UNIVERSITY UNDERGRADUATE COURSES TIMETABLING WITH GRAPH COLORING

Sakirudeen A. Abdulsalaam[1] and Khalid Saddiq[2]
Department of Mathematics, Federal University Birnin Kebbi, Nigeria
Mathematics Department, Federal College of Education (Special), Oyo, Nigeria
Email: sa.abdulsalaam@gmail.com

Abstract
Timetable scheduling is a paramount problem in every academic institution of higher learning. A schedule is a desired plan which combines resources like teachers, subjects, students, classrooms in a way to avoid conflicts satisfying various essential and preferential constraints. The timetable scheduling problem is known to be NP-complete but the corresponding optimization problem is NP-Hard. Hence a heuristic approach is preferred to find a near optimal solution within reasonable running time. Graph coloring is one such heuristic algorithm that can deal timetable scheduling satisfying changing requirements, evolving subject demands and their combinations. This study showcase the application of graph coloring to timetable scheduling problem.

Keywords: *Graph Coloring, Course Timetable Scheduling, Hard Constraints, Soft Constraints, Heuristics*

1.0 Introduction

Two of the most common problems in the institutions of higher learning are course timetabling and exam timetabling. It is considered one of the most difficult problems faced by universities and colleges. The university timetable problem is composed of a set of courses offered during the academic semester, a set of instructors teaching specific courses, a fixed number of room location and a list of students registered to a few courses. The problem is about assigning all this input into available timeslots. However, the timetable must satisfy a set of requirements and the solution must be clash-free where all the offered courses are scheduled.

While constructing schedule of courses, it is expected that courses taught by the same professor and courses that require same classroom must be scheduled at different time slots. In addition, a particular student or group of students may be required by a curriculum to take two different but related courses (e.g., Computer Science and Mathematics) concurrently during a semester. In such cases as well, courses need to be scheduled in a way to avoid conflicts. Thus, the problem of determining minimum or reasonable number of time slots that can successfully schedule all the courses subject to restrictions is a typical graph coloring problem (Nandhini, 2019). The University Timetabling problem is known to be an NP-hard optimization problem (de Werra, 1985). This means it cannot be solved in polynomial time by any deterministic algorithm. Reason being that the input is large, and the problem include a number of hard and soft constraints to be satisfied. Hard constraints are very sensitive in the sense that the absence of one such constraint can lead to the failure of the scheduler. Whereas soft constraints come at a rather more flexible level and can be treated with some tolerance at the time of failure. The objective here is to satisfy every single hard constraint while minimizing the violations to the soft ones as much as possible. A feasible solution is any possible assignment of courses to a possible timing and location without violating any of the hard constraints (Qu *et al.*, 2009).

1.1 Basic Graph Theory

A graph G is an ordered triplet (V(G), E(G), Φ), consisting of a non-empty set V of vertices or nodes, E; the set of edges and Φ is the mapping from the set of edges E to the set of vertices V. A simple graph G = (V, E) is Bipartite if we may partition its vertex set V to disjoining sets U and V such that there are no edges between U and V. We say that (U, V) is a Bipartition of G. A Bipartite Graph is displayed in Figure 1.

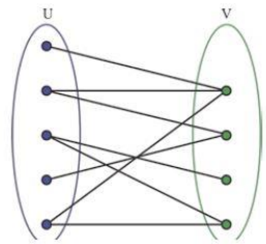


Figure 1: A Bipartite Graph

Given a graph G, a vertex coloring of G (Figure 2) is a function $\psi: V \to C$ where V is a set of vertices of the graph G and C is the set of colors. It is often both conventional and convenient to use numbers 1, 2, …, n for the colors. Proper k-coloring (Bondy, 1969) of G is a coloring function ψ which uses exactly k colors and satisfies the property that $\psi(x) \neq \psi(y)$ whenever x and y are adjacent in G. The minimum number of colors require to color G is known as its chromatic number $\chi(G)$. A graph that assigns properly k-colors is known as k-colorable. It is k-chromatic if its chromatic number is exactly k. The chromatic polynomial counts the number of ways a graph may be colored using no more than a given number of colors.

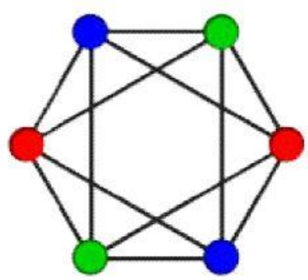


Figure 2: A Vertex Coloured Graph

Edge-coloring of graphs, on the other hand (Figure 3), corresponds to vertex-coloring. Given a graph G, an edge-coloring of G is a function ψ_0 from the edges of G to a set C of elements called colors. For every edge coloring problem there exists an equivalent vertex coloring problem (Deo, 2017)

1.3 Motivation

Effective timetable is vital to the performance of any educational institute. It impacts their ability to meet changing and evolving subject demands and their combinations in a cost-effective manner satisfying various constraints. In this paper, we have focused on Course Timetable Scheduling.

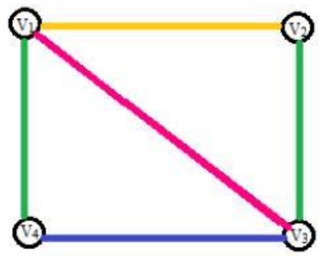


Figure 3: Edge Coloured Graph

After this introductory section, which prefaced the paper, the remainder of this article is organized as follows: We survey the literature in the next section while Section 3 deals with scheduling problem.

2.0 Review of Literature

Solving timetabling problems with the aid of computer applications has a long and varied history. In 1967, graph colouring was applied to the problem of timetable scheduling (K Associate Professor, 2011; T A Redl, 2009; Timothy A Redl, 2007)(Peck & Williams, 1966; T A Redl, 2009; Wood, 1969). In 1967, Welsh and Powell (Welsh & Powell, 1967) illustrated the relationship between timetabling and graph coloring, and introduced a new general graph coloring algorithm to solve the minimum coloring problem more efficiently.

They were also successful in coloring graphs that arise from timetabling problems, more specifically examination timetabling problems. In 1969, Wood's graph algorithm (Wood, 1969) operated on two $n \times n$ matrices, where n denotes the number of vertices in the graph; a conflict matrix C was used to illustrate which pairs of vertices must be colored differently due to constraint restrictions in the problem and a similarity matrix S was used to determine which pairs of vertices should be colored the same. Dutton and Bingham in 1981 introduced two of the most popular heuristic graph coloring algorithms. Considering each color one by one, a clique (Deo, 2017) is formed by continually merging the two vertices with the most common adjacent vertices. On completion, identical coloring is applied to all the vertices which are merged into the same. Carter (Carter et al., 1996), in his examination timetabling survey, refers to some of the above-mentioned graph coloring algorithms and heuristics and shows how graph theoretic approach to timetable scheduling is one of the most popular. It has been accepted and applied by many educational institutions to solve their examination timetabling problems. According to Carter in his survey, work of Mehta is significant as its objective of obtaining "conflict-free" schedules, given a fixed number of time periods turned out to be one of the most complex timetabling applications (Karp, 1972). In 1991, Johnson, Aragon, McGeoch and Schevon (Johnson *et al.*, 1991) implemented and tested three different approaches for graph coloring with a simulated annealing technique, observing that simulated annealing algorithms can achieve good results, but only if allowed a sufficiently large run time. In 1992, Kiaer and Yellen in a paper (Kaier & Yellen, 1992) describes a heuristic algorithm using graph coloring approach to find approximate solutions for a university course timetabling problem. In 1994, Burke, Elliman and Weare (Burke et al., 1994) introduced plans for a university timetabling system based on graph coloring and constraint manipulation. Graph coloring and room allocation heuristic algorithms were described along with an illustration of how the two can be combined to provide the basis of a system for timetabling. The authors also discussed the handling of several common timetabling features within the system, primarily with regards to examination timetabling. In 1995, graph coloring method was introduced aiming at optimizing solutions to the timetable scheduling problems (Miner et al., 1995). Bresina (1996) was among the early researchers who used this approach and made several modifications in the manual approach conducted at universities (Bresina, 1996). In 2007, for university timetabling, an alternative graph coloring method was presented that incorporates room assignment during the coloring process (Timothy A Redl, 2007). In 2008, the Koala graph coloring library was developed which includes many practical applications of graph coloring, and is based on C++ (Dobrowolski *et al.*, 2008). In 2009, automata-based approximation algorithms were proposed for solving the minimum vertex coloring problem (Torkestani & Meybodi, 2009). Akbulut and G. Yılmaz in 2013 (Akbulut & Yilmaz, 2013) proposed a new university examination scheduling system using graph coloring algorithm based on RFID technology. This was examined by using different artificial intelligence approaches. Also, in recent years, researchers have been exploring new alternative methods to deal with scheduling problems for obtaining better result.

3.0 The Scheduling Problem

The general idea of a scheduling problem is to allocate resources among a number of time-slots, satisfying various types of essential and preferential constraints, with the aim of creating optimized conflict-free schedule. Some of the typical scheduling problems include: Timetable Scheduling **(**Course Timetable or Exam Timetable), Aircraft Scheduling, Job Shop Scheduling, etc.

3.1 Course Timetable Scheduling

Course Timetabling is the scheduling of a set of related courses in a minimal number of timeslots such that no resource is required simultaneously by more than one event. In a typical educational institute, resources which may be required by courses simultaneously can be students, classrooms, or teachers.

3.2 Constraints

Constraints are the most vital aspect of any scheduling problem. These are the various restrictions involved in creating a schedule. Based on satisfaction of these, a schedule can be accepted or get rejected. Depending on the degree of strictness, constraints are broadly classified into- Hard and Soft Constraints

Hard constraints

Hard Constraints are those essential conditions which must be satisfied to have a legal schedule. If any of the hard constraints cannot be placed successfully by a schedule, then such a schedule is rejected. For example, no two subjects having common students can be scheduled in the same time-slot, courses cannot be assigned to more than maximum number of available time-slots or periods. In those scheduling datasets which involve resources like teachers and classrooms, no courses can be scheduled to the same classroom at same time-slot, more than one course taught by the same teacher cannot be assigned same time of the week.

Soft constraints

Soft Constraints on the other hand are those preferential conditions which are optional. Mostly, it gets difficult to incorporate all the soft constraints in a schedule. A schedule is still said to be legal even if it fails to satisfy soft constraints, provided all hard constraints are met. For example, a teacher may prefer to take practical classes only in the second half, honours and pass classes are preferred to be scheduled in non-overlapping time-slots, etc.

Temporal and Spatial Constraints

Constraints can also be viewed as time-related and space-related conditions. Time-related constraints are called Temporal constraints. For example, Computer Science theory and practical classes cannot be scheduled at the same time-slot or period because of common students. Also, there must be a fixed number of theory and practical classes scheduled in a week.

On the other hand, space-related constraints are called Spatial constraints. In course scheduling problem, spatial constraints mainly involve classroom related issues. Any educational institute has a fixed number of available rooms with specified capacity. Also, classrooms can be theory or laboratory based. While making schedules, courses having student capacity compatible to the classroom size is an essential condition. Courses which need specific classroom type need to be assigned accordingly.

Both temporal and spatial constraints are mainly hard type constraints whose fulfilment determines the effectiveness of a schedule.

4.0 Methodology

For solving Course Timetable scheduling problems using graph coloring, the problem is first formulated in the form of a graph where courses act as vertices. Depending on type of graph created, edges are drawn accordingly. One type is conflicting graph where edges are drawn between conflicting courses having common students. Other one is non-conflicting graph, where edges are drawn between mutually exclusive courses having no students in common. Sometimes it is found that creating a non-conflicting graph from the given input set and constraints is easier costing less time. This non-conflicting graph needs to be complemented to obtain the required conflict graph whose proper coloring provides the desired solution. This two-step method is efficient in certain cases, while in some conflict graph is created directly.

Some problems involve few resources while others may require many at a time. Courses can also conflict due to common teachers, common classrooms in addition to common students. In such cases, the conflict graph must consider course, teacher and room conflicts simultaneously.

As mentioned earlier, there can be various aspects of a scheduling problem. When teachers are involved in resources, other factors like availability of teachers, subject area preferred by each teacher acts as additional data inputs which needs to be provided for making a complete schedule.

Graph coloring algorithm

Input: The course conflict graph G thus obtained act as the input of graph coloring algorithm.

Output: The minimum number n of non-conflicting time-slots required to schedule courses.

Degree sequence is the array having degree of each vertices of the input graph G. Colors being used are stored in Used_Color array. And the chromatic number will be the total number of elements in the Used_Color array.
Step 1: Input the conflict graph G.
Step 2: Compute degree sequence of the input conflict graph G.
Step 3: Assign color1 to the vertex vi of G having highest degree.
Step 4: Assign color1 to all the non-adjacent uncoloured vertices of vi and store color1 into Used_Color array.
Step 5: Assign new color which is not previously used to the next uncoloured vertex having next highest degree.
Step 6: Assign the same new color to all non-adjacent uncoloured vertices of the newly colored vertex.
Step 7: Repeat step-5 and step-6 until all vertices are colored.
Step 8: Set minimum number of non-conflicting time-slots n= chromatic number of the colored graph=total number of elements in Used_Color array.
Step 9: End

5.0 Illustration

An undergraduate science student in a Nigerian Universities offer a variety of courses. Some courses are taken across department while some are for a specific programme. We illustrate the efficacy of this technique using a few year one courses. Suppose the students are to offer n number of general (departmental/core) course and m number of minor (borrowed) courses, the available number of periods, p, course timetable should be prepared satisfying some given constraints. The objective is to find minimum number of time slots to schedule all the courses without conflict.

Input Dataset: Table 1 shows the courses combination.

Major	**Courses Offered**
Mathematics	MAT101, MAT102, STA112, STA121, PHY101
Statistics	MAT101, MAT102, STA112, STA131, PHY101
Physics	MAT101, MAT102, STA112, PHY101, PHY121

List of constraints

Hard Constraints

- Courses having common student cannot be allotted at the same time slot on the same day.
- Total number of available periods is 8. (maximum)

Soft Constraints

- Minor and major courses must be scheduled in non-overlapping time-slots.

Solution

Each course is considered as a node and an edge exists between two nodes if and only if there is common student (see Figure 1).

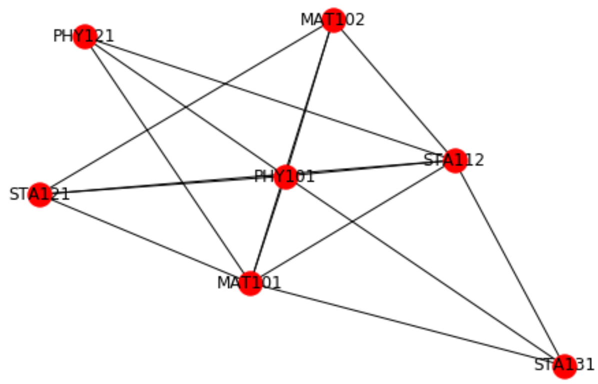
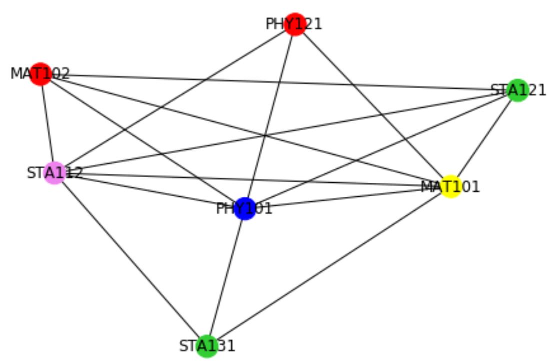


Figure 1: Course Conflict Graph

After applying graph coloring algorithm gives the graph in Figure 2. The graph is properly coloured with chromatic number 5. This is the minimum number of non-conflicting timeslots required to schedule all the given courses.



Result and Discussion

In the above solution, the hard constraints are satisfied properly. The resultant minimum number of time slots needed is 5 which do not exceed the total available 8 periods. No student overlapping is acceptable between all the courses except STA131, PHY102 and STA121 and separate timeslots must be used in these cases.

Conclusion

The complexity of a scheduling problem is directly proportional to the number of constraints involved. There is no fixed algorithm to solve this class of problem. Here we have studied a typical undergraduate course combination scheduling problem under university curriculum. Uniqueness and optimality are the main concerns in this problem. For the same chromatic number, there are many alternative solutions, and thus it is not unique. Although all the solutions can be claimed optimal when solved using minimum number of colors, a better schedule is one which maximizes satisfaction of soft constraints among its alternative solutions. The dynamic nature of scheduling problem pose challenges and warrant further experiment with large data sets and complex constraints. An algorithm which can evenly distribute resources among available timeslots without conflict, create unique and optimized schedule and satisfy all hard and maximum number of soft constraints can be called ideal. Finding such algorithm is surely an evolving area of further research.

References

Akbulut, A., & Yilmaz, G. (2013). University exam scheduling system using graphcoloring algorithm and rfid technology. *International Journal of Innovation, Management and Technology*, *4*(1), 66.

Bondy, J. A. (1969). Bounds for the chromatic number of a graph. *Journal of Combinatorial Theory*, *7*(1), 96-98.

Bresina, J. L. (1996). Heuristic-biased stochastic sampling. *AAAI/IAAI, Vol. 1*, 271-278.

Burke, E. K., Elliman, D. G., & Weare, R. (1994). A university timetabling system based on graph colouring and constraint manipulation. *Journal of Research on Computing in Education*, *27*(1), 1-18.

Carter, M. W., Laporte, G., & Lee, S. Y. (1996). Examination timetabling: Algorithmic strategies and applications. *Journal of the Operational Research Society*, *47*(3), 373-383.

de Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, *19*(2), 151-162.

Deo, N. (2017). *Graph theory with applications to engineering and computer science*. Courier Dover Publications.

Dobrowolski, T., Dereniowski, D., & Kuszner, L. (2008). Koala graph coloring library: An open graph coloring library for real-world applications. *2008 1st International Conference on Information Technology*, 1-4.

Johnson, D. S., Aragon, C. R., McGeoch, L. A., & Schevon, C. (1991). Optimization by simulated annealing: an experimental evaluation; part II, graph coloring and number partitioning. *Operations Research*, *39*(3), 378-406.

K Associate Professor, C. N. (2011). Timetable Scheduling using Graph Coloring. *International Journal of P2P Network Trends and Technology*, *1*.

Kaier, L., & Yellen, J. (1992). Weighted graphs and University timetabling. *Computers & Operations Research*, *19*(1), 59-68.

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations* (pp. 85–103). Springer.

Miner, S., Elmohamed, S., & Yau, H. W. (1995). Optimizing timetabling solutions using graph coloring. *NPAC, Syracuse University*, 99-106.

Nandhini, V. (2019). A Study on Course Timetable Scheduling and Exam Timetable Scheduling using Graph Coloring Approach. *International Journal for Research in Applied Science and Engineering Technology*, *7*(3), 1999–2006. https://doi.org/10.22214/ijraset.2019.3368

Peck, J. E. L., & Williams, M. R. (1966). Algorithm 286: Examination scheduling. *Communications of the ACM*, *9*(6), 433-434.

Qu, R., Burke, E. K., McCollum, B., Merlot, L. T. G., & Lee, S. Y. (2009). A survey of search methodologies and automated system development for examination timetabling. *Journal of*

Scheduling, *12*(1), 55-89.

Redl, T A. (2009). On Using Graph Coloring to Create University Timetables with Essential and Preferential Conditions. . Advances in Marketing, Management and Finances, Proceedings of the 3rd International Conference on Management, Marketing and Finances (International Conference on Computational and Information Sciences, University of Houston-Downtown, Houston, TX, April 2, 2-7.

Redl, Timothy A. (2007). University timetabling via graph coloring: An alternative approach. *Congressus Numerantium*, *187*, 174.

Torkestani, J. A., & Meybodi, M. R. (2009). Graph coloring problem based on learning automata. *2009 International Conference on Information Management and Engineering*, 718-722.

Welsh, D. J. A., & Powell, M. B. (1967). An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, *10*(1), 85-86.

Wood, D. C. (1969). A technique for colouring a graph applicable to large scale timetabling problems. *The Computer Journal*, *12*(4), 317-319.