I.J. Intelligent Systems and Applications, **2023, 3, 16-32**
Published Online on June 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2023.03.02

Graph Coloring in University Timetable Scheduling

Swapnil Biswas
Department of Computer Science and Engineering, Military Institute of Science and Technology Mirpur Cantonment, Dhaka 1216, Bangladesh
E-mail: swapnil.cse16@gmail.com

Syeda Ajbina Nusrat
Department of Computer Science and Engineering, Military Institute of Science and Technology Mirpur Cantonment, Dhaka 1216, Bangladesh
E-mail: hridiajbina@gmail.com
ORCID iD: https://orcid.org/0000-0002-9658-5570

Nusrat Sharmin*
Department of Computer Science and Engineering, Military Institute of Science and Technology Mirpur Cantonment, Dhaka 1216, Bangladesh
E-mail: nusrat@cse.mist.ac.bd
ORCID iD: https://orcid.org/0000-0003-2856-3561
*Corresponding Author

Mahbubur Rahman
Department of Computer Science and Engineering, Military Institute of Science and Technology Mirpur Cantonment, Dhaka 1216, Bangladesh
E-mail:nahbub@cse.mist.ac.bd
ORCID iD: https://orcid.org/0000-0001-6525-2274

Received: 24 July 2022; Revised: 03 October 2022; Accepted: 27 January 2023; Published: 08 June 2023

Abstract: Addressing scheduling problems with the best graph coloring algorithm has always been very challenging. However, the university timetable scheduling problem can be formulated as a graph coloring problem where courses are represented as vertices and the presence of common students or teachers of the corresponding courses can be represented as edges. After that, the problem stands to color the vertices with lowest possible colors. In order to accomplish this task, the paper presents a comparative study of the use of graph coloring in university timetable scheduling, where five graph coloring algorithms were used: First Fit, Welsh Powell, Largest Degree Ordering, Incidence Degree Ordering, and DSATUR. We have taken the Military Institute of Science and Technology, Bangladesh as a test case. The results show that the Welsh-Powell algorithm and the DSATUR algorithm are the most effective in generating optimal schedules. The study also provides insights into the limitations and advantages of using graph coloring in timetable scheduling and suggests directions for future research with the use of these algorithms.

Index Terms: Graph Coloring, Scheduling.

1. Introduction

University course scheduling refers to the process of assigning a number of time slots to the offered courses of a university for a particular semester [1]. The time slots need to be assigned in such a manner so no two courses sharing the common students and teachers are assigned with the same time slot. A manual implementation process by teaching staff or other responsible person is followed by most of the universities to schedule their courses [2]. The human driven manual process might be effective for a system consisting of a smaller number of courses, students and teachers however with the increasing number of courses, students and teachers this process can prompt a genuine misuse of human effort and time as well as it can lead to solution with conflicts. In order to address this problem, automation of course scheduling has been a popular research topic since 1970[3].

There are many existing algorithms for solving the problem of university timetable scheduling, such as:

- **Constraint-based algorithms:** These algorithms use constraint satisfaction techniques to generate schedules that satisfy a set of predefined constraints. These algorithms are highly customizable and can handle complex con- straints, but they can be computationally expensive and may not perform well to large-scale problems.
- **Genetic Algorithms:** These algorithms use genetic operators such as selection, crossover, and mutation to generate schedules. These algorithms are highly flexible and can handle complex constraints, but they can be computationally expensive and may not find the optimal solution.
- **Local Search Algorithms:** These algorithms use heuristics to search for schedules that are locally optimal. They are often faster than constraint-based and genetic algorithms, but they may not find the global optimal solution.
- **Backtracking Algorithms:** These algorithms use backtracking techniques to find a solution. They are easy to im- plement and provides the guarantee to lead to the optimal solution, but they have a huge exponential computational time.

The limitations of these existing algorithms are, they may not be efficient, they may not find the optimal solution, they may not be able to handle complex constraints, they may not scale well to large-scale problems and they may be computationally expensive.

In graph theory, the graph coloring problem has three major variants. They are vertex coloring, edge coloring and face coloring. Vertex coloring is characterized as the assignment of colors to the vertices of a graph to such an extent that no two adjacent vertices are assigned with the same color. University timetable scheduling can be modeled as vertex coloring problem as vertex coloring is often referred to as graph coloring because other graph coloring problems can be converted into vertex coloring problems and this convention has been followed throughout this entire study. For a graph G, k-coloring refers to the assignment of k number of colors to the vertices of G where no two adjacent vertices are assigned with the same color and the colors are represented by non negative integer numbers from 1 to k. The chromatic number of G is the minimum value of k for which there exists a k-coloring of G. Chromatic number of a graph G is represented by X(G) [4, 5].

The way toward preparing a functional course scheduling report manually meeting all the constraints is really a terrible and time consuming task for a teaching staff at Military Institute of Science and Technology (MIST). As a little change hampers the entire schedule so in most of the cases it is tiring to roll out certain improvements in the course plan and apply a better strategic plan. The main objective of this paper is to replace this tiring process and compare the effectiveness of five graph coloring algorithms (First Fit, Welsh Powell, Largest Degree Ordering, Incidence Degree Ordering, and DSATUR) in solving the problem. We evaluate the performance of each algorithm in terms of their penalty and the quality of the generated schedules. Additionally, we provide new insights into the limitations and advantages of using different graph coloring algorithms in timetable scheduling and suggest new directions for future research by comparing the Welsh-Powell algorithm and the DSATUR algorithm, which are found to be the most effective in generating optimal schedules.

This paper is organized as follows. In section 2, existing course scheduling techniques using graph coloring algorithms or heuristics are discussed. The problem statement is stated in section 3 while the problem representation is in section 4. In section 5, the methodology is described elaborately with the five algorithms. Section 6 contains experimental design including data setup, evaluation criteria, and experimental result with some concluding remarks in 7.

2. Literature Review

There are a number of studies directed over the years on graph coloring problems. In this section, we will discuss the solutions of graph coloring to address the course scheduling problem.

In [6], the authors proposed a comparison study of graph coloring algorithms based on their solution accuracy and exe- cuting times. In [7, 8], the authors proposed a course timetable scheduling system using a graph coloring algorithm. They focus on college course timetables with defined hard and soft constraints. They also studied a teacher-subject schedul- ing problem where two alternative graph coloring methods were applied and a complete solution was provided. The researcher in [9] proposed a concern with the problem of course timetable scheduling. They have adopted various graph colorings algorithms such as the Saturation algorithm, Degree of Saturation Algorithm, Simulated Annealing algorithm, and Greedy algorithm. Here graph coloring algorithm is designed to construct to minimize conflicting schedules.

In [10], the authors proposed a university examination scheduling system using a graph coloring algorithm based on RFID technology. This architecture developed exam timetabling problem is examined by using different artificial intelligence. The researcher in [11] proposed a novel approach of graph coloring to solve university course timetabling problems. They developed graph coloring with the Welch Powell algorithm that can produce class schedules. In [12], the author's proposed timetable scheduling using graph coloring. They develop a general system that can cope with the ever-changing requirements of large educational institutions and provides efficient scheduling of courses. In [13], the authors developed automatically generated college course timetables using a coloring scheduling algorithm. They proposed an algorithm that will consider all required constraints from both the students' and teachers' points of view. In [14], the authors proposed automata-based approximation algorithms compared with some well-known coloring algorithms and solved the minimum vertex coloring problem.

In [15] for alternative graph, coloring method was presented timetabling courses at a university can be modeled and solved using graph coloring techniques. They also present a university timetable that incorporates room assignments during the coloring process using an alternative graph coloring method. In [16], the authors proposed a graph coloring algorithm for Solving University Course Timetabling Problem. They proposed a new approach to solve course timetabling problems that are encountered by educational institutions frequently.

In [17], the author's proposed graph coloring method was developed for optimizing solutions to the timetabling prob- lem. In [18] a university timetabling system based on graph coloring developed, and several common timetabling features can be handled in the system. And they also develop common timetabling features that can be handled within the sys- tem. In [19] propose a new memetic teaching learning-based optimization algorithm combined with a robust tabu search algorithm to solve the graph coloring problem.

In the context of the present study, which aims to compare the performance of different graph coloring algorithms for solving the university timetable scheduling problem, a thorough literature review has been conducted to gain insight into the various approaches and techniques that have been proposed in the past. This review helped to identify the gaps in the existing knowledge and to formulate the research objectives and questions that guided the study. Furthermore, it also assisted to establish the theoretical foundations of the study, and to provide a basis for the selection of the most appropriate techniques and methods for the research. Overall, the literature review played a pivotal role in shaping the research methodology and providing a comprehensive understanding of the field, which ultimately enabled us to achieve the research objectives and draw meaningful conclusions from the study.

3. Problem Statement

In the Military Institute of Science and Technology (MIST) the undergraduate classes take place 5 days in a week from Sunday to Thursday. There are 6 time slots per day and the time slots are denoted from A to F sequentially. Duration of each slot is 1 hour. So, there are total 6 x 5 = 30 time slots per week. These 30 slots are represented with positive integer from 1 to 30 sequentially. The slot distribution per week is illustrated in Table-1.

The undergraduate students of MIST are categorized into 4 levels: Level-1, Level-2, Level-3, Level-4. If the level of a student is e then that means that the student is passing his/her e-th year at MIST. A student cannot belong to multiple levels and all the students of a particular level always register for the same set of courses. Each course has a unique 3 digit identifying code. The leftmost digit of the code indicates the level in which the course is offered. For example, if the code of a course is XYZ then the course is offered in Level-X where $0 \leqslant Y, Z \leqslant 9 and 1 \leqslant X \leqslant 4$.

Table 1. Slot Distribution for Course Scheduling Per Week at MIST

	A	B	C	D	E	F
SUN	1	2	3	4	5	6
MON	7	8	9	10	11	12
TUE	13	14	15	16	17	18
WED	19	20	21	22	23	24
THU	25	26	27	28	29	30

It means that if the leftmost digit of the code of two courses is the same then the courses are registered by the same set of students. A course can be conducted by one or multiple teachers. Each course is offered with a fixed contact hour that determines the number of required slots per week of the course. If the contact hour of a course is c then the course requires c number of time slots per week where c is a positive integer. The offered courses are classified in 2 categories: theoretical and sessional. The category of a course can be identified from its course code. If the rightmost digit of a course is odd then the course is theoretical otherwise sessional. For example, if the code of a course is XYZ and Z is odd then the course is theoretical where $0 \leqslant Y, Z \leqslant 9 and 1 \leqslant X \leqslant 4$.

4. Problem Representation

4.1. Terminologies

- The number of courses offered in a particular semester is **N**.
- Each course is identified by a unique integer course code of 3 digits as described earlier and it is denoted by **c**.
- A list is maintained containing all the unique course codes and denoted by **CL**. $|CL|$ = N.
- Each course is assigned with a contact hour that is a positive integer. A function **CH(c)** returns the contact hour of a course. A course **c** is splitted into **CH(c)** number of segments and each segment is assigned with a

unique integer **cId**. For example, If CH(c) = 3 then **c** will be splitted into 3 distinct segments. Consider that the **cId**s of the 3 segments are p,q,r where p, q and r are not equal. A function **f(cId)** maps the cId with a distinct course code in **C**L. For the above example, **c** is returned from f(p), f(q) and f(r). It is notable that **cId**s of two courses are generated in such a way that they do not match either completely or partially and **cId**s are generated sequentially starting from 1.

- The set of all the **cId**s is denoted by **CI** and $|CI| = n$.
- All the teachers are assigned with a unique **tId**. The set of all **tId**s is denoted by **TL**.
- The slot distribution is illustrated in Table-1 where each available slot is marked with a unique positive integer sequentially starting from 1. The set of all available slots is denoted by **SL**.

4.2. Constraints

The undergraduate course scheduling of MIST follows 2 types of constraints: hard constraints and soft constraints. If any of the hard constraints is violated in a solution then that solution is marked as infeasible. But a solution violating a soft constraint in fact all the soft constraints are accepted but it is preferred to adjust the soft constraints.

A. Hard Constraints

3 hard constraints HC1, HC2 and HC3 are followed during the course scheduling process of MIST. A solution is considered as feasible if

$$\text{HCl} \wedge \text{HC2} \wedge \text{HC3} = \text{TRUE}$$

otherwise, the solution is marked as infeasible.

- **HC1 = TRUE**: If the contact hour of a course **c** is **CH(c)** then the course needs to be assigned with exactly **CH(c)** number of slots per week.
- **HC2 = TRUE**: If two courses have any common teachers then they cannot be assigned in the same time slot.
- **HC3 = TRUE**: If two courses are offered in the same level then they cannot be assigned in the same time slot as they share common students.

B. Soft Constraints

A soft constraint SC1 is preferred to adjust in the course scheduling process after adjusting all the hard constraints.

- **SC1 = TRUE**: If the category of a course **c** is sessional and the contact hour of the course is **CH(c)** then the course is preferred to be assigned with **CH(c)** number of consecutive slots in a day.

4.3. Graph Representation

The course scheduling problem of MIST is represented as an undirected graph G where the set of vertices is denoted by **V** and the set of edges is denoted by **E**. G is represented by an adjacency matrix named **adj**.

A. Vertex

Each element of **CI** is represented as a vertex. Reason for this representation is because each course needs **ch(c)** number of slots to be assigned per week. That is why the course **c** is represented by **ch(c)** number of vertices rather than a single vertex. So, V = CL and |V| = n.

B. Edge

There is an edge between **i** and **j** if **i** and **j** share common students or common teachers where **i** $\in$ Cl and **j** $\in$ **Cl** and **i** $\neq$ **j**.

$$adj_{i,j} = adj_{j,i} = \begin{cases} 1\,if \;\; floor(i/100) = floor(j/100) \\ 1\,if \;\; TL_i \cap TL_j \neq \varnothing \\ 0\;Otherwise \end{cases}$$

i $\in$ Cl and j $\in$ Cl and $i \neq j$
TL_i returns the set of tIds considering i as cId
TL_j returns the set of tIds considering j as cId

C. Colors

The time slots are represented as color. After applying a graph coloring algorithm on **G** each vertex will be assigned with a color and that will represent the time slot of the corresponding course. From Table-1 the available time slots are 1,2,....,30. So if the number of required colors exceeds 30 by an algorithm then the algorithm is considered to be failed to distribute the slots among the courses. The set of colors are considered as a set of sequential integer numbers starting from 1. If the assigned color of a vertex is **i** then that means that the corresponding course will be allocated at **i**-th slot. For example, if for a vertex color-3 is assigned then the corresponding course will be allocated in slot-3. Slot-3 represents SUN-C from Table-1.

For example, let there be six courses and for simplicity assume that the contact hour of each course is one. So, the number of vertices in the graph will be six. The course codes are 101, 103, 201, 203, 205, 301. Course codes are marked with unique cid from 1 to 6 sequentially. Let there are a total five teachers who are marked with unique tid from 1 to 5. Now the assigned cid and tid for each course is illustrated in Table-2.

The graph representation of Table-2 is illustrated in Figure 1 where vertices are marked with cid. An edge exists between vertex-1 and vertex-2 because their corresponding course codes are 101 and 103 which are offered at the same level (level-1). For the same reason, edges exist among vertex-3, vertex-4 and vertex-5. Edges exist among vertex-1 and vertex-3 because their corresponding course codes are 101, 201, 205 which share a common teacher with tid = 1. Vertex-5 and vertex-6 share an edge for having a common teacher with tid = 5.

Table 2.

course code	cid	tid
101	1	1
103	2	2
201	3	1, 3
203	4	4
205	5	1,5
301	6	5

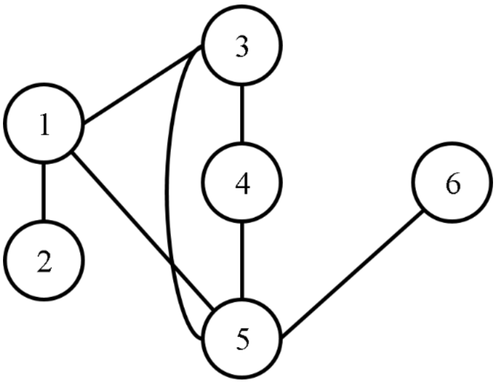


Fig.1. Graph Representation of Table-2

5. Methodology

The methodology section describes the five algorithms that are applied on the undirected graph G constructed from the dataset following the steps described in Section-IV.

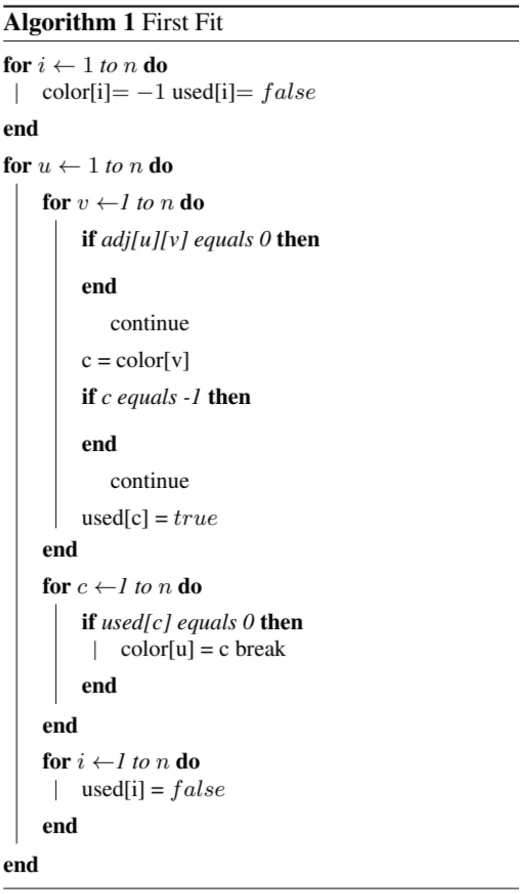
5.1. First Fit Algorithm (FF)

This algorithm employs a simple heuristic approach by assigning the least available color to a vertex, starting with the first vertex in the graph and progressing sequentially. First Fit Algorithm (FF) is easier as well as faster than any other greedy heuristics [20]. It is used as an online algorithm for dynamic graph coloring [21], i.e. refers to a scenario where requests arrive dynamically without any predictable [22]. The algorithm commences by initiating an array, "colors", to store the colors assigned to each vertex. Subsequently, for each vertex, v, in the graph, the process assigns the smallest available color to the vertex and declares the color as used. In Algorithm 1, we have presented the details algorithm in step by step.

Limitations – Though First-Fit is faster and simple it has some drawbacks too. This algorithm does not show good performance for every type of graph. There is some performance issue while dealing with some types of graphs. This bad performance comes from the lack of sensibility of the algorithm to the input sequence [21].

5.2. Welsh Powell Algorithm (WP)

Welsh Powell Algorithm works mainly on static graphs. The Welsh Powell algorithm, being a heuristic approach, employs a greedy strategy of assigning colors by giving priority to vertices with higher degree. Specifically, it prioritizes the vertex of highest degree in the graph and assigns a color to it. Then, it iterates over the vertices in descending order of their degree and assigns the smallest available color to each vertex. This algorithm prioritizes vertices with higher degree to minimize the chances of conflicts and increase the chances of optimal solution. The algorithm employs a strategy of assigning colors by giving priority to vertices with higher degree [23-25]. The details of the algorithms can be found in Algorithm 2.



Algorithm 1 First Fit

```
for $i \leftarrow 1$ to $n$ do
    color[i]= $-1$ used[i]= $false$
end
for $u \leftarrow 1$ to $n$ do
    for $v$ ←1 to $n$ do
        if adj[u][v] equals 0 then
        end
            continue
        c = color[v]
        if c equals -1 then
        end
            continue
        used[c] = $true$
    end
    for $c$ ←1 to $n$ do
        if used[c] equals 0 then
            color[u] = c break
        end
    end
    for $i$ ←1 to $n$ do
        used[i] = $false$
    end
end
```

Limitations – This is an efficient one in terms of giving an upper bound for a chromatic number of a graph but on the other hand, it does not always provide the minimum number of colors needed to color the graph. Hence it cannot give the optimal solution in graph coloring.

5.3. Largest Degree Ordering Algorithm (LDO)

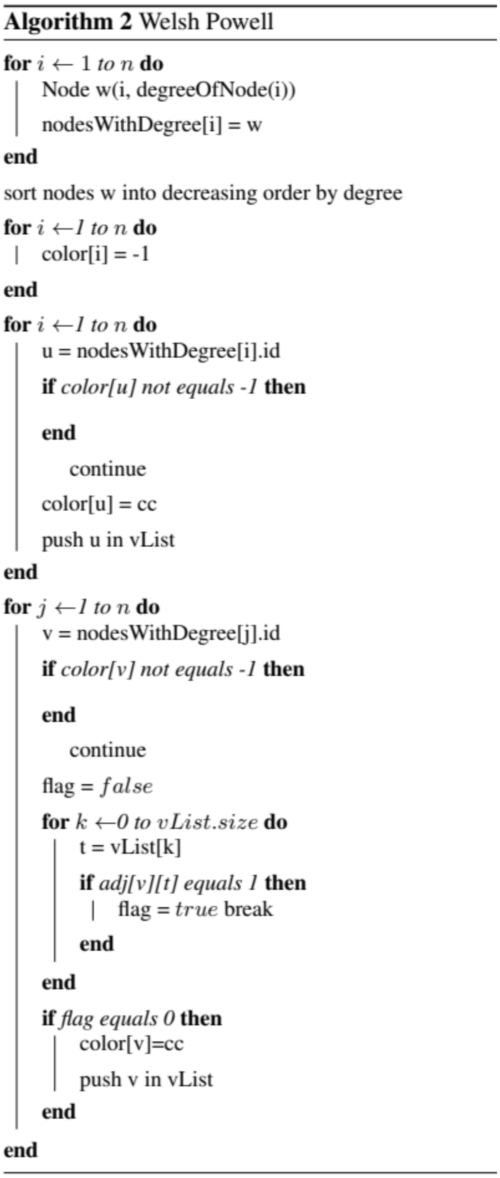
Degree-based ordering provides a better strategy for graph coloring. It does not simply pick a vertex and assigns color without any selection criteria. Largest Degree Ordering is one of the criteria for selecting vertices. This algorithm chooses a vertex which has the highest number of neighbors and colors it [26]. Then goes for the next one. The vertices having the most edges are hard to assign non-conflicting colors [27]. This provides better coloring than the First Fit algorithm. We refer algorithm 2 for details.

5.4. Incidence Degree Ordering Algorithm (IDO)

Another degree-based ordering algorithm is Incidence Degree Ordering. This Algorithm is a technique for determining the degree of a vertex in a graph. The procedure initiates by setting a variable "degree" to 0. Subsequently, it loops through each vertex "v" in the graph, and for each vertex, it iterates through each edge "e" in the graph. If the vertex "v" is one of the endpoints of the edge "e", the algorithm increases the "degree" variable by 1. In conclusion, the algorithm yields the "degree" variable, which signifies the degree of the vertex.

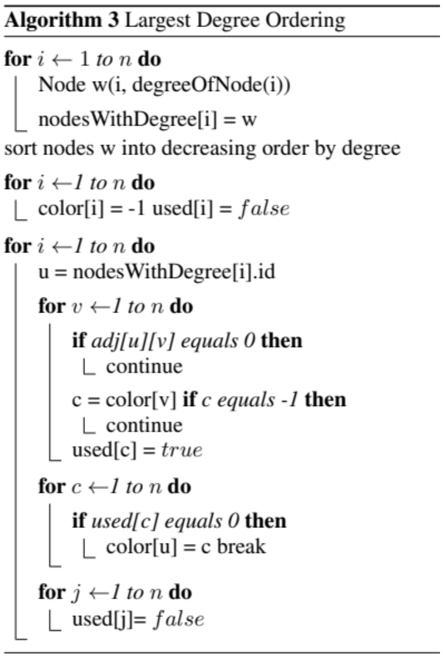
5.5. Degree of Saturation Algorithm (DSATUR)

Degree of Saturation is another degree-based ordering algorithm. This heuristic sequentially colors the uncolored vertices having colored neighbors who use the largest number of distinct colors[28]. The algorithm chooses vertex having maximal saturation degree[29]. This differs from Incidence Degree ordering in terms of choosing the saturation degree instead of incidence degree. There remains the least number of possible colors to choose from for the vertex with the largest number of saturation degrees. The highest degree vertices with the least number of color options are the most difficult ones to color and they should be colored next[27]. DSATUR also works as a branch and bound algorithm[30]. It divides a graph coloring instance into a series of subproblems. All of the subproblems are partial coloration of the graph. There is an upper bound on the number of colors required for the graph in each step[31].



Algorithm 2 Welsh Powell

```
for $i \leftarrow 1$ to $n$ do
    Node w(i, degreeOfNode(i))
    nodesWithDegree[i] = w
end
sort nodes w into decreasing order by degree
for $i \leftarrow$1 to $n$ do
    color[i] = -1
end
for $i \leftarrow$1 to $n$ do
    u = nodesWithDegree[i].id
    if color[u] not equals -1 then
    end
        continue
    color[u] = cc
    push u in vList
end
for $j \leftarrow$1 to $n$ do
    v = nodesWithDegree[j].id
    if color[v] not equals -1 then
    end
        continue
    flag = $false$
    for $k \leftarrow$0 to $vList.size$ do
        t = vList[k]
        if adj[v][t] equals 1 then
            flag = $true$ break
        end
    end
    if flag equals 0 then
        color[v]=cc
        push v in vList
    end
end
```



Algorithm 3 Largest Degree Ordering

```
for $i \leftarrow 1$ to $n$ do
    Node w(i, degreeOfNode(i))
    nodesWithDegree[i] = w
sort nodes w into decreasing order by degree
for $i \leftarrow$1 to $n$ do
    color[i] = -1 used[i] = $false$
for $i \leftarrow$1 to $n$ do
    u = nodesWithDegree[i].id
    for $v \leftarrow$1 to $n$ do
        if adj[u][v] equals 0 then
            continue
        c = color[v] if c equals -1 then
            continue
        used[c] = $true$
    for $c \leftarrow$1 to $n$ do
        if used[c] equals 0 then
            color[u] = c break
    for $j \leftarrow$1 to $n$ do
        used[j]= $false$
```

6. Experimental Design

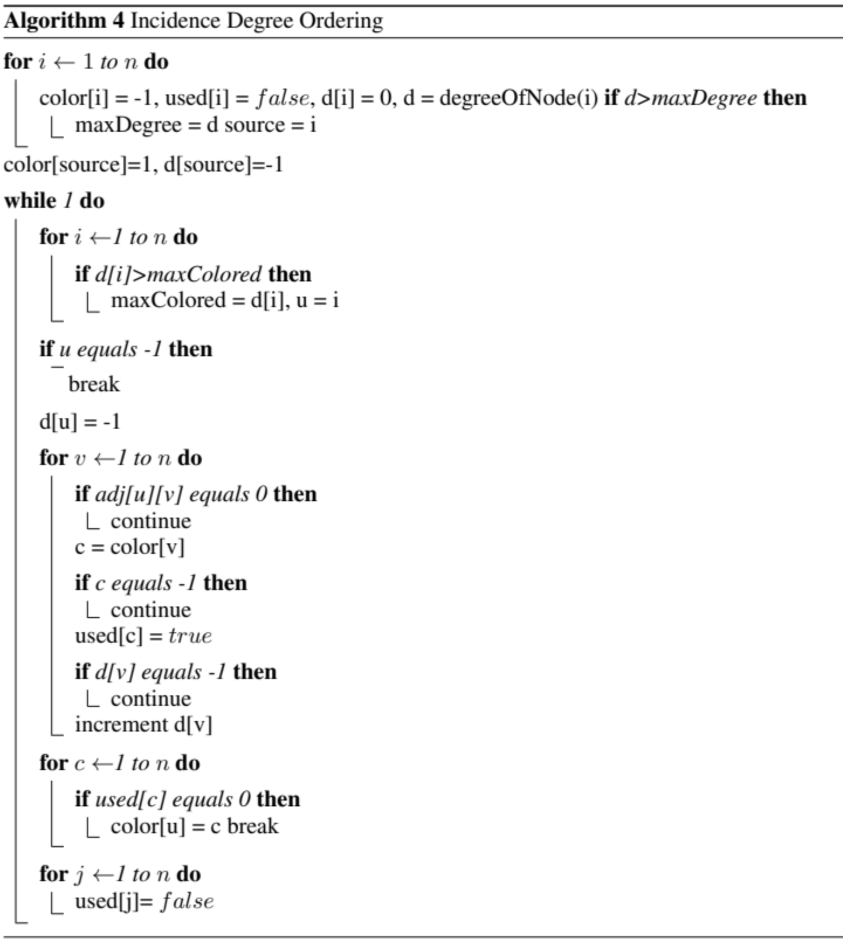
6.1. Data Setup

5 graph coloring algorithms that are described in the Methodology section have been applied on two different datasets illustrated in Table-3 and Table-4 respectively. The first dataset corresponds to the data of course distribution of Spring Term-2021 and the second dataset corresponds to the data of course distribution of Fall Term-2021 of the department of Computer Science and Engineering, MIST. The first dataset in Table-3 consists of 23 distinct courses (rows) mentioned in the first column. So, $|CL| = 23$. Contact hour of each course is associated in the corresponding row of the second column of Table-3. Sum of contact hour is 70. So, the graph consists of 70 vertices. $|V| = 70$. Each course can be associated with maximum of 6 teachers and minimum of 1 teacher. Total number of teachers is 32 and they are marked with tIds from 1 to 32 sequentially. tIds of each course is associated from column-3 to column-8. For the second dataset in Table-4, $|CL| = 31$. The sum of contact hour is 93 which implies that the total number of vertices for the second dataset will be 93. The max tId is 31. So the total number of teachers is 31 and they are marked from 1 to 31 uniquely. Each course can be associated with maximum of 7 teachers and minimum of 1 teacher. These two datasets have been derived in this numerical tabular format for the simplicity of understanding but generally the course distribution is performed in a excel file in more complicated format at MIST.

6.2. Evaluation Criteria

The satisfaction level of a solution of course scheduling problem at MIST is generally measured by 3 criteria.

- Adjusting the soft constraint as much as possible
- Minimizing the number of required slots per week
- Minimizing the number of engaged days of a teacher per week



Algorithm 4 Incidence Degree Ordering

```
for $i \leftarrow 1$ to $n$ do
    color[i] = -1, used[i] = $false$, d[i] = 0, d = degreeOfNode(i) if $d > maxDegree$ then
        maxDegree = d source = i
color[source]=1, d[source]=-1
while $1$ do
    for $i \leftarrow 1$ to $n$ do
        if $d[i] > maxColored$ then
            maxColored = d[i], u = i
    if $u$ equals $-1$ then
        break
    d[u] = -1
    for $v \leftarrow 1$ to $n$ do
        if $adj[u][v]$ equals $0$ then
            continue
        c = color[v]
        if $c$ equals $-1$ then
            continue
        used[c] = $true$
        if $d[v]$ equals $-1$ then
            continue
        increment d[v]
    for $c \leftarrow 1$ to $n$ do
        if $used[c]$ equals $0$ then
            color[u] = c break
    for $j \leftarrow 1$ to $n$ do
        used[j]= $false$
```

That is why 3 penalty functions P1(A), P2(A) and P3(A) are defined for an algorithm A to evaluate the level of satisfaction of a solution generated by that algorithm A.

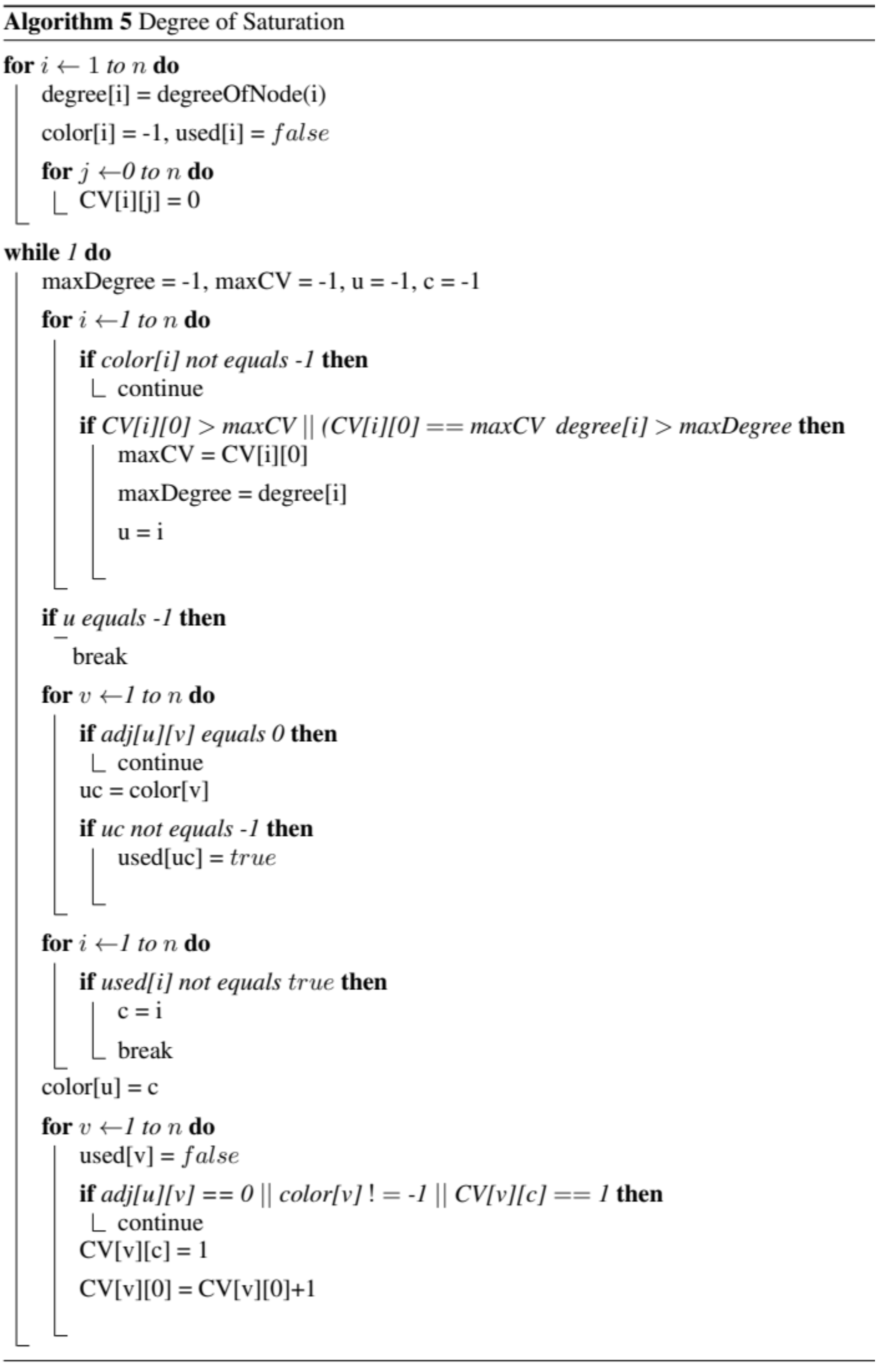
- **P1(A)** is defined as the number of violations of the soft constraint by the solution generated by A.

- **P2(A)** is defined as the number of slots required by the solution generated by A.
- **P3(A)** is defined as the sum of engaged day per week for all the teachers of a solution generated by A.

The penalty of A is denoted by TP(A) and calculated as

$$TP(A) = \frac{P1(A)}{|SC|} + \frac{P2(A)}{X(G)} + \frac{P3(A)}{|TL|} + \frac{P2(A) - X(G)}{X(G)}$$

Here SC is the set of sessional courses. For dataset-1, SC = 202, 204, 206, 302, 304, 306, 318, 402, 404, 460, 462. So, |SC| = 11 and for dataset-2, SC = 104, 106, 212, 214, 216, 220, 224, 308, 310, 316, 360, 414, 416, 444, 452. That means, |SC| = 15. X(G) denotes the chromatic number of G of the corresponding dataset. Backtracking algorithm has been used for calculating the chromatic number of the both of the dataset. For dataset-1, X(G) is 25 and for dataset-2, X(G) is 30. TL denotes the set of teachers (tIds) of the corresponding dataset. For Table-3, |TL| = 31 and for Table-4, |TL| = 32. The satisfaction of algorithm-A is measured from TP(A). The algorithm that generates a solution with lower amount of penalty points (TP) is considered as more satisfactory.



Algorithm 5 Degree of Saturation

```
for $i \leftarrow 1$ to $n$ do
    degree[i] = degreeOfNode(i)
    color[i] = -1, used[i] = $false$
    for $j \leftarrow 0$ to $n$ do
        CV[i][j] = 0
while 1 do
    maxDegree = -1, maxCV = -1, u = -1, c = -1
    for $i \leftarrow 1$ to $n$ do
        if color[i] not equals -1 then
            continue
        if CV[i][0] > maxCV || (CV[i][0] == maxCV  degree[i] > maxDegree then
            maxCV = CV[i][0]
            maxDegree = degree[i]
            u = i
    if u equals -1 then
        break
    for $v \leftarrow 1$ to $n$ do
        if adj[u][v] equals 0 then
            continue
        uc = color[v]
        if uc not equals -1 then
            used[uc] = $true$
    for $i \leftarrow 1$ to $n$ do
        if used[i] not equals $true$ then
            c = i
            break
    color[u] = c
    for $v \leftarrow 1$ to $n$ do
        used[v] = $false$
        if adj[u][v] == 0 || color[v] != -1 || CV[v][c] == 1 then
            continue
        CV[v][c] = 1
        CV[v][0] = CV[v][0]+1
```

6.3. Experimental Result

The 5 algorithms mentioned in the methodology section have been applied on the data of Table-3. The report generated by First Fit algorithm, Welsh Powel algorithm, Largest Degree Ordering algorithm, Incidence Degree Ordering algorithm and DSatur algorithm is presented in Table-5, Table-6, Table-7, Table-8, Table-9 respectively for dataset 1.

For dataset-2 the results are presented in Table-5, Table-6, Table-7, Table-8, Table-9 respectively for First Fit algorithm, Welsh Powel algorithm, Largest Degree Ordering algorithm, Incidence Degree Ordering algorithm and DSature algorithm.

Table 3. Data of course scheduling of Spring-2021, MIST

Course Code	Contact Hour	Teacher					
		Teacher-1	Teacher-2	Teacher-3	Teacher-4	Teacher-5	Teacher-6
101	3	1	2				
201	3	3	4				
202	3	3	4	5	6	7	
203	3	8					
204	3	9	3	1	10	11	12
205	3	13	10				
206	3	13	10	14	5	11	15
301	3	16	17				
302	3	16	17	18	11	19	
303	3	20	19				
304	3	21	20	19			
305	4	22	23				
306	3	22	23	2	16	3	24
317	3	25	26				
318	3	25	26	1	27		
323	3	5	18				
401	3	28	14				
402	3	28	14	13	15	27	
403	3	29	30				
404	3	29	30	13	24	27	
405	3	9	31				
460	3	32	22	4	18	12	
462	3	17	9	13	5	31	12
421	3	32	21	23			

Table-10 and Table-16 illustrate a comparison of these 5 graph coloring algorithms by their penalty for dataset-1 and dataset-2 respectively. In both of cases, Incidence Degree ordering achieves the highest penalty. The main reason for the maximization of the penalty is that IDO takes the highest number of slots and also generates the maximum amount of workload for the teachers in both of the cases. Though for dataset-2 IDO takes the same number of slots as the chromatic number for the dataset-1 it exceeds the chromatic number.

The most optimal algorithm for the considered scenario is both Welsh Powel and Largest Degree Ordering algorithm as they generate the lowest amount of penalty in both of the cases. The main reason behind the minimization of the penalty is adjusting the number of required slots with exactly the chromatic number of the corresponding graph. At the same time both of them generates the same amount of workload for the teachers which is minimum for both of the scenarios.

Regarding the First fit and DSatur algorithm the satisfaction level varies depending on the dataset. For the first dataset, DSatur performs better but for the second dataset the inverse relationship occurs. But the good side of DSatur is that it takes exactly X(G) number of slots in both of the cases where First Fit algorithm fails to adjust the number of required slots for the first dataset. P1 and P3 also varies in both of the scenarios.

The study found that all five algorithms were able to produce schedules with minimal conflicts in certain conditions, however, analyzing the penalty of each algorithm we can say that the schedules produced by Welsh Powell and DSATUR had a lower penalty than the others. This implies that Welsh Powell and DSATUR are more successful in resolving the university timetable scheduling problem. Figure 2 illustrates a comparison of penalty value for the mentioned 5 algorithms.

The methodology employed in the study helped to accomplish the objective of comparing the performance of graph coloring algorithms in solving the university timetable scheduling problem. By examining the algorithms using bench- mark problems and comparing the schedules generated, we were able to determine the most efficient algorithms for this problem.

Table 4. Data of course scheduling of Fall-2021, MIST

Course Code	Contact Hour	TID						
		Teacher-1	Teacher-2	Teacher-3	Teacher-4	Teacher-5	Teacher-6	Teacher-7
103	3	1	26					
104	3	1	26	21	25			
105	3	2	27					
106	3	2	27	14	17	19	20	
211	3	12						
212	3	12	24	11	14			
214	3	17	8	24	30	7		
215	3	19	25					
216	3	19	25	18	30			
217	3	18	20					
220	3	16	8	24	28			
224	3	23	27	29				
307	3	5	18					
308	3	5	18	23				
309	3	7	13					
310	3	7	13	9	16	19	26	
313	3	19	24					
315	3	11	16					
316	3	11	16					
319	3	10	21					
360	3	4	9	10	15	3	21	26
413	3	15	17					
414	3	15	17					
415	3	6	22					
416	3	6	20	22				
429	3	9	23					
443	3	3	10					
444	3	3	10	29	30			
451	3	14						
452	3	14	21					
417	3	31						

Table 5. Report generated by First Fit Algorithm

	A	B	C	D	E	F
SUN	101 201 301 401	101 201 301 401	101 201 301 401	202 302 402	202 302 402	202 302 402
MON	203 303 403	203 303 403	203 303 403	204 304 404	204 304 404	204 304 404
TUE	205 305 405	205 305 405	205 305 405	206 305	206 306	206 306
WED	306 462	318 460	318 460	318 460	323 421	323 421
THU	323 421	462	462	X	X	X

Chormatic number for the graph consisted from the data of Table-3 is 25. The value of penalty is illustrated in Table-4 for the performance comparison of these 5 algorithms.

7. Conclusions

The comparison of various graph coloring algorithms in university timetable scheduling is comparatively a novel area of research. The five algorithms employed in this study, namely First Fit, Welsh Powell, Largest Degree Ordering, Incidence Degree Ordering and DSATUR, while not guaranteed to yield optimal solutions, do so in a polynomial time but ensuring a conflict-free outcome. However, it must be acknowledged that the study is constrained by the limitations of the benchmark problems employed which may not fully encapsulate real-world scenarios. Additionally, the study's scope is limited to the comparison of these five algorithms alone. Future studies could expand the benchmark problem set and incorporate a wider range of graph coloring algorithms for a more comprehensive evaluation of their performance.

Table 6. Report Generated by Welsh Powell Algorithm

	A	B	C	D	E	F
SUN	203 306 462	203 306 462	203 306 462	206 318 460	206 318 460	206 318 460
MON	204 323 404	204 323 404	204 323 404	101 202 302 421	101 202 302 421	101 202 302 421
TUE	201 305 402	201 305 402	201 305 402	205 305 401	205 301 401	205 301 401
WED	301 405	304 405	304 405	304 403	303 403	303 403
THU	303	X	X	X	X	X

Table 7. Report Generated by Largest Degree Ordering Algorithm

	A	B	C	D	E	F
SUN	203 306 462	203 306 462	203 306 462	206 318 460	206 318 460	206 318 460
MON	204 323 404	204 323 404	204 323 404	101 202 302 421	101 202 302 421	101 202 302 421
TUE	201 305 402	201 305 402	201 305 402	205 305 401	205 301 401	205 301 401
WED	301 405	304 405	304 405	304 403	303 403	303 403
THU	303	X	X	X	X	X

Furthermore, incorporating other optimization techniques and constraints may also be considered to enhance the quality of the schedules generated. Additionally, the lack of a heuristic to adjust soft constraints and uniform time slot distribution over courses can be considered as potential avenues for future research.

Table 8. Report Generated by Incidence Degree Ordering Algorithm

	A	B	C	D	E	F
SUN	203 306 401	101 201 301 401	101 201 301 401	101 201 301 402	202 302 402	202 302 402
MON	202 302 403	203 303 403	203 303 403	204 303 404	204 304 404	204 304 404
TUE	205 304 405	205 305 405	205 305 405	206 305	206 305	206 306
WED	306 462	318 460	318 460	318 460	323 421	323 421
THU	323 421	462	462	X	X	X

In conclusion, this study has established the efficacy of graph coloring algorithms in university timetable scheduling, with the Welsh Powell and DSATUR algorithms being particularly efficacious. Nonetheless, further research is imperative to fully comprehend the capabilities and limitations of these algorithms in addressing real-world scheduling problems.

Table 9. Report Generated by Dsatur Algorithm

	A	B	C	D	E	F
SUN	203 306 462	203 306 462	203 306 462	206 318 460	206 318 460	206 318 460
MON	204 323 421	204 323 421	204 323 421	101 202 302 404	101 202 302 404	101 202 302 404
TUE	201 305 402	201 305 402	201 305 402	205 305 401	205 301 401	205 301 401
WED	301 405	304 405	304 405	304 403	303 403	303 403
THU	303					

Table 10. Table for Algorithm Wise Penalty for Dataset-1

A	X(G)	|SC|	|TL|	P1	P2	P3	TP
FF	25	11	15	2	27	94	7.61
WP	25	11	15	0	25	93	7.2
LDO	25	11	15	0	25	94	7.27
IDO	25	11	15	4	27	97	7.99
DSatur	25	11	15	0	25	93	7.2

Table 11. Report Generated by First Fit Algorithm

	A	B	C	D	E	F
SUN	103 211 307 413	103 211 307 413	103 211 307 413	104 212 308 414	104 212 308 414	104 212 308 414
MON	105 214 315 415	105 214 315 415	105 214 315 415	106 220 309 429	106 220 309 429	106 220 309 429
TUES	215 316 416	215 316 416	215 316 416	216 319 451	216 319 451	216 319 451
WED	217 310 443	217 310 443	217 310 443	224 313 452	224 313 452	224 313 452
THU	360 417	360 417	360 417	444	444	444

Table 12. Report Generated by Welsh Powell Algorithm

	A	B	C	D	E	F
SUN	106 220 360 415	106 220 360 415	106 220 360 415	105 212 310 444	105 212 310 444	105 212 310 444
MON	103 214 308 452	103 214 308 452	103 214 308 452	216 319 429	216 319 429	216 319 429
TUES	104 224 313 413	104 224 313 413	104 224 313 413	215 307 414	215 307 414	215 307 414
WED	217 315 443	217 315 443	217 315 443	211 316 416	211 316 416	211 316 416
THU	309 451	309 451	309 451	417	417	417

Table 13. Report Generated by Largest Degree Ordering Algorithm

	A	B	C	D	E	F
SUN	106 220 360 415	106 220 360 415	106 220 360 415	105 212 310 444	105 212 310 444	105 212 310 444
MON	103 214 308 452	103 214 308 452	103 214 308 452	216 319 429	216 319 429	216 319 429
TUES	104 224 313 413	104 224 313 413	104 224 313 413	215 307 414	215 307 414	215 307 414
WED	217 315 443	217 315 443	217 315 443	211 316 416	211 316 416	211 316 416
THU	309 451	309 451	309 451	417	417	417

Table 14. Report Generated by Incidence Degree Ordering Algorithm

	A	B	C	D	E	F
SUN	106 211 307 415	103 211 307 413	103 211 307 413	103 212 308 413	104 212 308 414	104 212 308 414
MON	104 214 315 415	105 214 315 415	105 214 315 416	105 215 309 414	106 220 309 429	106 220 309 429
TUES	215 316 416	215 316 416	216 316 429	216 319 451	216 319 451	217 310 443
WED	217 310 443	217 310 443	220 319 451	224 313 452	224 313 452	224 313 452
THU	360 417	360 417	360 417	444	444	444

Table 15. Report Generated by Dastur Algorithm

	A	B	C	D	E	F
SUN	106 220 308 444	106 220 308 444	106 220 308 444	105 212 310 413	105 212 310 413	105 212 310 413
MON	103 214 307 452	103 214 307 452	103 214 307 452	216 360 416	216 360 416	216 360 416
TUES	104 224 313 414	104 224 313 414	104 224 313 414	215 319 429	215 319 429	215 319 429
WED	217 315 443	217 315 443	217 315 443	211 316 451	211 316 451	211 316 451
THU	309 415	309 415	309 415	417	417	417

Table 16. Table for Algorithm Wise Penalty for Dataset-2

A	X(G)	|SC|	|TL|	P1	P2	P3	TP
FF	30	15	31	0	30	90	3.9
WP	30	15	31	0	30	87	3.81
LDO	30	15	31	0	30	94	4.03
IDO	30	15	31	6	30	94	4.43
DSatur	30	15	31	0	30	87	3.81

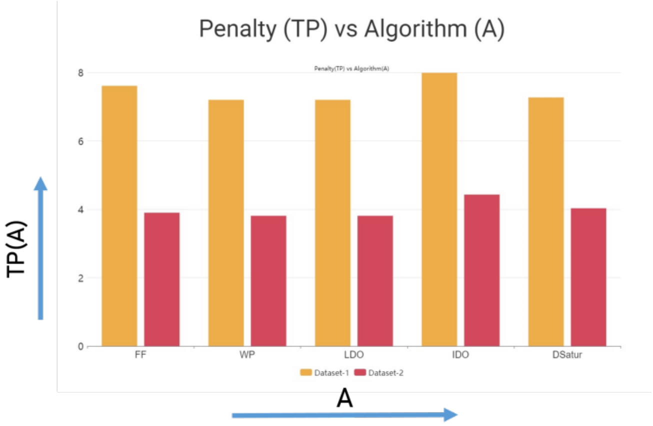


Fig.2. Dataset Wise Penalty of 5 Graph Coloring Algorithms

References

[1] Sadaf Naseem Jat and Shengxiang Yang. A hybrid genetic algorithm and tabu search approach for post enrolment course timetabling. Journal of Scheduling, 14(6):617–637, 2011.

[2] Wang Wen-jing. Improved adaptive genetic algorithm for course scheduling in col- leges and universities. International Journal of Emerging Technologies in Learning, 13(6), 2018.

[3] Raneem Gashgari, Lamees Alhashimi, Raed Obaid, Thangam Palaniswamy, Lujain Aljawi, and Abrar Alamoudi. A survey on exam scheduling techniques. In 2018 1st International Conference on Computer Applications Information Security (ICCAIS), pages 1–5. IEEE, 2018.

[4] D aniel Marx. Graph colouring problems and their applications in scheduling. Pe- riodica Polytechnica Electrical Engineering (Archives), 48(1-2):11–16, 2004.

[5] Dominic JA Welsh and Martin B Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. The Computer Journal, 10(1):85–86, 1967.

[6] Murat Aslan and Nurdan Akhan Baykan. A performance comparison of graph coloring algorithms. International Journal of Intelligent Systems and Applications in Engineering, pages 1–7, 2016.

[7] Narayan Poddar and Basudeb Mondal. An instruction on course timetable schedul- ing applying graph coloring approach. International Journal of Recent Scientific Research, 9(2):23939–23945, 2018.

[8] Runa Ganguli and Siddhartha Roy. A study on course timetable scheduling us- ing graph coloring approach. international journal of computational and applied mathematics, 12(2):469–485, 2017. Graph coloring in university timetable scheduling: A comparative study 23

[9] Walter Klotz. Graph coloring algorithms. Verlag nicht ermittelbar, 2002.

[10] Akhan Akbulut and G uray Yilmaz. University exam scheduling system using graphcoloring algorithm and rfid technology. International Journal of Innovation, Management and Technology, 4(1):66, 2013.

[11] Kristoforus Jawa Bendi, Theresia Sunarni, and Achmad Alfian. Using graph color- ing for university timetable problem. International Journal of Science and Research (IJSR), 7:1692–1697, 11 2018.

[12] NK Cauvery. Timetable scheduling using graph coloring. International Journal of P2P Network Trends and Technology, 1(2):57–62, 2011.

[13] P Nandal, Ankit Satyawali, Dhananjay Sachdeva, and Abhinav Singh Tomar. Graph coloring based scheduling algorithm to automatically generate college course timetable. In 2021 11th International Conference on Cloud Computing, Data Sci- ence Engineering (Confluence), pages 210–214. IEEE, 2021.

[14] J Akbari Torkestani and MR Meybodi. Graph coloring problem based on learning automata. In 2009 International Conference on Information Management and Engineering, pages 718–722. IEEE, 2009.

[15] Timothy A Redl. University timetabling via graph coloring: An alternative ap- proach. Congressus Numerantium, 187:174, 2007.

[16] Zafer Bozyer, M Sinan Ba sar, and Alper Aytekin. A novel approach of graph coloring for solving university course timetabling problem. Proceeding Number, 300:23, 2011.

[17] Sara Miner, Saleh Elmohamed, and Hon W Yau. Optimizing timetabling solutions using graph coloring. NPAC, Syracuse University, pages 99–106, 1995.

[18] EK Burke, DG Elliman, and R Weare. A university timetabling system based on graph colouring and constraint manipulation. Journal of research on computing in education, 27(1):1–18, 1994.

[19] Tansel Dokeroglu and Ender Sevinc. Memetic teaching–learning-based optimiza- tion algorithms for large graph coloring problems. Engineering Applications of Artificial Intelligence, 102:104282, 2021.

[20] Hussein Al-Omari and Khair Eddin Sabri. New graph coloring algorithms. Amer- ican Journal of Mathematics and Statistics, 2(4):739–741, 2006.

[21] Linda Ouerfelli and Hend Bouziri. Greedy algorithms for dynamic graph coloring. In 2011 International Conference on Communications, Computing and Control Applications (CCCA), pages 1–5. IEEE, 2011.

[22] NS Narayanaswamy and R Subhash Babu. A note on first-fit coloring of interval graphs. Order, 25(1):49–53, 2008.

[23] Stavros D Nikolopoulos and Charis Papadopoulos. On the performance of the first-fit coloring algorithm on permutation graphs. Information Processing Letters, 75(6):265–273, 2000.

[24] Dominic JA Welsh and Martin B Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. The Computer Journal, 10(1):85–86, 1967.

[25] Dahlan Abdullah, Marsali Yaton, Heri Sujatmiko, Sepyan Purnama Kristanto, Hendra Nazmi, Irma Lisa Sridanti, Andang Suhendi, Abdurrozzaq Hasibuan, Renny Kurniawati, Darmadi Erwin Harahap, et al. Lecture scheduling system us- ing welch powell graph coloring algorithm in informatics engineering departement of universitas malikussaleh. In Journal of Physics: Conference Series, volume 1363, page 012074. IOP Publishing, 2019.

[26] J Arthy and J Aiswarya. Classification of algorithms and largest degree ordering.

[27] Michael W Carter. Or practice—a survey of practical applications of examination timetabling algorithms. Operations research, 34(2):193–202, 1986. 24 Authors Suppressed Due to Excessive Length

[28] William Hasenplaugh, Tim Kaler, Tao B Schardl, and Charles E Leiserson. Or- dering heuristics for parallel graph coloring. In Proceedings of the 26th ACM sym- posium on Parallelism in algorithms and architectures, pages 166–177, 2014.

[29] Daniel Br elaz. New methods to color the vertices of a graph. Communications of the ACM, 22(4):251–256, 1979.

[30] Pablo San Segundo. A new dsatur-based algorithm for exact vertex coloring. Com- puters Operations Research, 39(7):1724–1733, 2012.

[31] Anuj Mehrotra and Michael A Trick. A column generation approach for graph coloring. informs Journal on Computing, 8(4):344–354, 1996.

Authors' Profiles

Swapnil Biswas, Lecturer, department of Computer Science and Engineering at Mil- itary Institute of Science and Technology (MIST) was born in Dhaka, Bangladesh in 1998. He rcompleted his B.Sc. in Computer Science and Engineering from Military Institute of Science and Technology (MIST), Dhaka, Bangladesh. Currently he is pursuing his M.Sc. in Computer Science and Engineering from the same institution. His interest in research includes Graph Theory, Data Structures and Algorithms.

Syeda Ajbina Nusrat, Lecturer, department of Computer Science and Engineering at University of Information Technology and Sciences (UITS) was born in Dinajpur, Bangladesh in 1998. She received her B.Sc. in Computer Science and Engineer- ing from Military Institute of Science and Technology (MIST), Dhaka, Bangladesh. Currently she is pursuing her M.Sc. in Computer Science and Engineering from the same institution. Her interest in research includes Blockchain, Machine Learning and Graph Theory.

Dr. Nusrat Sharmin, female, assistant professor at the Military Institute of Science and Technology. She did her Ph.D. in machine learning in neuroimaging and her master's thesis was based on computer vision and image processing. Her research interests include digital image processing, machine learning, and combinatorial op- timization problem. Her teaching interests include digital image processing, pattern recognition, and information and system design.

Dr. Md. Mahbubur Rahman, is working as professor at department of Computer Science and Engineering at Military Institute of Science and Technology. He did his Ph.D. in Information Science/Computer Science, 2004 from Japan Advanced Institute of Science and Technology (JAIST), Japan. His research interests include Image Processing, Network Security, Pattern Recognition, Health Informatics, AI and Ma- chine Learning, Data Analytics, Bioinformatics.

How to cite this paper: Swapnil Biswas, Syeda Ajbina Nusrat, Nusrat Sharmin, Mahbubur Rahman, "Graph Coloring in University Timetable Scheduling", International Journal of Intelligent Systems and Applications(IJISA), Vol.15, No.3, pp.16-32, 2023. DOI:10.5815/ijisa.2023.03.02