British Journal of Mathematics & Computer Science
21(3): 1-19, 2017; Article no.BJMCS.30024
ISSN: 2231-0851

SCIENCEDOMAIN *international*
www.sciencedomain.org

Resolving Timetable Scheduling Problem Based on Bio-inspired Genetic Algorithm

Ebinum Wallace Ossai[1*] and B. Souley[2]

[1]Department of Computer Science Education, Federal College of Education (Technical), Gombe, Nigeria.
[2]Department of Mathematical Sciences, Abubakar Tafawa Balewa University, Bauchi, Nigeria.

Authors' contributions

This work was carried out in collaboration between both authors. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/BJMCS/2017/30024
Editor(s):
(1) Wei-Shih Du, Department of Mathematics, National Kaohsiung Normal University, Taiwan.
(2) Sheng Zhang, Department of Mathematics, Bohai University, Jinzhou, China.
(3) Paul Bracken, Department of Mathematics, The University of Texas-Pan American Edinburg, TX 78539, USA.
Reviewers:
(1) V. S. Jorapur, University of Mumbai, India.
(2) J. Jayapriya, National Institute of Technology, Tiruchirapalli, India.
Complete Peer review History: http://www.sciencedomain.org/review-history/18420

Original Research Article

Received: 12th October 2016
Accepted: 8th December 2016
Published: 30th March 2017

Abstract

Timetabling is the task of assigning sets of events to periods of time, taking into account resource-constraints and preferences among assignments. This involves combinatorial optimization, time-based planning, in order to realize a highly constrained problems that addresses a multi-dimensional complexities. This paper investigated the use of activity matrix to reduce the complexity of timetabling and applying genetic algorithm to resolving Colleges of Education Timetabling Problem. In this study, Course, Rooms and Time slots are represented in the form of a multidimensional array. On this is applied certain genetic operators such as crossover operator in a manner that does not violate the hard constraints and then a local is performed to obtain an optimal solution. The fittest solution (optimum timetable) is then displayed as the final timetable. Based on the evaluation carried out on the completed system it was revealed that the completed system worked effectively well.

Keywords: Genetic algorithm; time table; subject; data.

**Corresponding author: E-mail: ebinum2@gmail.com, ebinum2@yahoo.com;*

1 Introduction

Resource allocation can be defined as the distribution of finite resources among competing entities with a goal to achieving optimal system functionality (Mohd, & Masri [1]). The efficient allocation of resources is an issue that impacts on all institutions, from small to large multinational companies. Some of the areas in which resource allocation problems exists in different forms include: health (Nurse and Doctors timetabling), transportation (flight and train timetabling), sports (timetabling of matches between pairs of teams), entertainment (scheduling network television programmes), industries (production scheduling), and education (course and exam timetabling) (Eugene [2]). The names Time-tabling problems or scheduling problems are sometimes used in referring to subclasses of resource allocation problems. However, the work described in this paper is concerned with solving Timetabling Scheduling Problem using Genetic Algorithm. Genetic algorithm is one of the powerful and widely used evolutionary computation methodologies for solving an optimization and search problem. Genetic algorithm does not create a problem specific result; rather it imitates the development of new and better populations among different species during evolution. Genetic algorithm proposes a general formulation for every problem by searching the solution space with a defined relevance calculation. It uses randomness and statistics to generate different solutions for different executions of the algorithm so that more than one trial can be made in order to get the best result. Unlike most standard heuristic algorithms, they use information of a population of individuals (solutions) when they conduct their search for better solutions and not only information from a single individual.

A study by Landa [3] reveals that most educational institutions always resort to manual generation of their timetables which even at the optimal stage is still not free from clashes. These few clashes are generally left for the lecturers taking the clashing courses to work out the logistics of the courses so as to avoid the clash. Nashwan and Talal [4] stated that most corrections and repairs are done in a manual timetable system after receiving feedback from staff and students. This is a typical situation in the School of Science Federal College of Education (Tech), Gombe, where timetable design and construction is done manually and priority is given to the non-conflict of different study groups in terms of courses, time slots and venues.

Naturally, the desire to avoid changes to school timetable is high in any academic institution, but changes seems to have become an unfortunate necessity in most schools due to the changes in the number of student groups and enrolment. This process of making changes or creating new timetable to satisfy a present reality in an institution is very tedious and laborious and often times could take a good number of days or even weeks to complete. This generally affects the startup of academic activities at the beginning of every semester. Despite the considerable time spent in the design and construction of manual timetable, it is still not free from errors due to clashes and course omission. Therefore, there is a need to provide an automated timetabling system that will resolve these challenging issues associated with manual timetabling.

2 Related Literature

2.1 Timetabling problems

A timetabling problem can be defined as the problem of assigning a number of events into a limited number of time periods. Schaerf [5] describes timetabling problems as the allocation of resources for factors under predefined constraints so that it maximizes the possibility of allocation or minimizes the violation of constraint. In the view of Luis and Oliveira [6], Timetabling problem (TTP) can be defined as the fixing in time and space, a sequence of meetings between teachers and students, in a prefixed period of time, satisfying a set of constraints of several different kinds. The constraints refer to here may include both hard constraints that must be respected at all cost and soft constraints which are used to evaluate the quality of a timetable.

During the last forty years, many papers related to automated timetabling have been published in conference proceeding and journals; some of these studies include that of Fang [7] who in his doctorate degree thesis,

investigated the use of genetic algorithm to solve a group of timetabling problems. He presented a framework for the utilization of genetic algorithms in solving timetabling problems in the context of learning institutions. The framework identified the following important points, which give considerable flexibility: a declaration of the specific constraints of the problem and use of a function for evaluation of the solutions, advising on the use of genetic algorithm since it is independent of the problem for its resolution. Grobner and Wilke [8] proposed a general purpose language which can be used to describe the basic structure of timetabling problems and its constraints. Luis and Oliveira [6] proposed another language known as Unilang. Unilang is intended to be a standard input language to any timetabling system. It enables a clear and natural representation of data, constraints, quality measures and solutions for different timetabling problems such as school timetabling, university timetabling and examination timetabling.

Several other works on timetabling using genetic algorithm have also been carried out as presented in (Manar and Fatima [9]). The reviewed literatures include the work of Yang and Jat [10] who presented a Guided Search Genetic Algorithm (GSGA) to solve the University Course Timetabling Problem (UCTP). The problem addressed is a university course timetable, where an event (course and lecture) is assigned to a timeslot. Also, a number of rooms with specific features are assigned to these events, and there are also a number of students that attend these events. The assignment of events to timeslots, rooms, and students is performed in such a way as to satisfy a number of hard and soft constraints. The proposed technique is a guided search strategy and Local Search (LS), which are integrated into a Steady State Genetic Algorithm (SSGA).

Sapru et al. [11] addressed the problem of educational timetabling by assigning each lecture for a particular student group to a specific room during a given timeslot. This is done while taking into consideration restrictions on time and the availability of faculty and other resources. The encoding scheme for the proposed GA is based on representing each individual in the population as a timetable for a 5-day per week, with 8 timeslots per day. Each timeslot holds information about subjects, faculty and rooms that are assigned to each slot. All this information is encoded in binary.

AlSharafat and M. S AlSharafat [12] developed a timetabling system to allocate first and second semester exams for a number of courses at Al-Bayt University. Three different forms of GAs were tried and compared in the work. The first is a Steady State Genetic Algorithm (SSGA) with overlapping populations; the second is an Enhanced Steady Genetic Algorithm (ESSGA), in which the crossover and mutation operators were enhanced using Fuzzy Logic (FL), and finally a Simple Genetic Algorithm (SGA) with non-overlapping populations. The results obtained from these researches showed a near optima solution, but the academic set up under which it was carried out is quite different from what is obtainable under this research. In the F.C.E.T Gombe, where most courses are double major, the different departments are completely interdependent on each another. This condition makes timetable construction in the F.C.E.T. Gombe more complicated when compared to course timetabling in the universities where different departments are relatively independent from each other.

Raghavjee and Pillay [13] proposed a Genetic algorithm to solve the school timetable problem for a South African primary and high school schedules. The overall process is a two phase approach. The first phase is a GA that focuses on producing feasible timetables, while the second GA phase improves the quality of timetables found during the first phase.

Abdullah et al. [14] investigated a genetic algorithm combined with a sequential local search for the curriculum based course timetabling problem. In this problem a timeslot and a room is assigned to all lectures of each course, while observing a set of hard and soft constraints.

Most of the work on educational timetabling tends to focus mainly on university timetabling problem which scenario is quite different from what is obtainable in the College of Education in Nigeria. Course groups in the Colleges of Education are highly dependent on one another. This situation leads to high level of complexities in the timetable system. The study presented in this paper uses activity matrix to identify the

complexities in the system in order to facilitate conflict checking before applying the principle of genetic algorithm to construct a solution to the problem.

2.2 Basic concept and the working principles of genetic algorithms

A genetic algorithm is a search algorithm based on natural selection and the mechanisms of population genetics (Holland [15]). A genetic algorithm (GAs) is different from other traditional approaches to solving optimization problems. This simple idea of GA search got its inspiration from the Darwin's theory of evolution commonly known as the survival of the fittest which is the biological processes of survival and adaptation (Goldberg [16]).

The process of searching among a collection of candidate solutions for a desired solution is generally referred to as searching in a "search space", which refers to the set of all possible solutions or some collection of candidate solutions to a problem. In solving problems, some solutions will be the best among others. The space of all feasible solutions among which the desired solution resides is called search space (Chakraborty [17]). In addition each point in the search space represents one possible solution. Each possible solution can be marked by its value (or fitness) for the problem. The GA therefore looks for the best solution among a number of possible solutions represented by one point in the search space.

Looking for a solution is equivalent to looking for some set of extreme values (Minimum or Maximum) within a pool of values called the search space. The process of finding an optimum solution in GA begins by using some set of values (candidate chromosomes) called the initial population. These individual solutions are then paired based on their fitness values to generate other possible set of values to form a new set of chromosomes for the next evolution. This is inspired by the principle that the new population that is generated has a high possibility to be better than the old population. Solutions are chosen according to their fitness value to form new solutions. This process is repeated until certain conditions are satisfied. Each iteration of this process is called a *generation*. An average of 50 to 500 runs or more are required for a standard (Gen and Cheng [18]). The entire set of generations is called a *run*. At the end of a run there are often one or more highly fit chromosomes in the population.

2.3 Methods of selection in genetic algorithm

There are many different techniques which a genetic algorithm can use to select the individuals to be copied over into the next generation. Some of the most common methods are mutually exclusive, but others can be and often are used in combination. In Alade et al. [19], the following are the methods of GA selection.

2.3.1 Elitist selection

Chromosomes with the best fitness values are copied into the next generation before the application of any genetic operator. This helps the GA to retain some of the best chromosomes at each generation, hence improving the performance of the GA.

2.3.2 Fitness-proportionate selection

Chromosome with the greatest fitness value has the highest probability of being selected than members with lesser fitness values.

2.3.3 Roulette-wheel selection

Roulette-wheel selection is a form of selection whereby the chance of any member being selected is proportionate to its fitness value being greater than or less than other competing members of the pool. This is generally represented in the form of a game of roulette where every member of the pool is assigned a space on the Roulette-wheel according to their fitness value. Members with greater fitness value stand a better chance of being selected each time the wheel is rotated.

2.3.4 Scaling selection

In scaling method certain objective functional values are assigned to each candidate chromosomes to scale their fitness values. These scaled values are then used to determine the probability of survival for each candidate chromosomes in the population. The purpose is to maintain a uniform selection pressure throughout the GA run as the method prevents a rapid domination by only highly fitted chromosomes.

2.3.5 Tournament selection

Some chromosomes are chosen at random out of the larger population and their fitness values are compared with each other. The fittest individual is then chosen as parent to reproduce in the next generation. The selection method can be improved by adjusting the size of the tournament.

2.3.6 Rank selection

Each member of the population is first ranked by assigning certain numerical values based on the members' fitness value. Thereafter, selection for the next generation is based on this ranking rather than depending purely on their fitness value. This method gives every member of the population a chance to be selected, hence preventing the dominance of only highly fitted chromosomes which help to preserve diversity.

2.3.7 Generational selection

An entire new population of chromosomes is formed from the offspring of the chromosomes that were selected for breading in each generation. In this method an entire population pool is replaced with a new population for each generation.

2.3.8 Steady-state selection

In each generation, some highly fitted chromosomes are selected and are copied back into the existing pool of values to replace some low fitted chromosomes to form a new set of chromosomes for the next generation. In this method, only a few individuals are replaced at any given time leaving the rest population to survive to the next generation.

2.3.9 Hierarchical selection

Candidate chromosomes are subjected to many rounds of selection processes at each generation. Generally, evaluation at the lower level of the tree are faster and less cumbersome when compared to the rigorous evaluation processes that surviving candidate chromosomes are subjected to at the higher level stage of the tree. This method enhances an efficient usage of computational time as many less promising candidate chromosomes had been weeded off at the early stage of evaluation.

3 Overview of the Existing System

FCE (T), Gombe is a fast growing institution; with six different schools offering divers' courses in different vocations. The School of Science Education is the largest of all the schools with a total number of six different departments and nine different course combinations. Currently a total of 125 courses are being offered in all the departments excluding practical courses. The timetable starts from 8:00 am and ends at 6:00 pm, which repeats in the same way for five working days of the week (Monday to Friday) making a total of 50 time slots. The traditional method of timetable construction in the School of Science involves the assignment of different courses to different cells in a table. The Timetable Officer carefully does this using pencil and paper to avoid clashes in terms of courses and venues. This process is sometimes facilitated by the use of Microsoft Excel to reduce paper works. Despite the considerable man hours spent using this traditional method of timetable construction, the timetable is still not free from clashes as some lecturers and course groups are often double booked. These clashes associated with the timetabling system are often left for the lecturers and student groups to sort out.

3.1 Analysis of the existing system

The current timetable used by the NCE program is prepared manually. This traditional method of timetable preparation ensures that there are no clashes for the very timetable produced – which is not always the case. Timetables are subject to changes, considering the large number of departments in the school of sciences, and the inter-dependency nature of the departmental courses on each other, extra care needs to be taken to avoid clashes. This method of timetable preparation as it is being done in the school of sciences is associated with a lot of problems.

3.2 Limitations of the existing system

i. It is time consuming as the process takes too much time of the timetable officer.
ii. Repeated time allocations may be made for a particular course thereby leading to data redundancy
iii. It generates a lot of paper work.
iv. Despite the time spent, it is still not free from clashes
v. A minor change to course allocation renders the timetable infeasible.
vi. Manual timetabling is not flexible, as a little adjustment can lead to multiple clashes.
vii. Individual lecturers cannot easily have their personal time tables.

4 Methodology

4.1 Data collection

To achieve the goal of conflict free and a good quality time table, different component parts of the school was investigated as follows:

i. From the department point of view: - what are the different courses taken by each department and the lecturers that take them. This involves getting data about the different courses taken by the department and the different student groups that take such courses.
ii. From the student group's point of view: - what are the different courses and combinations that are offered by each student groups. This implies getting data about courses taken by each students group and the departments that offer the courses.
iii. Finally the number of rooms available for lectures and their sizes. This involves getting data about the different rooms allocated to each department, their sizes in order to determine the courses that can be assigned to each room depending on the number of students groups that offer the course.

To start the construction of a school timetable, every necessary information or data that is required for its feasibility must be made available. In general, to solve a course timetabling problem the following set of data are required: the number of rooms available and their sizes, Courses and their credit unit, lecturers and the courses they teach, student groups and their course combination, which serve as an input data to the system. All these data can be obtained from the academic institution concerned.

4.2 Proposed system modeling

The idea behind genetic algorithms is, given a pool of parents (genes), select any two parents and initiate a crossover between the parents to generate two children which in turn can undergo mutation. These two children are then evaluated through an objective function to determine their fitness. This cycle continues until a minimum or maximum objective function is reached. Individual chromosomes with high fitness value have a higher probability of survival in the process of evolution. This implies that chromosomes with high fitness values will be more in number in the next pool of parents that will be used for subsequent generation while chromosomes with lower fitness values will gradually reduce in number until they become extinct from the population as the evolution continues. The implication of this is that an optimum solution (a highly

fitted chromosome) will be found as the population evolved. The fitness value of an individual chromosome is a determination of how far or close a chromosome is to a solution (Alade et al. [19]). The number of constraints that an individual chromosome satisfies determines the fitness value (goodness) of that chromosome. This is represented by a value which is the sum total of all violations with various scaling factor for each constraints, with hard constraints having higher values to ensure they influence selection more than soft constraint violations. In this research the formula: $fitness = \frac{1}{1+x} - \frac{\partial}{1+v}$ was used as a fitness function. Where x is the sum of all violated hard constraints and v is the sum of all violated soft constraint and ∂ is kept as zero until there are no hard constraints to consider again. This implies that soft constraints are not considered at the expense of violating even a single hard constraint. Fig. 1 shows the standard genetic algorithm process for course timetabling.

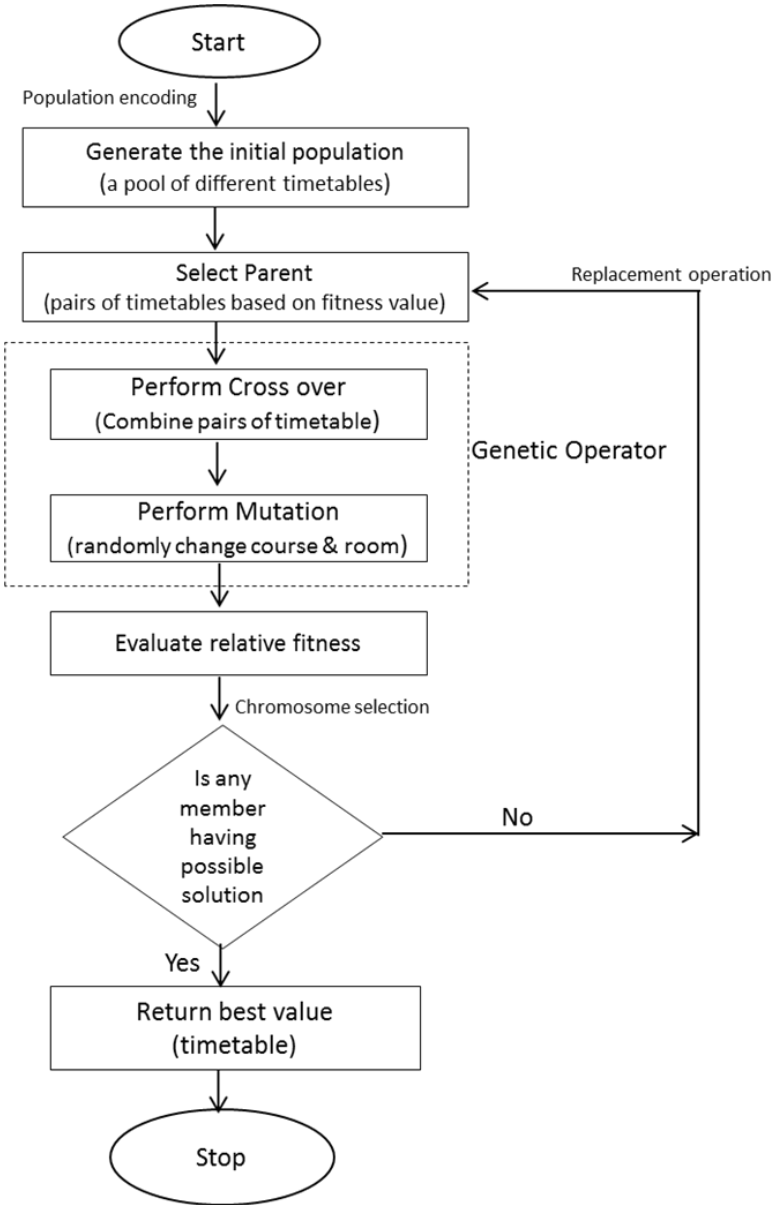


Fig. 1. Standard genetic algorithm flowchart

4.2.1 Activity matrix

The introduction of activity matrix is to serve as a guide in the process of space allocation to teaching events during timetable creation. This frame work is aimed at resolving a number of conflicting demands prior to time table creation. These conflicts often arise due to the peculiar nature of course combinations in most Colleges of Education in Nigeria, where most courses of studies are double major. This peculiarity made most of the departments in a school to be interdependent of one another. For the timetable officer as well as the quality of the timetable as it affects every staff, it is necessary to be able to pin point potential difficulties before timetable implementation. These potential conflicts can easily be resolved by the means of activity matrix.

Table 1. Activity matrix for course groups

Courses	MTH	CSC	PHY	CHE	ISC	BIO	EDU
MTH	1	x	x	x	x	1	x
CSC	x	1	x	1	1	1	x
PHY	x	x	1	x	x	x	x
CHE	x	1	x	1	x	x	x
ISC	x	1	x	x	1	x	x
BIO	x	1	x	x	x	1	x
EDU	x	x	x	x	x	x	1

The activity matrix in Table 1 shows all possible courses that can go together in the school of science of any college of education that offers the different course combinations as shown in the under listed courses. All 1s indicate courses that can be taken concurrently without clash due to student group course combination. On the other hand all X indicates courses that can never be scheduled together because it will result in a clash. Putting this condition as a constraint during program implementation will make a timetable more flexible and free from clashes.

1. Mathematics Computer
2. Physics Computer
3. Biology Integrated Science
4. Physics Integrated Science
5. Physics Mathematics
6. Chemistry Mathematics
7. Chemistry Integrated Science
8. Biology Chemistry
9. Mathematics Integrated Science

4.2.2 Chromosome representation

Before a GA can be used to solve any particular problem, a method must be devised to encode potential solution to that problem in a form that a computer can process. Since chromosome represent a candidate solution to the timetabling problem, it implies that chromosome representation should not be naïve but should be represented in a manner that the application of any genetic operator such as crossover and mutation operator will not render the timetable infeasible.

In this case study there are five (5) working days in the school, that is from Monday to Friday and ten (10) lecture hours in each day of the week (from 08:00 - 18:00). This scenario will give a total of $5 \times 10 = 50$ timeslots in a week as can be seen in Table 2.

In order to create a timetable for each room in a week, a multidimensional array called room timetable that adds up all the 50 timeslots available for each room end to end was designed as shown in Fig. 2. The first 50 elements of the multidimensional array (from 0 to 50) represents the 50 timeslots available for room #1 in a

week, the next 50 elements from 51 to 100 represents the 50 time slots available for room #2 in a week, this pattern continues until the last available room. Hence, the size of this array will be 50 x numbers of rooms available. Also another one dimensional array was designed to contain the list of all the courses that is taken in a semester. This array will be called Course List. The elements of this array also contain other information such the lecturer taken the course and number of hours allocated to the course (credit unit).

Table 2. Available time slots

Days / Hours	Monday	Tuesday	Wednesday	Thursday	Friday
8:00 – 9:00	1	11	21	31	41
9:00 – 10:00	2	12	22	32	42
10:00 – 11:00	3	13	23	33	43
11:00 – 12:00	4	14	24	34	44
12:00 – 13:00	5	15	25	35	45
13:00 – 14:00	6	16	26	36	46
14:00 – 15:00	7	17	27	37	47
15:00 – 16:00	8	18	28	38	48
16:00 – 17:00	9	19	29	39	49
17:00 – 18:00	10	20	30	40	50

Every element in the multidimensional array corresponds to each gene in a chromosome which also correspond to certain elements of the of the Course List array. For example each gene in a chromosome (each array element), say chrm [i], holds the index value of the room timetable element where the i^{th} course was scheduled to. Taking an instance from Fig. 2, the 2nd element of the chromosome chrm [ii] is 50, which correspond to Course #2 in the course list array, and was scheduled to the 50^{th} element in the rooms' timetable which is room #1. As can be seen from Table 2 also, the 50^{th} element corresponds to Friday starting at 17:00. The implication of this is that, Course #2 is taken on Friday at 17:00 in Room #1. Since each chromosome elements represents the starting time for each course, it can be deduced from the course list array the particular lecturer that takes a course and the numbers of hours allocated to the course. The Fig. 2 below depicts the chromosome representation of the School of Science timetable.

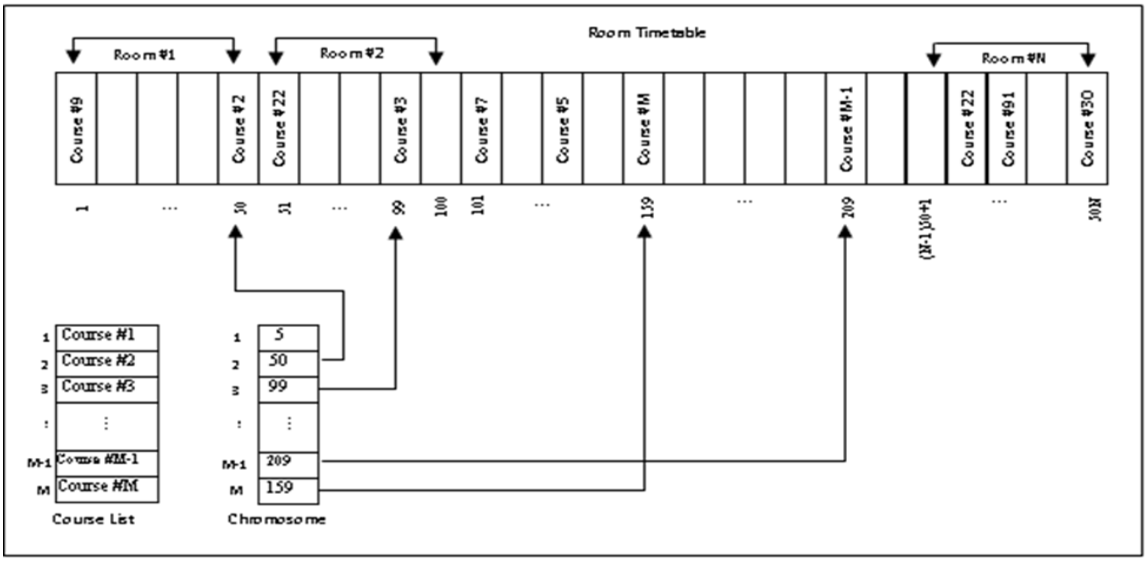


Fig. 2. Chromosome representation

4.2.3 Use case representation of the system

Fig. 3 below depicts a use case description of the program, showing the interaction within the system.

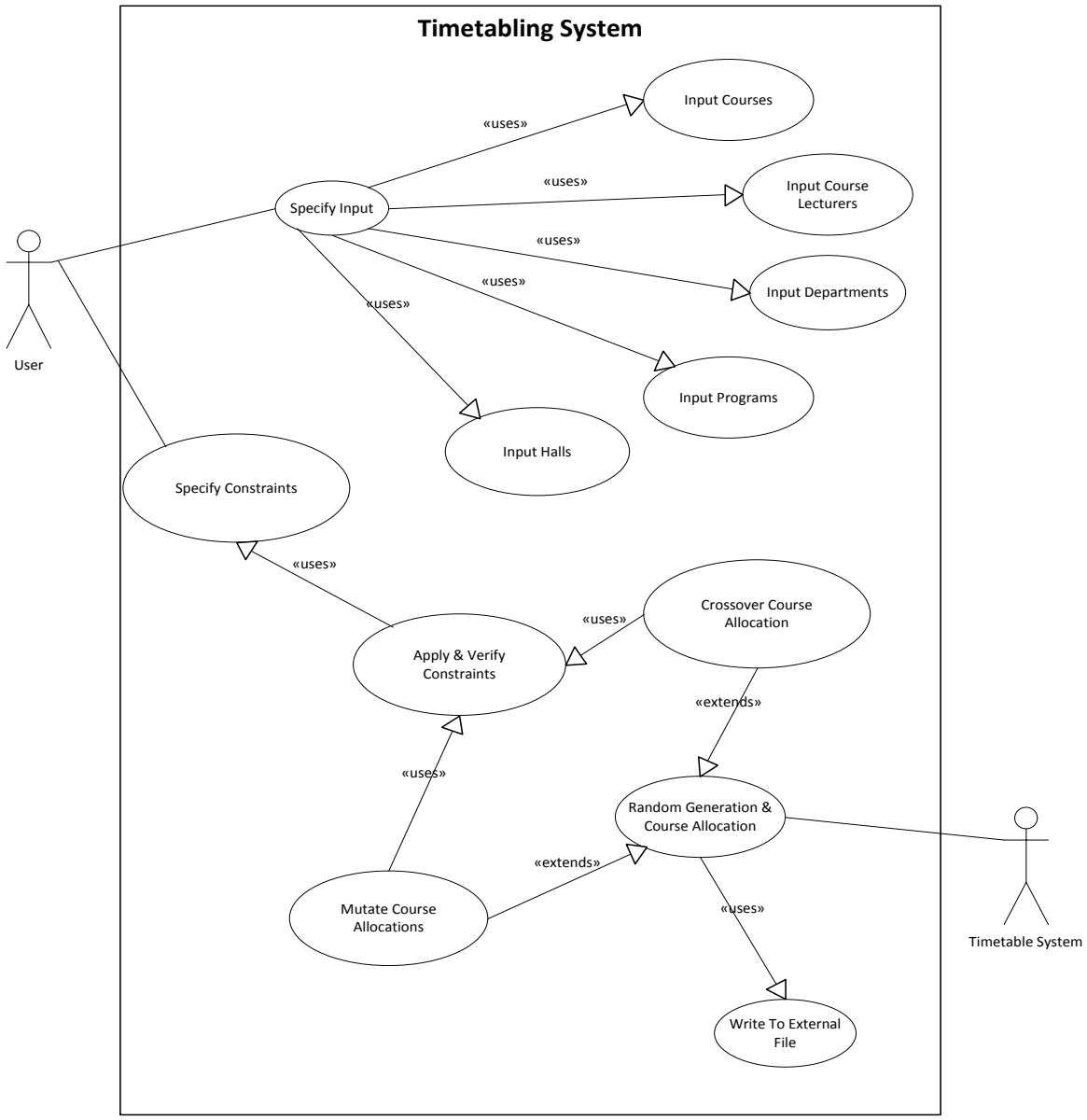


Fig. 3. Use case diagram of the system

4.3 System design

4.3.1 Design of interfaces

The purpose of the interface design is to provide the user with a comfortable and convenient means of interacting with the timetable system. The flexibility and convenience of the interface is a major factor to determining the friendliness of the system. An outline of the system interface is described below:

4.3.2 Output interface design

The output design shows the format of the value(s) that is expected to be generated by the system. The output of system determines its usability and reliability. The School of Science Timetable (SST) system is design to output the following as can be seen in Fig. 4:

- List of all courses
- Display the master timetable
- Display the individual lecturer timetable
- Display venue timetable

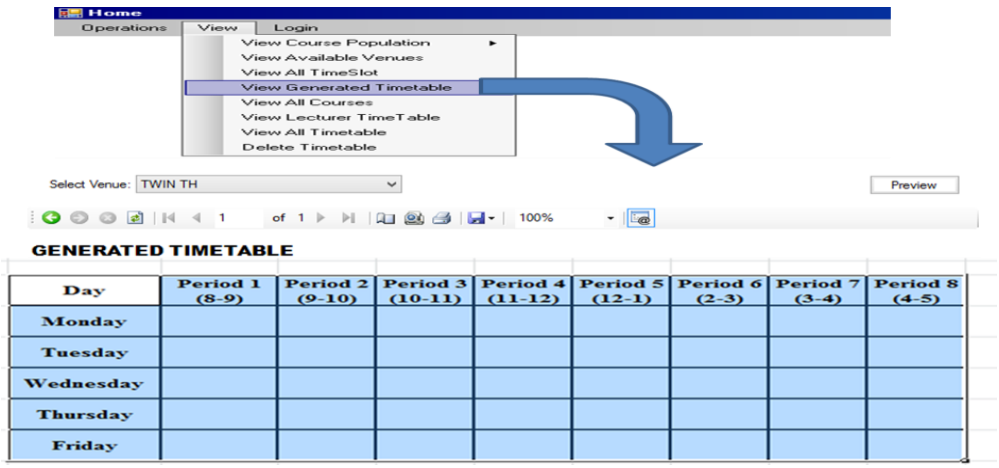


Fig. 4. Output options/display format

4.3.3 Input interface design

The input design displaces information in windows requesting the user to specify the basic information required for the construction of the timetable. The operation menu in Fig. 5 shows the different data that can be supplied into the system.

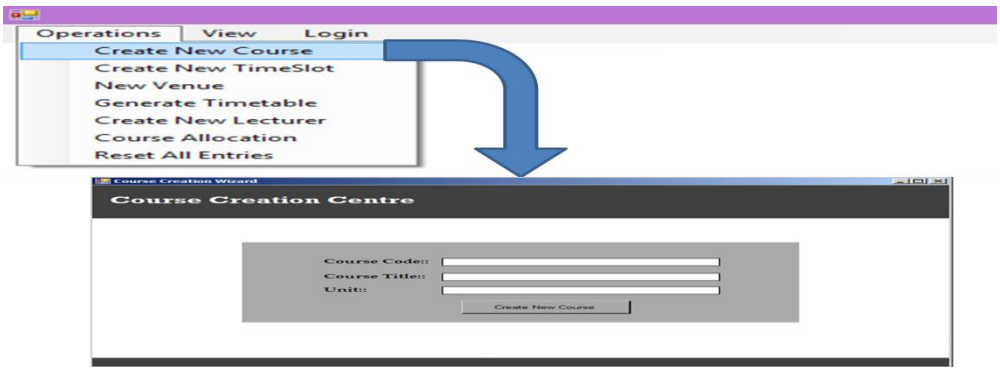


Fig. 5. Input options/display format

4.3.4 Program design

The first stage here is to create the classes for the system, the second was to experiment with the genetic algorithm concept which would then progress to stage three to improve it and finally to stage four where the aim was to create a standalone program.

One of the key requirements in designing the system was to make it extremely modular so that any aspect of the genetic algorithm could easily be changed without drastically affecting other areas of the system. Whilst there is plenty write up about the various methodologies that can be used to implement a genetic algorithm, there are no hard rules about which are best suited to a particular situation. Fig. 6 below depicts the class diagram description of the program development.

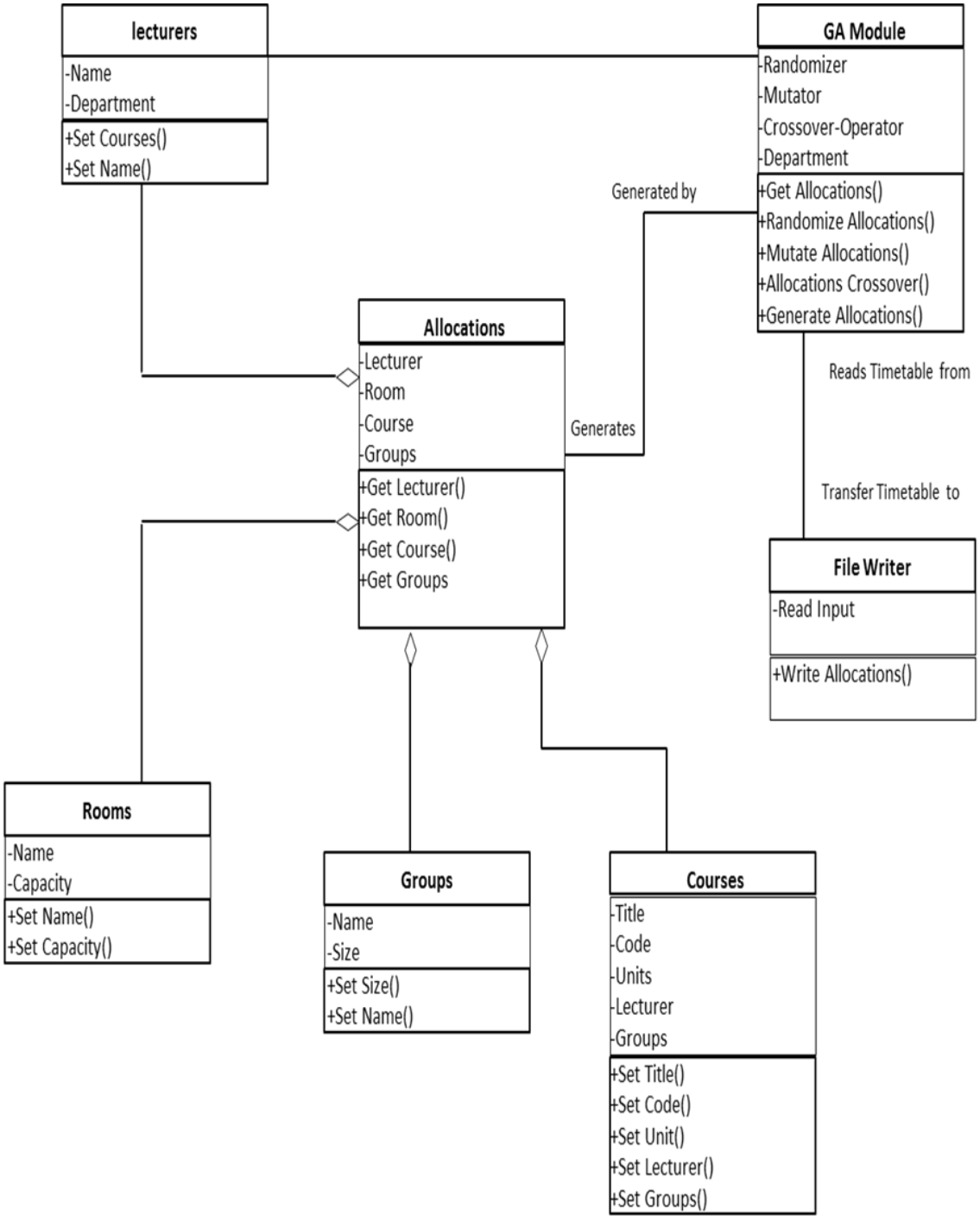
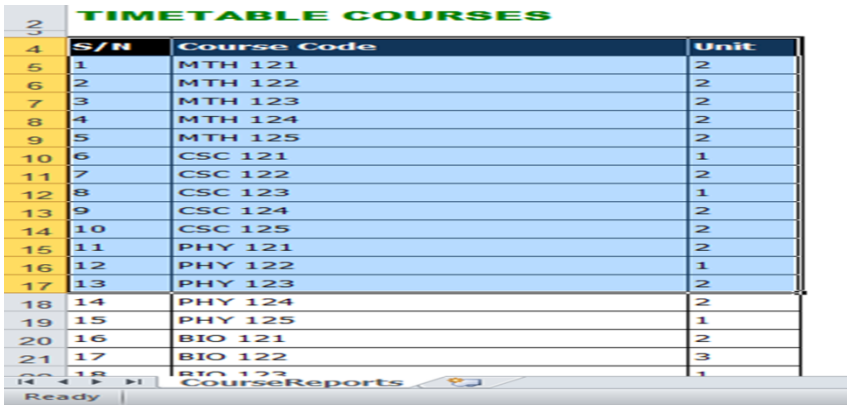


Fig. 6. Class diagram of the system

5 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points. The Tables 3, 4 and 5 below displays some real life data as obtained from the School of Science which was used to test the different functionalities of the entire system while Figs. 7, 8 and 9 give some sample output of some test cases carried out on the complete system.

Table 3. Course table



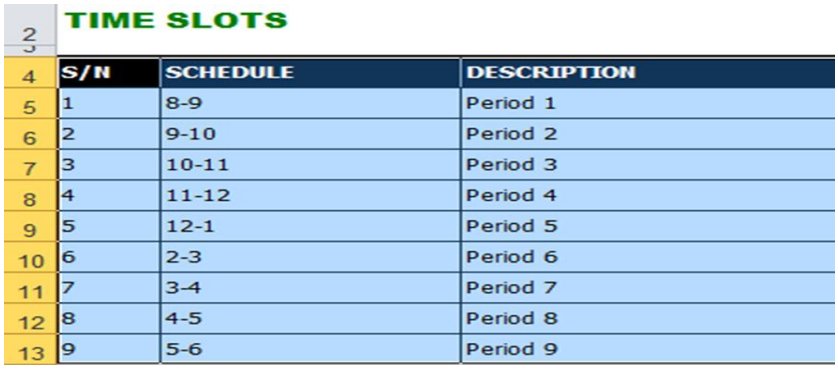
TIMETABLE COURSES

S/N	Course Code	Unit
1	MTH 121	2
2	MTH 122	2
3	MTH 123	2
4	MTH 124	2
5	MTH 125	2
6	CSC 121	1
7	CSC 122	2
8	CSC 123	1
9	CSC 124	2
10	CSC 125	2
11	PHY 121	2
12	PHY 122	1
13	PHY 123	2
14	PHY 124	2
15	PHY 125	1
16	BIO 121	2
17	BIO 122	3
18	BIO 123	1

CourseReports

Ready

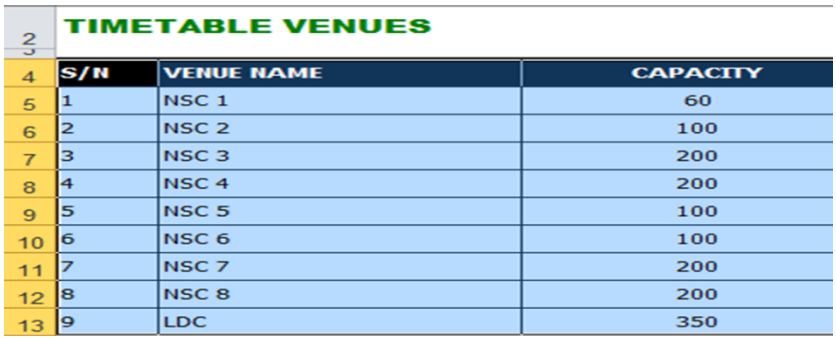
Table 4. Lecture time



TIME SLOTS

S/N	SCHEDULE	DESCRIPTION
1	8-9	Period 1
2	9-10	Period 2
3	10-11	Period 3
4	11-12	Period 4
5	12-1	Period 5
6	2-3	Period 6
7	3-4	Period 7
8	4-5	Period 8
9	5-6	Period 9

Table 5. Lecture venues



TIMETABLE VENUES

S/N	VENUE NAME	CAPACITY
1	NSC 1	60
2	NSC 2	100
3	NSC 3	200
4	NSC 4	200
5	NSC 5	100
6	NSC 6	100
7	NSC 7	200
8	NSC 8	200
9	LDC	350

5.1 Test case 1: Testing for number of clashes

The average number of related clash with regards to population increase was recorded, and the results were as displayed in the Figs. **7** and 8. The population size should therefore be large because a bigger population size leads to a better representation and a wider search. The downside of increasing population is that the calculation time increases proportionally.

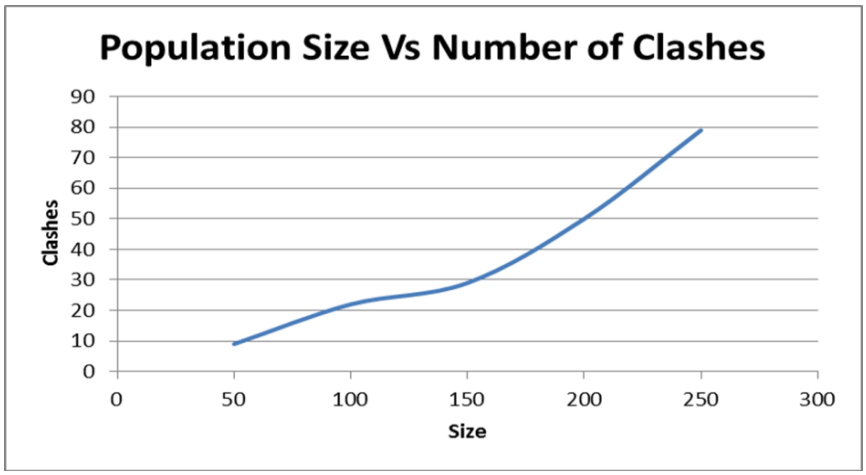


Fig. 7. Population size vs. number of clashes

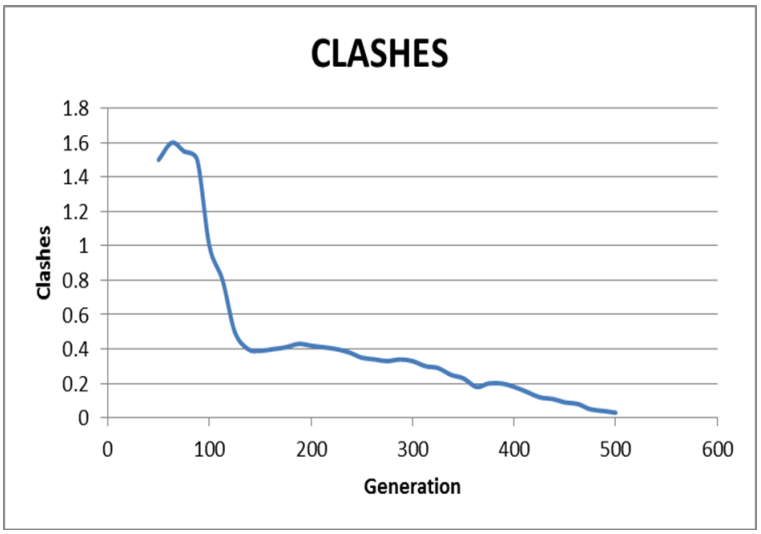


Fig. 8. Number clashes vs. number of generation

5.2 Test case 2: Testing for system convergence

This test was aimed at determining the convergence of the genetic algorithm with number of generation. The average number of fitness value with regards to increasing the number of generations was recorded, and the result were as displayed in the Fig. 9. From the graph of Fig. 9 the fitness value of the genetic algorithm tends to increase very fast at the early stage of generation, but this gradually eases off to nearly the same fitness value as the number of generation increases. However the fitness value never reached the value of one as it was very difficult to satisfy every soft constraint that is specified in the program.

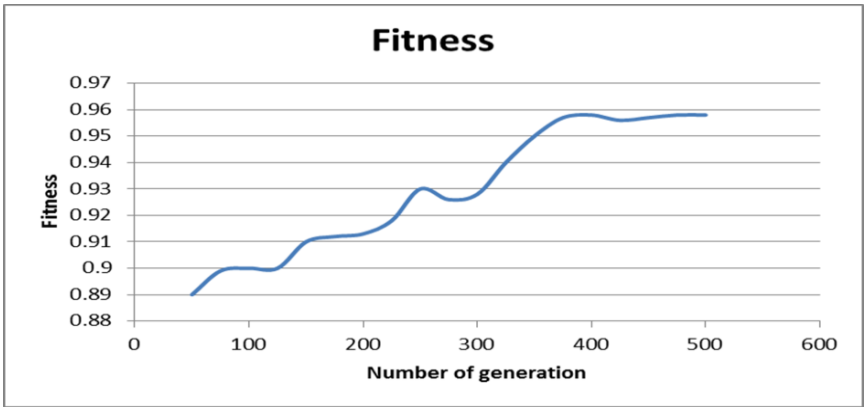


Fig. 9. Fitness vs. number of generation

5.3 Test case 3: Output of venue timetable

Fig. 10 displays the tests result for the lecture venue output. This test is to ensure that the different venue lecture timetable can be printed.

TIMETABLE REPORT

Select Venue: TWIN TH Preview

1 of 1 100%

GENERATED TIMETABLE

DAY	Period 1 (8-9)	Period 2 (9-10)	Period 3 (10-11)	Period 4 (11-12)	Period 5 (12-1)	Period 6 (2-3)	Period 7 (3-4)	Period 8 (4-5)
Monday	PES 121(TWIN TH)	PES 122(TWIN TH)	PES 123(TWIN TH)	PES 124(TWIN TH)	PES 126(TWIN TH)	PES 128(TWIN TH)	PES 129(TWIN TH)	EDU 121(TWIN TH)
Tuesday	PES 327(TWIN TH)	PES 328(TWIN TH)	PES 329(TWIN TH)	EDU 321(TWIN TH)	EDU 322(TWIN TH)	EDU 323(TWIN TH)	EDU 324(TWIN TH)	EDU 326(TWIN TH)
Wednesday	PES 121(TWIN TH)	PES 122(TWIN TH)	PES 123(TWIN TH)			PES 128(TWIN TH)		EDU 121(TWIN TH)
Thursday	PES 327(TWIN TH)		PES 329(TWIN TH)			EDU 323(TWIN TH)		EDU 326(TWIN TH)

Fig. 10. Test case 1 timetable for NSC 1

5.4 Test case 4: Output of individual lecturer timetable

This test was carried out to ensure that the individual lecturer timetable printed actually reflects the course allocated to them and that it is on the general time table.

Select Lecturer: HAMZA SOYE Preview

1 of 1 100%

GENERATED TIMETABLE

DAY	Period 2 (9-10)	Period 5 (12-1)
Monday	EDU 123(PESRM1)	GSE 122(PESRM1)
Wednesday	EDU 123(PESRM1)	

Fig. 11. Timetable for Hamza Soye

5.5 Test case 5: Error messages for a duplicate value

This test is carried out to ensure that no particular course or venue is entered more than once into the system. Duplicating a value can affect chromosome representation and the entire timetable.

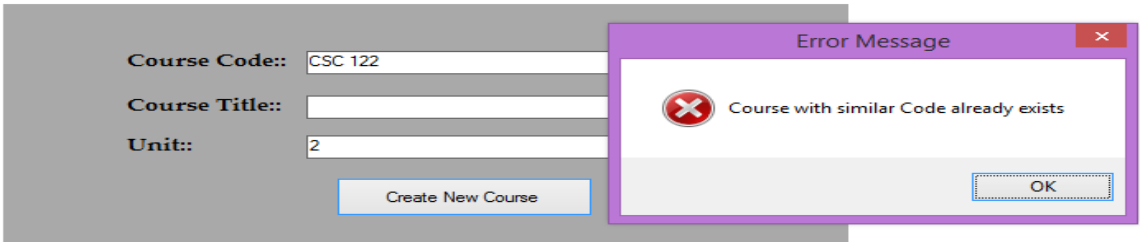


Fig. 12. Error message for duplicating a course

6 Discussion

After receiving initial data, such as lecturers and their courses, course groups and their courses, lecture venues and their capacities, the system was subjected to various tests. A system run of between 50 to 500 generations are set in the experiment, without considering probability of selection, crossover and mutation. The maximum best fitness value, that is 1, could not be achieved due to difficulties in satisfying all soft constraints. But all hard constraints were satisfied which implies that all courses can be allocated to different rooms and time-slots.

From Figs. 10 and 11 it is clear that every class meeting has their room. Hamza Soye teaches EDU on Monday at 9:00 to 10.00. The system is able to do this by looking at the room table in the database. It is easy to search for rooms by comparing the room size, their location etc, using the requirement of each class as specified in the database. After a room had been allocated, the status of that room is changed not available for that particular day and hour.

As observed in the graph of Fig. **7**, the number of clashes in a course timetabling increases in proportion with the population size. These clashes however reduces to a zero level as the number of generation increases indicating a continuous improvement in the fitness value of the generated parents over time as seen in Fig. 9. The population size should therefore be large because a bigger population size leads to a better representation and a wider search. The downside of increasing population is that the calculation time increases proportionally.

From the graph of Fig. 9 the fitness value of the genetic algorithm tends to increase very fast at the early stage of generation, but this gradually eases off to nearly the same fitness value as the number of generation increases. However the fitness value never reached the value of one as it was not possible to satisfy all the soft constraints specified in the program. It was observed that faster convergence rates normally lead to a greater number of populations being stalled at a local maximum. In others words faster convergence implies a narrower search and a wider search leads to a slower convergence. Rate of convergence can be seen graphically by observing how the fitness of the best chromosome evolves from generation to generation in Fig. 9. This result is similar to what was obtained in Kuldeep [20], Chohan [21], and Aydin [22] which shows that genetic algorithm exhibits similar characteristics when applied to solve timetabling problem as long as the chromosome representation is properly formed irrespective of the number of complexity.

7 Conclusions and Future Research

The timetabling problem may only be solved when the constraints and allocations are clearly defined and simplified thoroughly and more than one principle is applied to it. This is because an improper

representation of chromosomes in a population based heuristic search can lead to a high level violation of the problem constraints (Anam et al. [23]). Based on the evaluation carried out on the completed system it has shown that the completed system worked effectively well. Therefore using this approach will definitely ensure a clash free timetable for any college of education. This software makes the timetable production process faster, while the genetic algorithm ensure that only the fittest population got displayed as the final timetable. Hence, it can be concluded that with a correct input data, the genetic algorithm will generate "good" timetable for big schools no matter their complicated teaching plan.

In an attempt to create a valid timetable, most of the soft constraints were removed from the fitness function during implementation and were never re-introduced, mostly due to program complexity. In real world situations there are preferences that can make a timetable more flexible. The inclusion of certain soft constraint such as lecturers preferring some specific free periods or some part of days off will require a more efficient search technique.

Efficiency of the timetable could be further enhanced by embedding genetic algorithm with other methods such as particle swam algorithm, and fussy system etc. Embedding these algorithms with genetic algorithm may produce a better result in a more limited time. The timetable can also be made more flexible by making it an online system to enable lecturers and students have access to their timetable online. The research should be repeated with more recent meta-heuristic algorithm such as flower pollination algorithm because it has shown to be better than the genetic algorithm in solving optimization problems (Chiroma et al. [24-25]), hybridization of recurrent neural network and cuckoo search algorithm (Nawi et al. [26]) and lastly, hybrid cuckoo search (Chiroma et al. [27]).

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Mohd NMK, Masri Ayob. Resource estimation for optimized allocation of course timetabling. Proceedings of the International Conference on Electrical Engineering and Informatics Institute Technology Bandung, Indonesia; 2007.

[2] Eugene Ruben Ramire. Using genetic algorithm to solve high school course timetabling problem. Msc Thesis, San Diego State University; 2010.
Available: http://sdsudsapace. calstate.edu

[3] Landa S. The office space allocation – Problem overview; 2003.
Available: http://www.cs.nett.ac.uk/jds/research/spaceallocation.html
(Viewed 22/2/2013)

[4] Nashwan AA, Talal HA. Solving of lectures timetabling problem and automatic timetable generation using genetic algorithm. International Journal of Advanced Research in Computer and Communication Engineering. 2016;5(9).

[5] Schaerf A. A survey of automated timetabling. Artificial Intelligence Review. 1999;13(2):87–127.

[6] Luís Paulo Reis, Eugénio Oliveira. A language for specifying complete timetabling problems. Selected Papers from the Third International Conference on Practice and Theory of Automated Timetabling III. 2012;322-341
Available:http://paginas.fe.up.pt/~niadr/PUBLIC ATIONS/Unilang.pdf
(Viewed 2/3/2015)

[7] Fang HL. Genetic algorithms in timetabling scheduling. Ph.D. thesis, University of Edinburgh, Edinburgh, UK; 1994.

[8] Grobner M, Wilke P. A general view on timetabling problems. 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT); 2002.

[9] Manar Hosny, Shameem Fatima. A survey of genetic algorithms for the university timetabling problem. International Conference on Future Information Technology IPCSIT. IACSIT Press, Singapore. 2011;13.

[10] Yang S, Jat SN. Genetic algorithms and local search strategies for university course timetabling. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews. 2011;41(1):93-106.

[11] Sapru V, Reddy K, Sivaselvan B. Time table scheduling using genetic algorithms employing guided mutation, in Proc. IEEE International Conference on Computational Intelligence and Computing Research (ICCIC). 2010;1-4.

[12] AlSharafat WS, AlSharafat MS. Adaptive steady genetic algorithm for scheduling university exams, in Proc. Inernational Conference on Networking and Information Technology (ICNIT). 2010;70-74.

[13] Raghavjee R, Pillay N. Using genetic algorithms to solve the South African School timetabling Problem. In Proc. Second World Congress on Nature and Biologically Inspired (NaBIC). 2010;286-292.

[14] Abdullah S, Turabieh H, McCollum B, Burke EK. 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, Dublin, Ireland. 2009;727-731.

[15] Holland John. Scheduling adaptation in national and artificial systems. The University of Michigan press; 1975.
Available: http: /www.cs.qub. qc.wk/itc.2007/index. files/overview

[16] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc; 1989.

[17] Chakraborty RC. Foundamentals of genetic algorithms. A.I Course lecture notes, slides. 2010;39-40.
Available: www.myreaders.infor/html/artificial_intelligence.html
(Viewed 22/6/2014)

[18] Gen M, Cheng R. Genetic algorithms and engineering optimization. New York, NY, USA: John Wiley & Sons; 2000.

[19] Alade OM, Omidiora EO, Olabiyisi SO. Development of a university lecture timetable using. Modified Genetic Algorithms Approach, International Journal of Advanced Research in Computer Science and Software Engineering. 2014;4(9).
Available: http://www.ijarcsse.com/docs/papers/ Volume_4/9_September 2014/V4I9-0105. pdf
(Viewed 12/5/2015)

[20] Kuldeep K. Genetic algorithm approach to automate university timetable; 2012.
Available:https://www.researchgate.net/deref/http%3A%2F%2Fejournal.oitmhisar.com%2FPapers%2520for%2520E-Journal%2FPDF%2F7.pdf

[21] Chohan Ossam. University scheduling using genetic algorithm. Högskolan Dalarna; 2009. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:du-3791

[22] Aydin MA. Solving university course timetabling problem using genetic algorithm; 2008. Available: http://iheqexuxyl.tracktime.us/edu/master-thesis-genetic-algorithm

[23] Anam S, Tazeen A, Faizan B, Anand B. Genetic algorithm with meta- heuristic approach for generating timetable. International Journal of Advanced Research. 2016;4(2):300-304.

[24] Chiroma H, Shuib NLM, Muaz SA, Abubakar AI, Ila LB, Maitama JZ. A review of the applications of bio-inspired flower pollination algorithm. Procedia Computer Science. 2015A;62:435-441.

[25] Chiroma H, Khan A, Abubakar AI, Saadi Y, Hamza MF, Shuib L, Herawan T. A new approach for forecasting OPEC petroleum consumption based on neural network train by using flower pollination algorithm. Applied Soft Computing. 2016;48:50-58.

[26] Nawi NM, Khan A, Rehman MZ, Chiroma H, Herawan T. Weight optimization in recurrent neural networks with hybrid metaheuristic Cuckoo search techniques for data classification. Mathematical Problems in Engineering; 2015.

[27] Chiroma H, Abdul-Kareem S, Khan A, Nawi NM, Gital AYU, Shuib L, Herawan T. Global warming: Predicting opec carbon dioxide emissions from petroleum consumption using neural network and hybrid cuckoo search algorithm. Plos One. 2015B;10(8):e0136140.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://sciencedomain.org/review-history/18420