

**EXAM TIMETABLE FOR UNIMAS -
ENHANCEMENT OF SISTEM JADUAL WAKTU UNIMAS**

VOON TZE MIN
(Software Engineering)

This project is submitted in partial fulfillment of
the requirements for the degree of Bachelor of Computer Science with Honours

Faculty of Computer Science and Information Technology
UNIVERSITI MALAYSIA SARAWAK
2005

DECLARATION

No portion of the work referred to in this report has been submitted in support of an application for another degree or qualification of this or any other university or institution of higher learning.

.....

Voon Tze Min

9272

.....

Date

ACKNOWLEDGEMENT

First of all, I would like to express my gratitude to my supervisor, Dr. Rosziati Ibrahim, for her guidance, ideas and support throughout the development of this project and completion of this thesis.

Also, I would like to thank all the FIT staffs of being very helpful and never lacking in providing assistance to me at anytime.

Last but not least, many thanks to my family member who have demonstrated unwavering support and humor throughout this project. Finally, I would like to thanks all my friends for their great ideas, guidance and kindness during the development and implementation of this project.

TABLE OF CONTENTS

	PAGE
DECLARATION	ii
ACKNOWLEDGEMENT	iii
TABLE OF CONTENTS	iv
LIST OF FIGURES	viii
LIST OF TABLES	xii
ABSTRAK	xiii
ABSTRACT	xiv
Chapter 1: Introduction	
1.1 Introduction	1
1.2 Problem statement	2
1.3 Objective	3
1.4 Scope	4
1.5 Procedures/ Methodology	4
1.6 Expected outcome	5
1.7 Significant of research	6
1.8 Conclusion	6
Chapter 2: Literature review	
2.1 Introduction	8
2.2 Ant algorithm for University Course Timetabling Problem (UCTP) (applied in Universite Libre de Bruxelles)	8
2.2.1 Introduction	8

2.2.2	Algorithm description	9
2.3	Genetic algorithm (GA) based university timetabling system (applied in University of Nottingham)	10
2.3.1	Introduction	10
2.3.2	Characteristic of the Evolutionary Process	11
2.3.3	Basic outline of a Genetic Algorithm	12
	2.3.3.1 Sample implementation	13
2.3.4	Timetable generation	15
2.4	Hybrid Algorithm for the Examination Timetabling Problem (applied in University of Melbourne)	18
2.4.1	Constraint programming	18
2.4.2	Simulated annealing	20
2.4.3	Hill climbing	22
2.5	Comparison among the algorithm	23
2.6	Implementation tools	23
2.7	Proposed solution	24
2.8	Conclusion	25
 Chapter 3: Requirement Analysis and Design		
3.1	Object oriented Analysis (OOA) – Introduction	26
3.2	Business Process Modeling	26
3.3	Use Case Model	29
3.4	Object modeling	33
3.5	Data flow diagram (DFD)	34
	3.5.1 Context diagram	34
	3.5.2 Level Zero diagram	35

3.5.3	Level 1 diagram	38
3.5.4	Data dictionary	39
3.6	Object oriented Design (OOD) – Introduction	40
3.7	Access layer	40
3.7.1	Table class mapping	40
3.7.2	Access layer classes	42
3.8	View layer	43
3.9	Conclusion	56
Chapter 4: Implementation and Testing		
4.1	Implementation – Introduction	57
4.2	Main page	58
4.3	View timetable	58
4.4	Lecturer login	59
4.5	Register for course	60
4.6	Edit course	62
4.7	Administrator login	64
4.8	Timetable generation	64
4.9	Refresh the timetable	74
4.10	Comparison between Unimas Examination Timetabling System and Sistem Jadwal Waktu Unimas	76
4.11	Testing – Introduction	77
4.12	Test plan	77
4.13	Conclusion	83

Chapter 5: Conclusion and Future Works	
5.1	Introduction 84
5.2	Research finding 84
5.3	Contribution 85
5.4	Limitation 86
5.4.1	Course information 86
5.4.2	Repeated student 86
5.5	Future work 86
5.5.1	Allocation of generic courses 87
5.5.2	Examination duration 87
5.5.3	Allocation of venue 87
5.5.4	Involve of more than one lecturer 88
5.5.5	Edit the timetable 88
5.5.6	Manage lecturers' record 88
5.5.7	Manage course information 89
5.6	Conclusion 89
	REFERENCES 91
	APPENDIX A: DATA DICTIONARY 93
	APPENDIX B: SOURCE CODE 102

LIST OF FIGURES

	PAGE
Figure 2.1: The idea of genetic algorithm: mutation and crossover	24
Figure 2.2: SJWU uses sequential method to search for the possible solution	25
Figure 3.1: Activity diagram that shows the activities that can be performed by the student who is using Unimas Exam Timetabling System	27
Figure 3.2: Activity diagram that shows the activities that can be performed by the lecturer who is using Unimas Exam Timetabling System	28
Figure 3.3: Activity diagram that shows the activities that can be performed by the administrator who is using Unimas Exam Timetabling System	29
Figure 3.4: Use case diagram that shows the actors and use cases that can be performed by the Unimas Exam Timetabling System	30
Figure 3.5: Class diagram for Unimas Examination Timetabling System	34
Figure 3.6: Context diagram for Unimas Examination Timetabling System	35
Figure 3.7: Level zero diagram for Unimas Examination Timetabling System	36
Figure 3.8: Level 1 diagram which extends the register course process	38
Figure 3.9: Level 1 diagram which extends the edit course information Process	39
Figure 3.10: Mapping of class course with its database	41
Figure 3.11: Mapping of class timetable with its databases	41
Figure 3.12: Unimas Examination Timetabling System's access layer classes	42
Figure 3.13: Main page	43
Figure 3.14: View timetable	44

Figure 3.15:	Lecturer login	45
Figure 3.16:	Function list for lecturer	46
Figure 3.17:	Lecturer's function: register for course that will be taken in the examination	47
Figure 3.18:	Course registration status	48
Figure 3.19:	Lecturer's function: edit registered course	49
Figure 3.20:	Detail of the course that will be edited	50
Figure 3.21:	Course edition status	51
Figure 3.22:	Administrator login	52
Figure 3.23:	Function list for administrator	53
Figure 3.24:	Confirmation to generate the timetable	54
Figure 3.25:	Timetable generation status	54
Figure 3.26:	Confirmation to refresh the timetable	55
Figure 3.27:	Timetable refreshes status	55
Figure 4.1:	Function view() under class Timetable	58
Figure 4.2:	Connect to database and access the record from the selected table	59
Figure 4.3:	Disable the used session during previous login	59
Figure 4.4:	Compare ID and password with database. If match, session for ID and password will be enabled	60
Figure 4.5:	Function register_course() under class Course	61
Figure 4.6:	Verify the course code passed and the number of student passed	61
Figure 4.7:	Insert the new course into database	62
Figure 4.8:	Function edit_course() under class Course	62
Figure 4.9:	Compare course code passed and lecturer with database. If match, course detail will be displayed for edition	63

Figure 4.10:	Update course detail in the database	63
Figure 4.11:	Function generate() under class Timetable	64
Figure 4.12:	Retrieve the data from the course_detail table in the database	65
Figure 4.13:	Determine the year from the course code	65
Figure 4.14:	Looking for the possible available timeslot	66
Figure 4.15:	Store the possible available timeslot in a timeslot array	66
Figure 4.16:	Set preferred time for types (majoring) of course	67
Figure 4.17:	Randomly choose the timeslot if the preferred timeslot is not available	67
Figure 4.18:	Reinitialize the element for the timeslot array and decrease the array size	68
Figure 4.19:	Store the possible available day in a day array	68
Figure 4.20:	Reinitialize the element for the day array and decrease the array size	69
Figure 4.21:	Determine either BS or DK will be used	69
Figure 4.22:	Store the possible available venue in a venue array and randomly choose one of them	70
Figure 4.23:	Check whether the lecturer already has record in the database	71
Figure 4.24:	Check the status of the timeslot for the lecturer involved	71
Figure 4.25:	Allocate the lecturer of that timeslot and day with the course; allocate the venue of that timeslot and day with the course; allocate the timeslot with the course and the venue	72
Figure 4.26:	No clashing in the venue's timetable	73
Figure 4.27:	No clashing in the lecturer's timetable	73
Figure 4.28:	Function refreshall() under class Timetable	74
Figure 4.29:	Setting array for table year, venue and lecturer; set default data for table year and venue	75
Figure 4.30:	Delete all the record for all the tables	76

Figure 4.31: Insert the default data for table year and venue (bs and dk)

76

LIST OF TABLES

	PAGE
Table 2.1: Comparison among the algorithm	23
Table 3.1: Description for use case view timetable	31
Table 3.2: Description for use case register course	31
Table 3.3: Description for use case edit course	32
Table 3.4: Description for use case generate timetable	32
Table 3.5: Description for use case refresh timetable	32
Table 4.1: Comparison between Unimas Examination Timetabling System and Sistem Jadwal Waktu Unimas	77
Table 4.2: Test plan for Unimas Examination Timetabling System	78

ABSTRACT

For the current exam timetabling system in Unimas, some problems and weakness arise. The procedure of booking the venue for final exam is complicated and is done manually. In addition, current Sistem Jadwal Waktu Unimas has some weakness. A proper searching method and algorithm is not applied for that system. Hence, the Unimas Examination Timetabling System is developed. It is a computerized system to help the Academic department to simplify the process of arranging the examination timetable. The timetable will be generated through the adoption of the genetic algorithm. This is a web-based system which is developed using Object Oriented Methodology, PHP and MySQL. The main feature of this system is the automation generation of the examination timetable from the course information. Hopefully, this system will help to decrease and solve the problem such like clashing among timeslots, clashing among venue and clashing among lecturer's timetable.

ABSTRAK

Untuk sistem jadual waktu peperiksaan di Unimas sekarang, terwujudnya masalah dan kelemahan. Langkah untuk menempahkan tempat untuk peperiksaan akhir adalah rumit dan dilaksanakan secara manual. Selain itu, Sistem Jadual Waktu Unimas sekarang mempunyai kelemahan. Kaedah dan algoritma yang sesuai adalah tidak diaplikasikan untuk sistem itu. Selaras dengan itu, Sistem Jadual Waktu Peperiksaan Untuk Unimas telah dibina. Sistem tersebut ialah sistem berkomputer yang dibina untuk membantu pihak Akademik untuk meringkaskan proses penyusunan jadual waktu peperiksaan. Jadual Waktu tersebut akan dibina melalui penggunaan algoritma genetik. Ini adalah system berasaskan web yang dibina dengan menggunakan metodologi berorientasikan objek, PHP dan MySQL. Ciri utama system ini adalah pembinaan jadual waktu peperiksaan daripada maklumat kursus secara automatik. System ini diharapkan dapat membantu dalam mengurangkan dan mengatasi masalah seperti pertindihan antara jadual waktu, pertindihan antara tempat dan pertindihan antara jadual waktu pensyarah.

Chapter 1: Introduction

1.1 Introduction

Scheduling event plays an important goal in human daily life. Scheduling deals with the satisfactory allocation of resources over time to achieve organization's tasks (Heizer and Render, 1996). Timetabling is a general scheduling problem and can manifest itself in several different forms. The term timetabling is referring to the assignment of time slot subject to constraints, times to activities or events. Many complex constraints and objectives are involved upon the underlying combination problem. A feasible and good timetabling system will help in organizing timetable and the constraints are satisfied. It ensures that all events are assigned to a timeslot and a room. The constraints include no clashing or only one event is taking place in each place at a given time, and the room's capacity is suitable for the attending users and satisfies all the features required by the event. In addition, no users attend more than one event at the same time. Besides that, it simplifies the process of organizing the timetable, which replaces the manpower to do it manually and may take weeks. Sometimes it cannot avoid the occurrence of errors which is brought by last minute change.

The examination timetable required to schedule the examination in a given set of exam session (time slots). It is a difficult combinatorial optimization problem. Constraints have to be satisfied. For instance, no student can attend two exams at the same time, and no two exams can be hold at the same venue. In other words, no clashing is expected.

In this project, the timetable is used to organizing and allocating timeslots and venues for the examination for Unimas, which is enhancement of the existing timetabling system, Sistem Jadwal Waktu Unimas (SJWU).

1.2 Problem statement

The scheduling for examination timetable is not a simple affair. With examination scheduling, set of constraints have to be satisfied. For instance, clashing and conflict have to be avoided. Appropriate technique is required to place or allocate the exams in rooms.

Some of the most common forms of constraints include:

- Clashing
 - A student cannot sit for two exams at the same time
 - A venue should be prevented from having more than one exam at the same time
 - Lecturer should not involved in two examinations at the same time
- Capacity
 - Each venue or room must fix the number of students who sit for the exam. No over limit is allowed
 - There must have enough rooms or venue for holding the examination

With the current exam timetabling system in Unimas, some problems and weakness arise. This includes:

- The procedure of booking the venue for final exam is complicated. The staffs have to fill in the forms for booking the venue manually, and send to

the section in charge to be approved. An approval has to be confirmed before the timetable can be obtained. Since the process is done manually, it is time consuming and error-prone. Last minute changes sometimes are inevitable.

- Current SJWU has some weakness. After inputting data for allocating venue of lecture and tutorial, it takes a couple period of time in compiling and generating the result for getting a timetable. Appropriate search and optimization techniques are not implemented.
- The arrangement of booking the venue for final exam is done by manpower or manually. There is a risk that the staff will overlook in arranging the timetable. Thus, clashing may arise.

1.3 Objective

The objectives of this project include:

- To study, understand and identify the algorithm and search technique. Suitable technique and algorithm will be reviewed. The advantages and the disadvantages of each will be identified. Current SJWU will also be studied to identify the problems and weakness.
- To simplify the procedure of booking the venue for final exam. This will ensure less time and work is needed for the process of booking the venue for the final exam
- To find out the most suitable and best performance algorithm in searching and allocating the venue and timeslot. This will help in minimizing the time in generating the result (exam timetable) of the timetabling system.

- To design a system for final exam based on SJWU. The appropriate and most suitable technique and search algorithm will be proposed as well.

1.4 Scope

The algorithms and techniques that have been applied in other existing system will be studied. Comparison will be done for those existing system and the algorithm and technique used as well.

Current SJWU will be studied, enhanced and improved. Based on the current SJWU, the examination timetabling system will be developed. An algorithm or technique that helps in searching will be proposed and be adapted. This search algorithm will enable the timetabling system to generate timetable, especially allocating the venue and timeslot without clashing. Furthermore, it will be much efficient in generating the result for the system compared to the current SJWU (in term of the time used for generating the timetable and the allocation of timeslot).

1.5 Procedures/ Methodology

A good planning is required to ease the process of completing this project. The reuse-based development process model will be applied for this project. This approach is based on the existing of SJWU and integrating it to the exam timetabling system rather than develop the system from scratch. Object Oriented System Development Life Cycle (OOSDLC) will be applied to carry out this project. OOSDLC will be enable the system to be developed more easily and support greater design and easier to maintain. The main stages include object-oriented analysis (OOA), object-oriented design (OOD) and object-oriented implementation (OOI).

During the OOA, actors of the system will be identified. Actors are those people who will use the system. Use case is the functionality that provided by the system. The functionality will interact with the actors.

During OOD, the relationship among classes will be represented. View layer and access layer will be designed as well. View layer is concern about the user interface, while access layer will concern about the communication with the database.

In the OOI process, the system will be implemented using object-oriented programming language. User satisfaction and usability is important. Refine may have to be done if satisfaction is not reached.

Testing is also very important in OOSDLC. Incremental testing will be done through the whole process, which is from OOA until OOI. This is to ensure that each stage is going well during the process.

Unified Modeling Language (UML) will be used to model the system. PHP and MySQL will be applied to generate the result.

1.6 Expected outcome

At the end of this project, the system proposed will be able to organize the timetable for final exam for Unimas. A better performance algorithm can be applied or adapted and will help in reducing the processing of generating the timetable. Clashing problems may be minimized as well since it is no more done manually. Besides that, the procedures for booking the venue and allocating timeslot for final exam will be simplified as well.

The lecturer will input the relevant data. These include the course code of the examination and the number of student that will sit for the exam. After this, the data

will be retrieved and be generated. From the input data, the proposed algorithm and technique will search for the slot that is available and suitable for the number of student who sits for the exam. After all the slots have been finished allocated, the timetable will be generated.

1.7 Significant of research

This project will help solving the problems that are faced now. The system will:

- Simplify the procedure of booking the venue for final exam. Paper work may be less since the procedure will be done through the system.
- The staff does not have to arrange the timetable manually or by manpower. All the processes will be generated by the system.
- Avoid clashing in allocating the venue and timeslot. Since the timetable is generated by the system, error of wrongly allocating can be eliminated which may be overlook by doing it through manpower.
- The timetable will be generated through the adoption of the algorithm and the search technique. This will decrease the timetable generation time and also the increase the efficiency in term of timeslots allocation

1.8 Conclusion

The proposed system is important for generating the examination timetable for Unimas. It will be developed after reviewing some of the existing systems. The system will be generated by considering all the existing constraints. Finally, this

system will help in decreasing the current problems faced in term of arranging the timetable for examination.

Chapter 2: Literature review

2.1 Introduction

In order to enhance the SJWU and create an exam timetable for Unimas, a review and analysis on a set of benchmark real-world exam timetabling problems will be carried out. The behavior of different approaches with the solutions and objectives functions will be compared. The most suitable system will be chosen among the reviewed system and be proposed or adapted for creating the exam timetabling system for Unimas. The algorithm and search technique that are applied in other universities and will be reviewed here include the genetic algorithm, ant algorithm and hybrid algorithm.

2.2 Ant algorithm for University Course Timetabling Problem (UCTP) (applied in Universite Libre de Bruxelles)

2.2.1 Introduction

Ant Colony Optimization (ACO) is a metaheuristic. It is based on foraging behavior of real ants. The ant algorithm applies a lot of parameters that define its operation, in order to deliver best possible performance. Trial and error procedure is used in choosing the parameter. Limited time is used by each algorithm to run since the number of trials to find the best parameter is high. If the parameter fine-tuning done with run-times is shorter than actual problem solving runs, the suboptimal parameters may be chosen.

The University Course Timetabling Problem (UCTP) is a type of constraint satisfaction problem. It consists:

Events: $E = \{e_1, \dots, e_n\}$

Timeslots: $T = \{t_1, \dots, t_k\}$

Rooms: $R = \{r_1, \dots, r_{|R|}\}$

Furthermore, it consists of set of student S who attend the event and set of features F satisfied by rooms and needed by events. Every student is pre-assigned to a subset of events. There are two types of constraints that have to be satisfied which are hard constraint and soft constraint. The examples of hard constraint include:

- A student cannot attend more than one event at one time
- The room capacity can afford the number of student who attend the event
- Each place can be taken only for one event at a given time

On the other hand, the example of soft constraint is student has more than two events in a row. In a feasible timetable, the soft constraint violations are obtained to be minimized.

2.2.2 Algorithm description

Each of the m ants constructs a complete assignment C of events into timeslot and rooms every time the algorithm iterates. The ant will choose the timeslot and room for the given event based on the pre-ordered list of events and the stigmergic information, which is in the form of a matrix of pheromone value $\tau: E \times T \times R \rightarrow \mathbb{R}^+$, where E is the set of events, T is the set of timeslots and R is a set of rooms. To simplify the notation, the matrix becomes $\tau: E \times P \rightarrow \mathbb{R}^+$, where P is the set of k places, and $k = |P| = |T| \cdot |R| = i \cdot j$.

Specific knowledge or heuristic information will be applied in the algorithm. Choice will be made only if it is suitable and will not violate any hard constraint in

placing for an event. If there is no place available for allocating an event, a list of timeslot is extended by one, and then the event will be placed at this additional timeslot. As the timeslot from now on exceed i , it is infeasible. The pheromone matrix has to be extended by creating an extended set T' of I' timeslot, and new extended set P' of k' places. Now the pheromone matrix become $\tau: E \times P' \rightarrow R^+$, assumed that $i'=i$ and $k' = k$.

After constructing the events into places, local search routine will be used to improve the solution. Best solution of iteration will be compared with the global best solution. If the iteration best solution is better, then it will replace the global best solution.

Local search (LS) that is used can improve two major modules. First, it tries to improve an infeasible solution and does not contain any hard constraint violation and fits into i timeslot. This is defined as HardLS. Secondly, the LS is run only if feasible solution is available. It tries to reduce the number of soft constraint violation, called as SoftLS. It tries to rearrange the event in the timetable. Local search is very depend on the time the algorithm run, and also applies to other algorithm parameter. Best results are found from sets of parameter. Average best algorithm change if run-time change allowed.

2.3 Genetic algorithm (GA) based university timetabling system (applied in University of Nottingham)

2.3.1 Introduction

Genetic algorithm is a general purpose optimization tool and heuristic or approximate algorithm which takes principle of natural selection and evolution. (Ho