UTM UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

Timetable Scheduling System using Genetic Algorithm (tsuGA)

Lim Ying Ying & Hazinah Kutty Mammi
School of Computing, Faculty of Engineering
Universiti Teknologi Malaysia
Email: yylim1224@gmail.com; hazinah@utm.my

Submitted: 1/9/2021. Revised edition: 27/9/2021. Accepted: 26/10/2021. Published online: 15/11/2021
DOI: https://doi.org/10.11113/ijic.v11n2.342

***Abstract*—Current timetable scheduling system in School of Computing (SC), Universiti Teknologi Malaysia (UTM) is done manually which consumes time and human effort. In this project, a Genetic Algorithm (GA) approach is proposed to aid the timetable scheduling process. GA is a heuristic search algorithm which finds the best solution based on current individual characteristics. Employing GA and scheduling information such as rooms available and timeslots needed, it is shown that scheduling can be done more efficiently, with less time, effort and errors. As a testbed, a web application is developed to maintain records needed and generate timetables. Introduction of GA helps in generating a timetable automatically based on information such as rooms, subjects, lecturers, student group and timeslot. GA reduces human error and human efforts in the timetable scheduling process.**

***Keyword*—Timetable scheduling, genetic algorithm, web application**

I. INTRODUCTION

School of Computing (SC) is one of the school of Faculty of Engineering in Universiti Teknologi Malaysia (UTM). SC offered for both undergraduate and postgraduate programmes. In the year 2019, SC has a 1208 intake for undergraduate and 550 for postgraduate. Undergraduate programmes that been offered are Software Engineering (SCSJ), Computer Networks and Security (SCSR), Graphics and Multimedia Software (SCSV), Data Engineering (SCSP) and Bioinformatics (SCSB). In order to support the studies of all students, School of Computing provides ten computer labs, seven lecture rooms and other labs for education purposes.

Timetable scheduling is a process of creating a timetable to fit some conditions. In timetable generation, certain constraints must be followed, which include hard and soft constraints. Constraints for a timetable generation included hard constraints and soft constraints. Hard constraints are constraints that must be fulfilled and not violated while soft constraints are constraints that may be violated but satisfaction of the constraints are highly desirable to produce a good timetable [9].

Timetable scheduling is a nondeterministic polynomial time hard (NP-hard) problem which is difficult to solve by traditional methods [10]. In SC, this procedure is done manually by the academic office (AO). Process of trial and error is repeated to create a timetable that does not violate hard constraints. Manual scheduling consumed lots of efforts and time. Timetable Scheduling using Genetic Algorithm (tsuGA) is proposed to help in timetable scheduling for SC. It is applied in a web application which allows the manager and directors to add details of subjects, classroom, group of students and lecturers. Generation of a timetable will then be executed by tsuGA based on those input constraints by using GA. Generations of chromosomes which each represents a solution will be created and their fitness values will be analysed. Compared to manual scheduling, tsuGA provides advantages in terms of efforts and time.

II. PROBLEM BACKGROUND

Current existing timetable scheduling system for undergraduate programmes in SC is done manually by AO. This process is done by using trial and error to find a best timetable to fit for five programmes, which are SCSJ, SCSV, SCSR, SCSP and SCSB. AO assigns a lecture room to a lecture and makes sure that there is no classroom assigned with two lectures at the same time. If any of the timetable is found clashed, AO will then rearrange the timetable. The timetable is then passed to academic office administrator (AOA) to key in the timetable into the computer system and check possible clashing of lectures.

Director of each programme will then assign lecturers to each class.

Hard constraints are constraints that must be fulfilled and not violated while soft constraints are constraints that may be violated but satisfaction of the constraints are highly desirable in order to produce a good timetable. Hard constraints for a timetable included no lecture room is assigned to two lectures at the same time, no lecturer is assigned to two lectures at the same time. No soft constraint is introduced in tsuGA.

Existence of clashes will only be shown after the timetable is input into the computer system or getting comments from students or lecturers. AO may not know the existence of clashes during the process of manual scheduling. If any clashes are found, the process of trial and error will be repeated to rearrange the timetable. This procedure consumes time and effort. Multiple trials have to be done to get the most suitable timetable for each course and subject.

A web-based application, tsuGA is introduced to help in the current timetable scheduling system. It uses GA to find the best fit timetable by referring to constraints such as subject, lecture room and group of students. Even if any changes are needed, the procedure of checking for a new best fit timetable can be done without much labour effort and thus increases efficiency.

Hard constraints used by tsuGA:

(a) No lecturer should be teaching more than one class at the same time.

(b) No room is assigned to more than one class at the same time.

(c) No subjects are clashed in each course with the criteria of following conditions are not found:

- All sections of subject A have duplicate time with all sections of subject B.
- Either subject A or subject B only offer one section and the subject is not elective and subject A and subject B have duplicate time in any section.

III. CASE STUDY

As the project focuses solely on GA, case study of systems using GA for timetable scheduling process are chosen and studied. GA model proposed by each case study is studied and modified to be applied in tsuGA. As shown in Table I, three elements compared are hard constraints, soft constraints and optimization. In tsuGA, the chosen hard constraints focused on are class, classroom, classroom capacity, equipment, students and lecturers. Optimization chosen for tsuGA are tournament elimination selection and uniform crossover.

A. Making a Class Schedule Using a Genetic Algorithm [6] (CS1)

- A Java program introduced GA implementation in timetable scheduling with detailed explanation.

B. On Improvement of Effectiveness in Automatic University Timetabling Arrangement with Applied Genetic Algorithm [7] (CS2)

- University timetable scheduling with a proposed GA model.

C. Solving Timetable Scheduling Problem using Genetic Algorithm [11] (CS3)

- Introduce GA using C++ with Standard Template Library support.

D. A Utilization-based Genetic Algorithm for Solving the University Timetabling Problem (UGA) [1] (CS4)

- Utilized GA is proposed to get best room utilization for timetable scheduling.

TABLE I. COMPARISON BETWEEN EXISTING SYSTEMS

Features	CS1	CS2	CS3	CS4	tsuGA
Hard constraints					
(a) Class	✓	✓	✓	✓	✓
(b) Classroom capacity	✓	✓		✓	✓
(c) Equipment	✓				✓
(d) Student	✓	✓	✓	✓	✓
(e) Lecturer	✓	✓	✓	✓	✓
(f) Recess					✓
Soft constraints					
(a) Recess		✓			
Optimization					
(a) 3D structure representation			✓		✓
(b) Tournament elimination selection			✓		✓
(c) Uniform crossover			✓		✓

IV. PROPOSED GA MODEL

In this section, a GA model is proposed after study on current existing systems. It is modified to fit SC timetable scheduling process.

A. Chromosome Representation

Fig. 1 shows a three-dimensional chromosome used to represent the chromosome of tsuGA. Each dimension represents subject, room and timeslot id respectively. This is to ensure there is no subject assigned with a duplicate room and timeslot. It also eases the process to find for an available room and timeslot to be reassigned when clashing conditions occur.

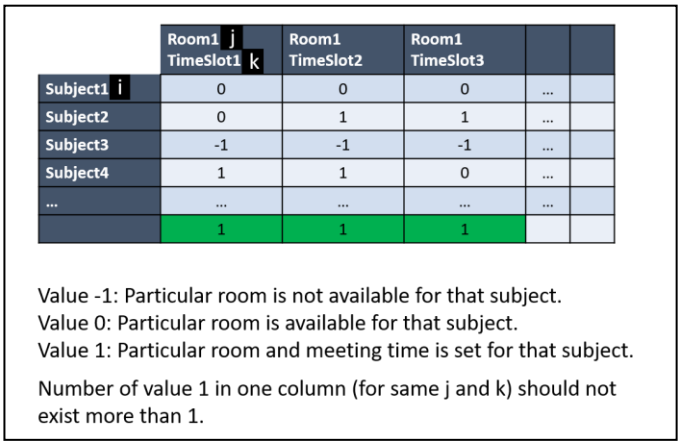

Fig. 1. Chromosome Representation

B. Fitness Function

The following equations, Eq. 1 and Eq. 2, show the number of conflicts calculation in tsuGA and the fitness value of tsuGA respectively. The number of conflicts is computed for each generated schedule and it is then used to calculate the fitness value of the schedule. Ideal fitness value is 1 when there is no conflict found in a schedule. No soft constraint is needed in this system. No weight is used in this system as all hard constraints are important and need to be resolved. Hard conflicts used in tsuGA are discussed in section II.

$$number\ of\ conflicts = \sum_{i=1}^{3} cost^{hard} \quad (1)$$

where,

$$cost_1^{hard} = \sum_{i=1}^{k} lecturer\_clash \quad (1.1)$$

$$cost_2^{hard} = \sum_{i=1}^{k} course\_clash \quad (1.2)$$

$$cost_3^{hard} = \sum_{i=1}^{k} room\_clash \quad (1.3)$$

where,

$$k = number\ of\ classes \quad (1.4)$$

$$fitness = \frac{1}{number\ of\ conflicts+1} \quad (2)$$

C. Crossover Operator

Crossover in tsuGA is designed to give better solutions by comparing between two parent schedules before any changes are made. Fig. 2 shows crossover pseudocode.

```
FOR each class in parent1
  IF class has no conflicts with other remaining class
    IF hit crossover rate
      take class room time of parent2 into parent1
    ENDIF
  ELSE
    put class room time of parent2 into parent1
    IF class has no conflict with other remaining class
      take class room time of parent2 into parent1
    ELSE
      heuristic search for available slot and room
  END
ENDFOR
```

Fig. 2. Crossover Pseudocode.

D. Mutation Operator

```
FOR each class in parent
    IF hit mutation rate
      heuristic search for available slot and room
    ENDIF
ENDFOR
```

Fig. 3. Mutation Pseudocode

Mutation is carried out to prevent schedules to have no diversity, which indicates no new genes are introduced in the current solution. In this system, mutation finds for new available time and room that has not been used by another class. Fig. 3 shows pseudocode of mutation. Fig. 4 shows illustration for heuristic search to get new available time slot and room.

	Room1 TimeSlot1	Room1 TimeSlot2	Room1 TimeSlot3		
Subject1	0	0	0	...	
Subject2	1	1	0	...	
Subject3	-1	-1	-1	...	
Subject4	0	0	0	...	
...	...	...	...	...	
	...	...	0		

Fig. 4 Heuristic Search Illustration

V. Design and Development

A. Requirement Analysis

Requirement analysis is carried out at the initial stage of system development to get user requirements for the system. An interview is carried out with AO, the clerk, who is in charge of the SC timetable scheduling process. Current workflow discussed in Section II.

B. Analysis Phase

Use case diagram is documented to show relationships between users and system. Total ten functions introduced in

tsuGA systems for five types of users, which are, students, lecturers, directors, manager and admin. Each of them having different privileges to access to the system. Use case diagram is shown in Fig. 5.

Fig. 5. Use Case Diagram for tsuGA

C. Design Phase

System architecture used in tsuGA is client server architecture. All clients, which included students, lecturers, manager, admin and directors will connect to the web server through the Internet. Web server will then request data or write data to the database.

D. Development Phase

Two applications are developed separately which are frontend tsuGA system and backend which provides RESTful API for communication between system and the database. Angular framework is used to develop tsuGA interfaces while Spring Boot framework is used to develop backend services. Java language is chosen for the GA process. Both applications are then deployed on Tomcat server.

VI. PROPOSED SYSTEM

A timetable must be generated by the manager before it is available for other users. Fig. 6 shows the interface when a timetable is being generated. A modal is shown to request the manager to wait for the process to be done.

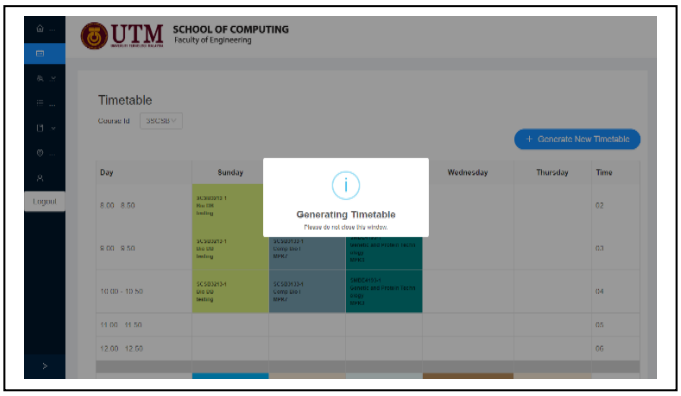

Fig.6. Generate Timetable Interface

Generated timetable is ready for viewing by students, lecturers, directors and manager. Fig. 7 and Fig. 8 show timetable view by students and lecturers respectively. Directors and manager will have the exact timetable view as students with additional feature to select courses. Besides, manager has unique feature to generate a timetable.

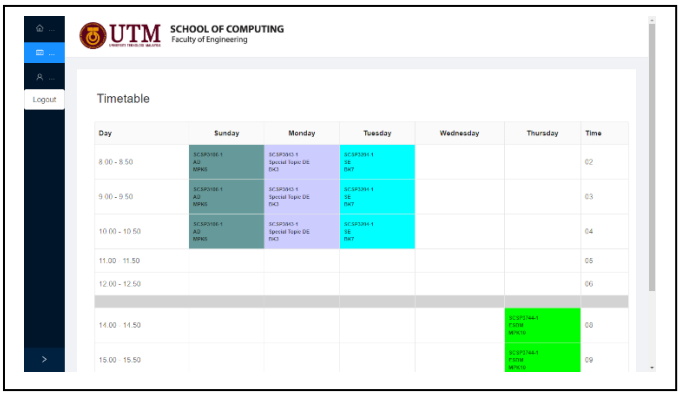

Fig. 7. Timetable View by Student

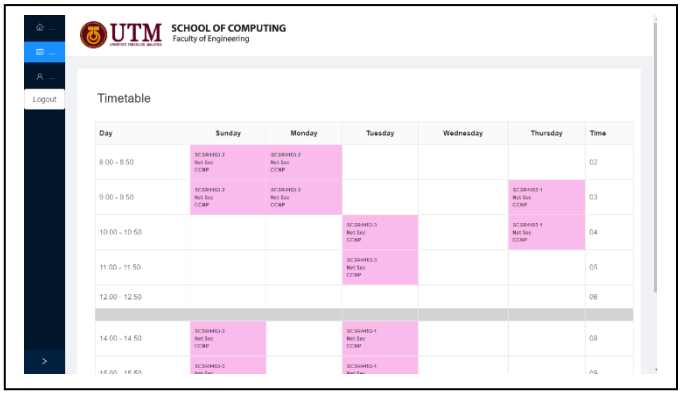

Fig. 8. Timetable View by Lecturer

VII. TESTING AND RESULTS

A. Testing Data

Details of testing data for timetable is shown in Table II. Users are then tested with schedules offered with details below. A collection of different types of subjects which include Generic which are subjects taken by all programs in SC; Core which are core subjects of the given programme; and Elective are programme specific subjects that students can choose from. Generic and Core subjects are compulsory subjects that must be taken by the students.

TABLE II. OVERALL SUBJECT OFFERED DETAILS

Subject Type	Number of subjects	Number of sections
Generic	2	3
Core	12	28
Elective	19	28
Total	**33**	**59**

B. Results

In general, tsuGA generates timetables which fulfilled hard constraints that have been defined in Section II. In Fig. 9, it is shown that 93% of students have no clashed timetable while there is one student having clashed timetable. This clashing condition only occurred between elective subjects.

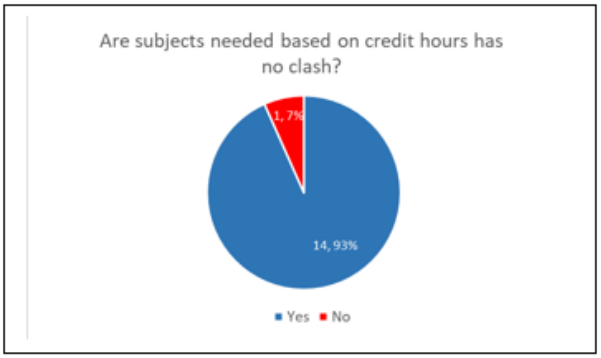

Fig. 9. Clashing Result - Student

This is due to restriction in timetable generation for elective subjects. Not all elective subjects will have no clash with each other. Example of clashing of elective subjects is shown in Fig. 10. It is unavoidable due to the large number of elective courses offered to a programme, which is SCSR in this example. A timetable has no enough timeslot to fit 14 courses offered for 3SCSR. Table III shows an example of clashing conditions of elective courses. Clashing will only occur when three elective courses are taken at the same semester (condition 4 and condition 5 in Table III). In this example, students can only take two out of three elective courses that face clashing condition (condition 1, condition 2 and condition 3 in Table III). Clashing condition of elective courses does not affect the conflicts of schedule as they are optional courses instead of mandatory courses to be taken in one semester.

Clashing conditions between elective courses can be improved by introducing soft constraints to the system. However, occurrence of clashing condition is unavoidable when there is a large number of elective courses being offered to a student group.

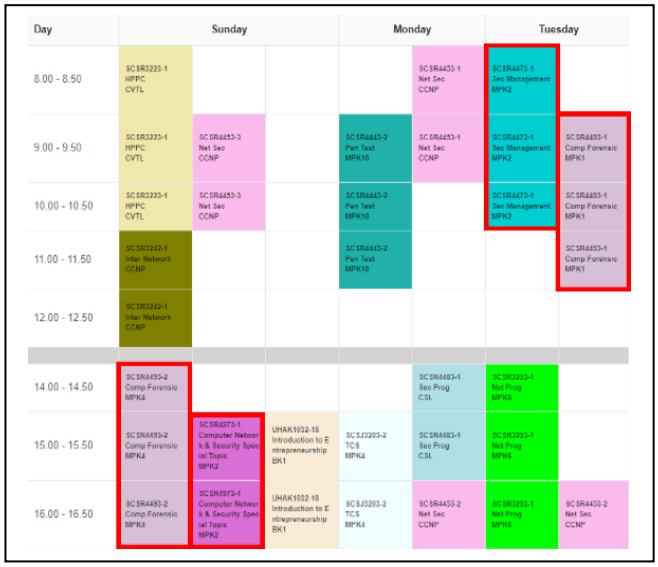

Fig. 10. Elective Subjects Clash in SCSR

TABLE III. TIMETABLE ELECTIVE CLASH SUMMARY

Condition / Taking	Subjects				Clashing Result
	SCSR4473	SCSR4483		SCSR4973	
	Section1	Section1	Section2	Section 1	
1	√		√		✕
2	√			√	✕
3		√		√	✕
4	√		√	√	√
5	√	√		√	√

A few generations of timetables are created to get running time required for a new timetable. Table IV shows time consumption of each generation and average time required for new timetable generation of third-year programmes. It is shown that the average time required is around 5 minutes 34 seconds. This shows that the proposed GA increases the efficiency of the current timetable scheduling system which took days to complete manual assignment of rooms and timeslots. Besides, timetable generated by tsuGA has no conflicts with hard constraints which means human errors can be eliminated during the timetable scheduling process.

TABLE IV. TIME CONSUMPTION FOR TIMETABLE GENERATION

No	Time Consumption
1	6 minutes 20 seconds
2	4 minutes 40 seconds
3	6 minutes
4	6 minutes 30 seconds
5	4 minutes 18 seconds
Average	**5 minutes 34 seconds**

VIII. CONCLUSION

In a nutshell, tsuGA helps the timetable scheduling process in SC to be more efficient in terms of human efforts and time. Proposed GA is implemented and successfully resolves defined hard constraints. It reduces time required and eliminates human errors during assignment of rooms and timeslots.

This system can be expanded to combine with the course registration system for future planning. It will allow students to go through the timetable for subjects they would like to register and then register through the system. GA used can be refined to create a better timetable by using soft constraints to make sure electives can be fulfilled at the best solutions. Furthermore, GA can be improved by redefining hard constraints and soft constraints with more details. By introducing soft constraints, the generated solution could satisfy more users and reduce the occurrence of clashing conditions between elective courses.

REFERENCES

[1] Abdelhalim, E. A. and Khayat, G. A. (2016). A Utilization-based Genetic Algorithm for Solving the University Timetabling Problem (UGA) [online]. Available at: https://www.sciencedirect.com/science/article/pii/S1110016816000703#b0350 (Accessed: 2 March 2020).

[2] Ambler, S. W. (2005). Feature Driven Development (FDD) and Agile Modeling [online]. Available at: http://agilemodeling.com/essays/fdd.htm (Accessed: 20 March 2019).

[3] Das P. (2016). Genetic Algorithm: An Implementation using JavaScript and HTML5. [online]. Available at: https://www.codeproject.com/Articles/1127321/Genetic-algorithm-an-implementation-using-JavaScri (Accessed: 15 March, 2019).

[4] Deoras S. (2019). 5 Languages to Use for Genetic Programming [online]. Available at: https://www.analyticsindiamag.com/5-languages-to-use-for-genetic-programming/ (Accessed: 25 March 2019).

[5] Garg, P. (2017). 9 Things You Must Know About FDD – Feature Driven Development [online]. Available at: https://www.openxcell.com/9-things-must-know-fdd-feature-driven-development (Accessed: 1 April 2019).

[6] Janković, M. (2008). Making a Class Schedule Using a Genetic Algorithm [online]. Available at: https://www.codeproject.com/Articles/23111/Making-a-Class-Schedule-Using-a-Genetic-Algorithm (Accessed 20 March 2019).

[7] Khonggamnerd, P. and Innet S. (2009). On Improvement of Effectiveness in Automatic University Timetabling Arrangement with Applied Genetic Algorithm. *2009 Fourth International Conference on Computer Sciences and Convergence Information Technology*, 1266-1270.

[8] Mallawaarachchi, V. (2017). Introduction to Genetic Algorithms—Including Example Code [online]. Available at: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3 (Accessed 9 March 2019).

[9] Patil, M. K., Subodh, R. S. and Pawar, P. A., Narendrasingh, N. (2014). Web Application for Automatic Time Table Generation. *International Journal of Current Engineering and Technology*, 1936-1938.

[10] Sapru, V., Reddy, K., Sivaselvan, B. (2010). Time Table Scheduling using Genetic Algorithms Employing Guided Mutation. *2011 IEEE International Conference on Computational Intelligence and Computing Research [online]*. Available at: https://ieeexplore.ieee.org/abstract/document/5705788 (Accessed: 9 March 2019).

[11] Sigl, B., Golub, M. and Mornar, V. (2003). Solving Timetable Scheduling Problem Using Genetic Algorithms. *25th Int. Conf. Information Technology Interfaces IT1 2003*, 519-524.

[12] Zukanov, V. (2018). Top Java Application Servers: Tomcat vs. Jetty vs. GlassFish vs. WildFly [online]. Available at: https://stackify.com/tomcat-vs-jetty-vs-glassfish-vs-wildfly/#wpautbox_latest-post (Accessed: 20 April 2019).