UTHM SECURITY UNIT SCHEDULING USING GENETIC ALGORITHM

Mohd Zaki Bin Mohd Salikon
Department of Software Engineering
Faculty of Information Technology and Multimedia
Universiti Tun Hussein Onn Malaysia
mdzaki@uthm.edu.my

Abstract – *Genetic Algorithm (GA) is one of the most popular optimization solutions. It has been implemented in various applications such as scheduling. The flows of GA are using selection, crossover and mutation operators applied to populations of chromosomes. Security Unit scheduling (SUS) should focus on aspects of time, place, time slot and person in charge in the slots. This workload has been the most difficult task faced by the institutions since it's requires many resources to solve various time slot matters. Scheduling is important for a smooth operation of the time slot and avoiding redundancies with any other time slot take place in the same time and same place. In one aspect, it deals with staffs such that it fulfils the process time slot. These aspects are important for the time slot arrangements can be done in a smooth way and no staffs can sit more than one duty in a same time slot. The other constraint is the staff workload should be arranged less than three duties in a row.*

Keywords: *Genetic Algorithm, Scheduling.*

1. Introduction

Scheduling problems is a difficult task in the artificial intelligence. It deals with the allocation of limited resources to tasks over time [8]. The process is to optimize one or more objectives. The measurement methods which is known as computationally NP (Nondeterministic Polynomial) is the most difficult parts that the researchers faced on the scheduling problems which is lacking with the elements of the polynomial time algorithm.

It is important to plan and manage the schedule for a better end results. For instance, in the higher learning institute, it is a normal practice for the administrator to prepare and design the class scheduling before students begin their new semester. This effort lead for an effective end results from a proper management of the time scheduling systems for the classes arrangements. Apart from scheduling the classes' time arrangements, the SUS also lead for an effective management systems in a university.

Scheduling problems contains a set of events such as duty shift and staff time slots. The constraints can be classified as hard and soft constraints with the purpose to avoid the redundancies. The hard constraints type is a situation when no staffs are allocated to more than one time slots at the same time. A soft constraint is a situation where there are no staffs with three time slots in a row. As a result of failure, penalty will be given for both of the constraints. The hard constraint will be used for the higher penalty while lower penalty will be initialized to the soft constraint.

Scheduling problems involve in feasible assignment of users to time slots that are distributed over a period of time, based on a set of constraints. Problems of time based planning and combinational optimizations, which tend to be solved with cooperation of search and heuristics to get optimal or near optimal solutions [1]. This problem should be solved to ensure the requirements and constraints are fulfilled within a limited time.

For this study, GA will be chosen in the prototype development. This method have been used in the science and engineering fields by adapting algorithms to solve practical problems and as computational models of natural evolutionary systems [6]. GAs can solve searching and optimization problems based on genetic process. According to [1], GA is a powerful techniques in optimization problems (mutation and crossover operators applied to populations of chromosomes) either from the domain to specific aspects of a problem (the evaluation function for the chromosomes).

GA is a search algorithm based on a simple idea from biology *survival of the fittest* [5]. GA performs a directed search of a solution space in order to find an optimal solution for some problem. They have been used for many different applications including scheduling, predicting the stock market and creating art.

In GAs method, the fittest is active based on the selection mechanisms and natural genetic in searching algorithms. This process happens among string of structures which represents the information structures. These structures will be changed based on non-randomly but stochastically in searching the algorithms.

In fulfilling the requirement of this study, UTHM (Universiti Tun Hussein Onn Malaysia) SUS will be preferred as a case study. Staffs are required to sit for not more than one time at same time slot. If possible, the prototype can arrange less than three slots in a row. The data consists of 39 staffs from with consists of 18 slots. This information will be used as a case study in designing and developing the prototype.

1.1 Problem Statement

Currently, SUS at UTHM is done manually, where the time task has been created once a year. The time consuming process require sequences of the manual steps. This process used the clustering and heuristic method. The duty is locating in the empty time slot and empty place randomly. This is to avoid redundancies and re-scheduling process. In addition, rescheduling process is time consuming and may affect on the lessons planning and all staff involved.

1.2 Objectives

The objectives of the study are as the following:

- To optimize the slot and arrangement scheduling process
- To develop an Security Unit Scheduling System (SUS) prototype in Java in the implementation of GA methods
- To evaluate the developed prototype in term of parameters setting

1.3 Scope

In this project, GA method will be implemented at UTHM. The real data were used to implement the SUS. The result of this study (timetable) is representing in the form of chromosomes.

2. Literature review

The constructions of schedules is very difficult with a lot of constraints should be followed. GAs has been used quite successful in the scheduling problem. The main objective is to find schedules to satisfy a numbers of hard and soft constraints.

GA was applied in that optimization problem because it's robust enough in such a huge problem [3]. They introduced a new set representation, which meets the demands better than previous. Students, teachers, lessons and classrooms have to be arranged optimally. There have been classified hard and soft constraints to be satisfied by the timetables. The method proved to be efficient in real life application of a secondary school. The set representation meets the demands better than former ones.

[2] presented a constructive evolutionary approach to school timetabling. It is process of fixing a sequence of meetings between teachers and students in a prefixed period of time, satisfying a set of constraints of various types. The problem is modelled as a basis to construct feasible assignments of teachers to classes on specified timeslots. This work presents an application of a Constructive GA to school timetabling problems. The results can be considered successful aiming the possibility of being in future an important component of administrative school tools.

According to [4], an experiment implementation was done as a C++ object oriented program. They investigate an approach for solving the university timetabling problem using a GA. It involves scheduling in the optimal way a given set of activities such that conflicts in using a given set of resources are avoided. The resulted schedule must be valid and must meet as much as possible an additional set of domain problem dependent soft constraints. It also presents 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.

The successful in their experiment in scheduling final examination timetable has been implemented [9]. This research investigates the use of GA for distance learning unit in Universiti Utara Malaysia (UUM). In this case, two constraints classified as hard and soft. Hard constraint must be fulfilled for the timetable to be feasible with no student clash. Meanwhile, soft constraint can be violated but still maintains the feasibility of the solution with no student with three exams in a row. They were developing a timetable system using a client-server model. The results show that the system is not only capable of producing several feasible solutions but also produces in a reasonable amount of time. The effect of varying GAs operators to the obtained feasible solution has been investigated.

Basically, the same structure as used in GA SUS system. In this case, the fitness function for a chromosome representing a schedule involves various problems such as clashes, instances of staffs having to take consecutive duty. GAs then performs crossover and produce schedules whose fitness values are evaluated and a schedule with no clashes can be finalized. GAs is chosen because of the efficiency and robust methods to solve the scheduling problems.

3. Methodology

The prototype developments are divided into four phases. The flows of phases are problem identification, theory building, system development and experimentation. Figure 1 illustrates the methodology approach adopted for this project.

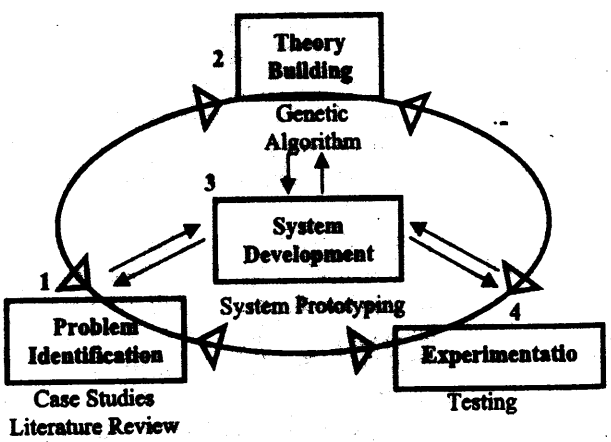

Figure 1: Methodology approach for SUS (Adapted from Nunamaker *et al.*, 1991)

3.1 Problem Identification

Problem Identification is the first step involved in the research methodology. In a particular problem, there are

constraints on whether certain events can appear at the same time, close together and others. The scheduling problem consists of a set of duties,

$$D = \{d_1, d_2, ..., d_{|D|}\} \quad (1)$$

and a set of timeslots,

$$T = \{1,2,...,|T|\} \quad (2)$$

The goal is to obtain an assignment where each duty in D is allocated to a timeslot in T.

In SUS problem, the events are duties and the times are separate timeslots. There are three timeslots per day. The problem identification is formulated based on GA and has been applied in this scheduling problem. In addition, the hard constraints and soft constraints should be obtained. The feasible candidate timetable is penalized for each occurrence of the constraints. It generates the data, optimizing a timetable to avoid staffs' clashes and maximize individual staffs' break period between duties.

3.2 Theory Building

Theory Building includes the development of methods or models. The GA has been used in this SUS prototype development due to a successful application in the previous literature. GA was investigated and relevant literature was examined in order to determine an effective scheme which represents timetables as chromosomes. Chromosomes are typically represented as simple strings of data and instructions. In this case, chromosomes had been chosen to represent a solution and duty shift had been chosen as a gene.

1. *Initialization*: Generate random population of n chromosomes (suitable solutions for the problem)
2. *Fitness*: Evaluate the fitness *f(x)* of each chromosome *x* in population
3. *New population*: Create a new population by repeating the following steps until the new population is complete
    a) *Selection*: Select two parent chromosomes from the population according to their fitness (the better fitness, the bigger chance to be selected)
    b) *Crossover*: With a crossover probability cross over the parents to form a new offspring. If no crossover was performed, offspring is the exact copy of parents
    c) *Mutation*: With a mutation probability mutate new offspring at each locus (position in chromosome)
    d) *Accepting*: Place new offspring in the new population
4. *Replace*: Use new generated population for a further run of algorithm
5. *Test*: If the end condition is satisfied, stop, and return the best solution in current population
6. *Loop*: Go to step 2.

Figure 2: Outline of Genetic Algorithm

(Adapted from Mitchell, 1999)

Figure 2 explains the framework for timetabling problem using GA. It is a general method for problem solving using evolution strategies and evolutionary programming.

Constraints should be defined before a fitness function can be attempted. It can be classified as a hard and soft one. A hard constraint is a major constraint involved in solving the problem. Any violation of this constraint will derive for an infeasible solution. The soft constraints is the case where the violations produce suboptimal (but feasible) solutions. For the purposes of penalty calculation constraints are classified into soft and hard categories.

Constraint type	Penalty
Hard	30
Soft	5

Table 1: Penalty values

Table 1 shows the given penalty values, the hard constraint value is 30 while the soft constraints value is 5. This value is to distinguish the constraint type used in this experimentation.

The fitness of each individual chromosome must be computed when populations of chromosomes are generated. The set of integer is used to represent the chromosome with the fulfilment of all of the constraints. All chromosomes in one generation are evaluated by a fitness function. Each chromosome in the population is allocated with a constant value which represents a maximum fitness score. Each staff's duty schedule is compared against all the chromosomes for any constraints violation during the evaluation phase. A penalty is given to a chromosome for each of the violate constraint. The penalty score from the constant value is subtracted to reduce the fitness value. The fitness of every chromosome in the population is obtained after the evaluation phase is completed.

During reproduction, chromosomes are selected from a combination exists in the population the roulette wheel selection algorithm has been used to select parents for the mating process. This means that the higher fitness strings represent the bigger parts of the wheel so that it will have a stronger change to be picked up. Genes from parents combine to form a whole new chromosome. It will produce offspring in the new population by crossover or replication of parents. Mutation will be applied to offspring after crossover and the fitness will be calculated to record their genealogy.

Good chromosomes with higher fitness strings consist of a high probability to be selected several times in reproduction. Simple reproduction allocates offspring strings according to fitness function. This is a way to choose members from the population of the chromosomes.

The selection operator selects chromosomes according to their fitness score. If the chromosome is more than random number (0.0 to 1.0), so the chromosome will be selected. If the new generations still

need the chromosome, the selection process will be conducted again. If the chromosome has been chosen, the probability of that chromosome is still the same. The wheel is remaining in its original place and will turn again until all new population is complete. The higher fitness has a great chance to be selected. However, it is not guaranteed that the fittest member goes to the next generation.

The crossover process using selected genes from the parents will produce better new chromosomes. The single crossover process is the easiest method usually implemented by previous researchers. Crossover may be occurs and operates on selected genes from the parent chromosomes. The new chromosomes will contain the best parts of parents' chromosomes and therefore a new better chromosome is produced.

First, one random point is chose to determine the crossover point. Then, all genes at the crossover point are copied from parents to offspring. As a result, these new chromosomes or offspring shares some similar features taken from their parents. The genes after crossover point are swapped between both parents. Single point crossover can be illustrated as Figure 3.

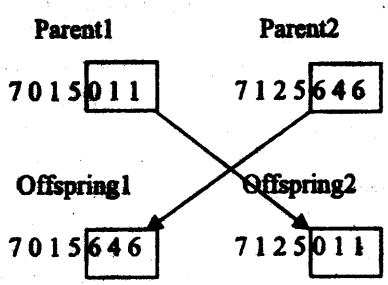

Figure 3: Single point Crossover

In this case, crossover rate has been created to produce more choices in parameters. If crossover is not applied, offspring are exact copies of parents. The crossover rate in this experiment has been set in a range between 0.75 (75%) and 0.95 (95%). According to [6], the crossover and mutation rate is followed from experimental approaches taken by the users to find the best parameters setting.

Each chromosome is now given the chance to mutate to any sequence after crossover. Mutation randomly modifies each gene with its probability to the offspring. If there is no mutation, offspring are generated immediately after crossover without any change. If mutation is performed, one or more parts of a chromosome are changed. Mutation operation randomly changes the offspring resulted from crossover. Figure 4 shows the mutation process change one of the duties' timeslot for duties 4 from 5 to 1.

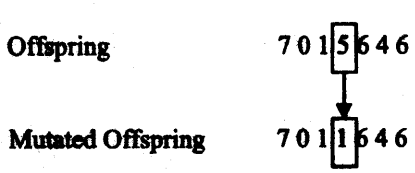

Figure 4: Mutation

Mutation is applied to genes by changing them with a very low probability. The mutation rates have been range between 0.005 and 0.05. This step is proposed to avoid falling of all solutions in the population into a local optimum. The mutation rates should not be high because it has the capabilities to create useless chromosomes representation.

When creating a new population by crossover and mutation, there is a big chance to lose the best chromosome. Elitism is the name of the method that copies the best chromosome to the population in the next generation. It can rapidly increase the performance because it avoids a loss of the optimal solution.

A new generation of chromosomes is produced after applying crossover and mutation. The process is continues, evaluate their fitness and reproduce by crossover and mutation again for next generation until sufficient solution is found.

3.3 System Development

System Development consists of prototyping, used as a proof of concept to demonstrate working ideas to construct the architecture of the system. Figure 5 below explains the SUS architecture. The architecture of the SUS prototype is standalone system. However, it is utilized on a database as mechanisms to store data, which are consist of staffs and duty shift. Connection to database is through Open Database Connectivity (ODBC).

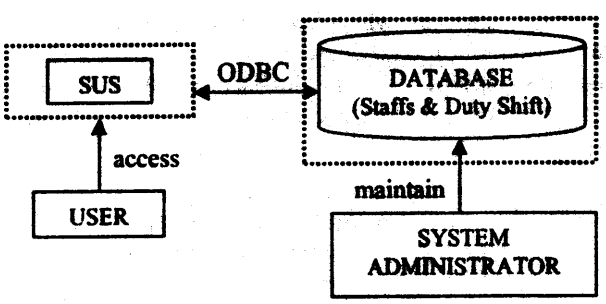

Figure 5: SUS architecture

User can access the system using SUS prototype while the system administrator is responsible to maintain the database. The purpose of using Java in the system is to examine the feasibility ideas of the projects. This effort can be done through an experiment with various parameters of GA elements.

The prototype provides basic functions to set parameters and how the output reports and schedules appear. The use of a number of tools and development resources would be also necessary to develop the SUS.

SUS will be tested using Pentium IV 1.7 GHz with Windows XP operating system that supports Java. The data was stored in Microsoft Access 2007 and it can retrieve using ODBC with Java Creator platform. Java Creator can directly compile or run Java program. The program is able to retrieve data from database using ODBC to calculate duty shift, staffs and generate the data.

3.4 Experimentation

Experimentation is the last phase in research methodology. A set of experiments were performed to assess the performance of a straightforward GA's on real UTHM SUS data. The experiments were conducted with varying degrees of crossover and mutation rates. The population size is also tested with a different size. However, the number of timeslot was maintained at 18. Elitism was applied in every run in this experiment to improve the performance.

This part is to find the optimal solutions with different parameters. The time taken to generate data in each test use a different parameters were recorded. Comparison with the results is to examine the effects of the optimal solution.

4. Findings/Results

In this study, results will obtain from the SUS prototype experimentation with different parameters. Initially, the purpose of the experiment is to find the optimal solution with different parameters with the similar number of time slots. In addition, the experiment is carrying out to examine the effects to the optimal solution obtained in respect to different kinds of crossover, mutation and population sizes.

The system prototype runs many times using different parameters. The number of time slots was fixed at 18 equal to 6 days. Each duty was allocated with a time slot ranges between 1 and 10. The default parameters set are shown in Table 3.

Parameters setting	Variable
Slot Number	10
Population Size	50
Elitism	True
Crossover Rate	0.8
Mutation Rate	0.05
Maximum Generation	1000

Table 3: Default parameters setting

Experiments will be done with a several testing methods using a different set of parameters. Combination of crossover rate and mutation rate should be obtained with an optimal solution while the other parameters were fixed. For the first testing, crossover and mutation rate is increased and decreased from default setting. This is because these two parameters can produced a different respond and effect.

Each testing with a different population size produces a good result. Assumption on the small population sizes can provide a quick calculation and less time require in generating the data. The best solutions are without clash and consecutives.

Therefore, this experiment will look into the effects from different slot size. The result shows that the increasing of slot size will give the shortest time to generate the data. The small generations provide an acceptable solution.

5. Significance

GA is a planning and combinational optimization method based on time to obtain the optimum solutions for developing SUS prototype. It can produce a different type of possibility with different sets of parameter. A comparison will be made between these possibilities in generating feasible and optimal results of the scheduling timetable. Thus, the prototype will be developed as an alternative approach to reduce cost and time in designing SUS for UTHM.

6. Conclusions

In this project, the SUS will be used as a tool for scheduling. By using the prototype, the scheduler only has to pick one suitable acceptable solutions. The prototypes require less time to generate data and produce several solutions. It can be used to get the optimal timetables to fulfil the hard and soft constraints.

The experiments using GA with different combination of crossover rate and mutation rate were the priority as the first testing. Apart from that, the population size was the second testing.

There is one limitation in this system. This system is can only be used by the trained users because the final output of the scheduling is represented by number of cell. It may confuse the naïve user. This can be one of the future works that can be done for better representation of the output such as display it in the way just like the manual scheduling.

The contribution of the study is to enhance the timing in producing the scheduling. The timetabling schedule requires a longer duration of time (one and a half month), but with the application of the proposed systems, the complete scheduling result is produced. This new solution method assists the scheduler in producing an efficient, effective and optimal schedule for the staffs.

In order to improve the capability of this prototype, future works can be undertaken to enhance the prototype as listed in the following:

i. Improve the number of staffs.
ii. Merge the other techniques (Hybrid System) such as fuzzy logic, expert system and neural network to improve the effectiveness and more efficient in scheduling problem.

In this project experiment, each testing will be produced using variation of parameters. The outcome of the experiment is generating at a different data and time. The different population size also generates effects in time length. If all parameters setting are fixed with a constant value, the result is also different when running the system in many times.

Population size, crossover rate and mutation rate typically interact with one another nonlinearly and cannot be optimized at one time. The feedback for the parameters setting comes from the population's success or failure on the fitness function [6].

7. Acknowledgement

The author would like to thank the UTHM for financial support for this work. This research is under UTHM Research Short Term Grant Vot 0345.

References

[1] Fang, H. L., "Investigating Genetic Algorithms for Scheduling". *Unpublished Thesis*: MSc. University of Edinburgh, 1992.

[2] Filho, G. R. and Lorena, L. A. N., "A Constructive Evolutionary Approach To School Timetabling". Retrieved December, 2004, from www.lac.inpe.br/~marcos/arsig2/CGA-timet-EVOCOP.pdf, 2000.

[3] Gy'ori, S. Petres, Z. and Varkonyi-Koczy, A. R., "Genetic Algorithms in Timetabling. A New Approach". MFT Periodika 2001-07. Hungarian Society of IFSA, Hungary, 2001. Retrieved February, 2007, from http://www.mft.hu/hallg/200107.pdf, 2001.

[4] Lalescu, L. and Badica, C., "Timetabling Experiments Using Genetic Algorithms". prezentata la TAINN'03, Canakkale, Turcia, 2003. Retrieved February, 2007, from http://software.ucv.ro/Cercetare/Inginerie_ Software/inginerie_software.html, 2003.

[5] Michalewicz, Z., "Genetic Algorithms + Data Structures = Evolution Programs; Third, Revised and Extended Edition". New York, New York: Springer-Verlag, 1999.

[6] Mitchell, M., "An Introduction to Genetic Algorithms", MIT Press, Cambridge, Massachusetts, London, England, ISBN 0-262-13316-4 (HB), 0-262-63185-7 (PB), 1999.

[7] Nunamaker, Jr. J. F. Chen. M. & Purdin, T. D. M., "System Development In Information Systems Research". *Journal of Management Information Systems/Winter 1990-91. Vol. 7, No. 3, pp. 89-106*, 1991.

[8] Pinedo, M., *Scheduling Theory, Algorithms, and Systems Second Edition*, Prentice Hall, Upper Saddle River. New Jersey 07458, ISBN 0-13-028138-7, 2002.

[9] Yasin, A. Puteh, N. and Tahir, H., "Examination Timetabling With Genetic Algorithms". *Jurnal Tek. Maklumat & Sains Kuantitatif, UiTM, Jilid 4, Bil (1), pp. 11-2*, 2002.