# A MICRO-GENETIC ALGORITHM APPROACH FOR SOFT CONSTRAINT SATISFACTION PROBLEM IN UNIVERSITY COURSE SCHEDULING

**ABD. HALIM BIN BOHADEAN @ BOHARI**

**(88485)**

**MSc. IT (By Research)**

**UNIVERSITI UTARA MALAYSIA**

**2013**

# A MICRO-GENETIC ALGORITHM APPROACH FOR SOFT CONSTRAINT SATISFACTION PROBLEM IN UNIVERSITY COURSE SCHEDULING

**This dissertation is submitted to the Centre for Graduate Studies to fulfill the requirement of Master of Science (Information Technology By Research) Universiti Utara Malaysia**

**Abd. Halim Bin Bohadean @ Bohari**
**(88485)**

**© Abd Halim Bin Bohadean @ Bohari, July 2013. Copyright Reserved**

## Permission to Use

In presenting this thesis as a major requirement for a post-graduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may take it freely available for inspection after being submitted for a year. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by my supervisor or, in his absence, by the Director of Centre for Graduate Studies. It is understood that any copying or publication or use of this thesis or part thereof for financial gain shall not be allowed without my written permission. It is also understood that recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use, which may be made of any material from my thesis.

Request for permission to copy or to make other use of materials in this thesis, in whole or in part should be addressed to:

Director
Center for Graduate Studies
Universiti Utara Malaysia
06010 UUM Sintok
Kedah DarulAman

# Abstrak

Masalah penjadualan kursus universiti adalah kombinasi masalah pengoptimuman. Masalah adalah lebih mencabar apabila satu set peristiwa perlu dijadualkan dalam slot masa, akan ditempatkan ke bilik yang sesuai, yang tertakluk kepada beberapa set kekangan keras dan lembut. Kesemua kekangan yang wujud sebagai peraturan-peraturan dalam setiap sumber untuk peristiwa tersebut perlu dipenuhi untuk mencapai tugas yang optimum. Di samping itu, reka bentuk jadual kursus untuk universiti adalah tugas yang sangat sukar kerana ia merupakan satu permasalahan "non-deterministic polynomial", (NP) keras. Masalah ini boleh dikurangkan dengan menggunakan pendekatan Algoritma Mikro Genetik. Pendekatan ini, mengekod perwakilan kromosom sebagai salah satu elemen penting untuk memastikan hanya sedikit bilangan kromosom individu yang tidak fisibel dihasilkan. Oleh itu, kajian ini mencadangkan pengekodan perwakilan kromosom menggunakan tatasusunan satu dimensi bagi menambahbaik pendekatan algoritma genetik mikro kepada masalah kekangan lembut dalam jadual kursus universiti. Sumbangan bagi kajian ini adalah dalam membangunkan perisian penjadualan yang efektif dan fisibel dengan menggunakan Algoritma Mikro Genetik yang mampu untuk mengurangkan pengeluaran kromosom individu yang tidak fisibel berbanding algoritma pengoptimuman sedia ada bagi jadual waktu kursus universiti, dimana data sampelnya ialah UNITAR International University. Algoritma Mikro Genetik yang dicadangkan telah diuji dalam ujian perbandingan dengan algoritma genetik biasa dan carian berpandukan kepada algoritma genetik sebagai penanda aras. Keputusan menunjukkan bahawa algoritma yang dicadangkan mampu untuk menjana bilangan minimum kromosom individu yang tidak fisibel. Keputusan eksperimen juga menunjukkan bahawa Algoritma Mikro Genetik mampu untuk menghasilkan jadual kursus terbaik untuk UNITAR International University.



**Kata kunci:** Mikro Genetik, Kekangan lembut, Pengoptimuman, Penjadualan

# Abstract

A university course timetabling problem is a combination of optimization problems. The problems are more challenging when a set of events need to be scheduled in the time slot, to be located to the suitable rooms, which is subjected to several sets of hard and soft constraints. All these constraints that exist as regulations within each resource for the event need to be fulfilled in order to achieve the optimum tasks. In addition, the design of course timetables for universities is a very difficult task because it is a non-deterministic polynomial, (NP) hard problem. This problem can be minimized by using a Micro Genetic Algorithm approach. This approach, encodes a chromosome representation as one of the key elements to ensure the infeasible individual chromosome produced is minimized. Thus, this study proposes an encoding chromosome representation using one-dimensional arrays to improve the Micro Genetic algorithm approach to soft constraint problems in the university course schedule. The research contribution of this study is in developing effective and feasible timetabling software using Micro Genetic Algorithm approach in order to minimize the production of an infeasible individual chromosome compared to the existing optimization algorithm for university course timetabling where UNITAR International University have been used as a data sample. The Micro Genetic Algorithm proposed has been tested in a test comparison with the Standard Genetic algorithm and the Guided Search Genetic algorithm as a benchmark. The results showed that the proposed algorithm is able to generate a minimum number of an infeasible individual chromosome. The result from the experiment also demonstrated that the Micro Genetic Algorithm is capable to produce the best course schedule to the UNITAR International University.



Keyword : Micro Genetic, Soft constraint, Optimization, Timetabling

## Acknowledgements

I would like to express my appreciation and thank to my supervisor, Prof. Madya Dr. Azman bin Yasin for being a good advisor in providing assistance, guidelines and support throughout this thesis. I also would like to thank to Dr. Yuhanis bt Yusof for her encouragement and insightful comments. A special thank to my wife Hartini Mohd Ashikin and my mother Aloha Mohd Diah for their understanding, prayers and blessing.

In addition, thanks to my friends:

Roshidi bin Din

Hanizan Shaker bin Hussain

Baharudin bin Osman

For your helps, ideas and support to complete this thesis.

Finally to the people who are keen to knowledge, perhaps this thesis contributes to the body of knowledge and enriches the information hereto.

# Table of Contents

# List of Tables

# List of Figures

# CHAPTER 1

# INTRODUCTION

In general, a university course timetabling problem usually refers to finding the exact allocated time within a limited time period for example a week, for a number of events (courses-lectures) and assignment of events to a number of resources (lecturers-rooms) in such a way that a number of constraints are satisfied.

Yang and Petrovic (2004) has defined the timetabling as the allocation of a set of subject into a classroom over a limited number of time periods to avoid the occurrence of conflicts of interests between two subjects or lecturers. A good scheduling technique that can lead to optimization is important to ensure it is able to produce all timetable for students and lecturers.

The main problem in the university timetable generation is to provide lecturers and lecture activities by matching all lectures involving the consumption a lot of time as well as the person responsible. The information required for the course schedule including room availability, time slots and several specific policy options. For example, information on room availability can be specified to the room capacity for certain events. In the domain of university timetable, it is often used to refer to the construction of schedule (with time slots) through the system by considering several numbers of constraints.

There are two categories of constraint which are hard and soft constraint have to be satisfied. Hard constraints are very strict constraints that must be satisfied. Examples of such constraints are the lack of resources such as lecturers and students, rooms to be assigned for different events at the same time, courses offered to a particular group of students need to be assigned at the same time.

On the other hand, the desire preferences to be fulfilled is called soft constraint. These soft constraints are as an extension of a good timetable. In a search algorithm, the soft constraints can also been defined as optimization objectives. The examples of soft constraints such as assigning an event for harmony and balance scattered for whole timetables and minimizing the interval time for the certain lecturers on the two consecutives class period. The accomplishment of constraints in university timetabling shows the feasibility of a solution, but to measure the quality of a timetable solutions, the soft constraints should also be taken into account. A Genetic Algorithm optimizer and evolutionary search algorithm have been applied in scheduling algorithm to produce the feasible timetable that fulfill the hard and soft constraints.

## 1.1 Motivation

The scheduling of courses and lecturers is a key sensible problem, which is faced by most educational institution. Substantial effort has been committed to develop effective and feasible timetabling software in the past six years. The problems handled by such procedures include classes and lecturers timetabling, in which a set of lectures is to be scheduled over a set of time periods, and class timetabling, where a set of classes must be scheduled over the length of an entire week and also at the most optimum usage of the rooms available.

University course scheduling can be a complicated problem and it cannot be solved with using only a few general principles. Every individual involved as administrators, teachers and students have their own objectives, and these objectives are usually contradict (Lai et al., 2006).

Timetabling problems are often complicated by numerous constraints; for instance, a lecturer should not be scheduled to teach two subjects at the same time. These constraints are typically considered as hard constraints, which must not be violated (in the class timetabling problem, a hard constraint might be that no lecturers is scheduled to teach two subjects at once), and soft constraints, which possess a penalty for being violated. Because of the number and a variety of constraints, such timetabling problems typically constitute NP-hard problems and local optima that are quite difficult to solve manually. This in turns has led to an increased emphasis on finding effective automated timetabling algorithms.

Hence, an effective and efficient timetabling system is very much needed. This thesis presents a work on utilizing a Genetic Algorithm technique in constructing a feasible course schedule.

## 1.2 Background

Most of the current timetabling systems using genetic algorithm (GA) have shown that the infeasible individual chromosomes will always be generated in the reproduction process especially during the crossover process. According to Xie and Pan (2010), during the reproduction of the offspring some feasible individuals which violate some constraints will be generated that is called the infeasible individuals. The infeasible individuals can also be generated in the mutation process where it could randomly change specific characteristics in the chromosome representing the individual (Karova, 2004).

The main idea for improving genetic operators was to prohibit the existences of new conflicts (infeasible individual) because the basic operators such as crossover did not take into account whether it makes individuals with more or with less conflict. Thus, the outcome of the algorithm utilizing such basic operators was noticeably poor. The decision was made to add some programming logic to the operators and to use a different selection algorithm. (Sigl, B. et al, 2003).

## 1.3 Problem Statement

When applying an algorithm to some scheduling problem, the crucial element is encoding (Perzina, 2007) and sometimes it could reach only a limited portion of the search space and thus it can converge to solutions close to local optima. (Burke and Newall, 1999). Most of the algorithms reviewed concern about the difficulties to obtain the optimal solution and therefore increasing the computational cost of generating feasible timetable. In global search technique by using GA may cause the possibilities to converge toward local optima and a new solution may fail in finding the global optimum. A good strategy has to be adopted to transform GA into an explorative algorithm. The feasible solution that achieve a fast convergence and better result can be produced by a good encoded chromosome representation for the search space. In many real timetable problems, the main objective is to obtain a solution that strictly satisfying hard constraints while soft constraints should be fulfilled as many as possible. Thus, by encoding a good chromosome representation, it may easy to obtain that all individuals in a chromosome will satisfy all hard constraints, then the soft constraint satisfaction need to be monitored in order to keep the individual chromosome to remain feasible even after the operation of crossover or mutation in Genetic Algorithm.

The infeasible individual in a chromosome can create some consequences that can affect the performance of the timetable system. Generally, it is very difficult to repair all infeasible individuals in a chromosome. The existence of the infeasible individuals in a chromosome will make the selecting or creating of an appropriate genetic operators becomes more complicated (crossover, mutation or selection). Thus the algorithm to

regenerate the chromosomes from the infeasible individuals need to be used for small populations.

Therefore, an algorithm should be applied in producing highly satisfied timetables which is fulfills the constraints as well as timetable's objectives. This study has decided to enhance a Micro-GA, proposed by Coello and Pulido (2001) which normally used a small population to represent the feasible solution of large populations by hope that it will take less time to process in selecting the appropriate genetic operators and should fulfill all soft constraints.

## 1.4 Research Objectives

The research objective is mainly to use a Micro-GA approach for soft constraint satisfaction problem in university timetable. In order to achieve this main objective, several sub-objectives have to be fulfilled. They are:

i. To enhance Micro-GA by introducing a new encoding chromosome representation.
ii. To introduce new fitness function according to soft constraint scale.
iii. To test and evaluate the effectiveness of the MGAT system based on soft constraint, fitness performance and end-user usability.

## 1.5 Research Scope

i. The soft constraint will only consider seven resources such as lecturers, classrooms, time periods, courses, lecture types, number of days and the number of different lectures.

ii. The timetabling in this study will only consider on the Micro-GA optimizer.

iii. The study of this timetabling only considers the environment of UNITAR International University, Petaling Jaya.

## 1.6 Significance of This Study

This study will affect the way on how the timetable is generated by considering more on the soft constraint to fulfill the lectures or lecturers preferences in terms of resources such as convenient time and classroom available. Furthermore, this study will also provide opportunities for learning institutions in order to manage an effective planning in scheduling all classes with various requirements to fit their institutional policies. In terms of Micro-GA application domain, this study may help other researchers to apply the construction of chromosome representation based on the genes generated in the search space.

## 1.7 Research Contribution

For the research community, this study will contribute in developing effective and feasible timetabling software. This is done by proposing a framework that can be used as a guide to other researcher when their intention is to use timetabling scheduler as an applied system.

The proposed framework is to be applied in the real environment to optimize the solution of soft constraint needs in order to produce the effective and efficient timetable.

## 1.8 Thesis Organization

In this research, the content is being organized into six chapters consisting an introduction, literature review, research methodology, data analysis, experimental results and conclusion.

The introduction chapter contains the overview and motivation of the timetabling system. This chapter also explains the objectives of this study in detail. Besides an overview of the study and the objectives, it also includes an outline of the research approach, the scope of the study, the expected outcome, the research method and the research design.

In Chapter Two, the literature review shows briefly the GA generic model, the basic steps in producing timetable especially with regard to the selection, crossover and mutation operations. Various approaches and rules have been presented in solving the existing timetable problems by using a Micro-GA.

.

Chapter Three contains the research methodology that shows the problem formulation and gives a theoretical background of standard Genetic Algorithm, Micro-GA and related technical information about the problem.

Chapter Four describes the analysis and result of testing and evaluation of constructing Micro-GA that was done from Chapter Three.

Finally, Chapter Five outlines the conclusions of the work, highlights key findings and results, and suggests some future works and recommendations.

The following, which is Chapter Three, provides information on research methodology used that provides guidance in data analyzing.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 Introduction

This chapter has several sections that review timetabling approaches, GA generic model, genetic operator, repairing method and GA approaches in timetabling and Micro-GA. All these sections will be discussed in detail and considered as a major component needed to be fully emphasized.

Timetabling has three major components which are resources, events and constraints that are always integrate into each other. The resources comprises time slots, rooms and teacher while events may includes the teaching session and constraints refers to the certain limitations need to be satisfied. During the process of assigning a set of specific resources for certain events, it will always subject to a number of constraints.

In the optimization, the constraints are usually divided into two groups which are hard constraints and soft constraints. In any circumstances, the hard constraints should not be violated, while the soft constraint should be as low as possible. The violation of hard constraints could lead to infeasible timetable while the violations of soft constraint will affect the quality of timetable thus it should be minimized. In a university course

scheduling problem, the available resources such as faculties, students, courses, time periods and rooms.

Using the genetic algorithm in the timetable system has been discussed by a lot of papers. If the search space of timetable is very large and has many limiting conditions, it will not produce desirable effects by directly using the genetic algorithm for timetable. According to the traits of timetable, many papers had modified the standard genetic algorithm. Burke and Newall Paper (1999) proposes to tackle timetable with many steps by dividing the maximum search space into several parts, and optimizes every one of them using the genetic algorithm.

However, the partial sub-optimal solution may not be the global sub-optimal solution. Paechter B., et al. (1994), proposed to change the chromosomes and give up the spaces relative to unchangeable courses or classrooms to reduce the search space and improving the searching efficiency. However, it is not so easy to remove these ineffective spaces. It needs deep analysis of the every problem and cannot be used universally.

According to Najdpour Feizi-Derakhshi (2010), by using a traditional crossover operation into a population, it could not satisfy all hard constraints. Thus, a genetic algorithms are easily trapped into local optimum. As an extension of the traditional crossover operation, a cycle crossover (CX) has been used to avoid this problem. In this

method, instead of producing only one offspring, the information will be exchanged in cycle starting with leftmost of the string and first two time slots, (Najdpour Feizi-Derakhshi, 2010).

Nia and Alipouri, (2009) introduced two methods which are Sequential Mutation Method and Circular Gene Method to increase the speed of the GA. They discussed that these methods could produce a better result by using a less cost function evaluations compared to the traditional GA.

In year 2009. Yang,. H., and Luo, D., successfully introduced Improved Genetic Algorithm (IGA) to overcome the defections that may occur in the evaluation process of GA. They stated that the defection can occur in the premature phenomenon due to the large difference among choromosome at the early in stage in evaluation process or when very small difference among choromosome can also cause low convergence efficiency.

Sapru, Reddy, and Sivaselvan (2010) addressed the scheduling problem by providing a timetable for each class for specific groups of students to a particular room while giving an appropriate time slot which is subject to the limited time and availability of faculty and other resources. For genetic algorithm coding scheme they proposed genetic algorithm which is based on the individuals in a chromosome who represents the population schedule for 5 days a week, with 9 time slots per day. Each timeslot contains information such as number of subjects, the allocated rooms and faculty. All information was coded in binary. For example, the 3-bit variable is used to represent

subjects that will be owned by this time slot. The scale between 0 and 1 were used in the fitness function and assuming that the chromosomes with fitness 0 have infinite constraint violations and chromosomes with fitness 1 shows no violation of constraints. In the crossover operator, the random selection of pairs of chromosomes and one point crossover is performed. Mutation is performed through the control of a novel mutation operator in which it will create a copy of the chromosome and chromosome toggles each bit of the chromosome with a relatively small probability.

A mutated copy of the chromosome fitness was compared with the original and copies of the mutated individual will replace the original if it has a better fitness. This is done to control mutation on all chromosomes in the population. The results indicated that the mutation contributes to improving the control of GA, the algorithm to focus on solutions that can be implemented in all cases, and does more quickly than the GA that using a conventional mutation operator.

AlSharafat and AlSharafat (2010) have developed a system of examination schedule for various courses at the University of Al-Bayt. They have tried and make a comparison between the three different forms of genetic algorithms. The first is a Steady State Genetic Algorithm (SSGA) with overlapping populations, the second is the increase in Strong Genetic Algorithm (ESSGA), the crossover and mutation operators have been enhanced by using Fuzzy Logic (FL), and the third is a Simple Genetic Algorithm (SGA) with non-overlapping population.

In the initial population, the individuals are first generated randomly. Each chromosome is encoded as a number of real values, each representing the input variables, such as courses, room size and number of classrooms. Initially, a fitness value is set to zero, and then the calculation of fitness is performed for each individual by adding a positive value for each constraint violation. The penalty values will be added for each violation depends on the importance of constraints. In ESSGA, FL rules were used to make the probability of crossover and mutation to adapt. The rules rely on the age of each individual in a chromosome, which is a counter that calculates the number of generation whereby this individual has been living in the population. The probability of crossover covering the aged of both parents, giving a priority to older individuals. In contrast, mutation probability will increase for young individuals. They found that ESSGA produce schedule with better fitness than the two other forms of GA tested in this study.

Abdullah, Turabieh, McCollum, and Burke (2009) have investigated the genetic algorithm that combines a sequential local search for the timetable problem. In this problem a time slot and a room is given to all the lectures for each course and observed set of hard and soft constraints. They divide the algorithm into two phases which are construction phase and improvement phase. The construction phase will generate the timetable, which represents an individual in the population. In this phase, the individual is enhanced by using three consecutive heuristic which are large degree heuristic, the neighborhood search and tabu search.

The timetable is generated through the construction phase is a feasible timetable that will be included in the population. In the improvement phase, GA is used by using the

single point crossover in the crossover technique. After the process of recombination, repair method is used to change the infeasible solutions to the feasible one. Finally, to improve the timetable the local search algorithm is used before moving to the next generation. The experimental results used as a benchmark to show that the proposed algorithm is able to produce the best results for 3 out of 21 example problems. An optimal solution for all resources must be grouped together in order to produce a class schedule by schedule manager. In general, the process to solve a university timetabling problem is refers to the process of the assignment of each lecturer to a room and a time slot which is subject to a number of hard and soft constraints. These constraints such as no lecturer can teach more than one class at the same time, no room can be scheduled more than one lecturer at the same time. This kind of problem is known to be NP-hard problems and hence it is known no algorithm for solving it in polynomial time.

## 2.2 Approaches To Solve Timetabling Problems

There are several approaches used in timetabling environment such as sequential, cluster, constraint based and meta-heuristic.

### 2.2.1 Sequential approach

This approach treats timetabling problems as graph problems. Gareyand Johnson (1979) introduces, given an undirected graph $G = (V, E)$. Vertices $v_1, \ldots, v_n$, and colors $c_1, \ldots, c_k$ are ordered. Initially, a vertex $v_1$ is assigned to $c_1$. Thereafter, vertex $v_i$ in turn is assigned to the "smallest" color that contains no vertices adjacent to $v_i$. The problem consists of finding a particular $V$ into a minimum number of color classes $c_1, \ldots, c_k$, where no two vertices can be in the same color if there is an edge between them. The edges between the vertices correspond to the conflicting events represents the conflicts between events, here the colors represent the time periods.

The method orders event using domain heuristics and the events are assigned sequentially into a valid time slots. The events that are most difficult to schedule are assigned into the first time slots (Carter, 1986), Thus a feasible coloring of the graph ensures a conflict free schedule.

In the survey by Carter and Laporte in 1996, Cluster Method and Sequential Methods were two of the four major categories of techniques for solving

examination timetabling problems and they use a strategy to select the next examination to add an initially empty timetable and construct the full timetable by assigning the examination one at a time (sequentially) to a selected period.

According to Carter (Carter et al, 1999) there are five heuristic methods inspired by the graph-coloring model of examination timetabling, for sequencing the examinations:

i. Largest Degree (LD): Largest number of conflicting examinations – highly conflicting examinations are difficult to schedule later in the construction process.

ii. Saturation Degree (SD): Number of periods in conflict - examinations with few remaining feasible slots should be scheduled sooner to avoid having no remaining feasible slots for the examination.

iii. Largest Weighted Degree (LWD): Number of students in conflict – similar to Largest Degree, but where each conflict (edge of the graph) is weighted by the number of students involved.

iv. Largest Enrolment (LE): Largest numbers of students enrolled for an examination– examination with a high enrollment are often difficult to be scheduled as they create many conflicts.

v. Random Ordering (RO): Select the next examination completely at random largely used for comparison purposes with the above techniques.

These sequential approach suggest how difficult it is to be scheduled the events, the most difficult event, according to the corresponding ordering strategy will be assigned first. This can result in difficulty to obtain the optimal solution and thus can increase the computation cost in the generating feasible timetable.

### 2.2.2 Cluster approach

The cluster method uses the technique that the events are divided into numbered of event sets (clusters) where any two events in a particular cluster do not conflict with each other, so that it satisfies all hard constraints. Then, the sets are assigned to real time slots to satisfy the soft constraints as well (White and Chan, 2002).

The main drawback of these approaches is that the clusters of events are formed and fixed at the beginning of the algorithm and that may result in a poor quality timetable (Newall, 1999). For example, the approach used by Vahid Lotfi and Robert Cerveney (1991) for scheduling final exams may be considered a cluster approach. In this approach, the solution may be found quickly, but in some cases they may result in poor timetable if there are many dependencies between events of different clusters.

### 2.2.3 Constraint Based approach

This approach fulfill a number of constraints by assigning a set of events with certain value. The events are denoted by a set of variables (Brailsford et al, 1999). According to Yakhno T., and Tekin, E., (2002), a number of rules are defined for assigning resources to events and when no rule is applicable to the current partial solution a backtracking is performed until a solution is found that satisfies all constraints; as the satisfaction of all constraints may not be possible, algorithms are generally allowed to break some constraints in a controlled manner in order to produce a complete timetable.

Usually constraints are not as simple as used in generating timetable and often are connected with each other.

The constraints are handled through a system of constraint propagation, which will reduce the domains of variables. Therefore constraint propagation does not remove all values that are in conflict with all constraints and its performance is measured as a trade-off between number of removed values and execution time. The main disadvantages of this approach such as the difficulties in expression for soft constraints and problems with improving the initial feasible solution.

### 2.2.4 Meta-heuristic approach

This approach is used to find a good feasible solution that is at least reasonably close to being optimal. A well designed meta-heuristic method can usually provide a solution that is nearly optimal (Colorni et al, 1990; Paechter et al, 2006).

These approach express constraints as some cost functions, in which to minimize the cost function by heuristic search of better solutions in a neighbourhood of some initial feasible solution. Meta-heuristic methods begin with one or more initial solution and employ search strategies to find optimal solution, trying to avoid local optima in the process (Valouxis, 2000).

The disadvantage of this approach is that it is usually designed to a specific problem type rather than a variety of applications and there is no guarantee that the best solution found will be the optimal solution.

Among all the approach discussed, the meta-heuristic approach will be used in this study by using the Micro-genetic algorithm to traverse the search space in the small population.

## 2.3 The GA Generic Model

Genetic Algorithm is an algorithm based on natural selection and evolution process. This theory attributes to Charles Darwin. As the basic idea of the heuristic, Genetic Algorithm starts with a set of chromosome (solutions), called a population. A new solution formed from a set of population which is considered has a better fitness compared to the among other population. According to their environment, a parent (selected solution) then form a new solution (offspring) that have higher chances to survive and reproduce. The process will keep repeated until it reach to a termination condition such as a number of iterations or a best solution obtained. The basic structure of the GA is shown in Figure 2.1.

Initialization of chromosome from the entire population is a first step in GA. In these step, the size of the population must be chosen. According to Goldberg (1989), if the size of the population is too large, the initialization process requires considerably more time and also the larger the size of the population the larger the convergence times. Different sizes are optimal to be chosen in this process and it is also rely on the available computing techniques. Altough the convergence times is quicker when a small population size is chosen it can cause a less of explorations from the global search space.

The next step in a GA is to evaluate the fitness of chromosome in a random population. In order to determine which chromosome is fitter than others, each chromosome must be

evaluated based on the knowledge about the environment of chromosome in which it survives. According to Gen and Cheng, (1994). this environment is the partially encoded (or partially decoded) description of the problem. There might be one or more rule used in evaluating a chromosome. Rich (1996), use a relative weighted to each rule as a method on evaluating a chromosome. One of the other method is by using the objective function cost from the penalties of individual chromosome if there is any collision to the constraints. The aim is to generate the population with a least costly (minimizing cost or maximizing fitness).

According to S. N. Jat and S. Yang (2009), The Guided Search Genetic Algorithm (GSGA) presented the framework of GA to solve the University Course Timetabling Problem (UCTP). The events are assigned to time slots, rooms, and students in which it is satisfy all hard constraint and a number of soft constraints. They also proposed the guided search strategy and Local Search (LS), which is integrated into a Steady State Genetic Algorithm (SSGA).

The initial population is generated randomly by assigning events to time slots and rooms in their research. The objective function is used by a calculation of weighted sum of hard constraints and soft constraint violations. The crossover operator exchanges time slots between the two parents, and tries to allocate rooms for events in each nonempty time slot. The mutation operator is based on selecting one of 3 different neighborhood operators (N1, N2 and N3). The N1 operator moves one event to a different time slot. The N2 operator swaps the time slots of two events, and the N3 operator chooses 3 events and orders them differently from their original order.

The role of local search comes after the mutation operator, and is performed in two steps: the first step tries to remove the hard constraint violations and the second step tries to remove the soft constraint violations. Both steps are based on the same neighborhood operators used in mutation.

In the GSGA proposed by them, the crossover operator creates offspring based on an external data structure that stores information about events. Specifically, the information stored is the list of room and time slot which were paired to each event that has a zero penalty value in which there are no constraint violation in this event. The advantage of using such data structure is to maintain useful information of good solutions, and pass this information to the new generations. Another version of the algorithm, called the Extended Guided Search Genetic Algorithm (EGSGA) was also proposed, by adding another local search method. The second local search method tries to reduce the number of violations for events involving a large number of constraint violations. The experimental results shown by the proposed GSGA and EGSGA are compatible with state of the art techniques for the UCTP, and can efficiently obtain optimal or near optimal solutions.

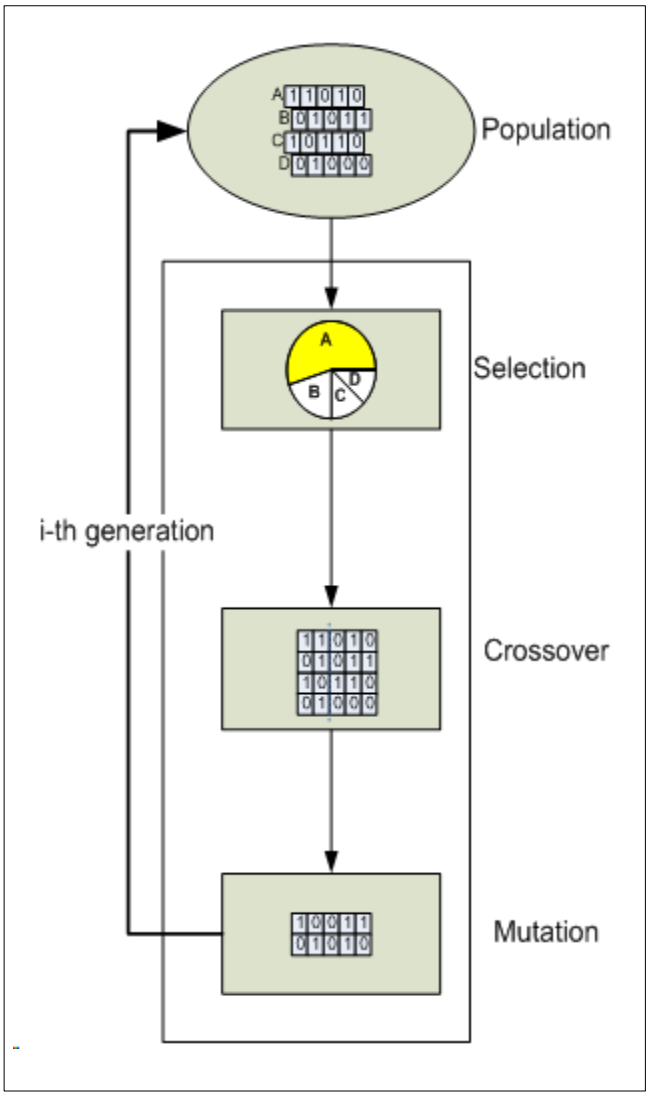



*Figure 2.1: Basic Structure of GA*

The basic algorithmic process of GA goes as shown in the Algorithm 1.

**Algorithm 1 : Genetic Algorithm**

1. START – Generate random population of n chromosomes (suitable solutions for the problem)

2. FITNESS – Evaluate the fitness f(x) of each chromosome x in the population.

3. NEW POPULATION – Create a new population by repeating following steps, until the new population is complete

    a. *Selection* – Select two parent chromosomes from a population, according to their fitness (the better the fitness, the bigger chance to be selected)

    b. *Crossover* – With a crossover probability, crossover the parents to form a new offspring (children). If no crossover is performed, the offspring is an exact copy or clones of the parents.

    c. *Mutation* – With a mutation probability, mutate new offspring at each locus (the position in the chromosome).

4. REPLACE – Use new generated population for a further run of the algorithm.

5. TEST – If the end condition is satisfied, STOP, and return the best solution found so far.

6. LOOP - Go to step 2.

## 2.4 Genetic Operators

A genetic operator is a process which is used to maintain genetic diversity. The process is similar to the variation of genetic which occur in the natural world such as: survival of the fittest, or selection; sexual reproduction (crossover, or recombination); and mutation. In the genetic algorithm, these common processes are selection, crossover and mutation. This common genetic operator as described in the following paragraphs.

### 2.4.1 Selection

Once the population has been initialized, Genetic Algorithm will select the chromosomes to be combined. This technique is important to determine at any given time the best two chromosomes will be combined to produce the best chromosome. The most common techniques used such as a roulette wheel, tournament, stochastic remainder and selection pressure.

Early GA used a replacement strategy which maintained a constant population by replacing two parents with their two offspring in each generation (Gen and Cheng, 1994). Tournament selection is another method for deciding which parents to be wiped out. In this method, (Rich, 1996) two parents are chosen and played off against each other- the winner is allowed to reproduce and/or the loser is selected for extinction.

In various Genetic Algorithms, the method of selecting parents for breeding is handled in different ways. Holland's original model uses a method where the healthiest are most likely to breed (Gen and Cheng, 1994). Other methods select any two chromosomes at random for breeding. According to Gen and Cheng, the selective breeding can be used in conjunction with or in the absence of an Elitist Natural Selection Operator.

### 2.4.2 Crossover

The crossover operator is to evaluate every chromosome on how close it is with the appropriate solutions. The chromosome will become closer to the best looking solution with the higher fitness in the population. This function must be built up carefully because it is a main driver in implementing GA. It is a method to combine two chromosomes to produce new chromosome so it can substitute the previous population. This method is shown in Figure 2.2.

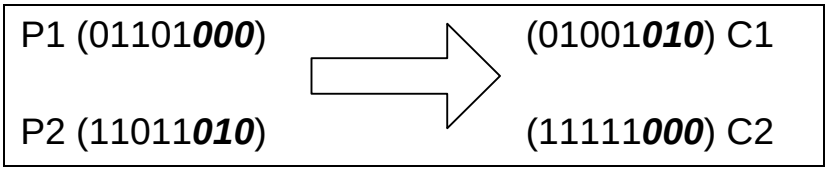



*Figure 2.2: Crossover Operation*

The process of crossover can be performed with more than one crossover point (Gen and Cheng, 1994). According to Heitkoetter and Beasley (2001), every point

can be chosen for crossover. They also suggested that one method of crossover, often used in multi-objective systems, is a unity order based crossover and also each gene has an equal probability of coming from either parent that there may be a crossover point place after each or any gene.

### 2.4.3 Mutation

The mutation operation needs only one chromosome in which a part of the chromosome will be changed. The changes will look different to the previous chromosome. This operator can be shown in Figure 2.3.

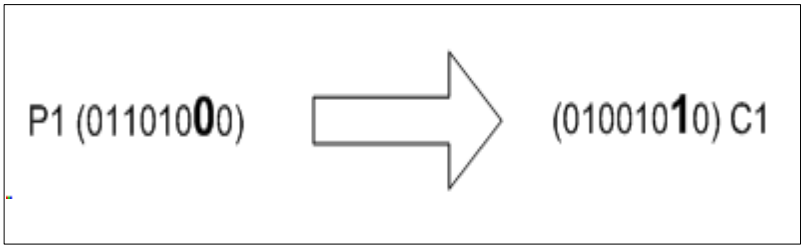



*Figure 2.3:Mutation Operation*

The mutation operators are applied to every chromosome with its probability. The probability of the mutation is calculated as the absolute value of the parameter of mutation.

## 2.5 Application of GA in Timetabling Problem

This study analyzed a few models based on GA that combine the advanced techniques with local search operators. The method to represent the encoding of timetable solution (chromosome) was the first requirement need to be considered in order to suit with the genetic operators when it is applied in the algorithm.

According to the literature, there are two different approaches to represent the chromosome which are direct and indirect representation. In a direct representation,

(Adamis and Arapakis, 1999), all events such as day, time slots, lecturers and rooms will be directly encoded. Using this representation approach, requires a very large search space to satisfy all constraints and the chromosome yielded frequently need to be repaired because it represents several invalid solutions.

Using the indirect representations, (Paechter et al., 2006), the chromosomes are usually represented by a sequential list of events. The encoded solution are located into the timetable follows to predefined method or "timetable builder". Then all events will be located into the timetable by using the combination of heuristics and local search.

According to Gy'ori and Varkonyi (2001), GA was applied in that optimization problem because of the robustness in a huge problem space. They have discussed optimization problem in timetabling by using GA. A new set of representation was introduced, in which meets the demands better than previous. Students, teachers, lessons and classrooms have to be arranged optimally. They also classified all hard and soft constraints and their method proven more efficient in a secondary school.

Filho and Lorena (2000) have presented an evolutionary approach to school timetabling. In their constructive approach, the problem can be modeled as a basis to build a feasible timetable of teachers and classes on specified time slots. It is a process of setting up a sequence of a teaching session in a specified time slot and need to satisfy in various types of constraints. The combination of teachers and classes are used to form conflict-free clusters for each time slot. Binary strings were used to represent the combination is grouped based on dissimilarity measurement. They also consider additional objective

such as waiting times between classes and teacher preferences. The computational tests consider four instances, corresponding to two typical Brazilian high schools and the results can be considered successful and it has become an important component of administrative school tools.

Lalescu and Badica (2003), investigate an approach for solving the university timetabling problem by using a GA. The experiment was done using an object-oriented program, C++. In the experiment, they reported that the valid schedule must satisfy as many as possible the soft constraints. They also presented an evolutionary program built on the skeleton of this GA, along with the obtained experimental results and conclusions. There were some improvements to the methods and parameters used in the previous experiments and they have concluded that the fastest converging program is obtained using the values 60% for mutation and 20% for crossover and according to them, this is consistent with the statement by Michalewicz , (1994) that crossover has a less role in evolution than it is accorded, and mutation is more important than it is usually thought.

## 2.6 Micro-GA

The standard GA is a one of the tools for many optimization problems. However because of it’s limitation such as time, a penalty is usually involved in the evaluation of fitness function for a large population. The term Micro-GA refers to a small-population genetic algorithm with reinitialization. There are several studies of Micro-GA has been

found in optimization domain used in University scheduling. The first idea of Micro-GA in term of theoretical aspect was proposed by Goldberg (1989) and noted that the small populations could be used successfully with GA that is called Micro-GA if the population is restarted with a sufficient number of times. Then, this idea was implemented by Krishnakumar (1989) to find the best string in GA population and most applications with Micro-GA use a population size of 5. An elitist strategy was used in his study to find the highest fitness as a winner in a population search space.

Abu-Lebdeh and Benekohal (1999) addressed on how to determine the best population size when using Micro-GA. The experiment done shows that mid-sized population (between 9 and 15) consistently exhibited lower internal variability defined as the variation in fitness value caused by changes in the initial population. There are also several related issues have been addressed in their work such as the issues of Micro-GA convergence behavior versus population size, the criteria for determining the best population size, Micro-GA internal variability and Micro-GA performance versus computational resources. They also proposed that some empirical approaches needed to determine the best population size.

After that, Knowles and Corne (2000) extend the Micro-GA for optimization in order to improve the performance of Pareto Achieved Evolution Strategy (PAES) version. Then, Coello and Pulido (2001), implemented the multi-objective optimization by using Micro-GA. They introduce Micro-GA with a small population and reinitialization process in multi-objective optimization problem that has computational cost.

Kazarlis, (Kazarlis et al, 2001) studied and proposed in their literature that Micro-GA as hill-climbing operator for Genetic Algorithm optimization. They combine a standard GA with the Micro-GA to produce a hybrid genetic scheme which is contrast to conventional hill climbers that attempt independent steps, and their Micro-GA operator performs the genetic local search. They also claimed that the Micro-GA operator is capable to evolve at random direction leading to better solutions in the search space. Sigl et al. (2003), use 3D cubes corresponding to rooms, days and timeslots to model the

timetable. In their method, every gene in an individual represents one class and each individual represent one timetable. They enhanced the performance of genetic algorithm by using modified genetic operators and tested their algorithm on small and large problem instances. The performance of the algorithm improved by using tournament eliminating selection which allowing greater population size.

Meanwhile Martins, (Martins et al, 2008) proposed Micro-GA as a "local search" operator for optimization with Genetic Algorithm. They employed Micro-GA and executed once in each iteration of Genetic Algorithm. Since the population of Micro-GA is a fraction of the Genetic Algorithm population, it is created according to a uniform distribution and if there is an enhancement of the current solution obtained, then at this point the current solution will be replaced with all other characteristics of GA are kept in the Micro-GA.

Selvi and Rajaram (2011) presented Micro-GA approach to window-constrained scheduling which is suitable in real time systems. They show that the Micro-GA algorithm is efficient for solving multi-objective window-constrained scheduling problem where multiple Pareto optimal solutions can be found in one simulation run. This approach is more suitable to find the solution of scheduling jobs with minimum number of consecutive jobs. By considering on a UNITAR environment, this approach cannot be applied because the jobs may assume as a teaching workload assigned to every lecturers in which it can have many consecutive jobs.

NguyenBa (2011) with their new approach is Hybrid GA-Bees Algorithm was introduced for solving real-world university timetabling in Vietnam. The hybrid algorithm is tested and the result demonstrates that their approach is able to produce high quality solutions. In their approach, they use the solution of the timetable by representing it as a fixed set of block elements in which each block element include course, first period, length and room. Using this approach, the mutation operation is done by choosing randomly one block element in an individual and then altering randomly new first period and new room. The mutation operation will take a long time to be completed with this approach if a new first period randomly selected is conflict to another block elements especially if there is exist a high length block element. This approach uses fewer variables which is in general is advantageous but it greatly increases the complexity of the constraint. In UNITAR environment, this approach is considered not very suitable because there is a block element with a high length to be assigned so that the first period randomly selected for a certain block element may simply conflict.

## 2.7 Conclusion

This chapter has reviewed both Genetic Algorithm and Micro-GA. In summary, Genetic Algorithm has been used for optimization problems. The optimization problems that concern to timetabling specifically when there are many too soft constraints on university still need to be explored because if there are many soft constraints to be fulfilled it will reach to become harden soft constraint. The further explanation of how the Micro-GA can be applied into the university timetable is presented in the next chapter. Thus, this study proposed a new encoding chromosome representation to enhance Micro-GA approach for soft constraint satisfaction problem in university timetable.

# CHAPTER 3

# RESEARCH METHODOLOGY

## 3.1 Introduction

This study has adopted a research framework proposed by Vaishnavi and Kuechler (2004) which is shown in Figure 3.1.

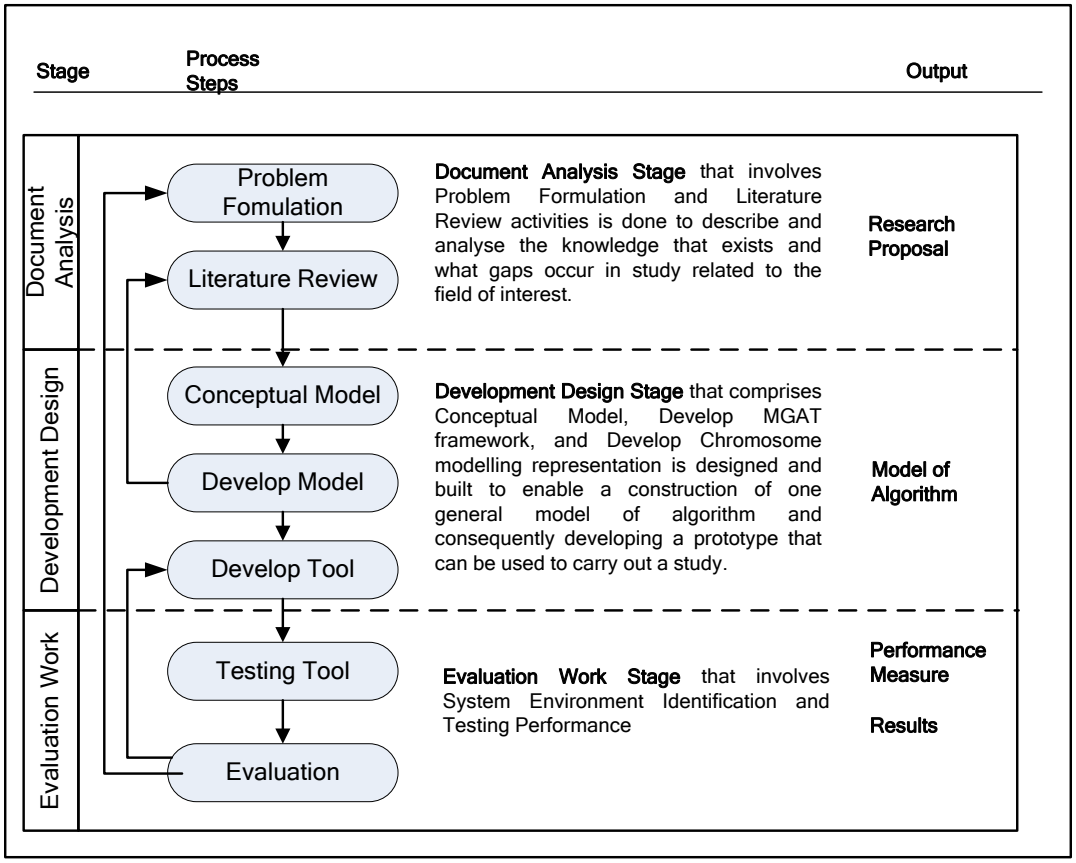



*Figure 3.1: Research Framework*

## 3.2 Design of Study

This study has been designed based on three stages which are document analysis stages, development stage and evaluation stage. These three stages are explained in details as following section.

### 3.2.1 Document Analysis Stage

The problem identification and application of Micro-GA are elaborated in the Document Analysis Stage. The increment of the performance of a time tabling system by minimizing the existing of the infeasible individuals in a chromosome will become a major concern of this study. In order to minimize the infeasible individuals in chromosome, this study will introduce a new encoding chromosome representation. One possible solution to cater the problem is by developing a conceptual of a Micro-GA Timetabling (MGAT) System. MGAT is an acronym for Micro Genetic Timetabling System, which is use micro genetic algorithm method in order to handle the soft constraint in time tabling problem. This study had proposed a Micro-GA approach for soft constraint satisfaction problems in a University Course Scheduling and some literature review had been done in order to fulfill the needs. In a literature review, the environment of time tabling system was explored in order to understand the fundamentals and challenger of the domain. Out of the detailed review and analysis emerged of the awareness of problems on which this study would focus: Will Micro-GA Timetabling System by minimizing the existing of the infeasible individual in a chromosome will affect the performance of a timetabling system?

### 3.2.2 Development Stage

The arrangements and the utilization of resources as well as the activity of gathering information and requirement needed in the Development Design Stage. From the review and information gathered, a tentative design had been done in order to develop a conceptual model.

#### a. Conceptual Model of MGAT system

This study proposed a conceptual model that will help to generate a best feasible timetable for UNITAR. This conceptual model is the modification of the existing model of a UNITAR timetable system to a new system that is called MGAT system as shown in Figure 3.2.

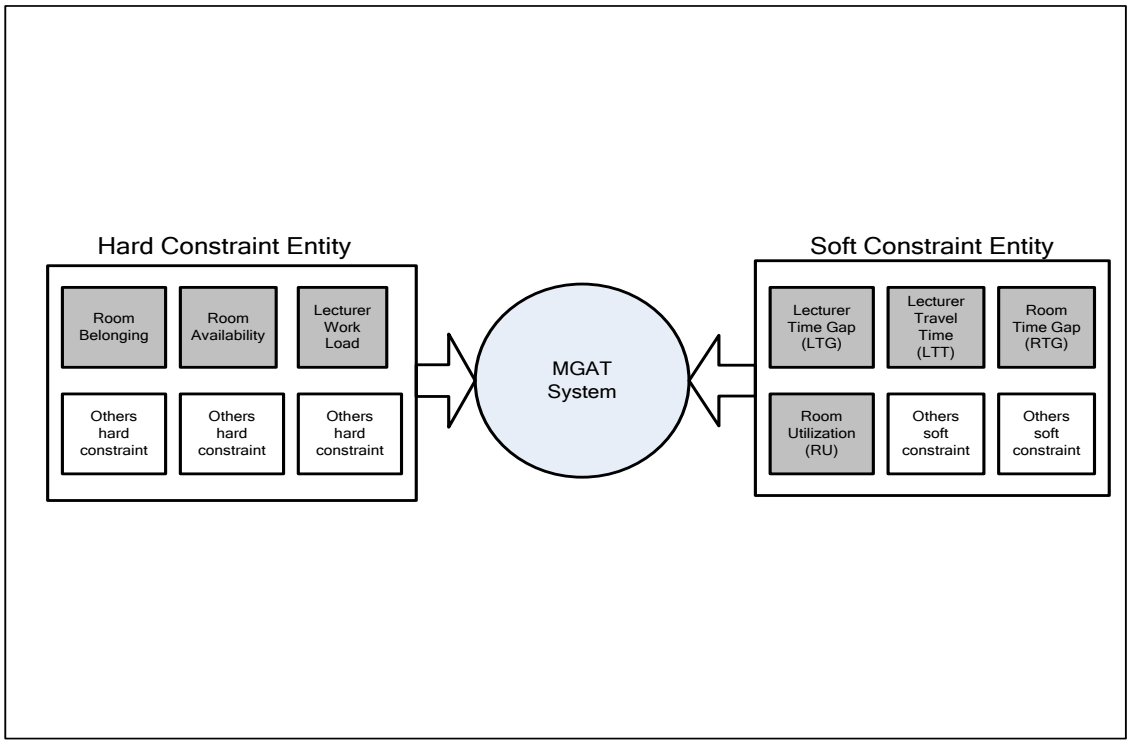



*Figure 3.2: A Conceptual Model of MGAT System*

In order to develop MGAT system successfully, a few components of the hard constraint and the soft constraint requirement should be organized to maximize the usage of resources. This study only considers four components of the soft constraint entity which are lecturer time gap; lecturer travel time; room time gap and room utilization while only three components of hard constraint will be mainly used such as room belonging, room availability and lecturers work load. The integration of all these components in a conceptual model of MGAT system is expected so that a good genetic algorithm on generating a timetable can be developed through this model.

**b. Develop MGAT framework**

The modeled process of MGAT has shown as Figure 3.3. There are four processes involved in this model, which are applying the GA, constructed a Micro-GA, implementation of Micro-GA and evaluation of Micro-GA.

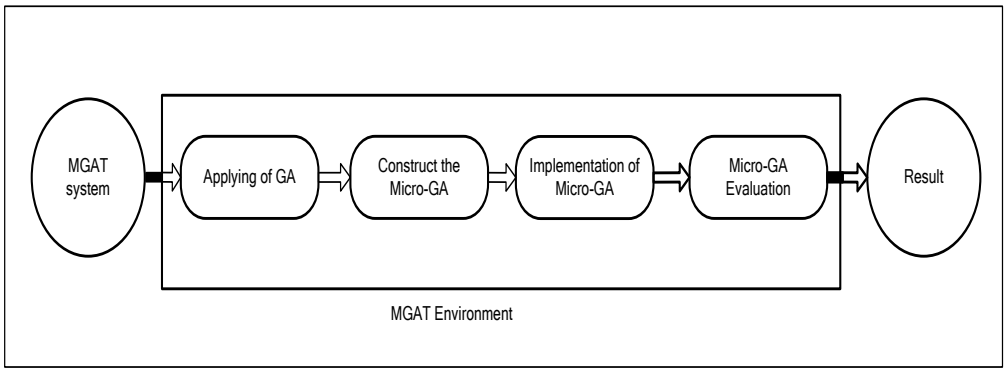



*Figure 3.3: MGAT Framework*

### c. Develop Chromosome modeling representation

This section shows how the genetic algorithm can be applied to the MGAT environment. The chromosome representation for population is also an essential factor of GA that affect the speed and quality of the final result as well as the efficiency and performance. Therefore, the most suitable chromosome representation for the population must contain all the information that represent about the solution to generate a good quality feasible timetable.

#### *i. Chromosome Representation for MGAT system*

The chromosome of rooms is chosen to represent a day of timetable solution is a 1-dimensional array *T*, *i*=l, ..*N* is shown in Figure 3.4 by assuming that the element of an array is represented as a time in 30-minute granules. The working days are Monday to Friday (5 days total) thus vector with size 19 slots (08.30am -05.30pm) $\times$ 5 $\times$ numbers of rooms, *N*. The constraints are always satisfied by its representation, for example, each lecturer cannot be assigned to more than one room for each period since there cannot be more than one element in each cell of the array.

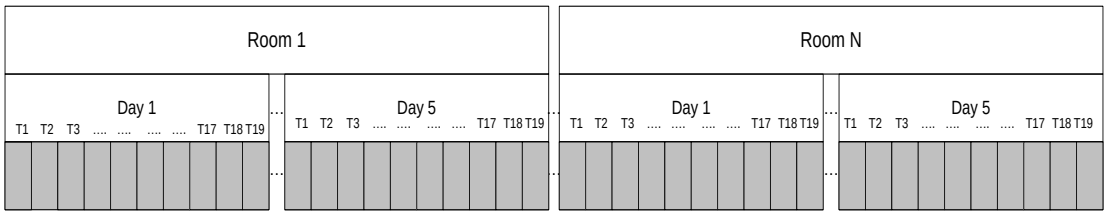



*Figure 3.4: One-dimensional array for chromosome representation*

This representation is chosen because:

i. The room assigned to the lecturer $L_i$ at time slot $T_i$, is subject to hard and soft constraint.
ii. $T_i$ can be lecture timetable where the element represents the classes at time and day.

The problem definition of this study has been identified that:

i) A set of *N* Lecturer's assigned subject mapped, $L = \{L_1, L_2, L_3 \cdots L_N\}$

ii) 95 time slots

iii) A set of rooms, *R*

It deals with a set of subject assigned to the lecturer, $L = \{L_1, L_2, L_3 \cdots L_N\}$ that need to be mapped to 95 time slots subject to a variety of hard and soft constraints. The objective of this problem is to satisfy the hard constraints and to minimize the violation of the soft constraints in order to construct a feasible schedule.

### *ii. Fitness Function for MGAT system*

The fitness function of each chromosome is evaluated by examining each soft constraint. Each soft constraint is assigned a penalty value and contributes to the fitness function for each constraint violated.

Fitness value $= \sum_{i=1}^{n} P(i)$ such that $P(i) = \sum_{j=1}^{m} \alpha$

where $P(i)$ denotes the summation of penalty values for constraint, *i*.

$\alpha$ denotes the penalty value associated with the violation at $j_{th}$ detected for constraint *i*

*n* is the number of constraints, and

*m* is the number of violations for constraint, *i*

The genetic algorithm employs a fixed number of generations with the population size the same from one generation to the next. When a feasible timetable is found or a preset number of generations was reached the algorithm will be terminated. Figure 4.4 shows the function to evaluate the soft constraint for generation.

### *iii. Soft constraint Value*

In order to construct Micro-GA time tabling that concern on soft constraints, the soft constraints are usually based on the request from lecturer or any requirement needed for the timetable could be more faithfully represented as preferences. The request from lecturers for their preferences suggests the use of preferences or in general as a soft constraint. The soft constraint value is the way to associate a certain element with a level of preferences, level of importance, weight or preference value. Such levels are usually ordered, and the order reflects that some levels are better than the other. For example, Table 3.1 shows setting of soft constraint value based on levels of preferences, a method to identify it and the operator used.

*Table 3.1: Setting of Soft Constraints Value*

| Type of Soft Constraint | Soft Constraint Requirement | Scale | Value | Description | Operator used |
|---|---|---|---|---|---|
| Soft Constraint 1, *SC1* | Maximize the time gaps within the schedule of lecturer, $L_i$ | 1 to 5 | 1= 1 hour gap<br>2 =2 hours gap<br>5= 5 hours gap | For each adjacent slot scheduled for lecturer,$L_i$ calculates the free time (in hours) available and chosen be made based on the maximum free time not greater than 5. | Max operator |
| Soft Constraint 2, *SC2* | Maximize the time gaps of room scheduled, $R_i$ | 0 to 2 | 0=none<br>1=1 hour<br>2=2 hours | For each adjacent slot scheduled, for rooms,$R_i$ calculate the free time (in hours) available and chosen be made based on the average free time. | Average operator |
| Soft Constraint 3, *SC3* | Maximize the travel time of lecturers and students between rooms within the campus | 0 to 15 | 0=none –assume because in the same room or adjacent room<br>5 = 5 minutes<br>10 = 10 minutes<br>15=15 minutes | For each lecturer session scheduled, calculate the time taken (in minutes) to travel from one location to another. Among the set time, chosen be made based on the maximum time taken. | Max operator |

For the soft constraint 1, the scale used is set from 1 to 5 in which 1,2,3,4 and 5 are referring to number of hour's gap respectively and the higher the scale, the better time gaps. For the soft constraint 2, each room scheduled may have time gaps between two sessions. The scale used is set from 0 to 2 hours gaps the rooms are vacant. The value 2 is the highest level and could be considered the better. For the soft constraint 3, the scale used are 5, 10 and 15 in which these number is referring to travel time for lecturers and

students between room in minutes. The higher level could be considered the better.

Thus, each of soft constraint can be defined as the following equations:

$$SC1_{\max} = \sum_{i=1}^{n} G_i \times s1$$

where

$s1 = 1,2,3,4 \text{ or } 5$

Soft constraint 2,*SC2* = average of $\sum_{i=1}^{n} G_i \times s2$ where $s2 = 0,1,\text{or } 2$

Soft constraint 3,*SC3* = maximizes of $\sum_{i=1}^{n} G_i \times s3$ where $s2 = 5, 10 \text{ or } 15$

Total soft constraint, TSC=*SC1+SC2+SC3*

### *iv. Calculation of Fitness Function*

The fitness function is committed in solving the suitability of a timetable represented by the chromosome. The calculation is based on the evaluation of "penalty" if it is breached. The fitness of a chromosome is penalized where it can be defined as $F(x) = \rho\alpha$

$\alpha$ is the total of infeasibilities for the timetable $S$

$\rho$ is the penalty value of infeasibility for the timetable $S$

The following is some of the infeasibility for the timetables $S$:

i. Various teaching time periods belonging to one lecture are not given in the same room.

ii. Various teaching time periods belonging to one lecture are not given on the same day (the time 114 is the last on Saturday, the time 115 is the first on Sunday).

iii. Lecture time periods are not in the sequence (for triple periods are optimal, 5+6+7 or 7+6+5, but it is wrong 4+5+7).

iv. Insufficient room space (the room for 15 people is not sufficient for the group of 20 people).

v. One room is scheduled for two lectures at the same time.

vi. One lecturer is scheduled for two lectures at the same time.

The objective of the function $F(x)$ is to have as minimum as possible in order to find the optimal solution of soft constraint needed of time tabling problems in UNITAR.

### *v. Generation of Initial Population*

The generation of initial population is a process to build individual chromosome to get the first generation. At this stage, the combination between lecturer and subject (LS) will be generated randomly to the

chromosome slots, which represents time slots and rooms. Algorithm to generate an initial population is shown in Figure 3.5.

START

1. Get the parameters such as number of generation, number of population and code in relation between lecturer and subject (LS) assigned in the workload table to get the size of the chromosome.
2. Input probability rate for crossover and mutation
3. Generate randomly LS to form initial population
4. Perform adjustment on initial population
5. Perform calculation of fitness on initial population

END

*Figure 3.5: Process to produce Initial Population*

The number of generations will determine the maximum numbers of generation to be generated by the Genetic Algorithm whereas the parameter of population size will determine the number of individuals in every generation. After calculating the fitness for initial population, the system will produce chromosome for the timetable with the most minimum cost of fitness function due to fail to fit the constraint.

### *vi. Calculation of penalties of lecturer clashes*

Lecturer clashes occur when the lecturer is scheduled to teach a subject for more than one room at any time slot. The calculation of the penalties of lecturer clashes as shown in Figure 3.6.

```
do until (LS<=max)
    do until (slot[x] in day[i]<= 5 '5 days i from 1 to 5
        if LS in slot[x] at day[i] = LS then
            y=getlectureID(LS)
        end if
        do until  (LS<=max)
            do until (slot[x+1] in day[i]<= 5 '5 days i from 1 to 5
                z=getlectureID(LS)
            loop
        loop
    loop
    if y=z then
        NumofClash=CountClashes(slot[x] at day[i])
    end if
loop
penalties=NumofClash *1000
```

*Figure 3.6: Calculation of Penalties*

In this study, every clashes will be penalized with a cost of 1000 per clash. This algorithm shows that the calculation for lecturer clashes at a sequential array in slot x. Let the slot $x^{th}$ is represents the first slot for every $i^{th}$ day is on Monday. Thus, a sequential array of slot $x^{th}$ represents the time from 8 to 9 for every Monday in every room. If there are three same *lecturerID* at the $x^{th}$ sequential array, the lecturer clashes occur in 2 times and the cost of the penalties will be calculated as 2 *1000=2000.

```
Function  EvaluateSoftConstraint()
For Every timeslot(i)
    numofClashes=Countclashes(time slot(i))
    numofLecViolate=CountLecViolate(time slot(i))
Next i
EvaluateSoftConstraint= numofClashes+ numofLecViolate 'return the value
End Function
```

*Figure 3.7: The Techniques of Soft Constraint Evaluation*

The genetic algorithm discussed in this chapter was evaluated using a set of data sets based on actual data from UNITAR.

### *vii. Constructed Micro-GA*

A proposed Micro-GA in this study is based on the Algorithm 2. Firstly, the general genetic population, P is to initialize the best ten populations (Coello and Pulido, 2001) and assign the generation limit as 1000 (Eiben, Hinterding, and Michalewicz, 1999). At each generation, *g* the archive population is augmented from non-dominated population of $P_i$. For reasons of computational efficiency, the non-dominated population of $P_i$ is then reduced to create $P_a$. Then, each of individuals is selected based on the fitness value which is defined by *EvaluatePopulation()* as shown in Figure 3.7.

**Algorithm 2 : Micro-GA structure code**

```
Private Sub microGA(N, g)
        ' N = size of Population g= number of generation M=population memory
        Initialize Population P of size N
        Select randomly from sorted Population (best of 10 Population)
        Create archive population  from non-dominated members of M
        For i =1 to g
          Perform crossover
          For j = 2 to N
            Mutate member j
          Next j
          EvaluatePopulation()
          Resort current population, Pi
          Eliminate dominated members from Pi
          Add non-dominated members from Pi to M
        Next i
End Sub
```

This process is done to maintain the best individuals from each generation because after performing the crossover and mutation, possible loss of the best chromosome is very high. Elitism will be applied by replacing the worst chromosome of the next generation with the best from previous generations of chromosomes when the chromosome of the newest is not worse than the old.

This study follows the approach proposed by Coello and Pulido (2001), in which a Micro-GA with dual search spaces: the population space, which is used as the source of varieties approach and the external space, which is used to archive member of the Pareto optimal set. Population space is respectively divided into two portions: a replaceable and a non-replaceable portion.

Figure 3.8 shows the process flow of using a Micro-GA. The first process is to generate an initial random population, then the population feed the population space, which is divided into two portions as mentioned before. The first portion of the population spaces will never change during the entire run in order to provide the required variety of random population for the algorithm. Then, the initial population is taken from both portions of the population space in order to get a larger variety of population. During each cycle, the Micro-GA undergoes conventional genetic operators that are tournament selection, one-point crossover, uniform mutation and elitism. After one cycle completed, Micro-GA will choose two non-dominated vectors from the final population and then compare it with the contents of the external space or the replaceable portion of the population space. In the comparison process, all the dominated vectors will be eliminated from the external memory. The end of this comparison process will produce the replaceable part of the population space which is tend to have more non-dominated vectors and some of them will be used in the initial population of the Micro-GA to start new evolutionary cycles.

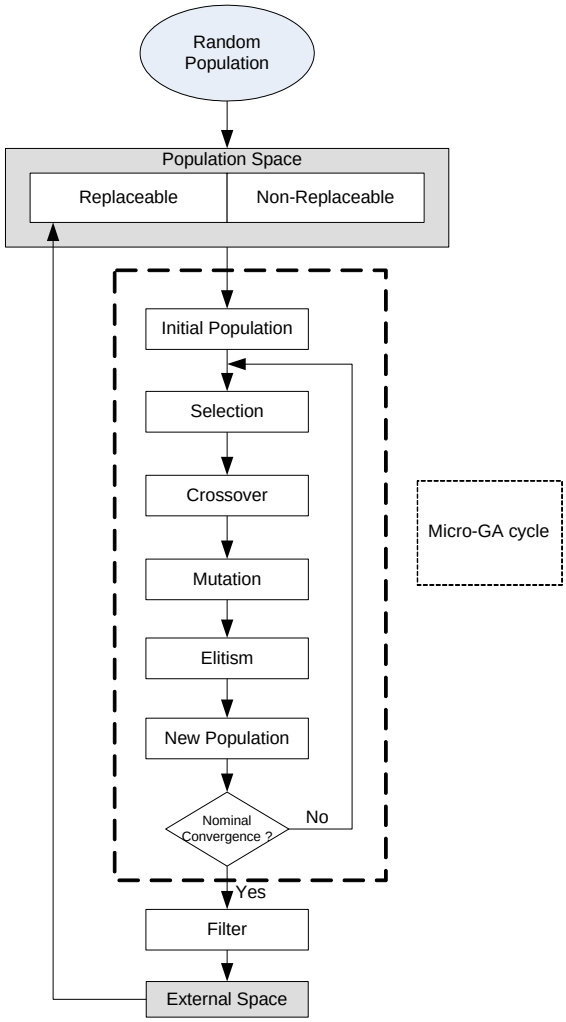



*Figure 3.8: The Process Flow of Using Micro-GA*

This approach also uses three types of elitism. The first type is based on the idea that the non-dominated vector produced from each cycle of the Micro-GA is stored so that any valuable information obtained from the evolutionary process will not be lost. The second type is based on the idea that the replacement of the population spaces by the nominal solutions, so the best solution found will gradually meet since the crossover, and mutation will have a higher probability of reaching the true Pareto front of the problem over time. This idea was implied by Goldberg (1989). The third type of elitism is applied at certain intervals. The intervals are defined by a parameter called “replacement cycle". By taking a certain amount of points from all the regions of the Pareto front generated so far and used it to fill in the replaceable space. Depending on the size of the replaceable portion, as many points chosen from the Pareto front as necessary to guarantee a uniform distribution.

This process is intended to use the best solutions generated so far as the starting point for the Micro-GA. This also avoids that the content of the replaceable portions will become homogeneous. The pseudo-code of the algorithm as shown in Figure 3.9.

```
START
    Generate starting population P of size N
    Store its contents in the population space M
    i=0
    While (i< Max)
        Get the initial population for the micro-GA (Pi)from M
        Do until(nominal convergence is reached)
            Apply binary tournament selection based on nondominance
            Apply one-point crossover and uniform mutation
            to the selected individuals
            Apply elitism (retain only one nondominated vector)
            Produce the next generation
        Loop
        Copy two nondominated vectors from Pi to the external memory E
        if full(E)
            call adaptive grid(indb)
        end if
        increment i
    Wend
END
```

*Figure 3.9: The pseude-code of Micro-GA*

### 3.2.3 Evaluation Stage

In the Evaluation Work Stage, it involves testing and evaluation activities to measure the performance of a proposed approach and make a conclusion based on the result. The output of this stage will support the conceptual model that has been developed. This stage will be divided into two parts namely system environment identification and testing performance. The activities involved in system environment identification are including types of university courses, availability of resources, rules for time tabling in the university, university time tabling resources and requirement of GA performance. In terms of testing performance, there are two activities will be measured, which are comparative test and usability performance.

### *a. System Environment Identification*

The system environment for the evaluation stage will be explained a detail such as types of university courses, availability of resources, rules for time tabling resources, university timetabling resources and requirement of GA, GSGA and Micro-GA performances.

#### *i. Types of University Courses*

A course offered by any faculty in the university may comprise of lectures and/or work in a lab session. The lectures are delivered by lecturers or teaching staffs, and they can choose to carry out the weekly requirement in single-period or multi-period sessions.

Time slots assigned for lectures, may be changed by the lecturers but need to be informed to the executive whose updating the resource utilization of the information. Lab session is usually part of a certain course, and the group of students is split into several sub-groups for training. In UNITAR, the courses offered in every semester are categorized as either core or electives. Time tabling becomes more complicated, given that some courses may overlap in which certain subject may be shared between faculty, and their schedule should be inconvenient for all students. On the other hand, different students may belong to lab sessions may pose many difficulties in controlling student schedules. The quality of rules placed by schedulers to make timetables easier for students have limited usage, since each student has his or her own schedule. However, in UNITAR, some considerations for

improving course attendance is by giving preferential treatment such as are generally accepted as ‘‘hard’’ courses.

### *ii. Availability of resources*

Duties of administrative work consume a number of periods during the week for everyone in the faculty. In each faculty, to make things more controllable, the faculty meetings are scheduled for a fixed day and time interval, so that courses for all participants are blocked to be scheduled for that time.

Availability of classrooms are defined by the system followed by the university. Some classrooms may be considered available during all time slots; However, other classrooms might have limited availability. For example, in UNITAR, it is common for several faculties to share some classrooms, so that one faculty has access to a given classroom some days of the week and other faculty the rest of the day. The sharing of classrooms is common for big labs, auditoriums or theater rooms.

### *iii. Rules for time tabling in the university*

Set up a timetable for UNITAR in each semester is a complicated process that administrators usually run a week to generate a complete timetable which can not be violated (hard constraints) and improve the quality (soft constraints).

Hard constraints for the university time tabling are regulated by the following basic rules:

i. Two or more courses can not be scheduled at the same period for the same lecturer or for same group of students.
ii. Two or more lecturers can not be assigned to the same group of students.
iii. Two or more classrooms can not be assigned to the same course and to the same group of students.
iv. A timetable has to be completed. A timetable is considered complete when all courses planned for every group of students appear in the timetable, with the right amount of time periods for every course and every portion of each course. Also, when every academic staff is assigned the total number of teaching periods that the faculty requires.
v. Any request for sessions of consecutive teaching periods scheme (single-period or multi-period) session must be catered. The timetable has to be scheduled for a given scheme request by lecturer such as

"2+2" or "4+0" and that the responsible lecturer may choose to follow.

Similarly, soft constraints for the university time tabling are regulated by the following quality rules:

i. Any request by lecturer for teaching in a specific time intervals should be obeyed as possible. Lecturer may prefer to teach a given course during the morning session or another during the afternoon or evening session.
ii. Any teaching session during lunch hours should be avoided if possible. Most lecturers and students prefer an empty slot around lunchtime, so they can rest before they go on with their classes.
iii. Minimize classroom changeovers for students if possible. Student may attend classes in room which is located at a same building block. It is preferable for them to stay in the same building block rather than they need to travel with long distance for make a change from one session to another.f

***iv. University Timetabling Resources***

In order to tackle time tabling problems, this study presented well-structured university timetabling resources as shown in Table 3.2.

*Table 3.2: University Timetabling Resources*

| No. | Resources | Value |
|---|---|---|
| 1 | Numbers of courses | 69 |
| 2 | Numbers of different lectures | 121 |
| 3 | Lecture types (theory/lab practices) | 2 |
| 4 | Numbers of lecturers | 23 |
| 5 | Numbers of rooms | 36 |
| 6 | Numbers of days | 5 |
| 7 | Numbers of time-periods (hours) within a day | 19 |

Based on Table 3.2, there are two attributes of university time tabling resources, which are the descriptions of resources and the quantity of the resources. The number of courses represents the total number of subjects to be taught by lecturers for each semester. There are 69 numbers of courses offered in a normal semester. In this sample 69 courses are to be distributed to 121 different slots of lectures (classes). Whereby this slot of lectures is categorized into 2 types of lectures, laboratory classes and normal face to face lectures. The time of the whole week is the natural number in the interval from 1 to 95. For example, the maximum time is 95, on Monday the first period is 1 and the last is 19, on Tuesday contain 20, …38, and so on.

There are five requirements of hard constraints have been identified based on Table 3.3. In addition, there are three soft constraints requirement has been identified as shown in Table 3.4.

*Table 3.3: Hard Constraints of the time tabling specification*

| No | Hard Requirement |
|---|---|
| 1 | No resource (lecturer or room) to be assigned to different events at the same time. |
| 2 | There are maximum numbers of time periods per day (19), which cannot be exceeded. |
| 3 | Each lecture to be held in a room belonging to a specific set of valid rooms for the lecture. |
| 4 | Each lecture to be assigned to a lecturer who belongs to a specific set of lecturers who can deliver the lecture. |
| 5 | Specific lectures must be rigidly assigned to specific lecturers. |

For all hard requirements, there are conditions to comply to. For example, any classroom or laboratory cannot be assigned for more than one event, which will cause redundant in timetable scheduling. The maximum timetable slots are tabulated into 19 slots per day and cannot be exceeded. Each lecturer must be assigned to specific classroom, with identified courses on prescribed time slots.

*Table 3.4: Soft Constraints of the time tabling specification*

| No | Soft Requirement |
|---|---|
| 1 | Maximize the time gaps within the schedule of each lecturer |
| 2 | Maximize the time gaps of each room scheduled |
| 3 | Maximize the travel time of lecturers and students between rooms within the campus |

***v. Requirement of GA, GSGA and Micro-GA Performances***

The GA system was written in Microsoft Visual Basic 6.0 and simulations were run on an Intel Dual-Core 1.86 GHz processor with 2GB of memory and Windows XP. The parameters used to test genetic algorithm's performance are listed in Table 3.5.

*Table 3.5: The genetic algorithm parameters*

| Parameter | Value |
|---|---|
| Num. of Timetables Generated | 50 |
| Initial Population Size for Algorithms | 1000 |
| Tournament Size for GA | 30 |
| Num. of swaps during mutation | 30 |
| Num. of Generations for GA | 70 |

The performance of the mutation operator for the soft constraint genetic algorithm was studied in detail for this purpose. By comparing the fitness of the offspring with the parent will ascertain whether the mutation operator was actually improving the quality of the timetable. It was found that in a number of cases, there was no improvement in the fitness of the offspring. The further swaps are not performed if the fitness is better, and the timetable is returned as the offspring. If there is no improvement, all swaps are performed and the offspring returned is the timetable on which all swaps were performed. The mutation operator for the hard constraints was also adapted to reproduce the parent if the offspring generated was not at least as fit as the parent. It was also

found that a decrease in tournament size (from 30 to 10) and an increase in the number of generations and mutations swaps (from 50 to 75) was more effective.

### *b. Structure of Testing Performance*

The structure of testing performance for the evaluation stage will be described such as a group of problem instances, data set used and parameter setting. There are three different groups of problem instances to be used for testing performances which are small, medium and large and this set of benchmark problem instances, which were proposed by Ben Paechter (2006) as shown in Table 3.6

*Table 3.6: Group of Problem Instances*

| | Number of lecturer session | Number of rooms | Number of lecturers | Number of soft constraints |
|---|---|---|---|---|
| Small | 1-150 | 1-15 | 1-30 | 1-10 |
| Medium | 151-300 | 16-30 | 21-50 | 11-20 |
| Large | 301-500 | 31-45 | 51-80 | 21-30 |

Based on the UNITAR environment, this study has also identified a several data set used for a comparative test between proposed Micro-GA and GSGA, which is shown in Table 3.7. These data sets are separated into three different groups such as small, medium and large.

*Table 3.7: Data set used*

| Data set | Problem Instance | | |
|---|---|---|---|
| | Small | Medium | Large |
| Number of lecture sessions | 100 | 250 | 400 |
| Number of Rooms | 10 | 20 | 40 |
| Number of Lecturers | 20 | 40 | 60 |
| Number of Soft Constraints | 5 | 15 | 30 |
| Maximum run time | 100 | 500 | 1000 |

The maximum run time was set for each problem instance and both algorithms were tested in 30 runs. The maximum run time was set to 100,500 and 1000 for small, medium and large instances respectively (Socha, Knowles, and Samples, 2002). The processing time will become longer as the larger of the problem instances will requires a larger search space. Besides, there are several parameters setting (Jat and Yang, 2011) used in this study, which is shown in Table 3.8. Both algorithms (Micro-GA and GSGA) are run by using all combinations of these parameters.

*Table 3.8: Parameters Setting*

| Parameters | Value | | | |
|---|---|---|---|---|
| Population size | 10 | 50 | 100 | 250 |
| Mutation Probability | 0.25 | 0.50 | 0.75 | 0.90 |
| Crossover Probability | 0.25 | 0.50 | 0.75 | 0.90 |

## 3.3 Conclusion

This chapter describes the research methodology that has been done in this study, which is divided into three phases.

The first phase is the document analysis phase. The result from the activity in this section is the research proposal. Several activities in this phase such as to propose Micro-GA approach for soft constraint satisfaction problems in a University Course timetabling and do the literature review regarding scheduling system environment.

The second phase discusses the design development phase which includes concept models, design models and design tools. Conceptual model has been developed to help in generating UNITAR through the proposed timetable MGAT system. A process model for MGAT also designed in which the model begins with the applying of GA, design of Micro-GA, implementation and evaluation of the Micro-GA. In the design tools of the Micro-GA, the selection of the representatives of the chromosome, the computation of the fitness function and the calculation of a number of soft constraint values are also discussed in this section.

Finally, in the third phase is the evaluation phase which involves testing and evaluation on the performance of the proposed approach.

# CHAPTER 4

# ANALYSIS AND RESULT

## 4.1 Introduction

For implementation of the proposed model, the testing and analysis results produced by MGAT system were conducted. The testing phase was divided into two phases. The first phase is to compare the performance proposed Micro-GA and GSGA (Jat and Yang, 2011) where GSGA will be used as a benchmark of this comparative test. The GSGA was chosen as a comparative study with regards to the similarities and the features offered by the algorithm which can be compared with the proposed algorithm as well as it can be tested by using the same system environment. Besides, the second phase is mainly focused on the testing performance of MGAT system. Actually, this testing performance which use all data sets is to show that the feasible timetable can be produced with the best soft constraint cost and the time taken by the MGAT system. Table 4.1 shows the summary of the Test and Analysis phases.

*Table 4.1: Phase of Testing*

<table>
<tr><td colspan="6">Test and Analysis Phases</td></tr>
<tr><td rowspan="3">Comparative Testing</td><td colspan="5">Testing the Performance of MGAT system</td></tr>
<tr><td colspan="3">Testing on soft constraint</td><td rowspan="2">Fitness Performance</td><td rowspan="2">End-user Usability</td></tr>
<tr><td>Measure of time taken</td><td>Effect of different crossover rate</td><td>Effect of different population size</td></tr>
</table>

## 4.2 Comparative Testing

The first testing has been done to evaluate the performance of the proposed Micro-GA and GSGA where GSGA has been used as a benchmark of this comparative test. Based on the Table 4.2, the objective value of proposed Micro-GA decreased gradually starting from first generation to generation-21 with crossover rate of 0.25. This is also occurred for GSGA with same crossover rate where the objective values are 728, 622, 527, 439 at generation 1, 5,9 and 13 respectively. Next, the decreasing of objective value for proposed Micro-GA is converged to 200, which is starting from generation-25 to generation-69 at crossover rate of 0.25. The objective value for GSGA at same crossover rate is seemly decrease parallel with the decreasing in proposed Micro-GA.

At crossover rate of 0.5, the objective value for both algorithm shows that the difference of the each value decreased are very small (range 5 to 10) starting from first generation to generation-69.

At crossover rate of 0.75, the objective values for proposed Micro-GA show a small decreasing (range 5 to 7) starting from generation-1 to generation-37 but in generation-37 to generation-41, it shows a drastic change of objective value from 426 to 320 and then remains with small changes (range 1 to 3) after generation-41 until generation-57. In contrast, although at same crossover rate, the objective value for GSGA shows that the decreasing value occurred with a gap around 6 to 10 starting from generation-1 to generation-37 and suddenly at generation-41 the objective value dropped from 525 to 421 where the distances are very big, and then it slowly decreased until generation-69.

At crossover rate of 0.90, the objective values for proposed Micro-GA show a decreasing value drastically from the value of 1213, 987, 845,561 at generation-1, generation-5, generation-9 and generation-13 respectively before it reaches at generation-17 with a value 554 and continually decrease at value 550 in generation-21. Then the value remains decreasing with very small difference for each generation starting from generation-25 to generation-69. Meanwhile, the rate of change of the objective value for GSGA shows a slowly decreased even it started with lower objective value than the proposed Micro-GA at the first generation.

*Table 4.2: Result of Objective Values of Micro-GA and GSGA with Different Crossover Rates*

| Generation | Objective Value | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Proposed Micro-GA | | | | GSGA | | | |
| | Crossover rate | | | | Crossover rate | | | |
| | MGA-CR=0.25 | MGA-CR=0.50 | MGA-CR=0.75 | MGA-CR=0.90 | GSGA-CR=0.25 | GSGA-CR=0.50 | GSGA-CR=0.75 | GSGA-CR=0.90 |
| 1 | 630 | 446 | 456 | 1213 | 728 | 446 | 556 | 860 |
| 5 | 527 | 440 | 450 | 987 | 622 | 437 | 550 | 777 |
| 9 | 435 | 423 | 435 | 845 | 527 | 417 | 535 | 660 |
| 13 | 350 | 426 | 436 | 561 | 439 | 417 | 536 | 661 |
| 17 | 350 | 426 | 436 | 554 | 436 | 414 | 536 | 654 |
| 21 | 356 | 433 | 434 | 550 | 439 | 418 | 534 | 650 |
| 25 | 289 | 420 | 430 | 289 | 369 | 402 | 530 | 393 |
| 29 | 290 | 421 | 430 | 292 | 367 | 400 | 530 | 392 |
| 33 | 293 | 420 | 430 | 291 | 367 | 396 | 530 | 391 |
| 37 | 290 | 415 | 425 | 290 | 361 | 388 | 525 | 390 |
| 41 | 289 | 310 | 320 | 272 | 357 | 280 | 420 | 372 |
| 45 | 286 | 311 | 321 | 270 | 351 | 278 | 421 | 370 |
| 49 | 286 | 308 | 318 | 250 | 348 | 272 | 418 | 350 |
| 53 | 285 | 305 | 315 | 249 | 344 | 266 | 415 | 349 |
| 57 | 260 | 300 | 310 | 230 | 316 | 258 | 410 | 330 |
| 61 | 258 | 248 | 300 | 224 | 311 | 203 | 400 | 324 |
| 65 | 257 | 245 | 280 | 220 | 307 | 197 | 380 | 320 |
| 69 | 250 | 241 | 281 | 221 | 297 | 190 | 381 | 321 |

Figure 4.1 shows the objective values as a quality of the solution for Micro-GA compared to GSGA. At some points, the objective values of proposed Micro-GA have drastically declined to reach the minimum objective value compared to the GSGA. In the example, at crossover rate of 0.90 for Micro-GA, starting at generation-13 the objective value has been drop 561 compared to 661 for GSGA and it continue occur at generation-21 then slowly to reach the minimum objective value.

This advantage of Micro-GA is the evidence that the Micro-GA can gives an improvement to the quality of individuals by storing part of previous good solutions, which is able to achieve smoother and faster the minimum objective value even at the small generation.

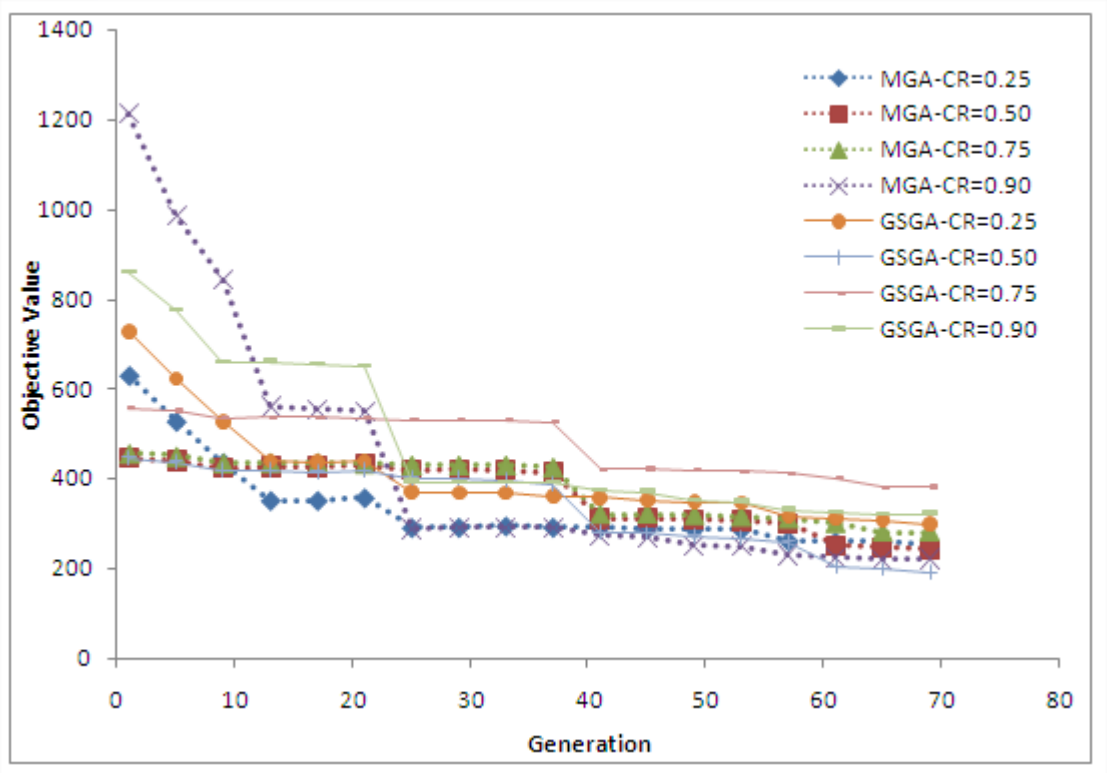



*Figure 4.1: Objective Value Over Generation For Different Crossover Rate*

### 4.3 The Effectiveness of MGAT System

The evaluation of MGAT system effectiveness are divided into three parts such as testing on soft constraint performance, fitness performance and end-user usability.

#### a. Soft Constraint Performance of Micro-GA

There are three types of testing that have been done throughout the soft constraint performance. The first test of soft constraint performance is to measure the time taken for each workload of lecturer which is shown Table 4.3.

*Table 4.3: Result of soft constraints performance over time taken*

| Data Set | Number of Soft Constraints | Time (mins) | Soft Constraints Cost |
|---|---|---|---|
| 1 | 10 | 7 | 181 |
| 2 | 11 | 10 | 226 |
| 3 | 12 | 10 | 253 |
| 4 | 13 | 18 | 278 |
| 5 | 15 | 21 | 322 |
| 6 | 18 | 21 | 160 |
| 7 | 20 | 25 | 146 |
| 8 | 22 | 24 | 172 |
| 9 | 23 | 26 | 165 |
| 10 | 24 | 28 | 150 |

Based on Table 4.3, the values of soft constraint cost increased from 181, 226, 253, 278 until 322 when the numbers of soft constraint increased from 10, 11, 12, 13 to 15, respectively. The time taken of the change has been started from minute 7 until minute 21. After the number of soft constraint reach 15, however the soft

constraint costs are decreased from 160 to 146 when the number of soft constraints increase from 18 to 20. Subsequently, the soft constraint cost increased from 146 to 172 with correspond to the number of constraints, which are 20 and 22, respectively. Then, the values of the soft constraint cost decreased to 165 and 150 when the number of soft constraint increased from 23 to 24. Thus, it is found that the best ranges of a soft constraint number are between 18 to 20 where the value of soft constraint cost are between 160 to 146.

Therefore, the result shows that the relationship between the number of soft constraint used, and the time needed for each workload of lecturer is a positive correlation. Thus, it is found that the best range of soft constraint number are between 18 to 20 where the values of soft constraint cost are between 160 to 146. Furthermore, the most important finding from the experiment is that the corresponding peak point is identified as a best solution for soft constraint cost.

The second test is to find out the effect of different level of crossover rate across a number of trials to the objective value of chromosomes. In this test, the initial population of size 10 (Beasly and Chu, 1996; Kwan et al., 1997; Khan 2003) at different crossover rate is used repeatedly. Based on Table 4.4, the fitness convergence of proposed Micro-GA can be shown by the objective value at a different level of crossover rate.

At crossover rate of 0.25, the objective value of proposed Micro-GA has declined quickly starting at 630, 527, 435 at generation 1, 5, 9 and 13 respectively. Next, the decreasing of objective value for proposed Micro-GA converged smoothly to value 200, which is starting from generation-17 to generation-69.

At crossover rate of 0.5, the objective value for proposed Micro-GA shows that the distance of the each value decreased are very small (around 5 to 10) starting from generation-1 to generation-37. Then the turning point of these smooth changes suddenly decrease to the low value which are 310 at generation-41 before continues to converge on the minimum objective value at generation-69.

At crossover rate of 0.75, the objective value for proposed Micro-GA shows that the decreasing patterns are quite similar to the decreasing objective value at crossover rate of 0.5.

At crossover rate of 0.90, the objective values decreased quickly starting at generation-1 to generation-9 and once again at generation-21 to generation 25. After that, the objective value again declines smoothly before it reaches at generation-17 with value 554 and then continues decrease smoothly with small distance at each generation starting from generation-29 to generation-69.

*Table 4.4: Result of different crossover rate to objective value*

| | Crossover rate | | | |
|---|---|---|---|---|
| | 0.25 | 0.50 | 0.75 | 0.90 |
| Generation | Objective Function | | | |
| 1 | 630 | 446 | 456 | 760 |
| 5 | 527 | 440 | 450 | 677 |
| 9 | 435 | 423 | 435 | 560 |
| 13 | 350 | 426 | 436 | 561 |
| 17 | 350 | 426 | 436 | 554 |
| 21 | 356 | 433 | 434 | 550 |
| 25 | 289 | 420 | 430 | 293 |
| 29 | 290 | 421 | 430 | 292 |
| 33 | 293 | 420 | 430 | 291 |
| 37 | 290 | 415 | 425 | 290 |
| 41 | 289 | 310 | 320 | 272 |
| 45 | 286 | 311 | 321 | 270 |
| 49 | 286 | 308 | 318 | 250 |
| 53 | 285 | 305 | 315 | 249 |
| 57 | 260 | 300 | 310 | 230 |
| 61 | 258 | 248 | 300 | 224 |
| 65 | 257 | 245 | 280 | 220 |
| 69 | 250 | 241 | 281 | 221 |

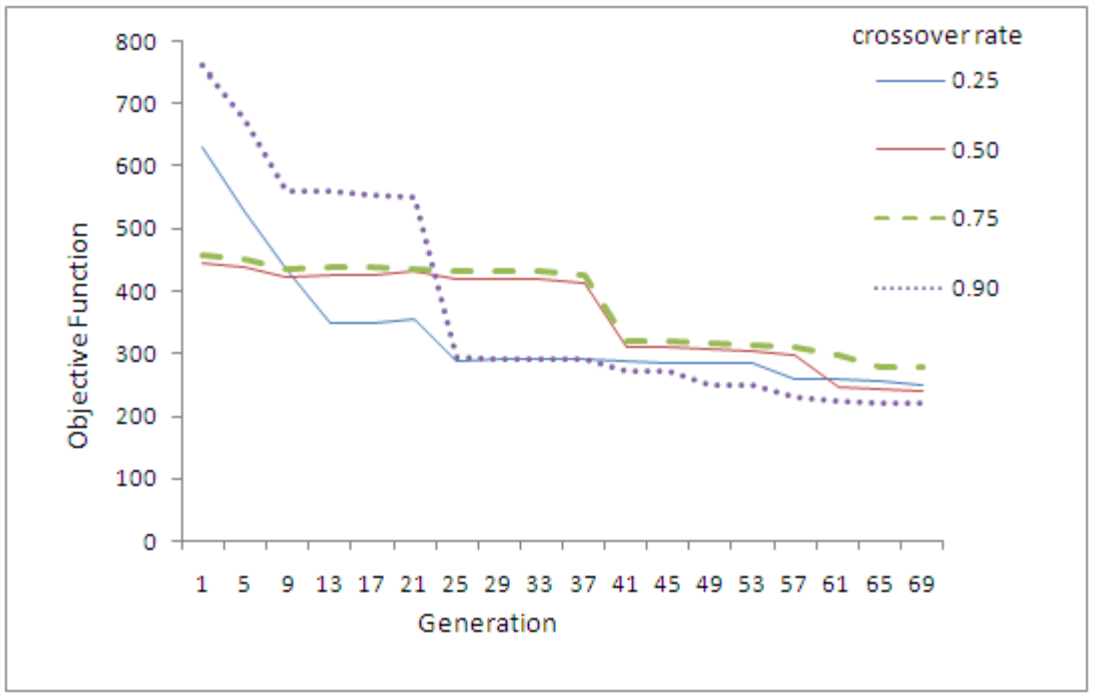



*Figure 4.2: Objective Function Over Generation For Different Varying Crossover Rate*

Based on Figure 4.2, the sequences of objective values are converged to the minimum objective value. The graph shows that the objective value has declined sharply at crossover rate 0.90. in generation-1 to generation-9 as well as at generation-21 to generation-25. At all other chromosome rate, the graph shows the smoothly decline of objective value to the minimum objective value. Particularly, at crossover rate from 0.75 to 0.9, it has an impact on the converge of the optimal result. This shows that the Micro-GA operations which can produce a chromosome with the objective values near to the optimal value are very quickly at certain generation. This occurs due to the different levels of constraints, the desired solutions found in the search space during searching or

the number of  the free period for a certain lecturer is quite small such as less than 0.5.

The third test is to determine the effect of different population size producing good results of a timetable to the search performance of Micro-GA in the MGAT system. Table 4.5 shows the performance of Micro-GA when using different population size. The size of population represents the diversity of the chromosome in a search space. A small population can reduce the computation in each generation, but it lacks sufficient diversity. The test was run to the different population sizes that are 10, 15, 20, 25, 30, 35, 40, 45,50, which result the time taken to reach the objective values are 0.0448, 0.0049, 0.0537, 0.0556, 0.0618, 0.0635, 0.0684, 0.0731 and 0.0734 respectively. From the result, the conclusion was made that on the small size of population, the Micro-GA can perform a fast calculation and less time is required to generate good solution.

*Table 4.5: Result of different size of population time taken to generate with corresponding objective function*

| Test | Population Size | Generation - th | Time (second) | Objective Function |
|---|---|---|---|---|
| 1 | 10 | 46 | 0.0448 | 760 |
| 2 | 15 | 54 | 0.0049 | 677 |
| 3 | 20 | 78 | 0.0537 | 560 |
| 4 | 25 | 65 | 0.0556 | 561 |
| 5 | 30 | 100 | 0.0618 | 554 |
| 6 | 35 | 90 | 0.0635 | 550 |
| 7 | 40 | 76 | 0.0684 | 293 |
| 8 | 45 | 110 | 0.0731 | 292 |
| 9 | 50 | 158 | 0.0734 | 291 |

**b. Testing of Fitness Performance for Micro-GA**

The efficiency of Micro-GA results from the use of small populations which lead to more rapid convergence and the frequent re-generation of random population members to ensure diversity during the search process. In this study, the Micro-GA is implemented as follows. One cycle of Micro-GA consists of evolving a small population of chromosomes of size *n*-based on the GA operations described earlier for a fixed number of generation *g*. At the end of this cycle, the chromosome with the highest fitness is identified. Then *n* chromosomes are generated based on mutating the fittest chromosome, and these newly generated chromosomes, together with the fittest individual from the previous Micro-GA cycle, constitutes a new population ready for the next stage of Micro-GA

operation. This cycling process continues until a desired level of performance criterion is achieved.

The fitness performance of Micro-GA compared to standard GA is shown in Table 4.6. For each iteration, the variation of fitness performance for Micro-GA and standard GA occurs with the small difference at the first iteration until iteration 3000th. The fitness value of Micro-GA remains smoothly increased but at a certain iteration, there are small declining before it reaches to the iteration 10000th. In contrast, the fitness values of standard GA are 490, 630 at iteration 3000th and 3500th respectively. The sudden and drastic increase of this value makes the difference of fitness performance between Micro-GA and standard GA become a larger distance in order to reach the best solution.

*Table 4.6: Result of different iteration over the fitness of standard GA and fitness of Micro-GA*

| Iteration | Fitness | |
|---|---|---|
| | Micro-GA | GA |
| 0 | 0 | 0 |
| 500 | 38 | 95 |
| 1000 | 90 | 150 |
| 1500 | 100 | 250 |
| 2000 | 230 | 300 |
| 2500 | 250 | 350 |
| 3000 | 280 | 490 |
| 3500 | 320 | 630 |
| 4000 | 356 | 670 |
| 4500 | 350 | 680 |
| 5000 | 358 | 685 |
| 5500 | 365 | 692 |
| 6000 | 370 | 695 |
| 6500 | 390 | 698 |
| 7000 | 380 | 700 |
| 7500 | 436 | 720 |
| 8000 | 457 | 725 |
| 8500 | 465 | 738 |
| 9000 | 470 | 746 |
| 9500 | 480 | 792 |
| 10000 | 483 | 798 |

The improvement in fitness evaluation of Micro-GA compared to standard GA is shown in Figure 4.3. In this example, after the execution about 10000 iterations, the fitness value of standard GA and Micro-GA approximately are 800 and 400 respectively. The advantage of Micro-GA is evident that the Micro-GA gives lower fitness prediction during the entire process of the optimization.

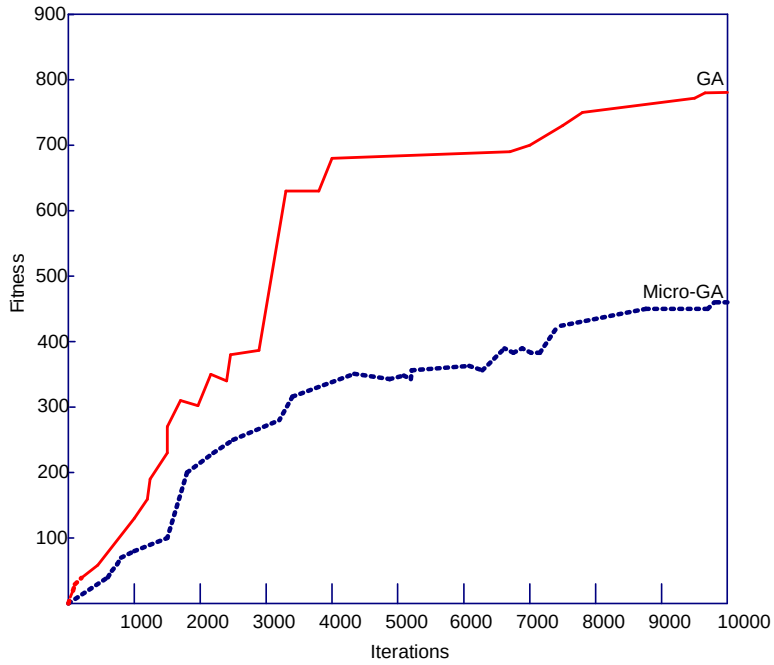



*Figure 4.3: Fitness comparison between GA and Micro-GA*

### c. Testing of End-User Usability

The usability testing has been performed to determine the level of usefulness, operability and ease of use of the MGAT system. This is conducted through the use of questionnaires given only to the people who involve in university scheduling. Based on recommendations by Neilsen (2006), the questionnaire should be given to 30 respondents, but in this research only 10 staff in the university academic department has been selected. Each of the respondents had been informed the objective, the description, the use the prototype and the functionalities of the system. This test is to obtain the level of satisfaction based on the ease of use, usability or operability of the prototype software. The measurement of the usability is based on the suggested by Davis (1989) and the sample questionnaire is attached in Appendix IV.

Table 4.7 shows the score point of the system aspect. The system aspects from B1 to B6 are about the perceived usefulness of MGAT system. (see Appendix IV) The system aspects from B7 to B12 were concerned with the ease of use of MGAT system. The overall satisfaction of using MGAT system by using the attributes of usability test represents in the aspect C1 to C6.

*Table 4.7: System Aspects*

| User Perceptions | | Number of Staff (frequency) | Score Point (%) | Average Score Point (%) |
|---|---|---|---|---|
| Perceived Usefulness | Aspect B1 | 10 | 80 | 87 |
| | Aspect B2 | 10 | 96 | |
| | Aspect B3 | 10 | 80 | |
| | Aspect B4 | 10 | 98 | |
| | Aspect B5 | 10 | 86 | |
| | Aspect B6 | 10 | 80 | |
| Perceived Ease of Use | Aspect B7 | 10 | 80 | 82 |
| | Aspect B8 | 10 | 80 | |
| | Aspect B9 | 10 | 86 | |
| | Aspect B10 | 10 | 80 | |
| | Aspect B11 | 10 | 80 | |
| | Aspect B12 | 10 | 88 | |
| Overall Satisfaction | Aspect C1 | 10 | 100 | 95 |
| | Aspect C2 | 10 | 100 | |
| | Aspect C3 | 10 | 100 | |
| | Aspect C4 | 10 | 80 | |
| | Aspect C5 | 10 | 94 | |
| | Aspect C6 | 10 | 98 | |

The collected data from the questionnaires had been analyzed using Microsoft Excel as a spreadsheet tool. Based on the score point collected, the score percentage of perceived usefulness (aspect B1 to aspect B6) stands at 87% as shown in Figure 4.4.

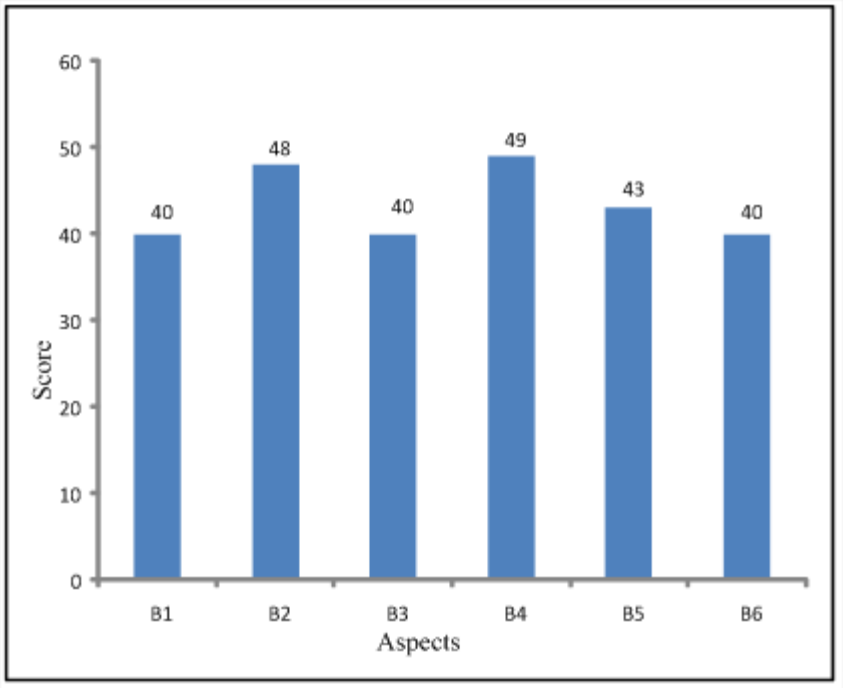



*Figure 4.4: System Aspect (Perceived Usefulness)*

In term of ease of use (aspect B7 to aspect B12), the score percentage stands at 82%. For the overall satisfaction (aspect C1 to aspect C6), the score percentage is about 92% in favor of this prototype.

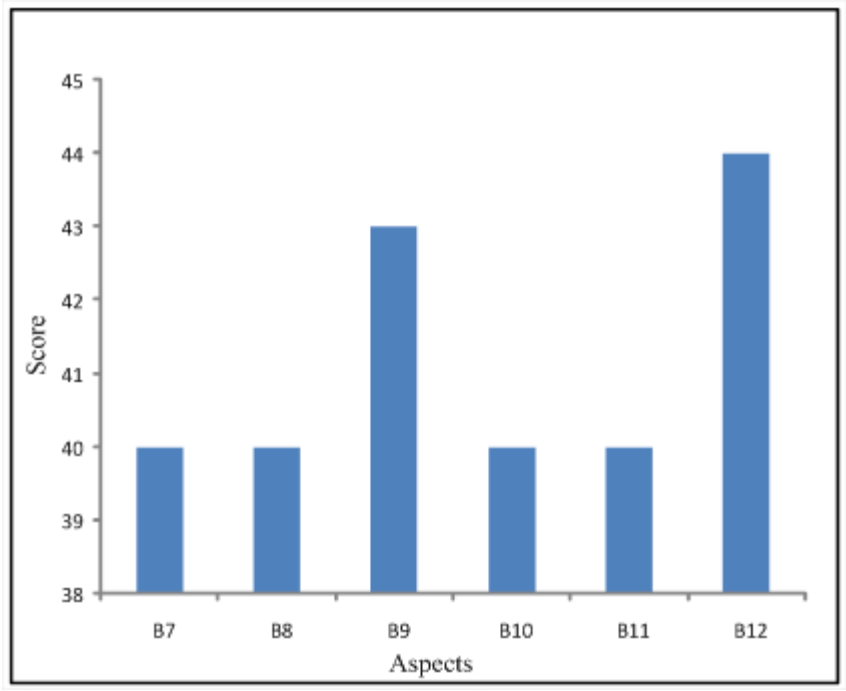



*Figure 4.5: System Aspect (Perceived Ease of Use)*

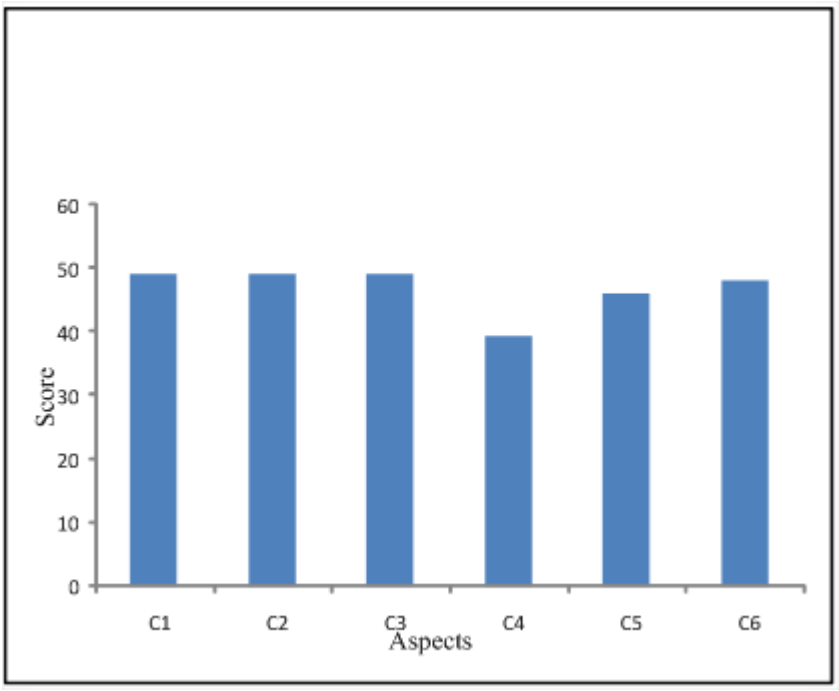



*Figure 4.6: Overall Satisfaction (Usability)*

## 4.4 Conclusion

This chapter describes two phases of the analysis and result that has been done in this study. The first phase is a comparative test between the proposed Micro-GA and GSGA. The second phase is about testing the performance of MGAT system in which it can be divided into three sections such as testing on soft constraint, testing the fitness performance of Micro-GA in MGAT system, and testing the end-user usability of MGAT system. The results of the test have been shown in tables or corresponding graphs.

The results of the first testing show the performance comparison between Micro-genetic algorithm, and GSGA is very encouraging. The sequence of objective values for micro genetic algorithm derived from the results are converged to the optimal value at a faster rate compared to the GSGA. This advantage can give the impression that the Micro-GA as a good algorithm to be remain used in MGAT system for UNITAR environment. The second testing has been done to measure the effectiveness of MGAT system in terms of the soft constraint performance. The analysis has shown that there is a good positive correlation between the numbers of soft constraint used, and the time need to generate the lecturer timetables taking in consideration of the soft constraint's cost for each lecturer. The encouraging result is also been obtained in the testing on how does the different level of crossover rate and population size can affect the convergence towards the optimal value. In terms of the testing on end-user usability, this test is actually to ensure the needs of the participant such as lecturers whose representatives have tested in the context of their work in UNITAR environment gives the positive feedback about the MGAT system.

# CHAPTER 5

# CONCLUSION AND RECOMMENDATIONS

## 5.1 Introduction

The entire research findings will be summarized in this chapter. By referring back to the problem statements, a brief introduction of the research has been discussed in Chapter 1. The literature review about the solving timetable problems and Micro-GA is elaborated in Chapter 2.

In Chapter 3, the methodology of the research is elaborated. The fundamental structures of a university timetable discussed along with the problem formulation and the conceptual model. The research then continues in the discussion on analysis and result of the testing data in Chapter 4 and it have been presented with a positive outcome. The performance of the algorithm using the Micro-GA was also presented.

Finally, the research was concluded and summarized in this chapter. The summary of the research as shown in Table 5.1. After going through the research process, the question in the research then has been answered.

*Table 5.1: Summary of the research*

| | | | |
|---|---|---|---|
| **Research problem (RP)** | Current timetabling systems using genetic algorithm (GA) that has the infeasible individual chromosomes will always be generated in the reproduction process may trapped into the local optima. | | |
| **Main Research question** | How to enhanced the functionality of the GA by introducing a new approach in repairing algorithm which could satisfactorily feasible to support soft constraint data? | | |
| **Sub questions (SQ)** | SQ1: What are the requirements needed to construct a system by using a new approach? | SQ2: How a new approach will be able to generate a feasible the individual chromosome? | SQ3: How a new approach will improves the individual chromosome? |
| **Main research objective** | is to use a Micro-GA approach for soft constraint satisfaction problem in university timetable. | | |
| **Sub objectives (RSO)** | SO1: To enhance Micro-GA by introducing a new encoding chromosome representation. | SO2: To introduce a new fitness function according to soft constraint scale. | SO3: To test and evaluate the effectiveness of the Micro-GA system based on soft constraint and end user usability. |
| **Research methods (RM)** | Study a problem formulation and literature review that describe and analyze the existing knowledge of Micro-GA. | Utilize all required resources, design a conceptual model of MGAT system, develop model process and tools. | Perform a comparative test between Micro-GA and GSGA, testing on soft constraint performance, fitness performance for Micro-GA and end-user usability. |
| **Outcomes (O)** | A new encoding chromosome representation be able to minimize the infeasible individual chromosome. | A new fitness function according to soft constraints scale can be used to produce infeasible individual chromosome. | The effectiveness of using Micro-GA system based on soft constraint, fitness performance and end user usability |
| **Contribution (C)** | A new framework and model for Timetabling System | | |

This study has been considered a highly constrained UNITAR timetabling problem and presented a successful algorithm to solve real-world problems as well as artificial test problems. The Micro-GA algorithm performs very well in the timetabling problems presented in this paper. This study has shown that the chosen parameter values were very good for the UNITAR timetabling problems.

The experimental result show that there are relationships between population size, number of generations converges, mutation rate, and crossover rate with the rate of random number injected into the population. The enhancement in producing the class timetabling in UNITAR can be achieved, which is one of the contributions of this study. Previously, to generate the timetable requires a long duration of time. By the application of proposed MGAT system, the complete scheduling result can be produced in a shorter time than before. This new solution will assists the scheduler to generate more efficient, effective and optimal timetable for every semester.

This study also describes on how Micro-GA can be used as a new framework and introduce an approach in scheduling for University course timetabling. An assessment of this new framework should be carried out to highlight the contribution made by other researchers where many interesting directions for future investigation such as more detailed and thorough analysis for improving should be explored.

## 5.2 Contribution

The contributions of this study is concern to the unique selection technique. In the selection technique, selects only a nondominated individual stored in the external space. The selected individuals are then cloned in a local search space. By using the nondominated selection, the Micro-GA realizes that the enhanced of the local search in the external space which is "less-crowded region" be a very good reason in computational efficiency.

## 5.3 Future works

This section will suggest several recommendations of future works for this research. In order to improve the capability of this prototype, future works can be undertaken as listed in the following:

i. The MGAT prototype can be expanded to generate class timetable for a whole UNITAR regional center where it was a very crucial need to be implemented.
ii. The MGAT prototype can be integrated to the UNITAR examination scheduling where there are many soft constraint need to be considered.
iii. Merge the other techniques (Hybrid System) such as Fuzzy Logic, expert system or Neural Network to improve the effectiveness and the efficiency of university timetable.
iv. Improve the additional function such as online application to let the lecturers do the predetermined of their preferences (soft constraint) so the system can summarize the variation of the soft constraint.

v. Improve the genetic operators by using a certain techniques to change the parameters of a Micro-GA may produce the better performance of Micro-GA for the university timetabling.

# REFERENCES


Abdullah, S., and Hamdan, A. R. (2008). A Hybrid Approach for University Course Timetabling, The International Journal of Computer Science and Network Security, Universiti Kebangsaan Malaysia, 8 (8), 127-132.

Abu-Lebdeh, G., and Benekohal, R.F. (1999). Convergence Variability and Population Sizing in Micro-Genetic Algorithms, Computer-Aided Civil and Infrastructure Engineering, Vol.14, No.5, pp. 321-334.

Abdullah, S., and Turabieh, H. (2008). Generating University Course Timetable using Genetic Algorithm and Local Search. A Proceeding Paper for the 3rd International Conference on Convergence and Hybrid Information Technology 2008, A Publication of IEEE, 254 – 260.

Abdullah, S., Turabieh, H., McCollum, B., and Burke, E. K. (2009). An Investigation of Genetic Algorithm and Sequential Local Search Approach for Curriculum-based Course Timetabling Problems, in Proc. Multildisciplinary International Conference on Scheduling: Theory and Applications (MISTA 2009), Dublin, Ireland (2009), pp.727-731.

Adamis, P., and Arapakis, P. (1999).Evolutionary Algorithms in Lecture Timetabling. Proceedings of the 1999 IEEE Congress on Evolutionary Computation (CEC ‘99), IEEE, 1999, 1145-1151.

AlSharafat, W.S., and AlSharafat, M.S. (2010)."Adaptive Steady Genetic Algorithm for Scheduling University Exams," in Proc.Inernational Conference on Networking and Information Technology (ICNIT), pp. 70-74.

Azimi, Z.N. (2004). Comparison of Metaheuristic Algorithms for Examination Timetabling Problem. Journal of Mathematics and Computing 16, 337–354.

Beasley, J.E., and Chu, P.C. (1996).A genetic algorithm for the set covering problem. European Journal of Operation Research, 94(2), 392-404.

BI WeiHong, REN HongMin and WU QingBiao (2006). A New Elitist Strategy in Genetic Algorithms. Journal of Zhejiang University(Science Edition),32-35.

Brailsford, S. C., Potts, C. N., and Smith, B. M. (1999). Constraint Satisfaction Problems: Algorithms and Applications. European Journal of Operational Research, 119, 557-581.

Burke, E. K., and Newall, J.P. (1999). Multistage Evolutionary Algorithm for the Timetable Problem. IEEE Transactions on Evolutionary Computation, 63-74.

Carter, M. W. (1986). A Survey of Practical Applications of Examination Timetabling Algorithms. Operation Research, 34, 193-202.

Carter, M. W., Laporte, G., and Lee, S.Y. (1996). Examination Timetabling: Algorithmic Strategies and Applications. Journal of Operational Research Society, 43(3), 373-383.

Coello, C. A., and Pulido, G. T. (2001). A Micro-genetic algorithm for multi-objective optimization. In Zitzler, E., Deb, K., Thiele, L., Coello, C. A. C., and Corne, D. W., editors, FirstInternational Conference on Evolutionary Multi-Criterion Optimization, pages 126–140. SpringerVerlag, Berlin.

Colorni, A., Dorigo, M., and Maniezzo, V.(1990). Genetic Algorithms – A New Approach to the Timetable Problem. In Lecture Notes in Computer Science – NATO ASI Series, F(82), 235-239.

Davis F. D. (1989), Perceived usefulness, perceived ease of use, and user acceptance of Information Technology.: MIS Quarterly (13 : 3).

Eiben, A.E., Hinterding, R., and Michalewicz,Z. (1999) "Parameter control in evolutionary algorithms," IEEE Trans. Evol. Comput., vol. 3, no. 2, pp. 124–141.

Filho, G. R., and Lorena, L. A. N. (2000). A Constructive Evolutionary Approach To School Timetabling. www.lac.inpe.br/~marcos/arsig2/CGA-timet-EVOCOP.pdf

Garey, M. R., and Johnson, D. S. Computers and Intractability.(1979).A guide to NP-Completeness. W.H. Freeman and Company, San Francisco, 179.

Gen, M., and Cheng, R. (1994). Evolution Program for Resource Constrained Project Scheduling Problem. In Proceedings of the first IEEE conference on evolutionary computation, 736-741.

Goldberg, D. E. (1989).Genetic Algorithms in Search, Optimization and Machine Learning, Kluwer Academic Publishers, Boston, MA.

Goldberg, D. E. (1989).Sizing populations for serial and parallel genetic algorithms, in Proceedings of the 3rd International Conference on Genetic Algorithms, Fairfax, V.I., 70–79.

Golub, M., Kasapovic, S. (2002). Scheduling Multiprocessor Tasks with Genetic Algorithms. Proceeding of the IASTED International Conference Allplied Infomatics, Insbruck, Austria, 273-278.

Gy'ori, S. Petres, Z., and Varkonyi-Koczy, A. R. (2001). Genetic Algorithms in Timetabling. A New Approach. MFT Periodika 2001-07. Hungarian Society of IFSA, Hungary.

Heitkoetter, J., and Beasley, D. (2001). The Hitch-Hiker's Guide to Evolutionary Computation: A list of Frequently Asked Questions (FAQ).

Jat, S. N and Yang, S. (2009). “A guided search genetic algorithm for the university course timetabling problem,” in Proc. 4th Multidisciplinary Int. Conf. Scheduling: Theory Appl., pp. 180–191.

Jat, S. N and Yang, S. (2011). "Genetic Algorithms and Local Search Strategies for University Course Timetabling," IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol. 41, no. 1, pp.93-106

Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evolutionary Computation,8(2):149 172, 2000.

Karova, M. (2004). Solving Timetabling Problems Using Genetic Algorithms. The proceeding of IEEE of the 27th International Spring Seminar on Electronics Technology, Bankja, Bulgaria.

Kazarlis, S.A., Papadakis, S.E., Theocharis, J.B. and Petridis, V. (2001). Microgenetic algorithms as generalized hill-climbing operators for GA Optimization. IEEE Transaction on Evolutionary Computation, 5(3): 204.217.

Khan, N. (2003). Population sizing in genetic and evolutionary algorithms. Chocago: University of Illinois at Urbana-Champaign.

Krishnakumar, K. (1989). Micro-genetic algorithms for stationary and non-stationary function optimization. SPIE, Intelligent Control and Adaptive Systems, pp. 289-296.

Krzysztof Socha, Joshua Knowles and Michael Sampels (2002). A MAX-MIN Ant System for the University Course Timetabling Problem. Proceedings of the Third International Workshop on Ant Algorithms, 1-13, September 12-14.

Kwan, A.S.K., Kwan, R.S.K., and Wren, A. (1997). Driver scheduling using genetic algorithms with embedded combinatorial traits (Technical Report 97.30). United Kingdom: University of Leeds.

Lai Lien-Fu, Hsueh Nien-Lin, Huang Liang-Tsung and Chen Tien-Chun (2006), An Artificial Intelligence Approach to Course Timetabling, a publication from the Proceedings of the 18th IEEE International Conference on Tools and Artificial Intelligence (ICTAI'06), IEEE Computer Society, 0-7695-2728-0/06.

Lalescu, L., and Badica, C. (2003). Timetabling Experiments Using Genetic Algorithms. Timetabling Experiments Using Genetic Algorithms, prezentata la TAINN'03, Canakkale, Turcia, 2003.Retrieved February, 2005, from http://software.ucv.ro/Cercetare/Inginerie_ Software/inginerie_software.html.

Michalewicz, Z. (1994). Genetic Algorithms + Data Structures = Evolution Programs (2nd ed.), Springer-Verlag.

Najdpour, N., Feizi-Derakhshi, M.-R. (2010), A two-phase evolutionary algorithm for the university course timetabling problem, Software Technology and Engineering (ICSTE), 2010 2nd International Conference on ,2, 266-271.

Newall, J. P. (1999). Hybrid Methods for Automated Timetabling, PhD Thesis, Department of Computer Science, University of Nottingham.

Nia, M.B., and Alipouri, Y. (2009).Speeding Up the Genetic Algorithm Convergence Using Sequential Mutation and Circular Gene Methods, Intelligent Systems Design and Applications, ISDA '09. Ninth International Conference on , vol., no., pp.31-36, Nov. 30 2009-Dec. 2 2009.

Nielson, J. (2006), Quantitative Studies : How many users to test Alertbox, citedon 3rd April 2009, fromhttp://www.useit.com/alertbox/quantitative_testing.html

Paechter, B. Cumming, A. Norman, L., and Luchian, H.(2006). Extensions to a Memetic Timetabling System. In E.K Burke and PM Rosse, Eds., Proceedings of the 1st Inter. Conf. on the Practice and Theory of Automated Timetabling.

Paechter, B., Cumming, A., Luchian, H., and Petriuc, M. (1994).Two Solutions to the General Timetable Problem Using Evolutionary Methods. IEEE World Congress on Computational Intelligence, 300-305.

Paechter, B., Rossi-Doria, O., Blum, C., Knowles J., Sampels, M., and Socha, K.(2002). A Local Search for the Timetabling Problem, PATAT 2002 Proceedings of the 4th international conference on the Practice And Theory of Automated Timetabling, Gent, Belgium, 124-127.

Perzina, R. (2007). "Solving Multicriteria University Timetabling Problem by a Self-adaptive Genetic Algorithm with Minimal Perturbation", ;in Proc. IRI, pp.98-103.

Rich, D.C. (1996). A Smart Genetic Algorithm for University Timetabling, In: E. Burke, P. Ross (Eds.): PATAT'95, LNCS 1153, Springer-Verlag, 1996, 181-197.

Sapru,V., Reddy, K., and Sivaselvan, B.(2010). "Time Table Scheduling Using Genetic Algorithms Employing Guided Mutation," in Proc. IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), pp. 1-4.

Sigl, B., Golub, M., Mornar, V. (2003). Solving Timetable Scheduling Problem Using Genetic Algorithms, Information Technology Interfaces, 2003.ITI 2003.Proceedings of the 25th International Conference, 519- 524.

Selvi, R., Muthu, and Rajaram, R. (2011). Multiobjective optimization to provide service guarantees for window-constrained scheduling using Micro-genetic algorithm. Proceedings of the 2011 International Conference on Communication, 457-461.

Vahid Lotfi, and Robert Cerveney, (1991). A Final Exam Scheduling Package. J. Opl Res. Soc. Vol. 42, 3, 205 – 216.

Vaishnavi, V., and Kuechler, B. (2004). Design Research in Information Systems,AISWorldNet, http://www.isworld.org/Researchdesign/drisISworld.htm, 20February 2004, last updated 5 June 2005 (accessed 15 October, 2005).

Valouxis, C., and Housos, E. (2000). Hybrid Optimization Techniques for the Workshift and Rest Assignment of Nursing Personnel. Artificial Intelligence in Medicine, 20, 155-175.

White, G. W., and Chan, P. W. (2002). Towards Construction of Optimal Examination Timetables. INFOR 17, 219-229.

Xie, S.T., and Pan, L.T. (2010). Selection of Machining Parameters Using Genetic Algorithms. The 5th International Conference on Computer & Education, Hefei, China. August,24-27.

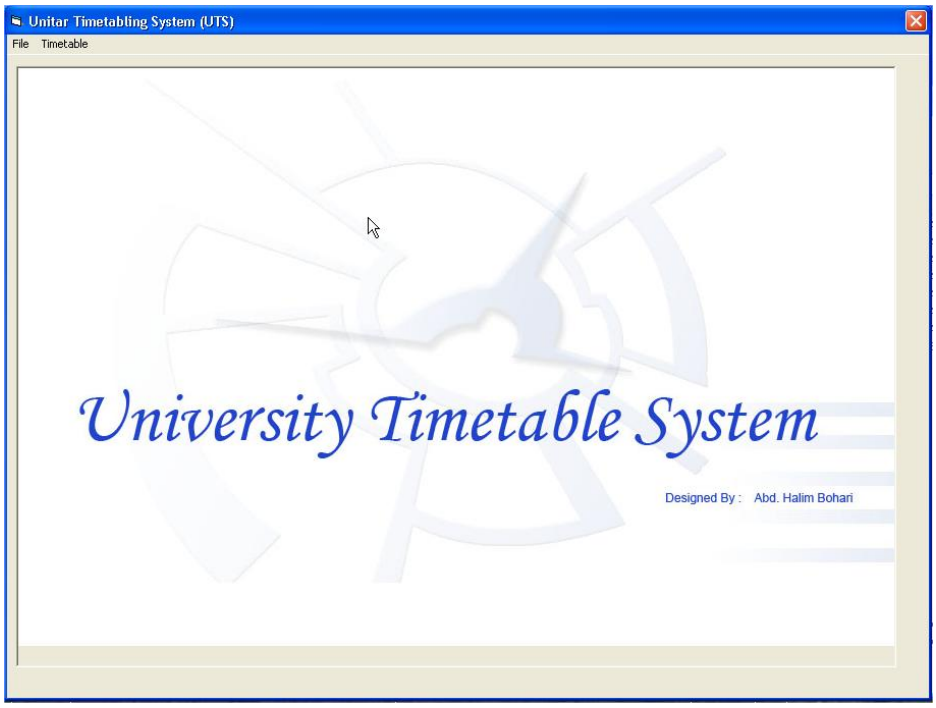
YakhnoT., and Tekin, E. (2002). Application of Constraint Hierarchy to Timetabling Problems, Proceedings of EurAsia-ICT 002, Springer- Verlag.

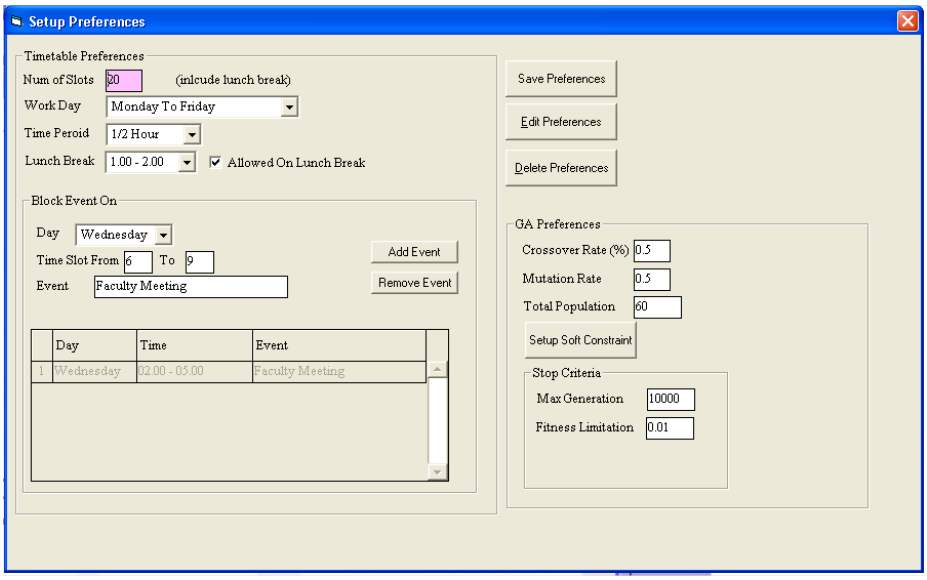
Yang, H., and Luo, D. (2009). Optimal Regional Bus Timetables Using Improved Genetic Algorithm. 2nd International Conference on Intelligent Computation Technology and Automation.

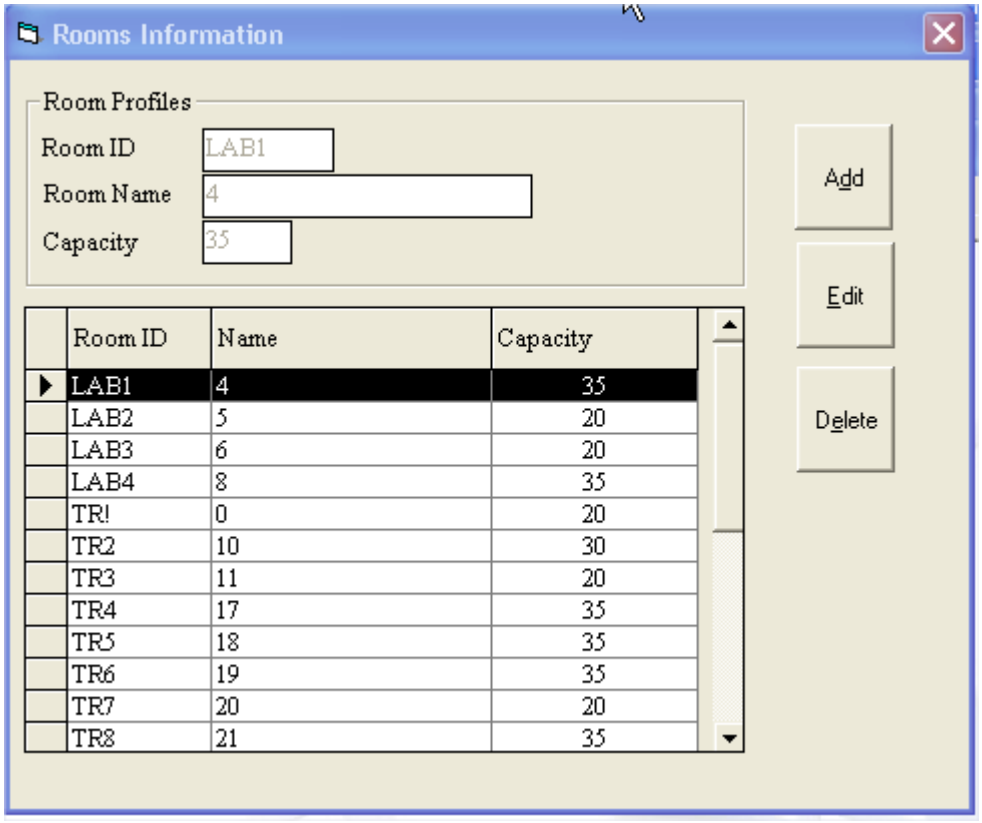
Yang, Y., and Petrovic, S.(2004). A Novel Similarity Measure for Heuristic Selection in Examination Timetabling. In: Burke, E.K., Trick, M. (Eds.), Selected Papers from the 5th International Conference on the Theory and Practice of Automated Timetabling (PATAT 2004) – The Theory and Practice of Automated Timetabling V, Lecture Notes in Computer Science, vol. 3616. Springer, Berlin, 247–269.

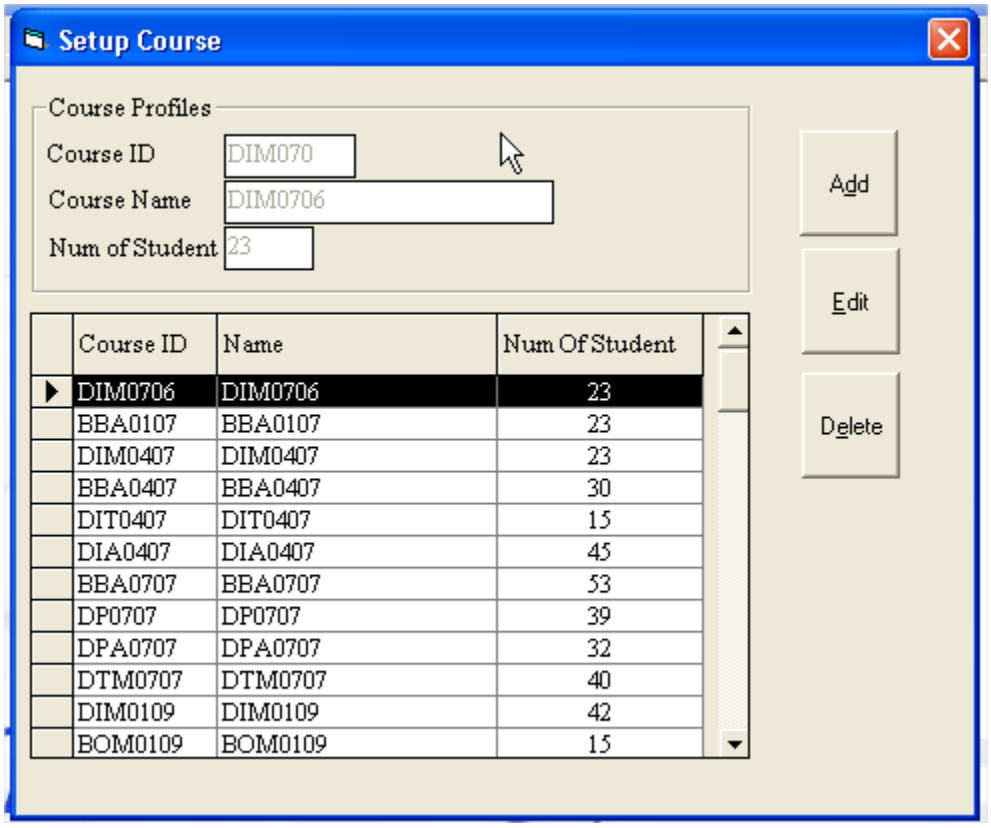
# Appendix

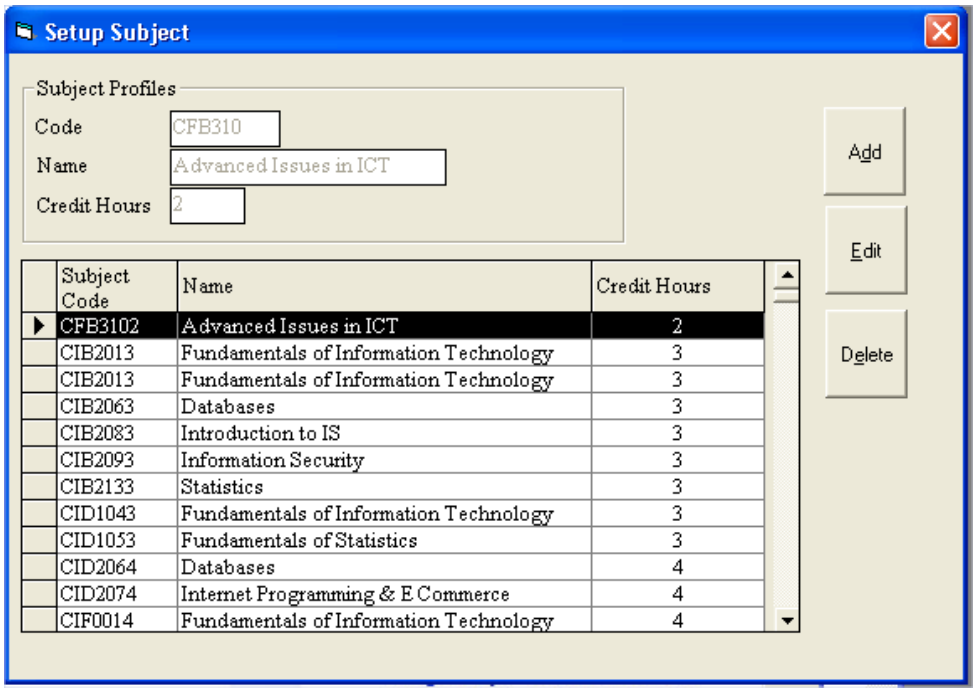
## Appendix I: University Timetable System Interface

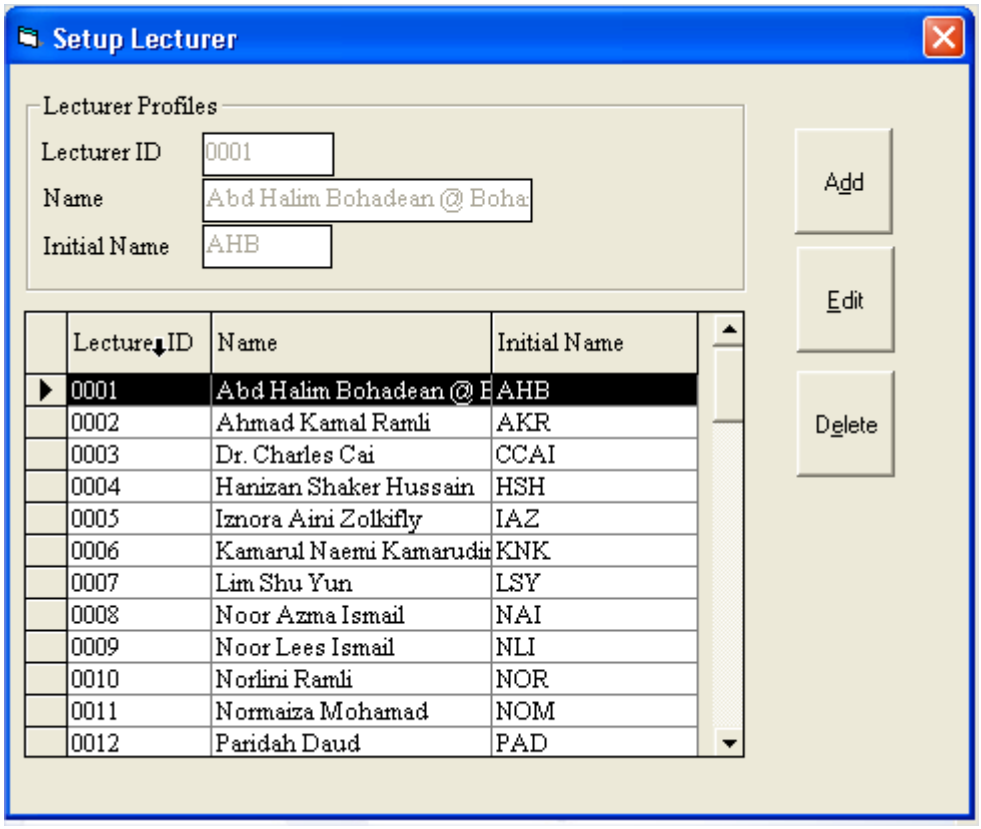


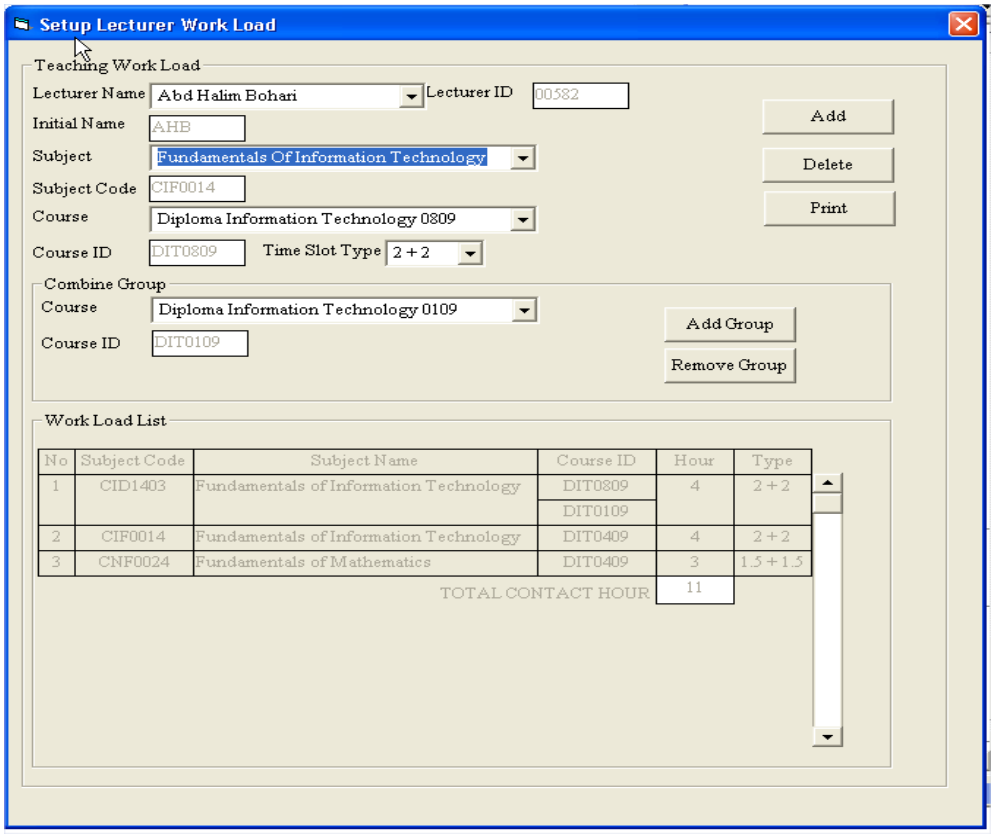




Rooms Information
Room Profiles
Room ID
LAB1
Room Name
4
Capacity
35
Add
Edit
Delete
Room ID
Name
Capacity
LAB1 4 35
LAB2 5 20
LAB3 6 20
LAB4 8 35
TR! 0 20
TR2 10 30
TR3 11 20
TR4 17 35
TR5 18 35
TR6 19 35
TR7 20 20
TR8 21 35

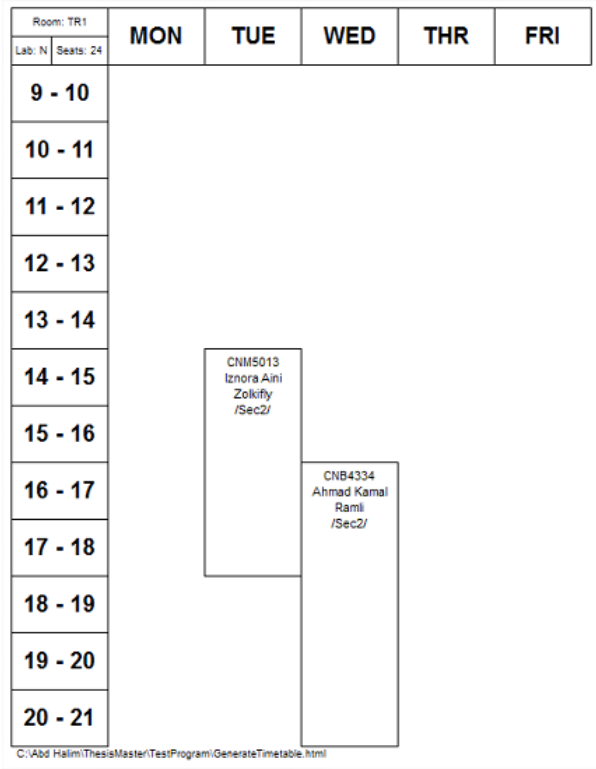

Setup Course
Course Profiles
Course ID
DIM070
Course Name
DIM0706
Num of Student
23
Add
Edit
Delete
Course ID
Name
Num Of Student
DIM0706 DIM0706 23
BBA0107 BBA0107 23
DIM0407 DIM0407 23
BBA0407 BBA0407 30
DIT0407 DIT0407 15
DIA0407 DIA0407 45
BBA0707 BBA0707 53
DP0707 DP0707 39
DPA0707 DPA0707 32
DTM0707 DTM0707 40
DIM0109 DIM0109 42
BOM0109 BOM0109 15


Setup Subject
Subject Profiles
Code
CFB310
Name
Advanced Issues in ICT
Credit Hours
2
Add
Edit
Delete
Subject Code
Name
Credit Hours
CFB3102
Advanced Issues in ICT
2
CIB2013
Fundamentals of Information Technology
3
CIB2013
Fundamentals of Information Technology
3
CIB2063
Databases
3
CIB2083
Introduction to IS
3
CIB2093
Information Security
3
CIB2133
Statistics
3
CID1043
Fundamentals of Information Technology
3
CID1053
Fundamentals of Statistics
3
CID2064
Databases
4
CID2074
Internet Programming & E Commerce
4
CIF0014
Fundamentals of Information Technology
4


Setup Lecturer
Lecturer Profiles
Lecturer ID
0001
Name
Abd Halim Bohadean @ Boha
Initial Name
AHB
Add
Edit
Delete
Lecturer ID
Name
Initial Name
0001 Abd Halim Bohadean @ B AHB
0002 Ahmad Kamal Ramli AKR
0003 Dr. Charles Cai CCAI
0004 Hanizan Shaker Hussain HSH
0005 Iznora Aini Zolkifly IAZ
0006 Kamarul Naemi Kamarudir KNK
0007 Lim Shu Yun LSY
0008 Noor Azma Ismail NAI
0009 Noor Lees Ismail NLI
0010 Norlini Ramli NOR
0011 Normaiza Mohamad NOM
0012 Paridah Daud PAD

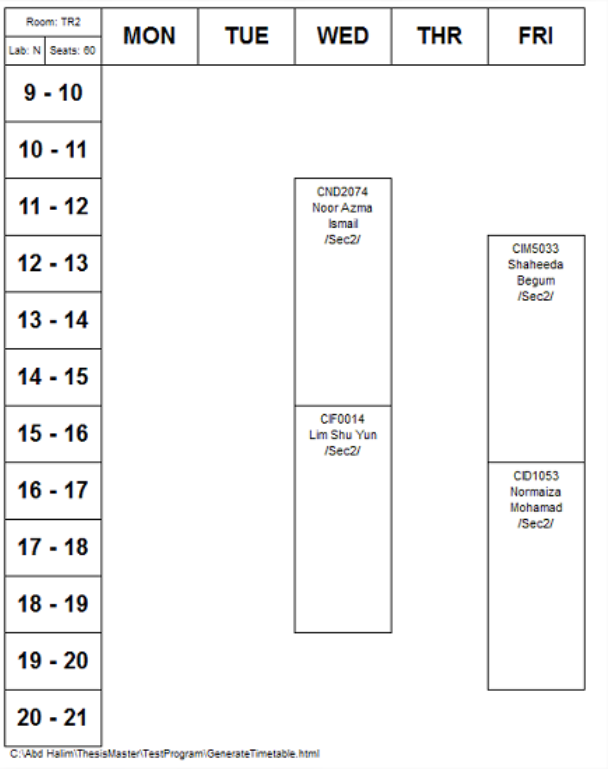

Setup Lecturer Work Load
Teaching Work Load
Lecturer Name
Abd Halim Bohari
Lecturer ID
00582
Initial Name
AHB
Subject
Fundamentals Of Information Technology
Subject Code
CIF0014
Course
Diploma Information Technology 0809
Course ID
DIT0809
Time Slot Type
2 + 2
Add
Delete
Print
Combine Group
Course
Diploma Information Technology 0109
Course ID
DIT0109
Add Group
Remove Group
Work Load List
No
Subject Code
Subject Name
Course ID
Hour
Type
1 CID1403 Fundamentals of Information Technology DIT0809 4 2 + 2
DIT0109
2 CIF0014 Fundamentals of Information Technology DIT0409 4 2 + 2
3 CNF0024 Fundamentals of Mathematics DIT0409 3 1.5 + 1.5
TOTAL CONTACT HOUR
11

## Appendix II:Output for UNITAR Timetable System



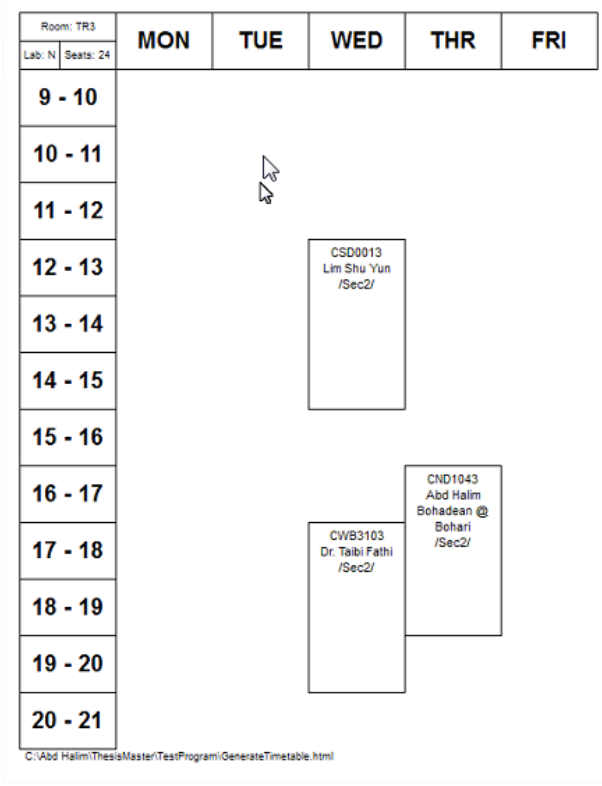
| Room: TR1 / Lab: N / Seats: 24 | MON | TUE | WED | THR | FRI |
|---|---|---|---|---|---|
| 9 - 10 | | | | | |
| 10 - 11 | | | | | |
| 11 - 12 | | | | | |
| 12 - 13 | | | | | |
| 13 - 14 | | | | | |
| 14 - 15 | | CNM5013 Iznora Aini Zolkifly /Sec2/ | | | |
| 15 - 16 | | | | | |
| 16 - 17 | | | CNB4334 Ahmad Kamal Rami /Sec2/ | | |
| 17 - 18 | | | | | |
| 18 - 19 | | | | | |
| 19 - 20 | | | | | |
| 20 - 21 | | | | | |

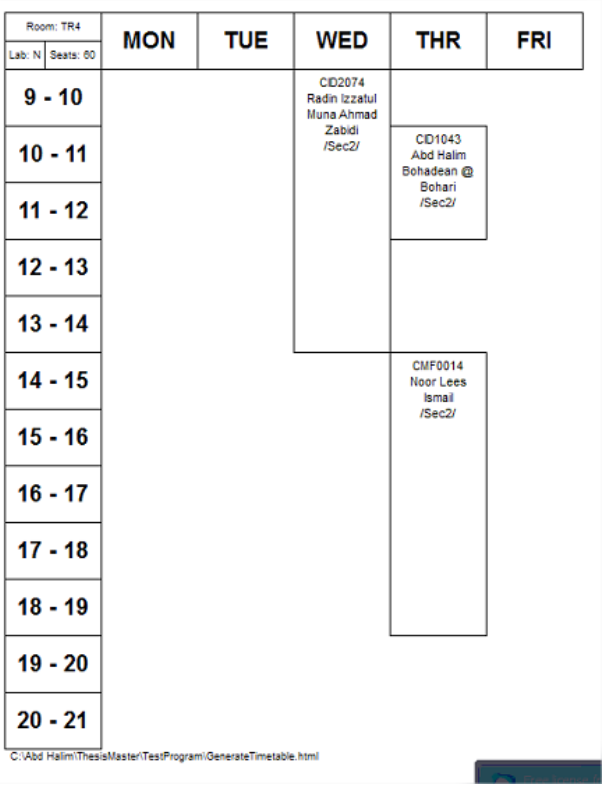
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

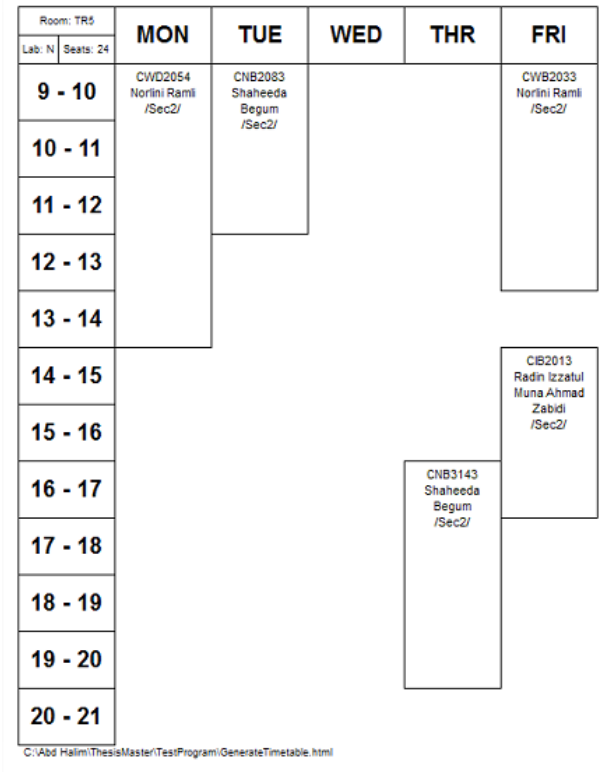


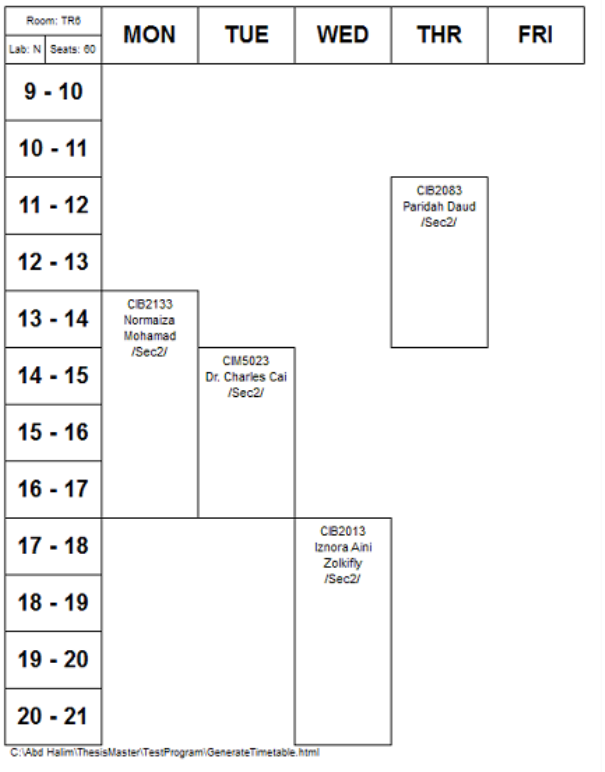
| Room: TR2 / Lab: N / Seats: 60 | MON | TUE | WED | THR | FRI |
|---|---|---|---|---|---|
| 9 - 10 | | | | | |
| 10 - 11 | | | | | |
| 11 - 12 | | | CND2074 Noor Azma Ismail /Sec2/ | | |
| 12 - 13 | | | | | CIM5033 Shaheeda Begum /Sec2/ |
| 13 - 14 | | | | | |
| 14 - 15 | | | | | |
| 15 - 16 | | | CIF0014 Lim Shu Yun /Sec2/ | | |
| 16 - 17 | | | | | CID1053 Normaiza Mohamad /Sec2/ |
| 17 - 18 | | | | | |
| 18 - 19 | | | | | |
| 19 - 20 | | | | | |
| 20 - 21 | | | | | |

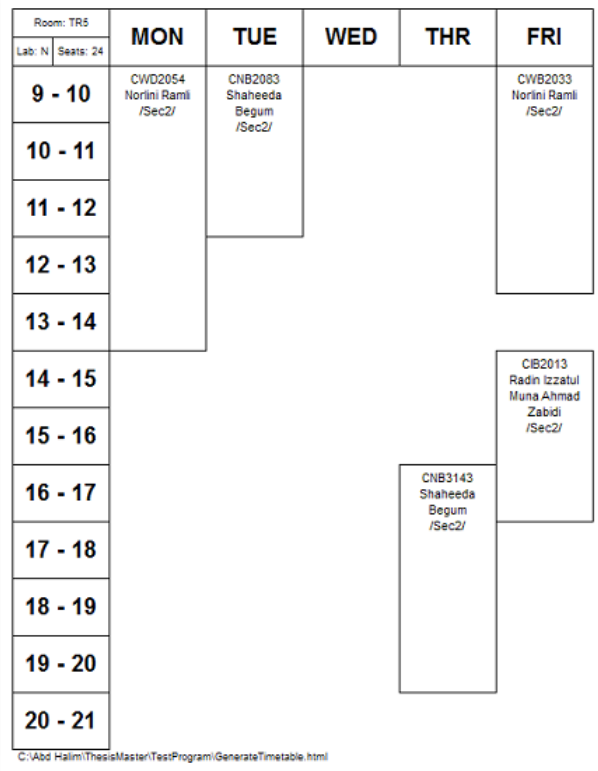
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

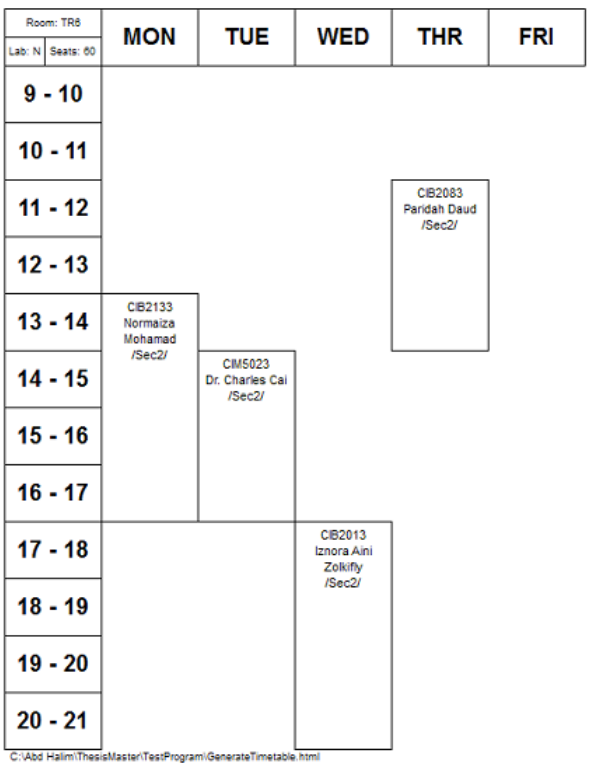

Room: TR3
Lab: N
Seats: 24
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CSD0013
Lim Shu Yun
/Sec2/
CND1043
Abd Halim
Bohadean @
Bohari
/Sec2/
CWB3103
Dr. Taibi Fathi
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

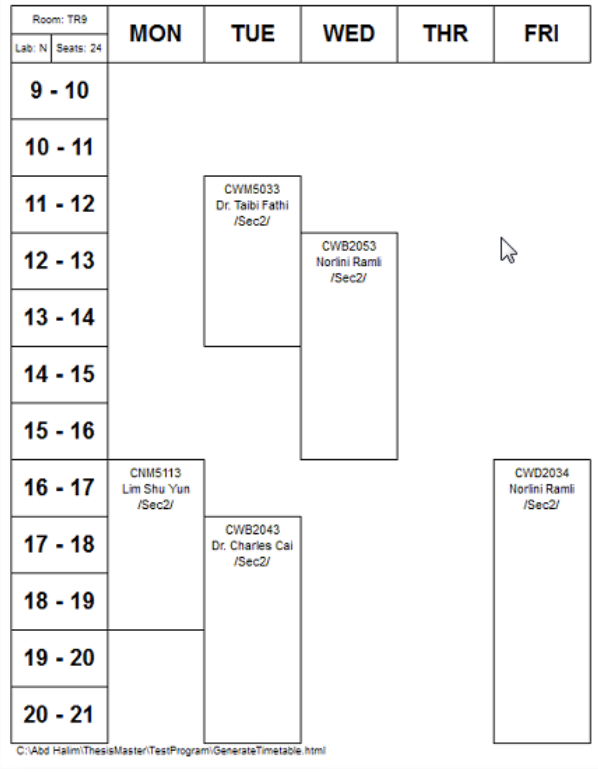

Room: TR4
Lab: N
Seats: 60
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CID2074
Radin Izzatul
Muna Ahmad
Zabidi
/Sec2/
CID1043
Abd Halim
Bohadean @
Bohari
/Sec2/
CMF0014
Noor Lees
Ismail
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

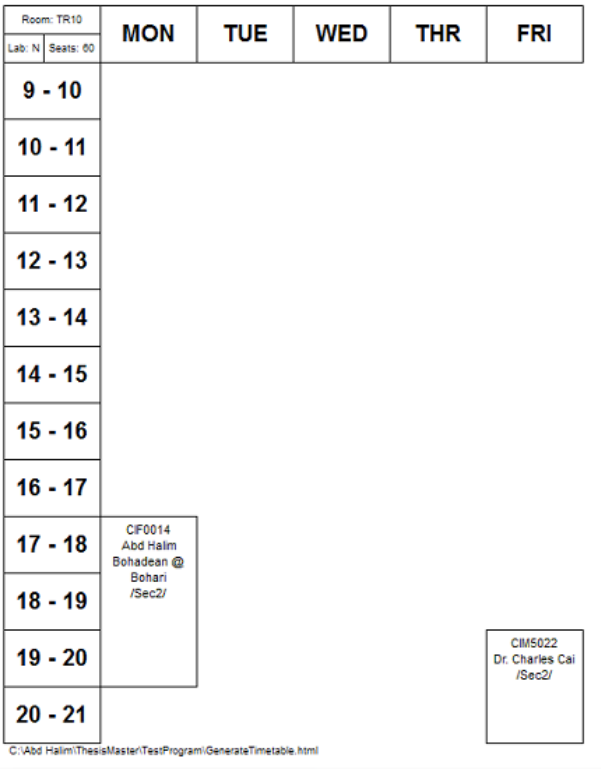

Room: TR5
Lab: N
Seats: 24
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CWD2054
Norlini Ramli
/Sec2/
CNB2083
Shaheeda
Begum
/Sec2/
CWB2033
Norlini Ramli
/Sec2/
CIB2013
Radin Izzatul
Muna Ahmad
Zabidi
/Sec2/
CNB3143
Shaheeda
Begum
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

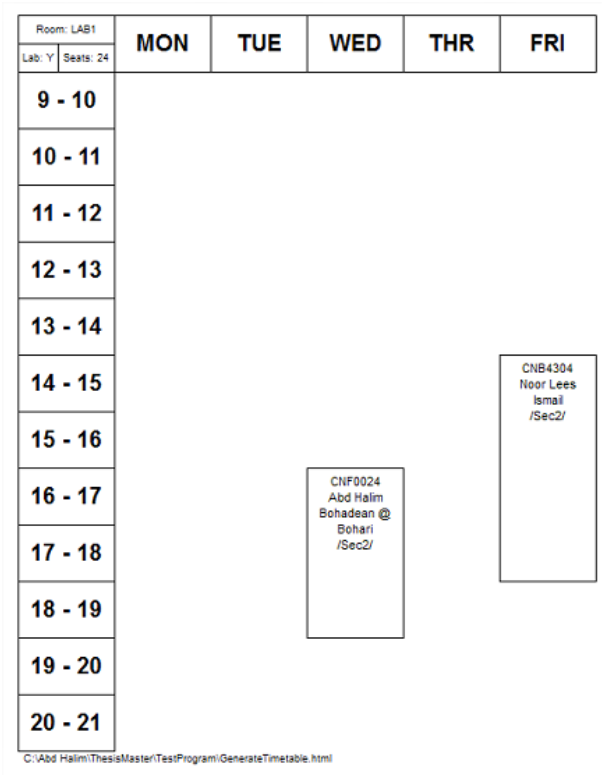

Room: TR6
Lab: N
Seats: 60
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CIB2083
Paridah Daud
/Sec2/
CIB2133
Normaiza
Mohamad
/Sec2/
CIM5023
Dr. Charles Cai
/Sec2/
CIB2013
Iznora Aini
Zolkifly
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

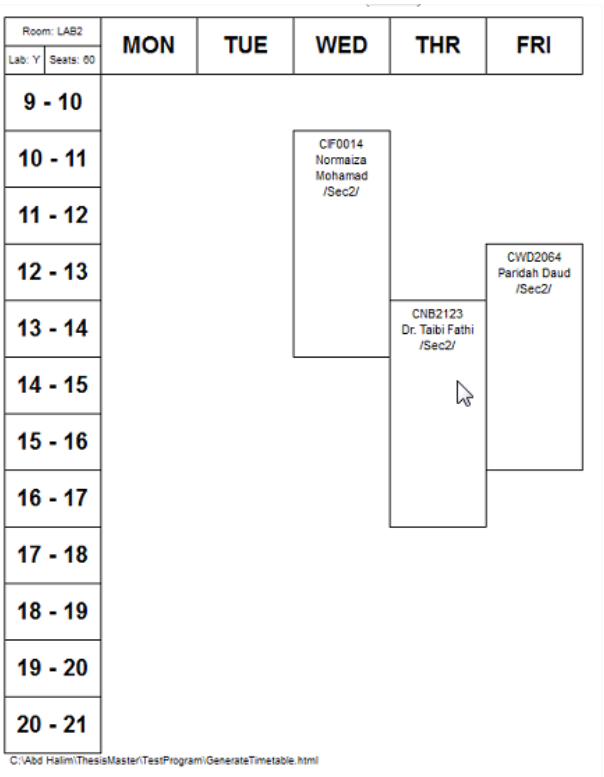

Room: TR5
Lab: N
Seats: 24
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CWD2054 Norlini Ramli /Sec2/
CNB2083 Shaheeda Begum /Sec2/
CWB2033 Norlini Ramli /Sec2/
CIB2013 Radin Izzatul Muna Ahmad Zabidi /Sec2/
CNB3143 Shaheeda Begum /Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

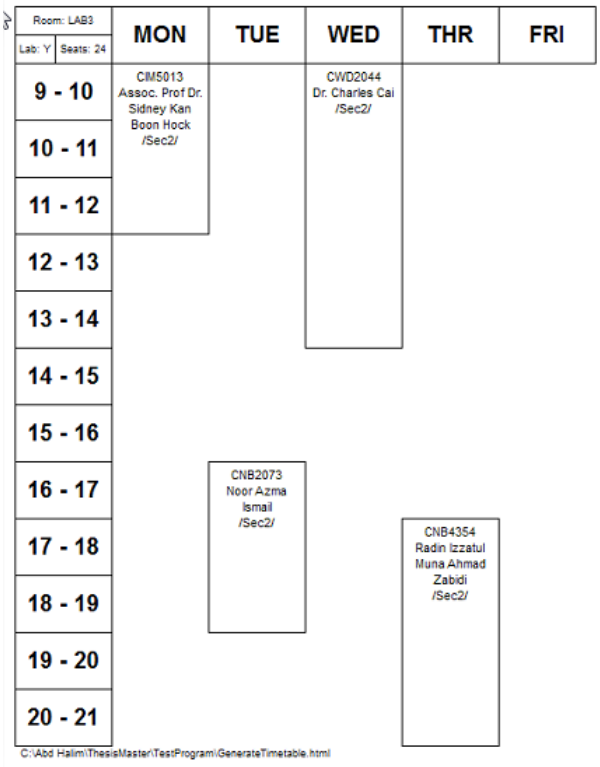

Room: TR8
Lab: N
Seats: 60
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CIB2083 Paridah Daud /Sec2/
CIB2133 Normaiza Mohamad /Sec2/
CIM5023 Dr. Charles Cai /Sec2/
CIB2013 Iznora Aini Zolkifly /Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

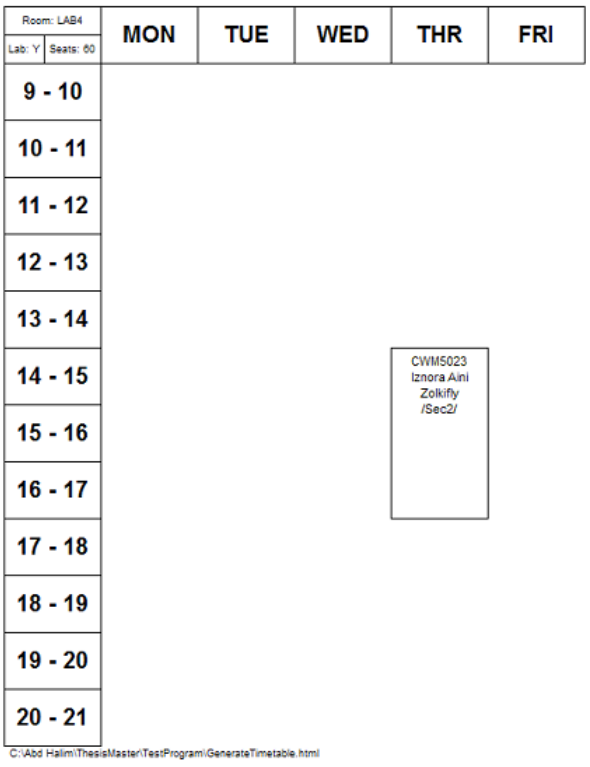

Room: TR9
Lab: N
Seats: 24
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CWM5033 Dr. Taibi Fathi /Sec2/
CWB2053 Norlini Ramli /Sec2/
CNM5113 Lim Shu Yun /Sec2/
CWB2043 Dr. Charles Cai /Sec2/
CWD2034 Norlini Ramli /Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html


Room: TR10
Lab: N
Seats: 60
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CIF0014 Abd Halim Bohadean @ Bohari /Sec2/
CIM5022 Dr. Charles Cai /Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html


Room: LAB1
Lab: Y
Seats: 24
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CNF0024
Abd Halim
Bohadean @
Bohari
/Sec2/
CNB4304
Noor Lees
Ismail
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html


Room: LAB2
Lab: Y
Seats: 60
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CIF0014
Normaiza
Mohamad
/Sec2/
CNB2123
Dr. Taibi Fathi
/Sec2/
CWD2064
Paridah Daud
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html


Room: LAB3
Lab: Y
Seats: 24
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CIM5013
Assoc. Prof Dr.
Sidney Kan
Boon Hock
/Sec2/
CWD2044
Dr. Charles Cai
/Sec2/
CNB2073
Noor Azma
Ismail
/Sec2/
CNB4354
Radin Izzatul
Muna Ahmad
Zabidi
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html


Room: LAB4
Lab: Y
Seats: 60
MON
TUE
WED
THR
FRI
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
16 - 17
17 - 18
18 - 19
19 - 20
20 - 21
CWM5023
Iznora Aini
Zolkifly
/Sec2/
C:\Abd Halim\ThesisMaster\TestProgram\GenerateTimetable.html

## Appendix III: Sample Code

```
‘Initializes chromosomes with configuration block (setup of chromosome)
Public SetupChromosome()
        ‘ reserve space for time-space slots in chromosomes code
        slots_size= DAYS_NUM * DAY_HOURS * GetNumberOfRooms()
        ‘ reserve space for flags of class requirements
        criteria_resize(GetNumberOfCourseClasses() * 5 )
End Sub
Public Sub ScheduleObserver(newChromosome)
SetSchedule(newChromosome )
End Sub

‘ Makes new chromosome with same setup but with randomly chosen code
Function Makenewchromosome(slots_size as integer)
        ‘ number of time-space slots
        size = slots_size
        ‘ place classes at random position
        While (i<=CourseClasslist.upperboundAndCourseClasslist[i]>const_iteration)
                ‘ determine random position of class
                nr = GetNumberOfRooms()
                dur =GetDuration()
                day = rand() % DAYS_NUM
                room = rand() % nr
                time = rand() % ( DAY_HOURS + 1 - dur )
                pos = day * nr * DAY_HOURS + room * DAY_HOURS + time
                ‘ fill time-space slots, for each hour of class
                for  j = dur - 1 to 0 step -1
                        newChromosome._slots.at(pos + i ).push_back( *it )
                Next  j
                ‘ insert in class table of chromosome
                newChromosome._classes.insert(pair.CourseClass)
                i=i+1
        Wend
        newChromosome->CalculateFitness()
        returnnewChromosome
End Function
```

```
‘Performes crossover operation using to chromosomes and returns pointer to offspring
Public Sub Crossover(parent2)
        ‘check probability of crossover operation
        If rand() % 100 > _crossoverProbability Then
                ‘ no crossover, just copy first parent
                return Schedule( parent1, false )
         End If
         ‘new chromosome object, copy chromosome setup
        Schedule* n = new Schedule( parent2, true )
        ‘number of classes
        size = _classes.size()
        ‘ determine crossover point (randomly)
        for  i= _numberOfCrossoverPoints to 0 step -1
                do while( TRUE )
                        p = rand() % size
                        If !cp[ p ] Then
                                cp[ p ] = true
                                Exit Do
                        End If
                Wend
        Next i
        hash_map.CourseClass, it1 = _classes.begin()
        hash_map.CourseClass, it2 = parent2._classes.begin()

        ‘ make new code by combining parent codes
        Dim first as boolean
        first=iif(rand() % 2 == 0,True,False)
        for  i = 0 to size

                If first Then

                        ‘insert class from first parent into new chromosome’s calss table
                        n._classes.insert( pair<CourseClass*, int>( ( *it1 ).first, ( *it1 ).second ) )
                        ‘ all time-space slots of class are copied
                        for(inti = ( *it1 ).first->GetDuration() - 1  i>= 0  i-- )
                                n->_slots[ ( *it1 ).second + i ].push_back( ( *it1 ).first )
```

```
                Else

                        ‘ insert class from second parent into new chromosome’s calss table
                        n->_classes.insert( pair<CourseClass*, int>( ( *it2 ).first, ( *it2 ).second ) )
                        ‘ all time-space slots of class are copied
                        for(inti = ( *it2 ).first->GetDuration() - 1  i>= 0  i-- )
                                n->_slots[ ( *it2 ).second + i ].push_back( ( *it2 ).first )
                End If

                ‘ crossover point
                If cp[ i ] Then
                        ‘ changesoruce chromosome
                        first = !first
                 End If
                it1++
                it2++
        Next i

        n->CalculateFitness()

        ‘ return smart pointer to offspring
        return n
End Sub

‘ Performs mutation on chromosome
Public Sub Mutation()
        ‘ check probability of mutation operation
        If rand() % 100 > _mutationProbability Then
                return
        End If
        ‘ number of classes
        intnumberOfClasses = (int)_classes.size()
        ‘ number of time-space slots
        int size = (int)_slots.size()

        ‘ move selected number of classes at random position
```

```
        For i= _mutationSize to 1 step-1
                ‘ selectranom chromosome for movement
                mpos = rand() % numberOfClasses
                pos1 = 0
                hash_map<CourseClass*, int>::iterator it = _classes.begin()
                ‘ current time-space slot used by class
                pos1 = ( *it ).second
                CourseClass* cc1 = ( *it ).first
                ‘ determine position of class randomly
                nr = Configuration::GetInstance().GetNumberOfRooms()
                dur = cc1->GetDuration()
                day = rand() % DAYS_NUM
                room = rand() % nr
                time = rand() % ( DAY_HOURS + 1 - dur )
                pos2 = day * nr * DAY_HOURS + room * DAY_HOURS + time
                ‘ move all time-space slots
                for d = dur - 1 to 0 step-1
                        ‘ remove class hour from current time-space slot
                        list<CourseClass*>& cl = _slots[ pos1 + i ]
                        Do Until list[it]<CourseClass
                                If  *it == cc1 Then
                                        cl.erase( it )
                                        Exit Do
                                End If

                it=it+1
                        loop
                        ‘ move class hour to new time-space slot
                        _slots.at( pos2 + i ).push_back( cc1 )
                Next d
                ‘ change entry of class table to point to new time-space slots
                _classes[ cc1 ] = pos2
        Next i
        CalculateFitness()
End Sub
```

**Appendix IV: Sample Questionnaire**

**Evaluation of Micro-GA Timetabling System**

**Introduction**

There are **THREE (3)** parts in this questionnaire.

PART A. General information

PART B. System aspects

PART C. Overall satisfaction

Intended Audience : Coordinator, Lecturer and Senior Executives, Scheduler.

**Please answer ALL questions in ALL parts.**

**Part A : General Information**

1. Gender : (Please tick)

   Male ☐

   Female ☐

2. Age : _____ Years

3. Educational Background : (Please tick)

   ☐ Diploma

   ☐ Degree

   ☐ Master

   ☐ PHD

4. How long had you worked in the education industry : (Please tick)

   ☐ Less then 1 year

   ☐ 1 – 2 years

   ☐ 2 – 5 years

   ☐ 5 years and more

**Part B : System Aspects**

This part is intended to obtain your view on some aspects of the Micro-GA Timetabling Systemfor a University Tun Abdul Razak.

Please mark [√] your answers.

1 = Strongly Disagree
2 = Disagree
3 = Natural
4 = Agree
5 = Strongly Agree

| | **Perceived Usefulness** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|---|
| B1 | Do you find that using the MGATto generate universitytimetable is quick? | | | | | |
| B2 | Do you think the MGAT improves the performance of time table scheduling? | | | | | |
| B3 | Do you think that using MGAT will improve your productivity? | | | | | |
| B4 | Will using the MGAT could enhance the effectiveness to carry out timetable tasks? | | | | | |
| B5 | Would using the MGAT will make it easier for you to carry out timetable tasks? | | | | | |
| B6 | Do you find the MGAT useful to carry out the timetable tasks? | | | | | |

| | **Perceived Ease of Use** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|---|
| B7 | Do find it easy when learning to use the MGAT? | | | | | |
| B8 | Do feel that the MGAT is easy for you to carry out what you want to do? | | | | | |
| B9 | Do you think that using MGAT is interactive and helps your understanding? | | | | | |
| B10 | Will using the MGAT be flexible to achieve your goals? | | | | | |
| B11 | Would you find it easy for you to become skillful at using MGAT? | | | | | |
| B12 | Do you find the MGAT user interface helps you feel that it is easy to use? | | | | | |

**Part C : Overall Satisfaction**

This part is intended to obtain your view on some aspects of the the Micro-GA Timetabling for a University Tun Abdul Razak.

Please mark [√] your answers.

1 = Strongly Disagree

2 = Disagree

3 = Natural

4 = Agree

5 = Strongly Agree

| | **Attributes of Usability** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|---|
| C1 | Are you satisfied with the number of steps included in the MGAT? | | | | | |
| C2 | Do you think it is easy to understand when interacting with the MGAT? | | | | | |
| C3 | Do you think that using MGAT will improve your productivity? | | | | | |
| C4 | Will there be a need for more instructions when using the MGAT to carry out tasks? | | | | | |
| C5 | Would using the MGAT is more complex than carrying out course scheduling manually? | | | | | |
| C6 | Do you find the MGAT easy to remember the steps of usage ? | | | | | |

**THANK YOU**