A Study on Heuristic Timetabling Method for Faculty Course Timetable Problem

Bong Chia Lih*, Sze San Nah† and Noor Alamshah Bolhassan‡
Faculty of Computer Science and Information Technology
Universiti Malaysia Sarawak
Sarawak, Malaysia
Email:*lihlih1206@gmail.com, †snsze@fit.unimas.my, ‡bnalamshah@fit.unimas.my

***Abstract*—This paper studies university course timetabling problem in a case study related to faculty management system, which is concerned with assigning students/lecturers to classes and time-slots. In order to increase flexibility choices of timeslot for "repeaters", the problem has become more tight in scheduling. A two-stage heuristic approach is presented, where the initial stage groups the courses that are able to conduct simultaneously. The second stage then assigns the weekly timeslots for each group of courses, followed by venue for each course. Computational results are presented for the proposed solution using real data. It shows that the proposed solution is effective to handle the faculty course timetabling.**

Keywords—automated timetabling; two-stage heuristic

I. Introduction

Course timetabling is well known as NP-complete [1] and has been studied extensively over the past decades [2], [3] and [4]. However, it is "ambitious" to develop a fully automated timetabling mechanism to meet the needs of all concerned. Eventually, the timetabling process requires a lot of manual effort.

In Universiti Malaysia Sarawak (UNIMAS), course timetabling is managed by two parties: faculty management and Centre of Applied Learning and Multimedia (CALM). Each faculty owns a number of lecture/tutorial rooms, while CALM manages four Centres of Teaching Facilities (CTF). Every CTF consists of one lecture hall (LH) with capacity of 540 students and six seminar rooms (SR) with capacity of 150 students. In the beginning of the timetabling setup, CALM will pre-assign each faculty a number of slots at CTFs based on past experience, to accommodate courses with large number of students. Together with the pre-assigned rooms allocation from CALM and own faculty's facilities, the responsibility of constructing faculty course timetable falls on each faculty management. This is mainly because each faculty has different preference or request from lecturers.

Previously, Faculty of Computer Science and Information Technology (FCSIT) course timetabling was handled by two experience planners. The planning process was overwhelmed with abundance of constraints and time consuming to even find a feasible solution. This paper presents a real solution to this faculty course timetabling by using a two-stage heuristic method.

II. Proposed Approach

In this timetabling problem, we consider a set of offered courses, $C = \{C_1, C_2, \ldots, C_n\}$ will be enrolled by a set of students (including lecturers), $S = \{S_1, S_2, \ldots, S_u\}$. Each course i has a list of potential students and lecturers. Let define the set of potential students and lecturers as $P_C = \{P_{C_1}, P_{C_2}, \ldots, P_{C_n}\}$, where $P_{C_i} \subset S$ and $C_i \in C$.

A two-stage heuristic is proposed to solve this course timetabling case study. This is due to its simplicity to cope with different hard and soft constraints in two stages.

A. Stage I (Course Grouping)

All the courses will be divided into a few groups based on the following constraints:

1) Students can max enrol one course in each group.
2) Lecturers can max enrol one course in each group.
3) Takes into account the repeating students to enable them to enrol in certain courses.

The main idea of course grouping is to cluster several courses into one group so that these courses can have lecture concurrently during the same timeslot in different venues by different lecturers and students. The task of this stage is to reduce the problem size, from n courses to m groups where $n > m$. Let define a set of group, $G = \{G_1, G_2, \ldots, G_m\}$ where $G_i \neq \emptyset$ and $G_1 \cap G_2 \cap \ldots \cap G_m = C$. The course grouping algorithm is shown as follow:

Step 1 Calculate each course's pairing ability, Y_i with other courses

Algorithm 1 Calculate pairing ability

```
n = number of courses
for i = 1 to n do
    Y_i = 0
    for j = 1 to n do
        if (P_Ci ∩ P_Cj = ∅) then
            X_ij = 1
        else
            X_ij = 0
        end if
        Y_i = Y_i + X_ij
    end for
end for
```

Step 2 Course with minimum pairing ability, Y_k, $C_k \in C$ is added into a new group, G_m with group number m. Then add courses in G_m. This step is repeats until all courses been assigned for a group.

Algorithm 2 Assign courses into groups

```
m = 1
while (all assign[C] = false) do
    k = index of minimum Y
    INSERT C_k ∈ G_m
    for j = 1 to n do
        if (X_kj = 1) then
            for every course i in G_m do
                if (every X_ij = 1) then
                    if (fulfil total venues capacity) then
                        INSERT C_j ∈ G_m
                        assign[C_j] = true
                    end if
                end if
            end for
        end if
    end for
    assign[C_k] = true
    m = m+ 1
end while
```

B. Stage II (Timeslot Allocation)

This stage takes into account time-related constraints:

1) One timeslot allocated for only one group.
2) Spread the lecture time for each major/program evenly throughout the weekdays.

Due to the limited availability of bigger venues given by CALM, a pre-processing algorithm is conducted. First, let us define a set of timeslot types, $Q = \{Q_1, Q_2, \ldots, Q_5\}$ (as in TABLE I), and a set of timeslot throughout a 5-day week (11 timeslot per day), $V_{i,j} = \{V_{1,1}, V_{1,2}, \ldots, V_{5,11}\}$. For each $V_{i,j}$ is assigned a timeslot type, $Q_i \in Q$ based on its venues availability. Meanwhile, each $G_k \in G$ is also assigned a timeslot type, $Q_i \in Q$ based on its need.

TABLE I. TYPE OF TIMESLOT

Timeslot types, Q	Number of room available	
	LH	SR
Q_1	1	2
Q_2	1	1
Q_3	1	0
Q_4	0	2
Q_5	0	1

In this stage, each of the group $G_k \in G$ from previous stage is assigned a timeslot. A new timeslot set $T = \{T_{1,1}, T_{1,2}, \ldots, T_{5,11}\}$ is proposed to update the timeslot allocation for each $G_k \in G$. Besides matching the group venue's need and timeslot ability, we consider the even distribution of the lecture of students.

Algorithm 3 Timeslot Allocation

```
for k = 1 to m do
    i = day with least lecture hours for the most junior course in G_m
    for j = 1 to 11 do
        if (V_i,j = G_k) then
            UPDATE timeslot for G_k, T_i,j = k
        end if
    end for
end for
```

Some groups with all small size courses do not need any big lecture room. These groups have not been labelled with any type and thus unallocated. They will be allocated into empty timeslots which are not occupied by any venue of CTF.

III. RESULT ANALYSIS

FCSIT had roughly 600 students at the semester 1 of academic year 2011/2012, taking five different programmes. The faculty owns ten tutorial rooms, TR (with capacity of 30 students) and one lecture theatre, MMT (with capacity of 120 students). On top of that, the faculty was given one lecture hall, LH and two seminar rooms, SR at CTF for certain timeslots. The venue availability is illustrated in TABLE II. In that semester, 49 courses were offered and conducted by 46 academics. The data was studied to evaluate the performance of proposed heuristic.

TABLE II. SUMMARY ON VENUE

Management Team	Venue	Number of Room Available	Student Capacity	Available Timeslot
CALM	LH	1	540	Limited
	SR	2	150	
FCSIT	MMT	1	120	All the time
	TR	10	30	

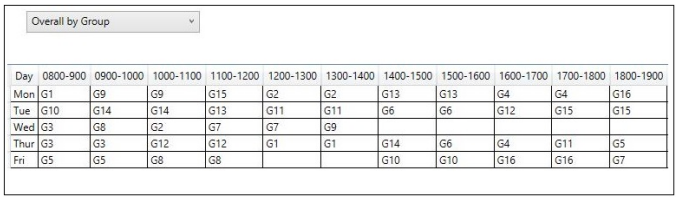
The heuristic experiment was coded in C# and all the experiments were carried out on a Toshiba i5 2.6GHz computer. All experimental tests can be computed in less than 2 minutes. While the manual course timetable planning took two to three weeks.

The grouping stage has successfully clustered 49 courses into 16 groups with minimum two and maximum five courses per group. The result shows that the grouping algorithm is able to average the courses into 16 groups; i.e. and the mod of statistic is also 3 (refer to TABLE III). The total number of the groups, 16 groups is the minimum number that the algorithm can generate. There are sufficient timeslots (48 hours weekly) to allocate 16 groups of courses. If the number of groups generated is large, it might not be able to get a feasible solution unless more timeslots available (night time or Saturday).

TABLE III. GROUPING SUMMARY

Number of Courses in a Group	Number of Groups	Number of Courses
1	0	1 * 0 = 0
2	4	2 * 4 = 8
3	8	3 * 8 = 24
4	3	4 * 3 = 12
5	1	5 * 1 = 5
TOTAL	**16**	**49**

The grouping method successfully avoids the clashes problem for all lecturers and students including the consideration of repeaters. Courses with common potential participants will never be grouped together. Each group is allocated with an available timeslot without overlap with another group. Fig. 1 shows different timeslots allocated for each group.



Day	0800-900	0900-1000	1000-1100	1100-1200	1200-1300	1300-1400	1400-1500	1500-1600	1600-1700	1700-1800	1800-1900
Mon	G1	G9	G9	G15	G2	G2	G13	G13	G4	G4	G16
Tue	G10	G14	G14	G13	G11	G11	G6	G6	G12	G15	G15
Wed	G3	G8	G2	G7	G7	G9					
Thur	G3	G3	G12	G12	G1	G1	G14	G6	G4	G11	G5
Fri	G5	G5	G8	G8			G10	G10	G16	G16	G7

Fig. 1. Display Timetable by Group

Fig. 2 presents the number of lecture day from 20 generated timetables (5 programs for 4 years). Most of the students, which is 70% or 14 of them having lecture every weekday. The rest of them have only 3 to 4 lecture days per week. This is due to fewer courses required to be enrolled by final year students.

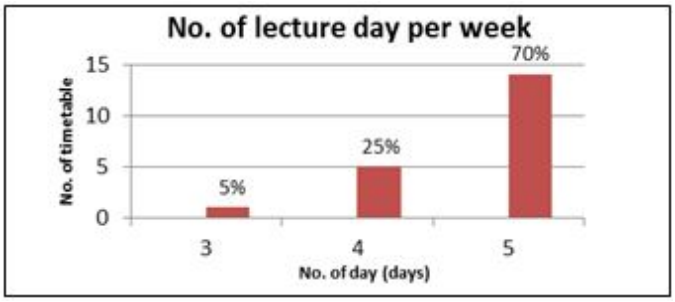


Fig. 2. Number of lecture day per week

On the other hand, Fig. 3 illustrates the total lecture hour of a day. There are 100 days in total for 20 timetables. The total lecture hours in a day are range from 0 hours to a heavy lecture day of 8 hours. The result presents that only 8% of the days are having lecture more than 5 hours in a day. This proves that the algorithm is able to distribute the lecture evenly throughout the week.

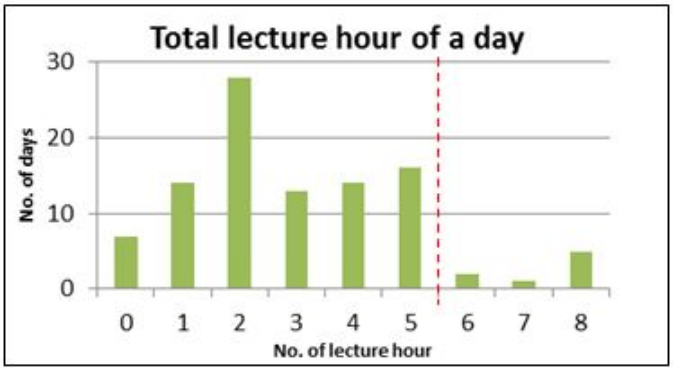


Fig. 3. Total lecture hour of the day

IV. CONCLUSION

In this research, two-stage heuristic is proposed to solve faculty course timetabling. Courses are initially clustered into different groups then followed by time and room allocation. Real data from FCSIT, UNIMAS was test by proposed two-stage heuristic. The computational result illustrated that the proposed solution can handle timetabling efficiently and effectively. Furthermore, the lectures are mostly evenly distributed.

ACKNOWLEDGMENT

The authors would like to thank to FCSIT, UNIMAS for the opportunity given to learn, focus and deeper understanding towards research area.

REFERENCES

[1] R. Lewis, *A survey of metaheuristic-based techniques for University Timetabling problems*, OR Spectrum, 30(1), pp. 167-190, 2008.

[2] Andrea Schaerf, *A survey of automated timetabling, Artificial intelligence review*, 13(2), pp. 87-127, 1999.

[3] S.A. MirHassani and F. Habibi, *Solution approaches to the course timetabling problem*, Artificial Intelligence Review, 39(2), pp. 133-149, 2013.

[4] R. Qu, E.K. Burke and B. McCollum, L.T. Merlot and S.Y. Lee, *A survey of search methodologies and automated system development for examination timetabling*, Journal of scheduling, 12(1), pp. 55-89, 2009.