Genetic Algorithm for Solving University Timetabling Problem

U. M. Modibbo[1], I. Umar[2], M. Mijinyawa[3], R. Hafisu[4]
[1,2,3]Department of Statistics and Operations Research, Modibbo Adama University of Technology, P.M.B. 2076 Yola Nigeria
[4] Department of Statistics, College of Science and Technology, Adamawa State Polytechnic Yola Nigeria
[1]umarmodibbo@mautech.edu.ng, [2]iuchiroma1@gmail.com, [3]m.mijinyawa@gmail.com, [4]rafiyatuhafisu@gmail.com

Abstract:
This study focused on solving the University Timetabling Problem through a procedure, based on Genetic Algorithm for generating timetable in the "Department of Statistics and Operations Research Modibbo Adama University of Technology Yola". A proposed Genetic Algorithm timetabling problem was modeled using "Microsoft Visual C++ integrated environment". We also developed our selection process in such a way that the main program would need to run crossover and mutation process in less amount of time, which increase the convergence rate of new schedules and decrease the new scheduling generation time.

Keywords: Genetic Algorithm, Timetabling, Microsoft Visual C++, Crossover and Mutation.

1. Introduction

Timetabling means "the allocation, subject to constraints, of given resources to objects being placed in space-time, in such a way as to satisfy as nearly as possible a set of desirable objectives, (Wren, 1996)". A more realistic definition of time-tabling can be refer to as an assignment of "lecturers, vehicles, public events, etc." to scarce resources with conflicting hard or soft constraints having some priorities over a long period of time. (Aydin, 2008)

According to Burke et al. (1994), "Timetabling problem contains four factors which is a finite set of resources, times, meetings and constraints. Timetabling problems come in several types including nurse scheduling, transportation timetabling, educational timetabling and sport timetabling. All of these have shown an important problem and challenging tasks for the researchers. Educational timetabling is widely studied among all the timetabling problems. The main factor of affecting a wide range of various stakeholders is the quality of timetabling. There have been relatively close problems between course and exam timetabling".

Allocation of events or activities to people or machines, etc. can be viewed as various categories of real life time-tabling problems. Mostly, constructing a promising and workable timetable is a hard problem and very difficult to achieve as a result of the scarce and yet competing resources (time, place, people, etc.). Therefore, developing efficient and effective workable timetable becomes a crucial problem of concern for most researchers in this field. (Aydin, 2008)

An inevitable problem that every University has to solve by any means is thetimetabling problem. It is considered as Non-Deterministic Polynomial (NP) problem. More specifically, individual departments are responsible for doing this task. Usually, the problems are addressed in different context by gearing both capital and human resources in most institutions. Different solution approaches and techniques for solving this kind of problem were introduced such as optimization algorithm concepts, multi objective among others (Kazarlis 2005). Genetic algorithm (GA) according to Abramson & Abela (1992), was "developed from evolution of living things to survive on the Earth under such changeable environment". In evolutionary studies, the next generations are formed from parental chromosomes using the GA operation. Desired results in pairs and groups are obtained by solving the problems through construction of many parents' chromosomes. (Paitoonwattanaki and Vongvipaporn 2000) and (Junnatat, 2005).

To reach a solution of high quality and satisfactorily it deserved rarely difficult and hard, university timetabling problems are viewed as Non-deterministic Polynomial. (Gyõri, Petres, & VárkonyiKóczy, 2001), (Srinin, Rojanavasu, & Pin-nguen, 2005) and (Ozcan & Alken, 2002). A GA is an algorithm-wise and powerful tool in finding an optimal solution (Srinin et al., 2005) and (Burke et al., 1994). Hence, is employed to solve these problems

2. Review of Related Work

According to Schaerf (1999), Timetable can be classified as subject, school and examination timetables respectively. Brownlee (2005) investigated the problem of class timetabling and attempted to reproduce three different approaches to solve it using mimetic algorithm. In a study conducted by Norberciak (2006), highly large constrained problems of timetabling were described universally in different domain. The framework of evolutionary algorithm and Tabu search were employed to fasten the process. Operating parameters of the method were established using hyper heuristics concept. Results look appealing but it needs improvement in the algorithm such as employing some form of local search.

A "multi-agent system for University course timetable scheduling" was reported in Oprea (2007). The benefits of the approach were analyzed incorporating the negotiation, cooperation and communication processes. An architecture of a multiple agent-based system was disclosed.

Also, Abdullah (2008), reviewed the current manual timetable systems and developed a web-based timetable

system using genetic algorithm. Khonggamnerd and Innet, (2009) proposed an automated model of genetic algorithm for university timetable assignment. Crossover, mutation, and selection were used as the three genetic operators in the model. Optimized results were obtained with crossover probability of 0.70 for the hard constraints. However, modifications are required to satisfy the soft constraints.

Nabeel (2009) established based on the concepts of GA and Great Deluge (GD) a newly hybridized algorithm for solving university lecture courses allocation problem. He applied the model on standard benchmark problems which gave better results. The proposed method produced acceptable solution domain. Moreover, it gave consistently good results across various benchmark problems.

3. Statement of the Problem

University Timetabling is the arrangement to reach congruence and relation among students, subjects, lecturers, rooms, periods and time space (Nuntasen & Innet, 2007). To gain effective timetable, problems have to be solved. Such problems are able to be revealed by constraints, which can be hard or soft or both as the case may be, (Burke et al.1994 & Gyõri, 2001). In timetabling, hard constraints are not acceptable they need robust and compulsory treatment. These problems are not allowed to happen. For example, a lecturer cannot lecture for more than one subject in the same period of day, also he cannot teach the same subject for different level at the same time period. Soft constraints on the other hand can be accepted within a minimum frequency. Perfectiveness of timetabling will be maximized, e.g., in a particular schedule, students' nearby subject should be loaded without a time space of two or more periods. Both of them have been considered in this study. Exams Officers of various Department found it difficult to produce lectures and exams timetable without clashes, to the extent some students missed lectures and/or exams as a result of continuous tempering of the released timetable.

4. Methodology

System analysis development life cycle was use in the research design which includes the following steps:

i. *Preliminary Investigation:* Information on the department timetable and how it is design was obtained through interview and observation.
ii. *System Design:* Input, output, storage and processing requirement of the proposed Genetic Algorithm Timetabling system was described.
iii. *System Development:* Microsoft Visual C++ integrated developmentenvironment was used as the development tools for the proposed Genetic Algorithm Timetabling system.
iv. *System Implementation:* At this stage the proposed system developed was tested using the record collected from the Department of Statistics and Operations

5. Data Collection and Analysis

Data such as number of lecturers in the Department, their Qualification, experience, etc, number of classes, number of courses and their units, lecture days among others were collected in the "Department of Statistics and Operations Research, Modibbo Adama University of Technology, Yola" and Microsoft Visual Studio was used for the analysis.

6. Algorithm

A genetic algorithm was applied to solving university timetabling problems under some certain constraints. The algorithms are:

1. parents and chromosomes (P_1 & P_2) were produced as prototypes
2. children chromosomes (Ch_1 & Ch_2) are produced by crossover process
3. Two best chromosomes that give highest fitness value was selected using fitness function.
4. Crossover process or mutation process by randomizing a probability value was decided whether to be used.
   - When crossover process was chosen, steps 3 – 4 were repeated.
   - When mutation process was employed, each chromosome "fitness value" with the "fitness function" was examined, and only the chromosome that gave highest fitness value to do the mutation process was selected. The chromosome was put back into the pool and step 4 was repeated.
5. The process continued until the "fitness value" is achieved or the user terminated the process.

Genetic algorithm uses only four chromosomes in the mating pool employed in the crossover process or mutation process. This reduces utilization of computation memory. Figure1. Shows the Steps of Genetic Algorithm Application for Timetabling and Figure2. Shows how desired chromosomes were produced from their parents**.**

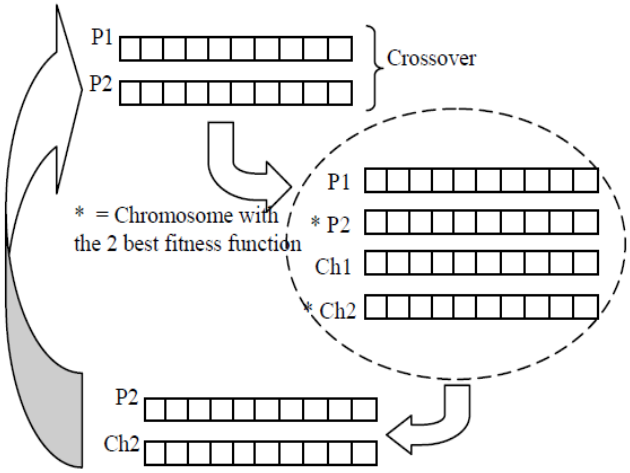

Fig. 1. Steps of Genetic Algorithm Application for Timetabling (Nuntasen & Innet, 2007)

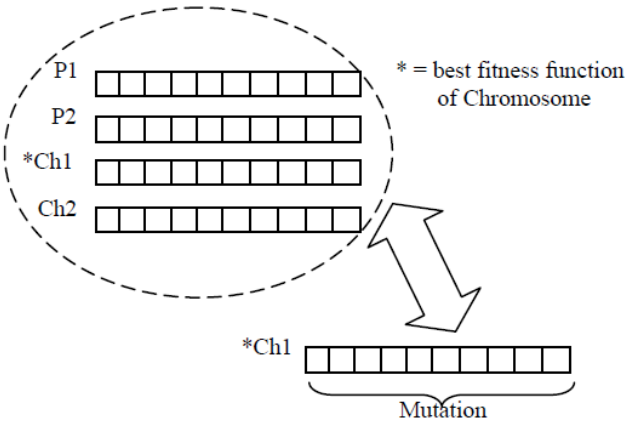

Fig. 2. The best chromosome selection for mutation (Nuntasen & Innet, 2007)

6.1 Constraint Identification:
Bachelor's degree of Statistics and Operations Research regular course timetable, 2016-2017 academic years of Modibbo Adama University of Technology Yola, was used to identify constraints. Six periods a day from Monday to Friday was adopted in the schedule. They were explained in details in the next subsection.

6.1.1 "Hard Constraints"
These are avoidable limitations in any timetabling which must not be breach. For instance, Different lectures cannot be delivered in two different places at the same time by the same teacher. Four of such constraints are considered here:

Lecturer or Room cannot be allocated for different tasks at the same period of the day.

i. No two subjects are allowed to be allocated to a room at the same period of a day.
ii. Post graduate class is not allocated to any undergraduate course.
iii. Specific courses arestrictly assigned to specific lecturer.
iv. The capacity of room(s) is ensured to match with student size.

6.1.2 "Soft Constraints"
These limitations can be broken with a minimum breach. E.g, booked a class(s) at a preferred day and time. These constraints are:

i. In a particular schedule, students' nearby subject should be loaded without a time space of two or more periods.
ii. A day in a week should be off classes for the lecturers to have a research time.
iii. Every day in the schedule, period 3 and 4 should be reserve for lunch
iv. There should be no more than four continued lecture periods for students per day.
v. There should be no more than four continued lecture periods for lecturers per day.

6.2 Genetic Algorithm applied in "Timetabling Chromosomes"
According to Nuntasen & Innet (2007) and Junnatat (2005) "Timetabling Chromosomes applied by GA were constructed into tuple or group of elements (E) which were spaces for selected input. In this study, elements comprised of three types of data; which are lecturer (L), Subject (S) and Room (R). In each element, it was presented in a form of E = {L, S, R}. A set of E represented as constituent elements of chromosomes. So, E = {E_1…, E_n} represented as the elements of input to timetabling while each of E(1-n) comprised of consequent set of L, S, and R. Each of them had a subset of its own. It was shown as L = {L_1, L_2, L_3…, L_n}, which represented lecturers in that semester. In the same way of a set of Subject and a set of Room in that semester was done as S = {S_1, S_2, S_3…, S_m}and R = {R_1, R_2, R_3…, R_n}, respectively.

A chromosome was stringed following the sequences of periods that yielded the length of that chromosome as shown in equation (1)

$$lChrom = G * P(n) \quad (1)$$

where:

lChrom: the length or number of bits of each chromosome

G: Group of students who register the subjects following the plan in each major

P(n): number of periods in the timetabling.

Figure.3 shows timetabling chromosomes resulting from equation (1).

Bit	1	2	…	…	9	9	…	17
Chro	E1	E1	…	…	E2	E2	…	E9

Fig. 3. Timetabling chromosomes (Nuntasen & Innet, 2007)

6.3 Fitness Function
Fitness function was used to measure appropriateness of timetabling and search to find the best one, which is capable to be the raise problems. Fitness function constructed from timetabling conditions. It can be shown as the following equations:

Fitness function =

$$W_1 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \sum_{R=0}^{L=n} \text{Bound Lecturer Clash}[((L_{L,}S_s)P_P)G_G]$$

$$+W_2 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \sum_{R=0}^{L=n} \text{Bound Room Clash}[((R_{R,}S_S)P_P)G_G]$$

$$+W_3 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \text{Break not more than 2 period}[((R_R,P_p)G_G)]$$

$$+W_4 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \text{Break Period 3 or period 4}[((S_s, P_p)G_G)]$$

$$+W_5 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{L=0}^{L=n} \text{Lecturer Load}[((L_L, P_P)G_G)]$$

$$+W_6 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \text{Student Load}[((L_L, P_p)G_G)]$$

$$+W_7 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \sum_{R=0}^{L=n} \text{Lecturer Study day}[((L_L, P_P)S_S)\ G_G]$$

$$+W_8 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \sum_{R=0}^{L=n} \text{Bound Room Size}[((R_R, S_S)P_P)G_G]$$

$$+W_9 \sum_{G=1}^{G=n} \sum_{P=1}^{P=40} \sum_{S=0}^{S=n} \sum_{R=0}^{L=n} \text{Bound Room for Undergraduate}[((R_R, S_S)P_P)G_G] \quad (2)$$

where weight of condition for each constraint is as follows:

W_1: weight of condition which a lecturer is not allowed teaching different subjects in the same period.

W_2: weight of condition which a room is not allowed to be used to teach different subjects in the same period.

W_3: weight of condition which a group of students should not have more than two periods free before any subjects in timetabling schedule.

W_4: weight of condition which the period 3-4 should be free for lunch time in a day.

W_5: weight of condition which a lecturer student should not have classes more than 4 continuing periods.

W_6: weight of condition which a group of students should not have classes more than 4 continuing periods.

W_7: Weight of condition which a Lecturer should have a day free to study

W_8: Weight of condition which the capacity of rooms must match with student size

W_9: Weight of condition which no lectures should be allocated to PG class

In addition, the value of each function can be explained as seen below:

Bound Lecturer Class $[((L_L, S_s)\ P_P)\ G_G]$
- Equal 0: when P are as L_L (no lecturer) or there is the same L_L and the same S_s in the same P
- Equal 1: when the same L_L happens in the same P but different S_s

Bound Room Clash $[((R_R, S_S)\ P_P)\ G_G]$
- Equal 0: when as others
- Equal 1: when the same R_R happens in the same P but different S_s

Break not more than 2 periods $[((R_R, P_p)\ G_G)]$
- Equal 0: when as others
- Equal 1: when SS happens in P and P-4 or S happens in P and P+4

Break Period 3 or period 4$[((S_s, P_p)\ G_G)]$
- Equal 0: when Ss happens in P3 or when Ss does not happen in P3 and P4
- Equal 1: when Ss happens in P3 and P4

Lecturer Load $[((L_L, P_P)\ G_G)]$
- Equal 0: when P happens in LL while $P \leq 4$
- Equal 1: when P happens in LL while $P > 4$

Student Load $[((L_L, P_p)\ G_G)]$
- Equal 0: when Ss happens in P of each G while $Ss \leq 4$
- Equal 1: when Ss happens in P of each G while $Ss > 4$

Lecturer Study day $[((L_L, P_P)\ G_G)]$
- Equal 0: when P are as L0 (lecturer has no lectures) and no Ss happen in P
- Equal 1: when P are as L1 (Lecturer has lectures) and S happen in P

Bound Room Size $[((R_R, S_S)\ P_p)\ G_G]$
- Equal 0: When RR happen in P for particular Ss and $R \geq G$
- Equal 1: when $R < G$ for a particular P and SS

Bound Room for Undergraduate $[((R_R, S_s)\ P_P)\ G_G]$
- Equal 0: when particular SS and P does not happen in RR
- Equal 1: When SS and P happen in RR

The Best answers of timetabling by GA application comprises the minimum target function, which is not allowed to have the hard constraintsand the one which should yield minimize soft Constraint".

6.4 Genetic Operator

A genetic process begins from the two initial chromosomes called parents and produces subsequent ones Crossover and Mutation known as operators following the genetic evolution, including process development of the Crossover and Mutation. Data of L, S, and R was examined in level of element for timetabling. Crossover and Mutation process in timetabling was ran within every chromosome's string, which was occupied by students' groups.

6.4.1 Crossover Operator

"Crossover is an exchange element of two parents' chromosomes. The crossover operation combines data in the hash maps of two selected parents and creates a vector of slots according to the content of the new hash map. The operation splits hashmaps of both parents in parts of random sizes which are defined by the number of crossover points which are defined in the chromosomes parameters. Then, it alternately copies parts from parents

to the new chromosome and forms a new vector of slots". This is shown in figure 4 and 5 respectively.

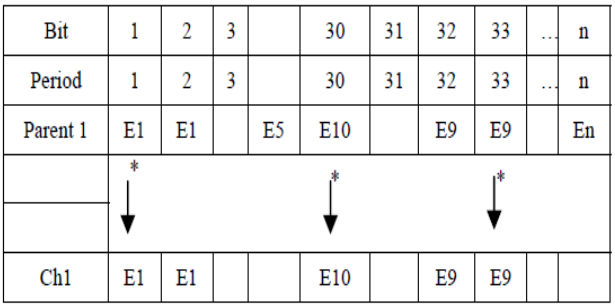

Fig. 4. Chromosome random and selection for crossover (Nuntasen & Innet, 2007)

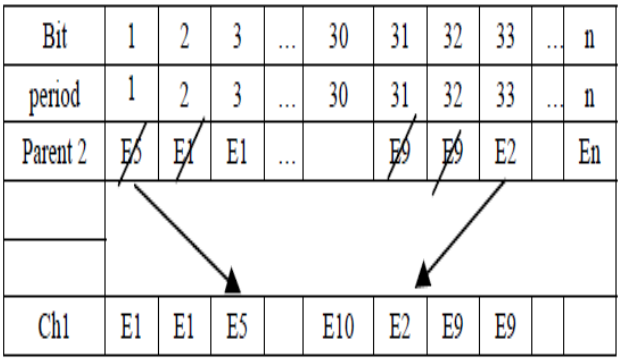

Fig. 5. Delete and Select Element from the Second Parents of Chromosome (Nuntasen & Innet, 2007)

6.4.2 Mutation Operator

The mutation operation randomly takes a class and moves it to another chosen slot. The number of classes which are to be moved in a single operation is defined by the mutation size in the chromosomes parameters.

The process of chromosome mutation would be done by altering the elements randomly, as shown in Figure 6.

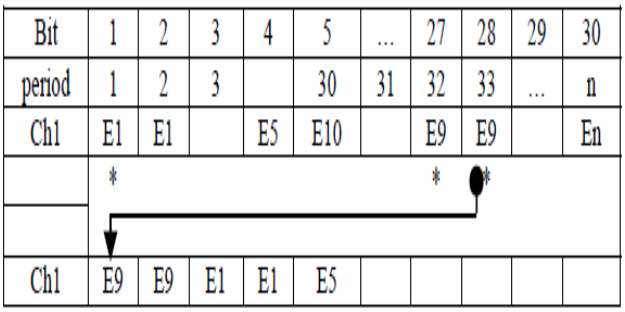

Fig. 6. Chromosome random and selection for Mutation (Nuntasen and Innet, 2007)

7. System Analysis and Design

The Genetic Algorithm Timetable System was designed using Microsoft Visual C++ integrated development environment. The data entered to the system with the use of configuration file by clicking on the file menu on the status bar and clicking on open configuration, then selecting the configuration from the application folder and clicking open. The timetable can be generated by clicking File + Start Solving in the status bar.

The fitness function computed by the generation of the timetable would be displayed at the top of the timetable and likewise the total generation that lead to the final timetable or optimal generation.

7.1 Constraint Data

We store sufficient details about the department before we test each type of the two constraints. Information relating to classrooms and lecturers has to be considered and maintained. Below are the way we incorporated each of these data types into the design of the system;

Three fields are structured and stored as a class, a predictable size for each class; a number indicating a lecturer and a number indicating related group code.

Lecturers, classes and classrooms were stored as arrays. Each of these is uniquely identified by its position as an element in the relevant array. The overall information on the three classes is referred to as the set of constraint data.

A single room time table is given by two arrays, each field of the array described the genetic information aspect in respect of number of breaches. A null allocation with zero value is given to a time where no class holds.

7.2 Breeding Timetable

A breeding process is used to randomly select a timetable from the population. Priority is not allowed for filtering timetables. Unit order performances are carried out on the parents by means of crossover to create a child timetable. This implies that each parent in the population has equal chance (probability) of providing a gene.

7.3 Mutating Timetables

The probability of any gene to be a part of the mutation is twice the mutation rate divided by one thousand approximately. If a mutation rate is equal to ten, then on the average twenty per one thousand genes have chances of being mutated. The entire problem can be scaled through this method. The proposed system design has no effect on increasing the number of constraints. It means that system can be extended to solve even larger or complex timetabling problem.

7.3 System Description

Upon approaching or starting the software for the first time, assuming no generation was made, or the configuration was not loaded to the system. The user would experience the following sequence of event:

Figure 7 shows a plain outline of the way the application looks like before the configuration file is loaded to the system.

Figure 8 and 9 shows the process of loading the configuration file and the application view after loading the configuration file to the system for timetable generation.

Figure 10 shows the process of starting the generation of timetable. While

Figure 11 and 12 shows the timetable generated in all the classrooms, it contains the different time available for the courses and number of days for receiving lectures, which is from Monday to Friday.

It is very important to note that the timetable is generated based on lecture rooms. For example, the left side of the frame shows the courses allocated to lecture room 1(CR1) and the time for the courses while the right side shows the courses allocated to lecture room 2 (CR2), and as in figure 12 for lecture room 3 (CR3) and CR4 respectively.

Figure 8 also shows the size of the lecture halls and indicate whether or not each one of them contains a laboratory that can be used for theoretical lab work, it is indicated by Y or N (where Y= Yes and N= No). Based on the algorithm configuration and design, a class that requires laboratory cannot be fixed in lecture room without one or without indication.

On the top left side of figure 11, a value is calculated and given for the fitness of any timetable generated. The fitness function ranges from 0 to 1. The closer to 1 the fitness function the better the timetable generated and the more it satisfies all parties involved. The fitness value generated in figure 11 is 1.000000. This means that the timetable generated will greatly satisfies all constraints.

Scrolling down the frame in figure 11 will show all the other lecture rooms, the courses and time allocated to them.

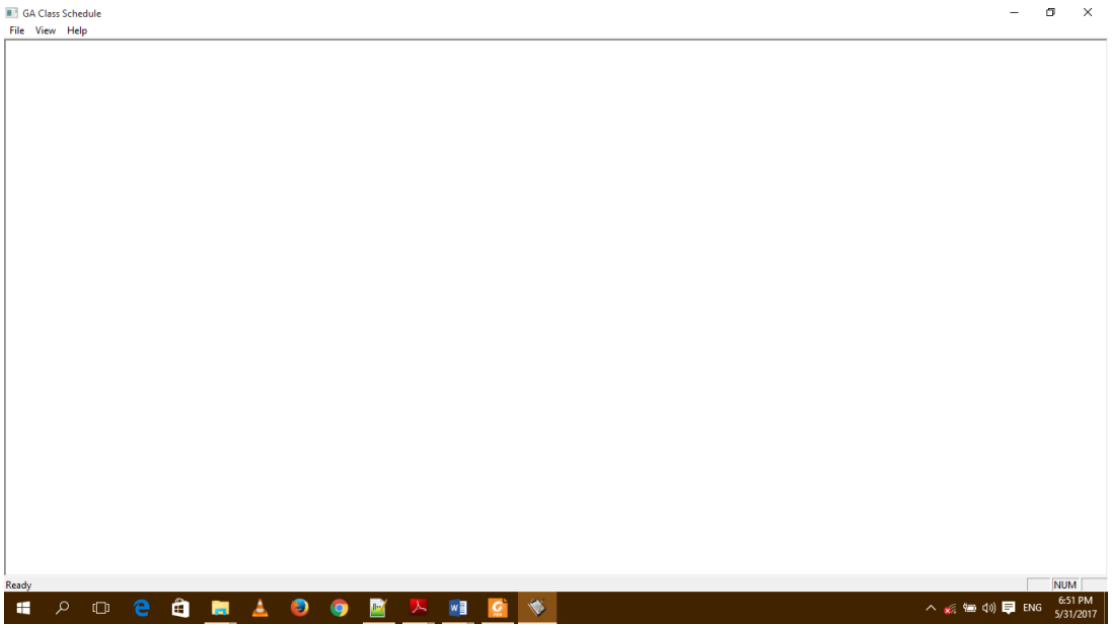

Fig.7. Application View after starting the Application Before loading the configuration file

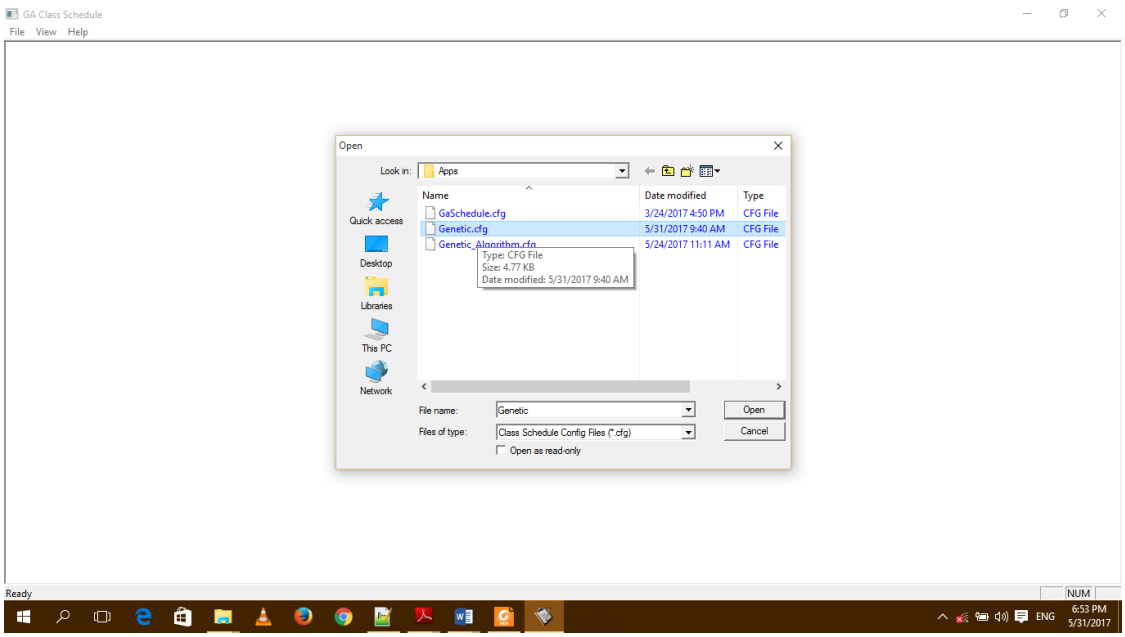

Fig. 8. Loading the configuration from file to the system

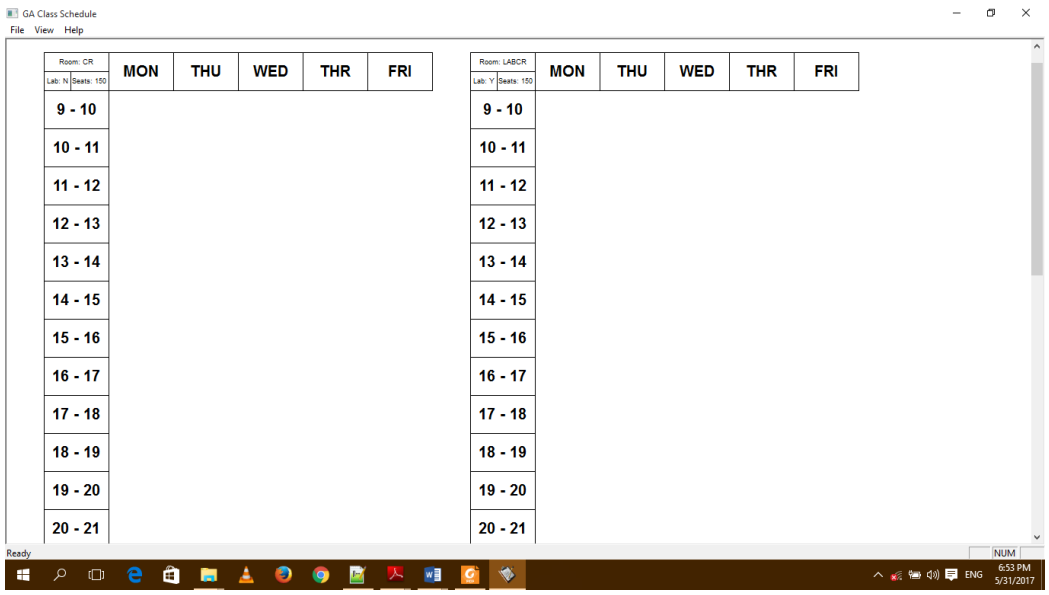

Fig. 9. Application View After loading the Configuration

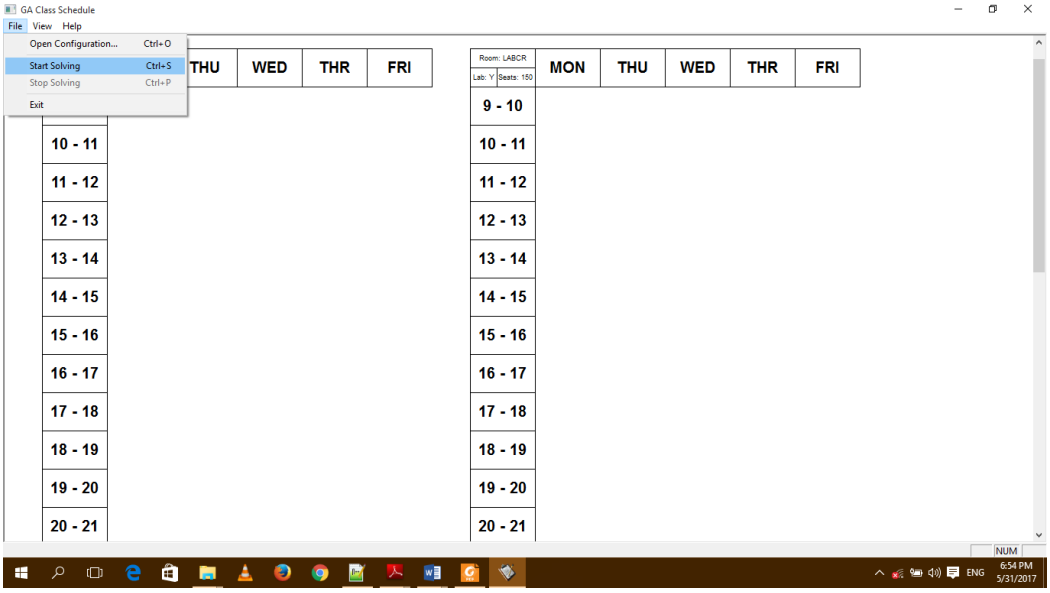

Fig.10. Starting the Generation of the Timetable

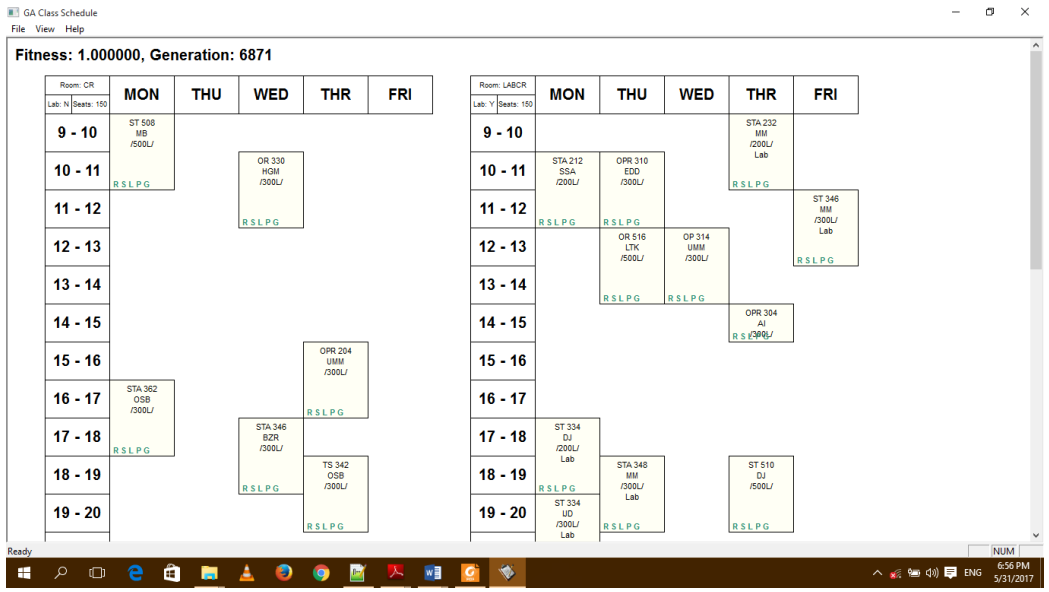

Fig. 11. View of the Timetable Generated for CR1 and CR2

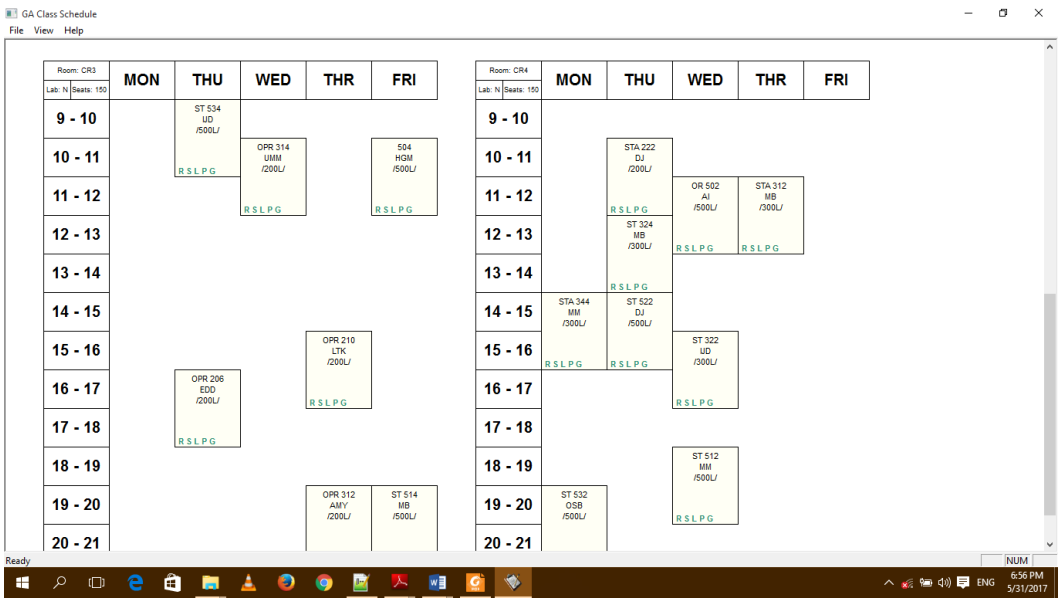

Fig. 12. View of the Timetable generated CR3 & CR4

7.4 Systems Requirement

Hardware /software requirement for the proposed system include:

- ✓ 700 MHZ (minimum) Pentium 3 or faster processor
- ✓ Microsoft Visual Studio
- ✓ Microsoft window XP (with server pack 2) or other later version of window operating system.

8. Conclusion

The problem studied in this work, belongs to Combinatorial Optimization known as NP hard problem where a set of constraints are satisfied. The efficiency of these algorithms are realized at the beginning of each semester yearly. Our proposed algorithms are a prototype for the preparation of automated timetabling. A Genetic Algorithm concept was used as a solution method in defining our problem. The automated system generator was constructed fundamentally with GA based on mathematical models of the problem. The feasible (possible) and infeasible (impossible) solutions are represented by matrixes with integer number as their elements.

REFERENCE

[1] Abdullah, A. B. (2008, May). Timetable Management System Using Genetic Algorithm. Technical Report, submitted at University of Malaya.

[2] Aydin, M. A. (2008). Solving University Course Timetabling Problem Using Genetic Algorithm. istanbul.

[3] Norberciak, M. (2006). Universal Method for Solving Timetabling Problems Based on Evolutionary Approach . Proceedings of the International Multiconference on Computer Science and Information Technology, (pp. pp. 149 – 157).

[4] Nuntasen, N., & Innet, S. (2007). Genetic Algorithm for Solving University Timetabling Problems. Proceedings of the 7th WSEAS International Conference on Simulation, Modeling, and Optimization. Beijing, China.

[5] Oprea, M. (2007). MAS_UP-UCT: A Multi-Agent System for University Course Timetable Scheduling. International Journal of Computers, Communications and Control, Vol. II , pp. 94-102 .

[6] Schaerf, A. (1999). A Survey of Automated Timetabling, Artificial Intelligence review. Retrieved October 28, 2016, from http://www.diegm.uniud.it/satt/papers/Scha99.pdf

[7] Wren, A. (1996.), Scheduling, Timetabling and Rostering. A special relationship. In Practice and Theory of Automated Timetabling.