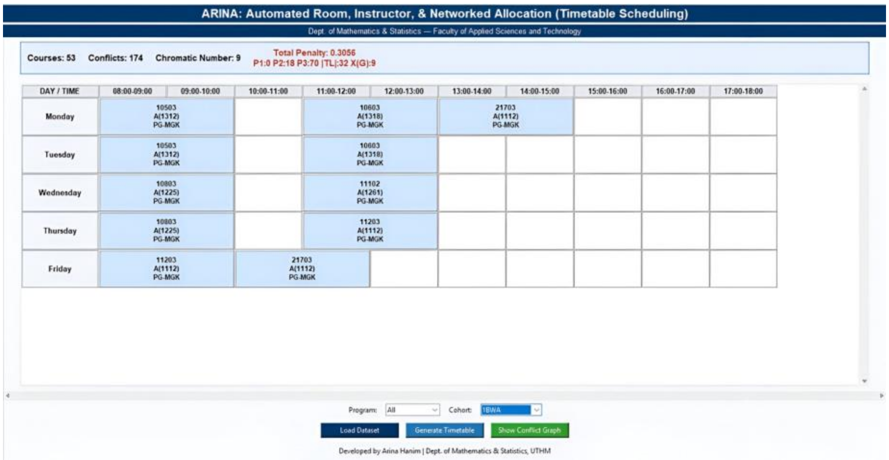


Fig. 4 Generated Timetable for 1BWA

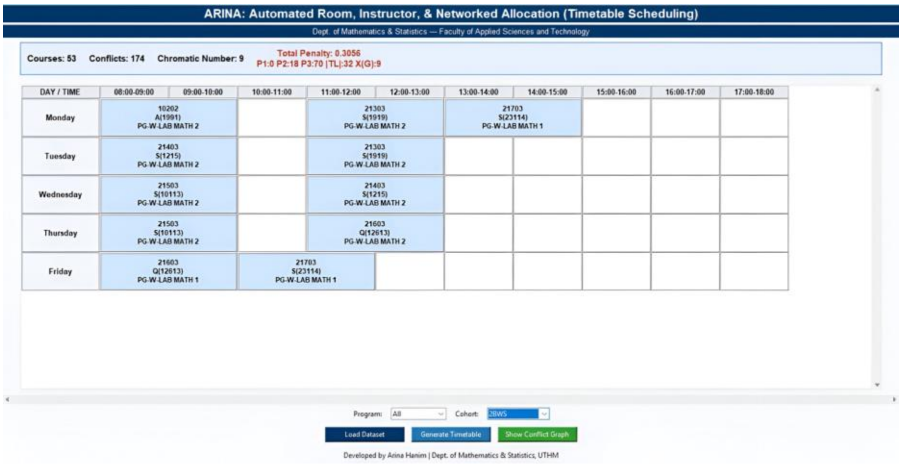


Fig. 5 Generated Timetable for 2BWS

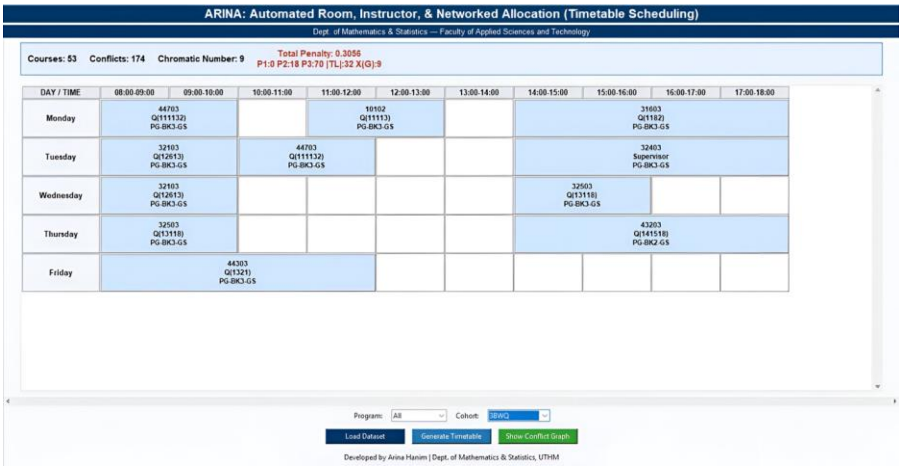


Fig. 6 Generated Timetable for 3BWQ

The GUI considers the credit hours required by all students, course requirements, instructors' distribution, and student capacity for classroom allocation. Every course with 3 credits requires two 2-hour lecture sessions, while courses with 1 or 2 credits require one 2-hour lecture session over a semester. Courses with laboratory sessions have two sessions with two student sections. This is to make sure students get a proper hands-on practice with instructor guidance. Furthermore, the classroom allocation is automatically based on capacity matching between the number of students and classroom size. Overall, the GUI is designed to ensure consistency in graduation requirements across academic programs and institutions, while ensuring the generated timetables are aligned with the academic structure.

3.2 Penalty Score

The generated timetable was analysed by calculating the total penalty score to validate the effectiveness of the GUI. A perfectly balanced timetable should have a score of TP in the range $0.2 \leq TP \leq 0.4$. This range is used as a practical benchmark based on a comparative evaluation between the previous semester's timetable and the GUI timetable. The GUI timetable was compared with the previous semester's timetable, and the results can be seen in Table 2.

Table 2 Comparison of Total Penalty Score

Timetable	P1 (Soft Constraint)	P2 (Slot Utilization)	P3 (Engagement Days)	Total Penalty (TP)
Previous Semester's Timetable	139	39	64	1.3011
GUI Timetable	0	18	70	0.3056

The penalty score is a combination of three components that reflect the quality of the timetable. Based on the comparison, the GUI timetable shows a lower score of total penalty, 0.3056, indicating the proposed algorithm generated a perfectly balanced and efficient timetable with zero soft constraints, a lower number of slot utilization, and a moderate number of engagement days. Meanwhile, the previous semester's timetable scored a total penalty of 1.3011, which is higher than the GUI timetable.

4. Conclusion

An academic timetable is considered good quality if all hard and soft constraints are eliminated. The use of graph colouring technique is very important to make sure all conflicting courses get different colours. This study proves that the Welsh-Powell algorithm has successfully produced a conflict-free academic timetable. By integrating the graph colouring technique into the GUI, the process has become significantly faster and deterministic, which eliminates both hard and soft constraints. The results show a fair distribution of course loads, proper clustering of sessions, and systematic avoidance of restricted time ranges. The penalty score has confirmed measurable improvements from the previous UTHM timetable. Therefore, this GUI is efficient and reliable, making it practical for departmental scheduling.

To further improve this study, future work it is recommended to apply hybrid models that incorporate metaheuristics like Genetic Algorithm or Ant Colony Optimization to handle complex problems. In addition, it is recommended to create a standardized departmental database for courses and instructors to ensure the GUI can provide deterministic and automated updates for each semester.

Acknowledgement

The authors would like to thank the Faculty of Applied Sciences and Technology, Universiti Tun Hussein Onn Malaysia, for the support.

Conflict of Interest

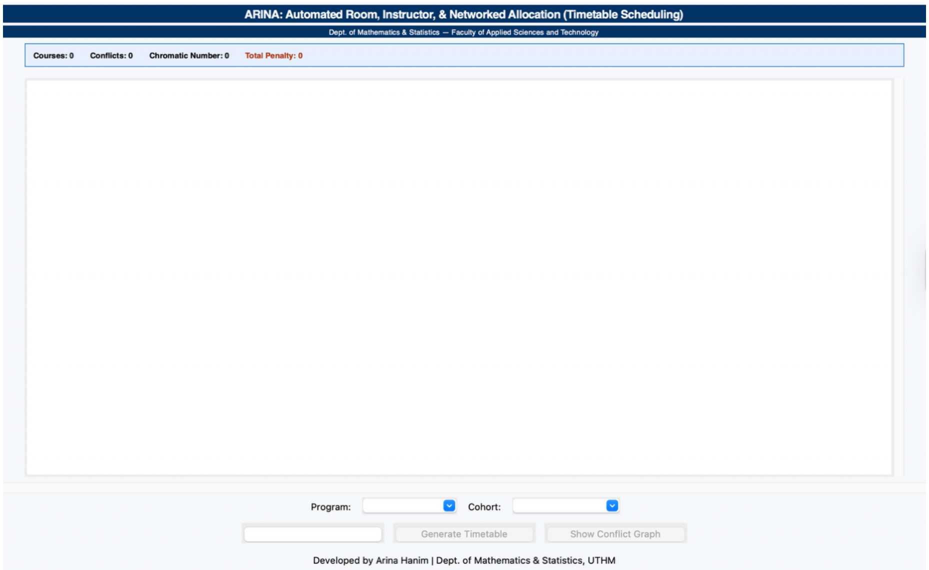
Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Arina Hanim Yahya;* ***data collection:*** *Arina Hanim Yahya;* ***analysis and interpretation of results:*** *Arina Hanim Yahya, Wan Nor Munirah;* ***draft manuscript preparation:*** *Arina Hanim Yahya, Wan Nor Munirah Ariffin. All authors reviewed the results and approved the final version of the manuscript.*

Appendix

ARINA: Automated Room, Instructor & Networked Allocation (Timetable Scheduling) – Copyrighted ©



References

[1] Biswas, S., Nusrat, S. A., Sharmin, N., & Rahman, M. (2023). Graph coloring in university timetable scheduling. *International Journal of Intelligent Systems and Applications*, *15*(3), 16–32. https://doi.org/10.5815/ijisa.2023.03.02

[2] Gangrade, A., Agrawal, B., Kumar, S., & Mansuri, A. (2022). A Study of Applications of Graph Coloring in Various Fields. *International Journal of Statistics and Applied Mathematics*, 7(2), 51-53. https://doi.org/10.22271/maths.2022.v7.i2a.795

[3] Malviya, A., Agrawal, B., & Kumar, S. (2024). Graph Coloring Algorithm for Course Time Table Scheduling Problem. *Shodhkosh Journal of Visual and Performing Arts,* 5(3). https://doi.org/10.29121/shodhkosh.v5.i3.2024.2182

[4] Chen, Y., Liu, H., & Zhang, Q. (2022). Hybrid Ant Colony and Graph Coloring for Dynamic University Timetabling. *IEEE Access*, 10, 34567-34579 https://doi.org/10.1109/ACCESS.2022.3161890

[5] Yu, Y. (2021). A Research Review on Job Scheduling Problem. *E3S Web of Conferences*, 253, 02024. https://doi.org/10.1051/e3sconf/202125302024

[6] Rina, I., Sulistiowati, D., & RaudhatulOktavi, D. (2022). Graph Coloring Applications in Scheduling Courses using Welsh-Powell Algorithm – A Case Study. *IEEE Xplore*, 131-135. https://doi.org/10.1109/isitdi55734.20229944511

[7] Nandal, P., Satyawali, A., Sachdeva, D., & Tomar, A.S. (2021). Graph Coloring Based Scheduling Algorithm to Automatically Generate College Course Timetable. *2022 12th International Conference on Cloud Computing, Data Science & Amp; Engineering (Confluence),* 210-214. https://doi.org/10.1109/confluence51648.2021.9377151

[8] Abdipoor, S., Yaakob, R., Goh, S. L., & Abdullah, S. (2023). Meta-heuristic approaches for the University Course Timetabling Problem. *Intelligent Systems with Applications*, *19*, 200253. https://doi.org/10.1016/j.iswa.2023.200253

[9] Abdullahi, M., Aliyu, S., & Ibrahim, A. (2023). A comparative study of graph coloring and metaheuristics for academic timetabling. *Journal of Operational Research*, 74(1), 88–105. https://doi.org/10.1016/j.jor.2023.100234

[10] Assi, M., Halawi, B., & Haraty, R. A. (2018). Genetic Algorithm Analysis using the Graph Coloring Method for Solving the University Timetable Problem. *Procedia Computer Science*, *126*, 899–906. https://doi.org/10.1016/j.procs.2018.08.024

[11] Badoni, R. P., Kumar, S., Mann, M., Mohanty, R. P., & Sarangi, A. (2023). Ant colony optimization algorithm for the university course timetabling problem using events based on groupings of students. *Modeling and Applications in Operations Research*, 1–36. https://doi.org/10.1201/9781003462422-1

[12] Owens, G., Khusanov, J., Zholdoshov, A., & Dzholbunov, A. (2021). Graph theory, scheduling problems, and its modern applications. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3911654

[13] Gangrade, A., Agrawal, B., & Kumar, S. (2024). An Application Of Graph Coloring: Nurse Scheduling Problem. *ShodhKosh Journal of Visual and Performing Arts*, *5*(3). https://doi.org/10.29121/shodhkosh.v5.i3.2024.2521

[14] Sunanto, U. a. M., Napitupulu, H., Carnia, E., & Supriatna, A. K. (2022). Course Scheduling of Undergraduate Study Program of Mathematics Universitas Padjadjaran Case Using Graph Coloring with Modified Algorithm. *International Journal of Mathematics Trends and Technology*, *68*(1), 61–69. https://doi.org/10.14445/22315373/ijmtt-v68i1p507

[15] Meharwade, S., Mahindrakar, S., & Khator, A. (2024). Graph Theory. https://www.researchgate.et/publication/385482507_Graph_Theory