An Application of Graph Coloring Model to Course Timetabling Problem

Wathsala Samarasekara

Department of Information Technology, Sri Lanka Institute of Advanced Technological Education, Sri Lanka

Abstract: ***This paper studies the problem of developing an automated timetable using graph coloring model which is applicable to course timetabling problem in higher education institutes.Constructing a satisfactory conflict-free semester-long timetable of courses is often a difficult problemthat any institute faces each semester. Construction of such type of timetable can be modeled as a graph coloring problem. In a graph coloring model, the timetabling problem is usually represented as a graph where events are represented as vertices, while conflicts between the events are represented by edges. Edges of a graph are colored and each time slot in the timetable corresponds to a color in the graph coloring problem. After finding time slots, the possible courses can be conducted in each time slot are assigned according to the graph. Finally, separate timetables for each academic year are filled by assigning each subject for available time periods according to the proposed algorithm. The automated timetabling system is based on the proposed algorithm of the graph coloring model. The proposed system can generate not only one solution for the timetabling problem, but also many possible solutions. Therefore this timetabling system exhibits a much better quality of solution.***

Keywords: Timetabling Problem, Graph Coloring, Algorithm, Automated

1. Introduction

The Timetabling problem consists of allocating a number of courses to a limited set of resources such as time slots, set of lecturers, lecture rooms, labs and group of students to satisfy predefined constraints. The constraints can be divided into two groups: hard constraints and soft constraints. A timetable has to satisfy all hard constraints in order to be feasible and it should satisfy as much as possible soft constraints in order to be good quality.

The SLIATE is one of the leading educational institutions in Sri Lanka for higher education which conduct the courses of Higher National Diplomas and National Diplomas. It is mandated to establish ATIs' in every province. ATI Kurunegala is one of them where the study has been applied. There are nearly 600students, 15 lecturers and four departments namely HNDIT, HNDM, HNDE and HNDA. Constructing a satisfactory conflict-free semester-long timetable of courses is often a difficult problem that management of ATI's faces each semester as it takes more days.

The approach of this research is to develop a mathematical model for solving ATI Timetabling problem using the Graph Coloring techniques that incorporates the satisfaction of both hard constrains (i.e. constraints that must be satisfied in order to produce a feasible timetable) as well as soft constraints (i.e. additional constraints that need not necessarily satisfied to produce a legitimate timetable, but if satisfied may very well produce a more acceptable timetable for lecturers and/or students).

It is mainly considered courses, subjects for each course, lecturers for each subject, available lecture hours and practical/tutorial hours for each subject, lecture rooms, lecture room capacity and number of students for each subject for the relevant semester in each academic year. In addition to that lecture preferences are also considered.

Finally, a system is developed to implement the model as automated timetable. Using this system, we can generate feasible timetable/timetables within considerable time duration.

2. Related Work

During the last thirty years, many researchers have been widely studied the timetabling problems in the educational sector and many papers related to automated timetabling have appeared in conferences and journals. In addition, several applications have been developed and employed with a quite good success.

Many researchers have been proposed and developed algorithms for solving course scheduling problems [2, 4, 5, 7]. One researcher[7] uses graph Coloring approach and presented a "largest degree first: fill from top" examination scheduling algorithm. The objective of this algorithm [7] is to assign m courses in n time periods while not scheduling more than a predefined number of students during any one time period. The algorithm[7] scans the list of exams and then places all the exams as possible in the first time slot (lowest numbered color). Then go back to the top of the list again and fill the second time slot, and so on. In this research [7], the researcher has been considered only two parameters and the proposed algorithm can be applied only for examination scheduling problem.

Another two researchers [8] introduced plans for a university timetabling system based on graph Coloring and constraint manipulation. The authors also discussed graph Coloring and room allocation heuristic algorithms and the handling of several common timetabling features within the system. This study has been shown that graph Coloring can be used to find better solution for timetabling problems with many constraints.

There is a research called "A Study of University timetabling that Blends Graph Coloring with the Satisfaction of Various Essential and Preferential Conditions". In this research [9] relevance of university timetabling problems as a natural and practical application of graph coloring has been discussed and also developed a new mathematical and computational model for solving university timetabling problems using graph Coloring.

Some researchers have also developed several metaheuristics for solving the timetabling problems. For example, a research work has been found using Tabu Search algorithm (a metaheuristic local search algorithm) for large-scale timetabling problems. [10] And also another research has been found using genetic algorithm in the course scheduling problems. [7]

Genetic algorithm also can be used to solve timetabling problems with many parameters and constraints. But graph Coloring can be selected as the suitable approach for this problem by considering number of parameters, relevancy of those parameters in my problem and complexity of the model. Therefore edge Coloring approach of graph Coloring is selected as the most suitable approach for this research work.

3. Problem Statement

Presently, the ATI timetable is created manually by HODs'. Using the manual way to generate workable timetables is not preferable in this case as it takes more time with the problem size increases. Therefore it is needed an automated timetable for this purpose, because generating timetables through automatic methods seems to be attractive, effective and alternative to manual approach.

The ATI timetabling problem consists of a set of courses that are to be scheduled into a set of time periods and a set of lecture rooms and also consist of a set of constraints that are expected to be satisfied.

The main goal of creating a timetable is to provide lecturers and students with a schedule that is not only conflict-free but also has a good quality. Different aspects of this goal in terms of lecturers, students and labs are investigated as follows. From lecturers' point of view, the resulting timetable should be conflict-free at first and then if there are preferences for some time periods, it should be satisfied as much as possible. From students' point of view, the timetable must be conflict-free for all groups of students. It should minimize idle hours between subjects in a day, but it should give 30 minutes break for lunch between 12.30 p.m. and 1.00 p.m. Finally, from lecture rooms' point of view, the student group size should fit the lecture rooms' capacity and two different subjects should not be scheduled at the same time in a lecture room.

4. Proposed Methodology

4.1Overview of the Methodology

Many problems in timetabling can be solved using Graph Coloring as it is easy to determine the minimum number or a reasonable number of time slots needed to schedule all the subjects to restrictions.

The model involves creating a conflict graph from the assembled input course data, properly coloring the conflict graph using edge-Coloring, and transforming this coloring into a conflict-free timetable.

Therefore, to create this model, initially I have selected graph Coloring approach. After creating the graph, it was a bipartite graph. Therefore, I selected the edge Coloring approach of bipartite graph as it was the most suitable approach to create this model.

The heart of the problem is the constraint that is there exist restrictions on which courses can be scheduled simultaneously. These constraints can be identified as hard constraints and soft constraints.

Hard Constraints:

- Two or more subjects taught by the same lecturer cannot be scheduled for the same time slot.
- Subjects which enroll the same set of students cannot be scheduled for the same time slot. (Conflict- free schedules for years one and two)
- Each subject must be scheduled for exactly one time slot and one location (class room, laboratory or audio visual room), to remain constant throughout the scheduling period or semester.
- Each subject must be scheduled in an available location (class room, laboratory or audio visual room) that can accommodate its size.
- Two or more subject that requires the same location cannot be scheduled for the same time slot.
- Two or more subjects scheduled for the same time slot cannot be assigned the same location.
- Some subjects are required to meet a certain fixed number of times per week (for example, 3 times per week, 2 times per week, or 1 time per week).
- Predefined location requirements or type for a particular subject, if any (for example, a computer lab or audio visual room with projection screen) should be taken into account.

Soft constraints:

- A lecturer may have a time preference as to when a subject is scheduled to meet (for example, in the morning or in the afternoon).
- A lecturer may have a specific location (for example, a computer lab or audio visual room with projection screen) request for a subject, beyond the scope of the outlined location requirements specified above.
- Rooms (or locations) should be just large enough to hold the courses in them, in order to eliminate the presence of unused empty space.
- Try to schedule sufficiently large subjects at times that "minimize pain for students".

Some other assumptions present in our timetabling model will be introduced and described in subsequent sections of this chapter.

Let's take a simpler version of the timetabling problem according to our model.

Let L1, L2, . . . , Lm are m lecturers and S1, S2, . . . , Sn are n subjects to teach. Given that lecturer Li is required to teach subject Sj for hij hours. The teaching requirements are represented first by an m × n matrix H = [hij] shown in figure 1, where the entry hij is the number of periods Lecture Li is required to teach Subject Sj. One period consists of two hours as it is assumed that the continuous two hours lecture period is effective for students. Then the matrix H is transformed into a bipartite graph G with bipartition (L, S), where L = {L1,L2, . . . ,Lm}, S = {S1, S2, . . . , Sn}, and vertices Li and Sj are joined by hij edges.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \cdots & h_{2n} \\ h_{31} & h_{32} & h_{33} & \cdots & h_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{m1} & h_{m2} & h_{m3} & \cdots & h_{mn} \end{bmatrix}$$

Figure 1: Teaching requirements matrix

Figure 2, figure 3 and table 1 below illustrate a solution to a 3-Lecturer, 5-Subjects example where
L = {L1, L2, L3}, S = {S1, S2, S3, S4, S5}, and

$$H = \begin{bmatrix} 2 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

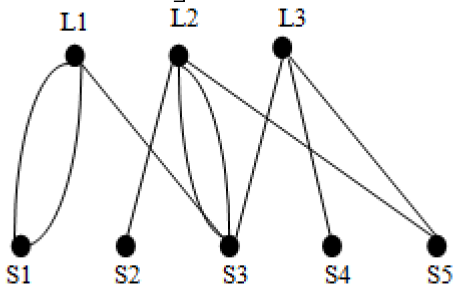


Figure 2: A Bipartite graph G (3-Lecturer, 5-Subjects timetabling example)

In the bipartite graph G in figure 2, the vertex set S represents subjects and the vertex set L represents lecturers. Edges represent each conflicting pairs of subject and its lecturer according to lecture hours. Then we can Color the graph G using edge-Coloring approach as shown in figure 3 applying the König's theorem.

Theorem 1: If G is bipartite, then $\chi'(G) = \Delta(G)$, where $\Delta(G)$ denotes the maximum degree of G and $\chi'(G)$ is the edge chromatic number. [1]

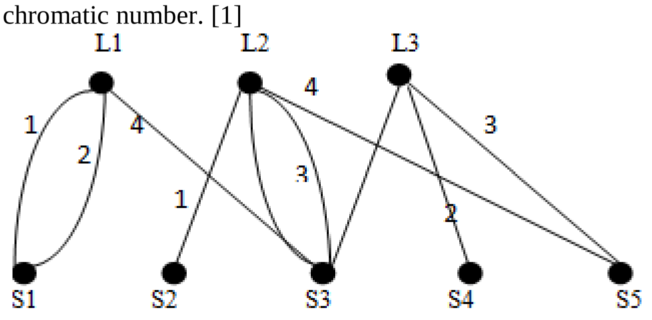


Figure 3: A Edge-Coloring graph G (3-Lecturer, 5-Subjects timetabling example)

Table 1: A 4-period Timetable (3-Lecturer, 5-Subjects time tabling example)

Subject Code / Lecturer	IT2001	T2002	IT2003	IT 2004	IT2005	IT2006	IT4001	IT4002	IT4003	IT 4004	IT 4101	IT 4102	IT 4103
L1	6	-	-	-	-	4	-	6	-	-	-	-	-
L2	-	6	-	4	-	-	-	-	2	-	-	-	4
L3	-	-	-	-	4	-	-	-	-	6	-	4	-
L4	-	-	4	-	-	-	2	-	-	-	6	-	-

To Color the edges "largest degree first" method is selected as it gives minimum number of Colors quickly and easily. According to the figure 3, the edges with the same Color represent the subjects that can carry out in parallel.

Then according to the Colored graph, subjects are assigned to two hour timeslots as in table 1. One Color represents one time slot.

Table 2: Teaching requirements

Time slot (color) / Lecturer	1	2	3	4
L1	S1	S1	-	S3
L2	S2	S3	S3	S5
L3	S3	S4	S5	-

From the edge-coloring of bipartite graph G and its associated timetable we conclude that four periods are both necessary and sufficient to schedule a conflict-free timetable for the above example.

In ATI timetable, there are twenty periods (each period consists of two hours) from Monday to Friday. And also when we consider one department (for example, department of Information Technology), there are two major groups of students as first year and second year. Therefore subjects should be assigned for two timetables, first year and second year, by considering those two groups of students.

After obtaining this timetable we can then assign courses to classrooms, audio visual rooms and laboratories based on room capacity and availability.

4.2Assembling Course Data

As mentioned in chapter 4.1, we assume that we have a set of subjects for each course to be offered during a particular semester by particular lectures and that their subjects are assigned to be appropriate time periods considering a number of essential & preferential time tabling conditions in our time tabling model.

In this study, I have selected second semester in HNDIT department as a sample.

4.3 Creating a Graph

Once we have gathered all of the necessary ATI course data for a given semester as described above, as a trial, we have tabulated the necessary data as shown in table 2 to obtain the teaching requirement matrix (figure 4).

$$\begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 2 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 3 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 3 & 0 & 0 \end{bmatrix}$$

Figure 4: Teaching requirement matrix

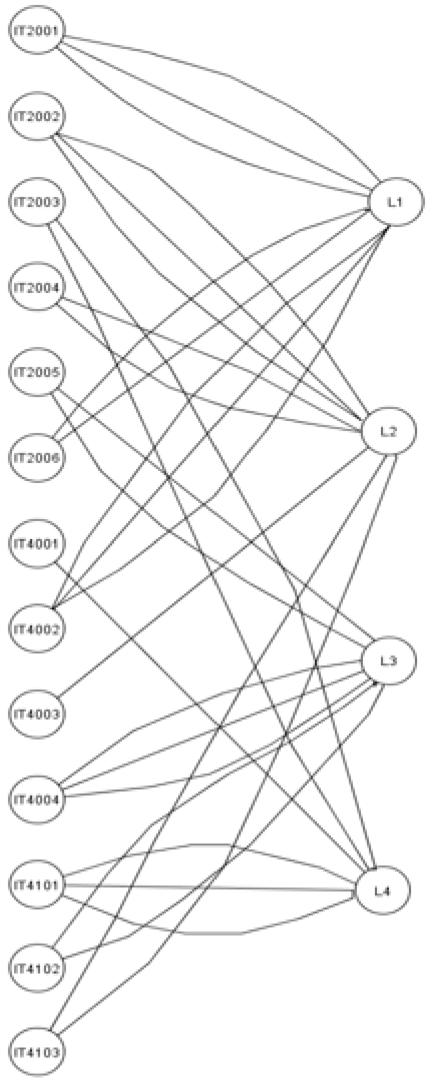


Figure 5: A conflict bipartite graph

Once the teaching requirement matrix is formed, the conflict bipartite graph can be constructed with the help of it for the selected course. Because constructing one graph for all courses manually is more complex. But using the automated system we can construct one graph for all courses or separate graphs for each course such as HNDIT, HNDM etc. In the conflict bipartite graph as shown in figure 5, one vertex set represents subjects (dividing lectures, tutorials and practical as separate subjects and using sub_codes as the names of vertices) and the other vertex set represents lectures (using lec_ids as names of vertices). Edges connect each conflicting pairs of lectures and their subjects according to lecture hours. Since the lecture hours of each subject in SLIATE is multiple of two (i.e. minimum lecture hours of a subject is two), each edge in our graph represents two hours. With this practice we can insert subjects into time periods easily. Then we can minimize pain for students to learn subjects as we believe that the effective teaching hours of a subject is two.

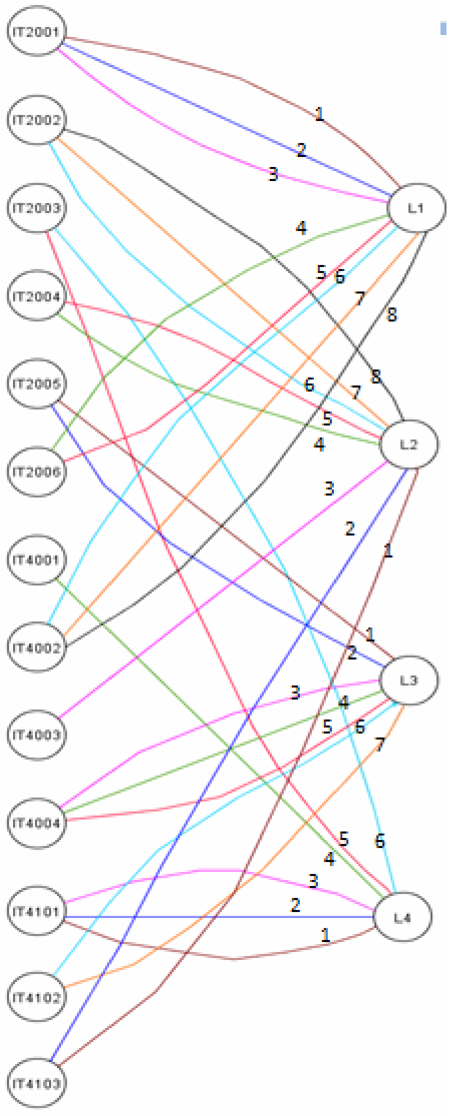


Figure 6: An Edge-Coloring bipartite graph

4.4 Coloring the Graph

Once constructing the conflict graph as described above, we can properly Color its edges using edge Coloring approach. Recall that an edge Coloring of a graph G is a Coloring of the edges of G so that adjacent edges receive different Colors. Edges that do not adjacent with another edge may be colored with different Colors or they may be colored with same Color.

Since our graph is a bipartite graph, we used largest degree first method to Color edges to obtain minimum edge Coloring (i.e. Coloring edges using smallest possible number of Colors). We can use any method to label Colors such as naming system (for example, red, blue etc.) or numbering system (for example, 1, 2, …). I have used numbers 1, 2, . . for the colors in the graph as shown in figure 6 as it is often both conventional and convenient to use.

4.5 Transforming the Coloring to a Timetable

Once Coloring the graph, the next process in our time tabling is to transform the Colored graph to a conflict - free course time table.
In this process, as the first step, we group all the subjects that are colored using a same Color as shown in table 3.

Table 3: Summary of graph Coloring and subject

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
IT2001	IT2001	IT2001	IT2006	IT2006	IT4002	IT4002	IT4002
IT2005	IT2005	IT4004	IT4004	IT4004	IT4102	IT4102	IT2002
IT4101	IT4103	IT4003	IT4001	IT2004	IT2002	IT2002	
IT4103	IT4101	IT4101	IT2004	IT2003	IT2003		

Before allocating subjects into time periods in the time table, we have to satisfy the student constraint also. That is, subjects which enroll the same set of students cannot be scheduled for the same time slot. Therefore, according to the color groups, subjects should be assigned again into new time slots by separating the subject in the same year as shown in table 4. Then we can find minimum number of timeslots to create the timetable.

Table 4: Timeslot Allocation

1	2	3	4	5	6	7	8	9
IT2001	IT2005	IT2001	IT2005	IT2001	IT4003	IT4101	IT2006	IT2004
IT4101	IT4103	IT4103	IT4101	IT4004			IT4004	IT4001

10	11	12	13	14	15	16	17
IT2006	IT2004	IT2003	IT4002	IT4102	IT4002	IT4102	IT2002
IT4004			IT2002	IT2003	IT2002		IT4002

In ATI timetable, there are two main sessions namely morning and evening as well as a lunch break. Morning session is from 8.30 a.m. to 12.30 p.m. and evening session is from 1.00 p.m. to 5.00 p.m. Therefore there are four time periods per day and twenty time periods per week in the timetable and each time period consists of two hours.

According to the timeslots, we can assign subjects into time periods in the timetables of first year and second year while satisfying lecture and student constraints. The table 5 and table 6 show the first year and second year conflict-free timetables respectively.

Table 5: First Year Timetable

Time	*Monday*	*Tuesday*	*Wednesday*	*Thursday*	*Friday*
8.30a.m-10.30a.m.	IT2001	IT2005	IT2001	IT2005	IT2001
10.30a.m-12.30p.m.			IT2006	IT2004	IT2006
12.30p.m-1.00p.m.	Lunch Break				
1.00p.m-3.00p.m.	IT2004	IT2003	IT2002	IT2003	IT2002
3.00p.m-5.00p.m.		IT2002			

Table 6: Second Year Timetable

Time	*Monday*	*Tuesday*	*Wednesday*	*Thursday*	*Friday*
8.30a.m-10.30a.m	IT4101	IT4103	IT4103	IT4101	IT4004
10.30a.m-12.30p.m	IT4003	IT4101	IT4004	IT4001	IT4004
12.30p.m-1.00p.m	Lunch Break				
1.00p.m-3.00p.m			IT4002	IT4102	IT4002
3.00p.m–5.00p.m	IT4102	IT4002			

4.6 Development of an Algorithm

To create edges and set color_ids for edges

```
1) Begin
2) Collect all lecturers into an array L
   {L1, L2,…..,Ln}
3) Set 1 as the index of L.
4) For all lecturers Li in the array
   4.1. Collect the subjects of ith lecturer, Li into an array S
   {S1, S2,…,Sn}
   4.2. Count the degree of Li
   4.3. If the index is odd number, then
   4.3.1. Set 1 for ColorId
   4.4. Else
   4.4.1. Set degree for ColorId
   4.5. For all subjects, Si in the array
       4.5.1. Find the number of hours of Si
       4.5.2. For hours/2 of Si
       4.5.2.1. Create new edge for Si
       4.5.2.2. Set the ColorIdas the Color id of the edge
       4.5.2.3. If the index is odd number, then
       4.5.2.3.1. Set ColorId+1 for ColorId
       4.5.2.4. Else
       4.5.2.4.1. Set ColorId-1 for ColorId
       4.5.3. Repeat step 4.5.2
   4.6. Select the next subject and repeat step 4.5
       1. Set index+1 for index
       2. Select the next lecture and repeat step 4
       3.  End
```

To set time slots for edges

```
1) Begin
2) Collect all edges to an array E {E1, E2,…, En}
3)    Sort E by ColorIdand then by year of the subject of E
4) Set 1 for the time slot Ts
5) For all edges Ei in E
   5.1. If the edge is E1, then
       5.1.1. Set Ts for the timeslot of E1
       5.1.2. Set Color id of E1 as ColorId
       5.1.3. Set year of E1 as the Year
   5.2. Else if the Color id of Ei is ColorId, then
       5.2.1. If the year of Ei is Year
       5.2.1.1. Set Ts+1 for the timeslot of Ei
       5.2.2 Else
       5.2.2.1. If the number of timeslots of Tsis less than 2
       5.2.2.1.1. Set Ts for the timeslot of Ei
       5.2.2.2. Else
       5.2.2.2.1. Set Ts+1 for the timeslot of Ei
       5.2.2.2.2. Set year of Ei as the Year
5.3. Else if the Color id of Ei is not the ColorId, then
5.3.1. Set Color id of Ei as ColorId
5.3.2. Set Ts+1 for the timeslot of Ei
5.4. Set year of Ei as the Year
     1. Select the next edge and repeat step 5
     2. End
```

4.7 Obtaining Other Timetables

Once obtaining the master timetable for subject according to the lecture, we can consider location or room constraint further. That is predefined location or room requirements or type for a particular subject, if any should be taken into account.

According to the room details, there are three computer labs and two audio visual rooms. Computer labs are used for computer related practical subjects and audio visual rooms are used conduct some lectures. If any lecture has any room requirement, he/she can make a request before creating the timetable. Classrooms are available any time for each student group in any course as there are separate classrooms for them.

Assume that we need a computer lab for every practical subject in IT. To satisfy this constraint, we have to allocate any computer lab for each practical subject. Along with this we can also satisfy one of one of soft constraint that is, rooms should be just large enough to hold the courses in them, in order to eliminate the presence of unused empty space.

Thus to allocate computer labs for every practical subject, the following steps can be followed.

For the first year timetable:

```
1) Begin
2) Select the first subject, S, in the first time period of the first year timetable
3) If S has practical hours
    3.1. Count the number of practical time periods, P, and theory time periods, T, by dividing the total hours by 2
    3.2. Find the number of students, N, registered for S
    3.3. Select the first year timetable
    3.4. Ignore first T time periods of S and select the next time period, Tp
    3.5. Select the first lab, L, from lab details which has number of seats just large enough to N
    3.6. If L is free for Tp(If it is already not allocated)
        3.6.1. Allocate L for S
    3.7. Else
        3.7.1. Select the next lab, L, from lab details which has number of seats just large enough to N
        3.7.2. Repeat step 3.6
    3.8. If any lab does not free to assign S, thendivide N by 2 (to make two groups) and set it as N
        3.8.1. Repeat step 3.5
        3.8.2. Assign the same subject S to another nearest unallocated time period considering both first year and second year timetables for the other group
    3.9. Select the next Tpof S
    3.10. Repeat step 3.5
    3.11. Select the next subject, S, in the first year timetable
4) Repeat Step 3
5) End
```

The above mentioned steps can be repeated to assign laboratories for the subjects of second year timetable also. According to the steps, we can allocate labs for each practical subject as shown in table 7.

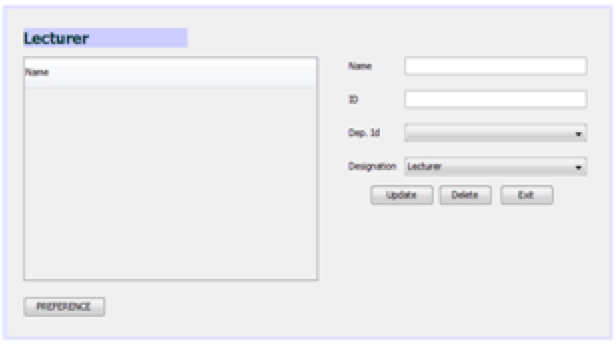
Table 7: Lab Allocation

Room Name	*Time period*	*Subject code*
Main Lab	T3	IT2001
Main Lab	T5	IT2001
Main Lab	T15	IT2002
Main Lab	T17	IT2002
Main Lab	T14	IT2003
Main Lab	T11	IT2004
Main Lab	T8	IT2006
Main Lab	T10	IT2006
Main Lab	T13	IT4002
Lab 1	T15	IT4002
Lab 1	T18	IT4002
Lab 1	T5	IT4004
Lab 1	T19	IT4004
Lab 1	T8	IT4004
Lab 1	T20	IT4004
Main Lab	T1	IT4101
Main Lab	T4	IT4101
Lab 1	T14	IT4102
Main Lab	T2	IT4103

Once allocating labs as given above, we can construct the lab time tables. The table 8 and table 9 shows the time tables for Main lab and Lab 1 respectively for the selected case.

Table 8: Main Lab Timetable

Time	*Monday*	*Tuesday*	*Wednesday*	*Thursday*	*Friday*
8.30a.m–10.30 a.m	IT4101	IT4103	IT2001	IT4101	IT2001
10.30a.m-12.30p.m			IT2006		IT2006
12.30p.m–1.00p.m.	Lunch Break				
1.00p.m.–3.00 p.m.	IT2004		IT4002	IT2003	IT2002
3.00p.m–5.00 p.m		IT2002			



Designation: Lecturer

Table 9: Lab 1 Timetable

Time	Monday	Tuesday	Wednesday	Thursday	Friday
8.30a.m-10.30 a.m					IT4004
10.30a.m–12.30p.m			IT4004		
12.30 p.m–1.00p.m	Lunch Break				
1.00 p.m–3.00p.m				IT4102	IT4002
3.00 p.m–5.00p.m			IT4002	IT4004	IT4004

4.8 System Design

System design involves a systematic and accurate approach to design of systems. It is a process of defining the architecture, components, modules, interfaces and data for a system to provide a technical solution that satisfies both the user requirements and the expectations.

The logical design of a system refers to an abstract representation of the data flows, inputs and outputs of the system and often conducted by means of modeling. The physical design of a system refers to the actual input and output processes of the system. User interface design, Data design and process design are three sub-tasks in the physical design.

4.9 System Development

In the system development phase, the actual system is constructed according to the system design. Internal development of customized systems and the creation of database systems are included in this system development process.

Appropriate platform, software and language for the implementation of the system is selected by considering the important factors such as support for frameworks, performance, object oriented support, compatibility and user friendliness.

Interface Development
User Interface is the place where the user and the system are interacting together.

Some sample user interfaces are shown in following figures.

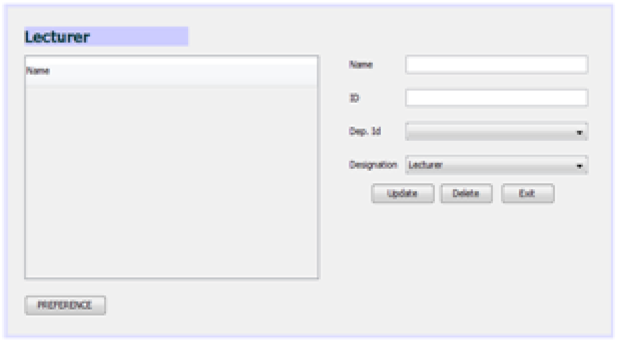


Figure 7: Interface for Lecture Information

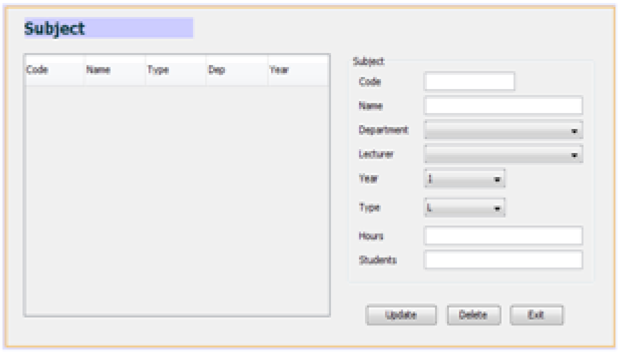


Figure 8: Interface for Subject Information

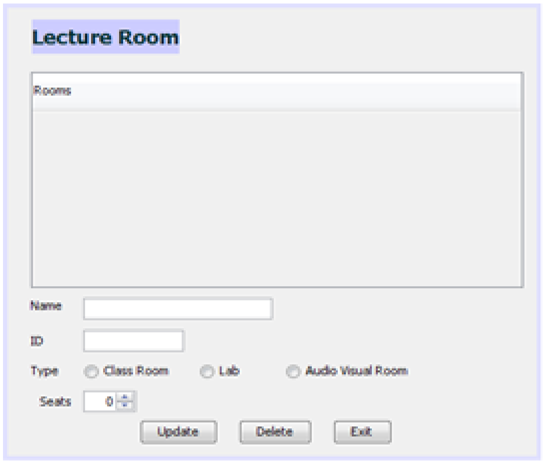


Figure 9: Interface for Lecture Room Information

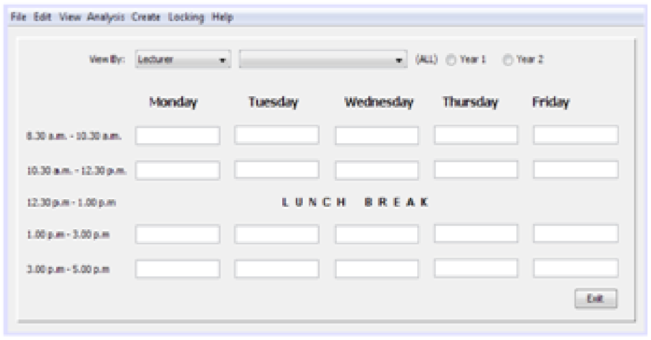


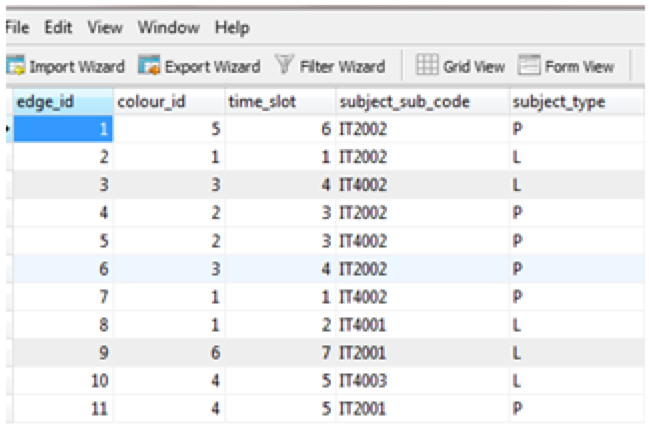
Figure 10: Main Interface for Timetable

5. Results and Discussion

5.1 Automated Timetabling Results

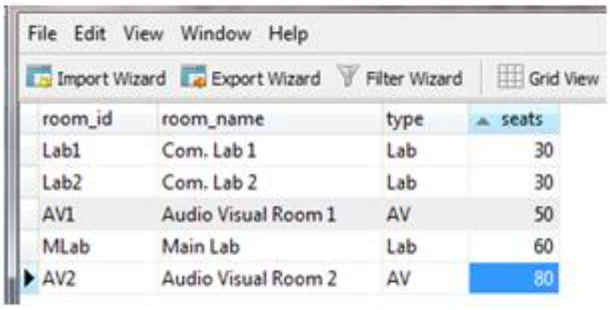
The designed algorithm has tested on part of ATI's real world data. The goal was to check whether the algorithm could actually generate acceptable timetables for ATI weekly courses. The accuracy of the result is compared with the manual solution. The following timetables show the results of two test problems as experimental suite.

In the first test problem, the program used subject and room specifications in figure 11 and figure 12 as input. The manually edge Colored bipartite graph is shown in figure 13 and the generated edge table is shown in figure 14. The system output is shown in figure 15 and figure 16.


File Edit View Window Help
Import Wizard
Export Wizard
Filter Wizard
Grid View
Form View

edge_id	colour_id	time_slot	subject_sub_code	subject_type
1	5	6	IT2002	P
2	1	1	IT2002	L
3	3	4	IT4002	L
4	2	3	IT2002	P
5	2	3	IT4002	P
6	3	4	IT2002	P
7	1	1	IT4002	P
8	1	2	IT4001	L
9	6	7	IT2001	L
10	4	5	IT4003	L
11	4	5	IT2001	P

Figure 11: Subject Table - Test Problem 1


File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View

room_id	room_name	type	seats
Lab1	Com. Lab 1	Lab	30
Lab2	Com. Lab 2	Lab	30
AV1	Audio Visual Room 1	AV	50
MLab	Main Lab	Lab	60
AV2	Audio Visual Room 2	AV	80

Figure 12: Room Table - Test Problem 1

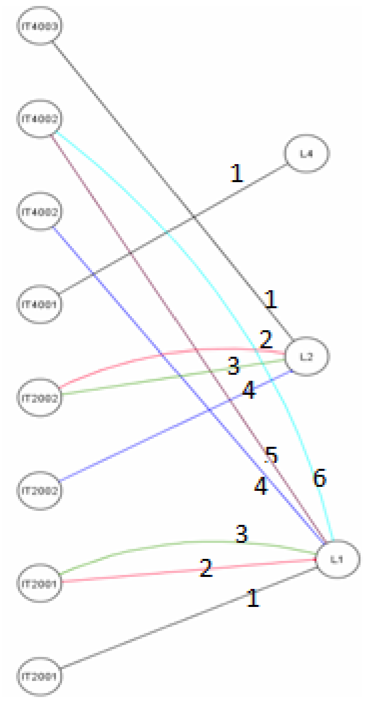
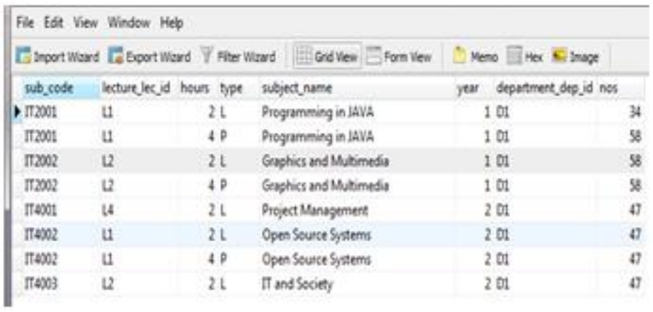


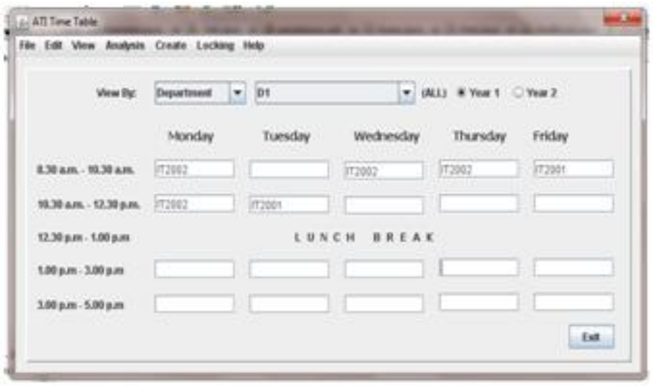
Figure 13: Edge Colored bipartite graph - Test Problem1


File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View Memo Hex Image

sub_code	lecture_lec_id	hours	type	subject_name	year	department_dep_id	nos
IT2001	L1	2	L	Programming in JAVA	1	D1	34
IT2001	L1	4	P	Programming in JAVA	1	D1	58
IT2002	L2	2	L	Graphics and Multimedia	1	D1	58
IT2002	L2	4	P	Graphics and Multimedia	1	D1	58
IT4001	L4	2	L	Project Management	2	D1	47
IT4002	L1	2	L	Open Source Systems	2	D1	47
IT4002	L1	4	P	Open Source Systems	2	D1	47
IT4003	L2	2	L	IT and Society	2	D1	47

Figure 14: Edge Table - Test Problem 1

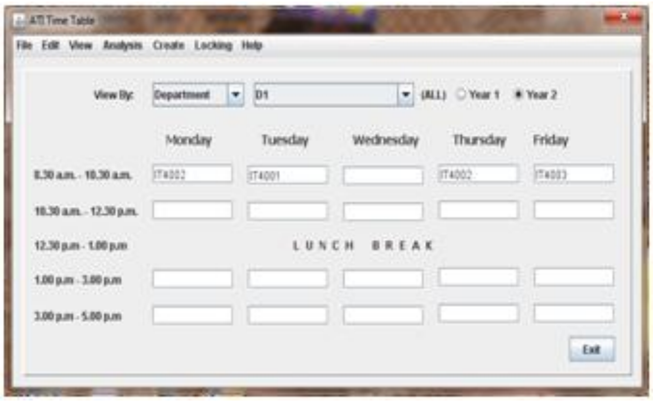
According to the figure 13, there should be six Colors to Color edges for minimum edge Coloring. The system generated edge table also shows six Color_id's for edges. Figure 15 and 16 shows the year wise generated timetables according to the assigned timeslots.



View By: Department D1 (ALL) [✓ Year 1] Year 2

	Monday	Tuesday	Wednesday	Thursday	Friday
8.30 a.m. - 10.30 a.m.	IT2002		IT2002	IT2002	IT2001
10.30 a.m. - 12.30 p.m.	IT2002	IT2001			
12.30 p.m - 1.00 p.m	L U N C H B R E A K				
1.00 p.m - 3.00 p.m					
3.00 p.m - 5.00 p.m					

Figure 15: First Year Timetable - Test Problem 1



View By: Department D1 (ALL) Year 1 [✓ Year 2]

	Monday	Tuesday	Wednesday	Thursday	Friday
8.30 a.m. - 10.30 a.m.	IT4002	IT4001		IT4002	IT4003
10.30 a.m. - 12.30 p.m.					
12.30 p.m - 1.00 p.m	L U N C H B R E A K				
1.00 p.m - 3.00 p.m					
3.00 p.m - 5.00 p.m					

Figure 16: Second Year Timetable - Test Problem 1

5.2 Discussion

Since the timetabling problems are belongs to the class of NP hard problems, the algorithm provides many possible solutions. Therefore, after obtaining the edge table, we can generate many possible timetables by assigning timeslots to different time periods. Figure 15 and figure 16 shows only one example among all possible solutions.

6. Conclusions and Future Work

This paper presents an application of graph Coloring model for SLIATE course timetabling problem. Graph coloring is a powerful tool whose theory maintains a central position in discrete mathematics and there is of great interest for its many useful applications. The proposed method is based on edge Coloring of a bipartite graph and it is capable of generating conflict free course timetables, lecture room timetables and lecture's personal timetables. Also, the presented algorithm works well on the real-life large scale course timetabling problem at SLIATE.

The major contributions of this work are: we have collected course data and then create the bipartite graph of the subjects and its lectures. Then we have properly colored that bipartite graph using minimum Colors by applying edge Coloring approach. Next, we have developed the graph Coloring algorithm which is capable as we believe of solving timetabling problems in all departments for any possible set of data. Once the master timetable has been constructed by using the designed algorithm, we have used it to generate lecture room timetables as well as lecture's personal timetables.

In general we would think of a good quality timetable as one that is (firstly) feasible and that (secondly) satisfies the soft constraints. In this study we were able to satisfy all hard constraints and some of soft constraints. The experimental results show that this method finds more than one possible solution for a given set of data other than the optimal solution. That is, we can generate more than one possible timetables while satisfying hard and soft constraints with this method.

The computational results of this project show that the proposed method is able to generate better solutions as compared to the allocations scheduled by the departments of SLIATE manually. Therefore, overall, our proposed method produced good results for course timetabling problem.

References

[1] Burke, E.K., Petrovic, S., Recent Research Directions in Automated Timetabling, *Eu. J. Ope. Res*., 2002.
[2] Carter, M.W., Laporte, G., Recent developments in practical course timetabling, 1998.
[3] Schaerf, A., A Survey of Automated Timetabling, Artificial Intelligence Review, **1999**.
[4] Wren, A., Scheduling, timetabling and rostering — A special relationship?,**1996**.
[5] De Werra, D., An Introduction to Timetabling ,*Eu. J. Ope. Res*., **1985**.
[6] Brailsford, S.C., Potts, C.N., Smith, B.M., Constraint Satisfaction Problems: Algorithms and Applications, *Eu. J. Ope. Res*. 119 557-581, **1999**.
[7] Peck, J. E. L., Williams, M. R., Algorithm 286: Examination scheduling,http://dl.acm.org/citation.cfm/id=365713 (Accessed 5 December 2016).
[8] Burke, E.K., Elliman, D.G., Weare, R., A university timetabling system based on graph coloring and constraint manipulation, *J. Res. Com. Edu.*, **1993**.
[9] Timothy Anton Redl, A Study of University Timetabling that BlendsGraph Coloring with the Satisfaction of Various Essential and Preferential Conditions, **2004**.
[10] Herts A., Tabu Search for large scale timetabling problems, *Eu. J. Ope. Res*., **1991**.
[11] Yu, E., Sung, K.S., A genetic algorithm for a university weekly courses timetabling problem. *International Transactions in Operational Research*, 9(6), pp. 703-717, **2002**.

Author Profile

Wathsala Samarasekara received the B.Sc. in Applied Sciences degree with a First Class from Rajarata University of Sri Lanka and M.Sc. in Computer Science degree from University of Peradeniya, Sri Lanka in 2008 and 2013 respectively. She is an Assistant Lecturer in Information Technology in Sri Lanka Institute of Advanced Technological Education, Sri Lanka.