International Journal of Engineering & Technology, 8 (1.9) (2019) 220-223

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper



Solving University Course Timetabling Problem (UCTP) Using Depth First Search (DFS) Algorithm and Rule Base Systems

Fatchurrochman[1], Zainal Abidin[2]

[1,2]*Informatic Department, Maulana Malik Ibrahim State Islamic University of Malang, Malang, Indonesia*
**Corresponding author E-mail: fatchur@ti.uin-malang.ac.id(*

Abstract

University course timetabling are administrative activities undertaken by the universities to manage their resources for education process. The resources are classrooms and lecturers. In this research, the university course timetabling or course scheduling is seen as the search process. Searches conducted on unused slots to be filled with lectures classroom. slot is a term used to denote an entity that can be filled with classroom lectures. Lectures classroom are objects to be scheduled. DFS is used to prepare candidates for the class schedule of a lecture classroom. Subsequently checked whether the candidate's course schedule does not violate the lecturers constraint, classroom constraints, dhuhur prayer time constraint, Friday prayer time constraint, and in accordance with the preferences day of lecturer. If the candidate does not violate all constraint it will be stored as class schedules and if violated the constraint it will look for another schedules candidate. The system testing showed that of the 131 lectures classroom can all be scheduled at 6 classroom without breaking the constraints that have been set. There are 20 lectures classroom that are not scheduled when only 5 classroom used, this indicates that the DFS is not complete and is not optimal, this algorithm requires sufficient classrooms in order to work properly. In terms of computing time, this study provides the big picture of time required to complete the preparation of lecture schedules using DFS.

Keywords: *Schedule Lectures, Depth First Search, Object Oriented Approach*

1. Introduction

University course timetabling are administrative activities undertaken by the universities to manage resources that consist of classrooms and lecturer in the preparation of the learning process.

Arrange the university course timetabling is not a simple job. This job takes time to complete in order to meet all the applicable rules at the university. There are general and specific rules to each university. This resulted in the scheduling system in a particular university is not necessarily in accordance with the prevailing system in other universities. It is true that the differences between institutions are many, especially as far as quality criteria are concerned. That makes the use of one application that was implemented for one university inappropriate for other (Kyriakos & Panagiotis, 2001).

Research problem in this paper is arrange schedule in Islamic university considered specific hard constraint are dhuhur prayer times and Friday prayers times beside general hard constraint like lecturer constraint, classroom constraint, and lecturer preference day to teach.

Research question are 1) how to arrange the schedule so it can be meet all constraint for all lectures classroom. 2) how to implement Depth First Search (DFS) Algorithm to solve UTCP.

Research objective are 1) develop software to generate schedule that considered lecturer constraint, classroom constraint, lecturer preference day to teach, dhuhur prayer times and Friday prayers times. 2) understanding performance of DFS algorithm to solve university course timetabling.

2. University course timetabling problem (uctp)

UCTP are old issues and still being investigated in line with the changing needs and technological developments. The basic problem in UCTP is how to construct a lecture schedules with little energy but successfully manage limited resources so that the learning process can run smoothly. Settlement with a small energy gave rise to the need for automated scheduling software.

There are many approaches have been made to complete the UCTP automatically. This research seeks to understand the fundamental problem of scheduling lectures and finish it with a simple approach. In this study, the preparation of the lecture schedule is seen as a search problem (Schaerf, 1999). It is expected with this approach the issue of scheduling lecture can be better understood and the various constraints can be added as needed. University course timetabling problem is considered by researchers as a NP-hard combinatorial optimization that does not have analytical solution methods (Najlaa & Ash, 2015). The system is built using the object-oriented paradigm

Copyright © 2019 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

for this approach provides software reusability and maintainability (Chuan, 2003). The facility was easier for researchers to further develop the software according to the needs.

Pray on time and in congregation for moslem is a high virtue, so the schedule of lectures to provide opportunities for students and lecturers in performing the prayers on time will be very useful for a moslem. This research used dhuhur prayers constraint and friday prayers constraint, namely that the lecture schedules may not coincide with dhuhur prayer times and Friday prayers times. Differences dhuhur prayers constraint and friday prayers constraint in the course timetabling problem is the amount of time slots used for prayers, where dhuhur prayer requires one time slot and friday prayers require two time slots.

This paper seeks to resolve the problem of lectures scheduling at the Islamic university that is predominantly moslem. In this university, besides the hard constraint of classroom and lecturer, there is also a special hard constraint that fardlu prayer time and friday prayer time. In this research fardlu prayer is represented of dhuhur prayer.

3. UCTP Using DFS algorithm and rule base systems

How the system works in this study is shown in Figure 1. Input system is a lecture classroom that contains the data subject, class name, and lecturers. The data is merged with the classroom, time slot, and the name of the day obtained using DFS algorithms into one candidate schedules. Candidates for the schedule needs to be checked for compliance with the specified constraints. In case of violation of the constraint it will look for the new schedule, and in the absence of constraint violations, the candidates that schedule will be kept to a schedule.

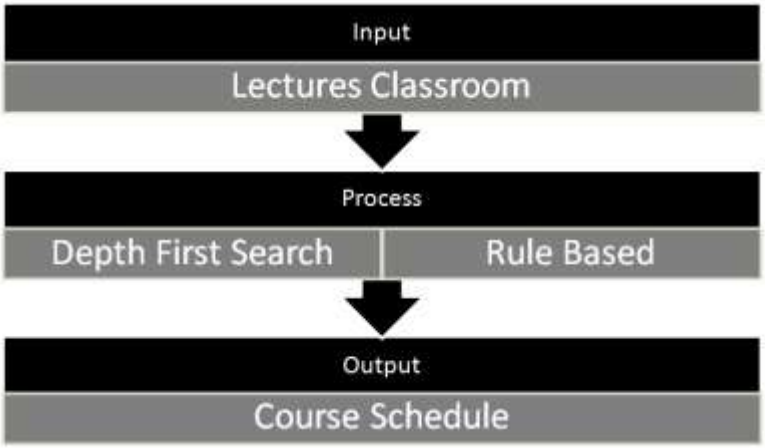


Fig.1: System Design UCTP using Depth First Search Algorithm dan Rule Based

Algorithm for for UCTP using Depth First Search Algorithm dan Rule Based describe in the following way :

```
1       Get Lectures Classroom from database
2       for each record in Lectures Classroom
3           sks ← getCredit(subject_code)
4           timeslot[] ← getTimeSlot(sks,dhuhur_time)
5           preferenceday[] ← getPreference(lecturer_code)
6           classroom[] ← getClassroom(departmen_code)
            //Depth First Search
7           for i ← 0 to classroom length
8               for j ← 0 to timeslot length
9                   starttime ← getStart()
10                  Endtime ← getEnd()
11                  for k ← 0 to preferenceday length
12                      if(day ← 5)
13                      Else
14                          room_cons ← roomConstraint() // rule based
15                          lect_cons ← lectureConstraint() // rule based
16                          If(room_cons ← true or lect_cons ← true)
17                          Else
18                            saveSchedule()
```

DFS work in preparing the schedule of lectures conducted by retrieving data lectures classroom consisting of subject, classes and lecturers. Based on this information, the system will search for information on the amount of credits of the subjects to be determined corresponding slot. The next step is to find classroom, time slots and days. These data are candidates for the class schedule for a particular lecture.

The next step is the examination of the constraints on candidate schedules have been prepared. Examination constraint is an important step that must be done in order to schedule composed of DFS process in accordance with the applicable rules. Constraint lecturer is basically guarantee the absence of slices of teaching time a lecturer in the day and the same hour. Constraint of space also ensures that no slice the use of space on the day and the same hour. Constraint teaching preference is today expected by lecturers in teaching. The fulfillment of this constraint is expected to be made comfortable in doing the lecturer teaching.

In this study the constraint embodied in the rule based on the constraints of lecturer and classroom constraint. Here are some examples of rule for the constraint Lecturer, namely:

1. If a lecturer teaching at a specific time slot then the lecturer should not be scheduled in the other room on the day and time slot.

2. If a lecturer teaching at a specific time slot then the lecturer should not be scheduled in another room on the same day and the start is at the same time slot.

3. If a lecturer teaching at a specific time slot then the lecturer should not be scheduled in another room on the same day and the end is at the same time slot.

Other rules are arranged in the same way to create classroom constraint, the preferences teaching constraint, dhuhur prayer constraint, and the Friday prayers constraint.

As an illustration of how this system works the following is a simple example of the application of the above algorithm. Suppose we want to schedule a software engineering course 2 credits with the course code 0765309. This course is conducted as many as four classes namely class A, B, C, D.

lecturer of this course has a code 65002. Lecturer preferences for teaching are Tuesday, Wednesday and Thursday.

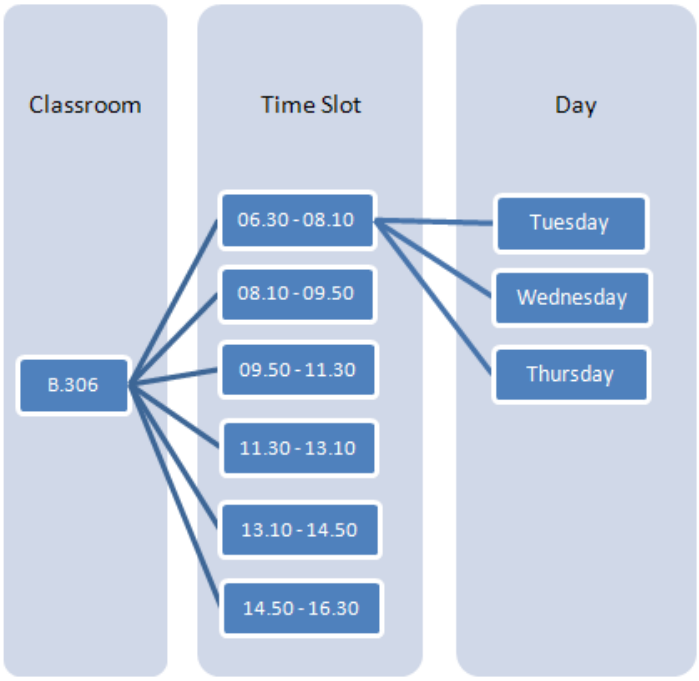


Fig.2: Ilustration for Process Scheduling with DFS

Data preparation is done at the start of the algorithm. These data consist of classrooms, lecture time slot and teaching preferences. These data can be described as Figure 2. The next step is to develop a schedule using DFS approach. Search for a solution starting from room B.306, time slot 06.30 – 08.10, Tuesday to subject software engineering A class. When no constraint is violated then the schedule will be saved. Furthermore looking for schedule for class B in the room B.306, time slot 06.30 - 08.10, Tuesday. Because the class has been occupied by a class A, then resumed the search for solutions to Wednesday. And so on as table 1.

Table 1: Results of System Testing

Code	Class	Classroom	Time Slot	Day
0765309	A	B.306	06.30-08.10	Tuesday
0765309	B	B.306	06.30-08.10	Wednesday
0765309	C	B.306	06.30-08.10	Thursday
0765309	D	B.306	08.10-09.40.	Tuesday.

Class D is scheduled in room B.306, time slot 08.10 – 09.40, Tuesday. Because this room at 06.30 Tuesday time slot has been used it must find another time slots.

4. Discussion

Testing is done using a blackbox testing to determine whether the system has been built to work as expected. Computing environment for testing the system using the operating system Windows Vista Home Basic SP1, Intel Core 2 Duo 2 GHz, and 1GB of memory. The programming language used is Java and DBMS MS Access to store data. At the time of testing, the database was filled schedule of lectures in previous years as many as 2140 records.

Tests conducted for 131 lectures classroom with varying credits between 1 to 3. This data is stored in the database through applications that have been built. The system will retrieve the data's classroom lectures one by one starting from the first record to the last record, look

for the amount of credits in subjects table's records are selected, and then get a time slot data based on the amount of the credits. Teach preference data obtained from the field KodeDosen connected with tables PreferensiMengajar.

Each record in the table PlotMengajar will be given classroom, then the specified time slot, and the latter was given the day according to the preference of teaching. The next step is to check this schedule is already present in the database or not. If this schedule is not in the database then the schedule will be kept and if there are then the schedule will be rearranged to change the day in the schedule to another day. These step were taken for the whole PlotMengajar so as to form the lecture schedule.

The total number of credits for the 131 lectures classroom is 303 credits. This means that theoretically required minimum of 303 slot time for all lectures classroom can be scheduled.

Table 2: Results of System Testing

Testing number	classroom lectures	classroom	lecturer constraint violated	classroom constraint violated	lecture preference violated	dhuhur prayer time violated	jumat prayer time violated	not schedule	Process Duration (Minute)
1	131	7	0	0	0	0	0	0	57
2	131	6	0	0	0	0	0	0	48
3	131	5	0	0	0	0	0	20	40

Testing for 7, 6 and 5 classrooms, shows that there is no constraint is violated (Table 2). On 7 and 6 classrooms, there is no constraint violations also no lectures classroom are not scheduled. For the amount of classroom as much as 5, 20 lectures classroom are not scheduled. This is due to the amount of classroom as much as 5 to have more slots than the small needs. If in one day there are 12 time slots so one room in one week (5 working days) there will be 60 slots. Then 5 classrooms will produce 300 slots. At the trial over, 131 lecture classes require 303 slots. The number of courses that are not scheduled is also influenced by the lecturer preferences are not met.

Testing also noted the computing time during the process of drafting the schedule runs. It appears that the shortest time to complete the preparation of the schedule is 40 minutes and the longest time is 57 minutes. The processing time is obtained because the database has been loaded 2140 scheduling data records of previous years. When the application is executed on the condition of each database is empty, the time required to prepare a schedule is about 1 minute. This is influenced by the process of finding an empty space in the database, where the more data the computing process will be longer.

5. Conclusion

From the above it can be concluded that sistem pejadwalan perkuliahan telah dibangun. Constraint is considered in the preparation of the schedule is a lecturer constraint, the constraint of classrom, dhuhur prayers constraint, friday prayers constraint and teaching preferences constraint. A system built to schedule a 131 lectures classroom on six classroom.

The DFS algorithm can be used to perform automatic scheduling lecture well when the number of time slots provided sufficient. When the number of time slots is inadequate, there will be a lectures classroom that are not scheduled. In the testing that has been done there are 20 lectures classroom that have not been scheduled when five spaces provided. The use of the optimization algorithm is required to reduce the amount of lecture classes that are not scheduled when the amount of space available is limited. In the above case, the use of the optimization algorithm is expected to reduce the amount of lectures classroom that are not scheduled to be smaller than 20 because mathematically lacking only 3 time slots.

Further development of this system can be done by adding various constraint line with the rules in university.

Computational processes can also be improved by rearranging various looping process or replace it with a matrix or using right data structures.

References

[1] Ateeq, Najlaa., & Bakar, Abu. (2015). An Autonomous Software Approach to Enhance Information Sharing in University Course Timetabling Planning. Journal of Theoritical and Applied Information Technology, Vol 73 No 1.

[2] Chuan, Swee Tan. (2003). An Object Oriented Timetabling Framework. International Journal of Information Technology, Vol 9 No 1.

[3] De, Broes., Machiels, Christophe., Janssens, Gerda., & Denecker, Mark. Regularity Requirements in University Course Timetabling.

[4] Grobner, Matthias., Wilke, Peter., & Buttcher, Stefan. (2003). A Standard Framework for Timetabling Problem. Springer-Verlag, Berlin Heidelberg.

[5] Schaerf, A. (1999). A Survey of Automated Timetabling. Journal Artificial Intelligence Review, Vol 13, Issue 2.

[6] Zervoudakis, Kyriakos., & Stematopoulos, Panagiotis. (2001). A Generic Object Oriented Constraint-Based Model for University Course Timetabling. Springer-Verlag, Berlin Heidelberg.