DON BOSCO INSTITUTE OF TECHNOLOGY

Premier Automobiles Road, Kurla (W), Mumbai-70

Approved by AICTE, Govt. of Maharashtra

&

Affiliated to the University of Mumbai

B.E. PROJECT REPORT "I"

On

"Automatic Timetable Generation Using Genetic Algorithm"

Department of Computer Engineering

University of Mumbai

2019-2020

Project Title	Automatic Timetable Generation Using Genetic Algorithm
Organization	Don Bosco Institute of Technology
Address	Premier Automobiles Road, Kurla (W), Mumbai-70
Project Team Members	1. Olivia D'sa 2. Denzil D'souza 3. Prayuj Pillai
Internal Guides	Dr. Phiroj Shaikh
Address	Department of Computer Engineering, Don Bosco Institute of Technology, Premier Automobiles Road, Kurla (W), Mumbai-400070

_____

INTERNAL GUIDE (s)

_____

HEAD, COMPTUER ENGINEERING

CERTIFICATE

This is to certify that the following project members of final year class have satisfactorily completed the PROJECT- I report on "Automatic Timetable Generation Using Genetic Algorithm" in the partial fulfillment of the Computer Engineering course of Bachelor of Engineering of the Mumbai University during July to October 2019.

Project Team Members:

1. Olivia D'sa B. E. – (Roll No: 18.)
2. Denzil D'souza B. E. – (Roll No: 21)
3. Prayuj Pillai B. E. – (Roll No: 49)

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Most colleges have several different courses and each course has several subjects. There are limited faculties, each faculty teaching more than one subject. The Timetable is needed to schedule the faculty at provided time slots in such a way that their timings do not overlap, and the timetable schedule makes best use of all faculty subject demands. We use a genetic algorithm for this purpose. Many Colleges uses Timetable software for scheduling lectures, practical, tutorial and academic activities. This software is proprietary and not open source, a fee must be paid for using this software legally. Due to this fact, an open source "Automatic Timetable Generation" software is the way to go. The project provides an architecture to design such a system to generate a timetable based on specific hard and soft constraints which need to be satisfied for a given organization.

BACKGROUND

Our body is made up of trillions of cells. Each cell has a core structure (nucleus) that contains your chromosomes. Each chromosome is made up of tightly coiled strands of deoxyribonucleic acid (DNA). Genes are segments of DNA that determine specific traits, such as eye or hair color. You have more than 20,000 genes. A gene mutation is an alteration in your DNA. It can be inherited or acquired during your lifetime, as cells age or are exposed to certain chemicals. Some changes in your genes result in genetic disorders.

Natural Selection: Darwin's theory of evolution: only the organisms best adapted to their environment tend to survive and transmit their genetic characteristics in increasing numbers to succeeding generations while those less adapted tend to be eliminated.

GA is Inspired from Nature

A genetic algorithm maintains a population of candidate solutions for the problem at hand, and makes it evolve by iteratively applying a set of stochastic operators. Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or the genotype or the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves to a better solution. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are selected from the current population (based on their fitness) and modified (recombined and possibly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

Table of contents

ABBREVIATIONS

HTML	Hypertext Markup Language
GA	Genetic Algorithm
MTTS	Manual Timetable Scheduler

CHAPTER 1: INTRODUCTION

Timetable Scheduling is an NP-hard problem and hence polynomial time verifiable using genetic algorithms. It a typical scheduling problem that appears to be a tedious job in every academic institute once or twice a year. In earlier days, timetable scheduling was done manually with a single person or some group of people involved in task of scheduling, which takes a lot of effort and time. Planning timetables is one of the most complex and error-prone applications.

The lecture-timetable scheduling is a Constraint satisfaction problem in which we find a solution that satisfies the given set of constraints. Automatic Timetable Generator is a software used to generate timetable automatically. It will help to manage all the periods automatically. The maximum and minimum workload for a Faculty for a day and week will be specified for the efficient generation of timetable.

Timetable Scheduler targets to develop software for colleges in order to handle the “Timetable Formation” for the staff. The head of every Department has problems in delegating work to their subordinates and response for the work position. This work resolves the problem by permitting the lecturers to see their assigned subject and timetable. This software helps to handle the particulars of the timetable of staff. The process of preparing a timetable involves beneficial employment of resources which needs to be confronted each year by every educational institute.

Classroom object comprises of week objects. Week objects comprise of days, comprises of timeslots. Timeslot has an address in which a subject, student gathering going to the address and educator showing the subject is related.

CHAPTER 2: LITERATURE SURVEY

2.1 Survey Existing System

Sr No.	Paper Title	Authors	Summary	Conclusions
1	Solving Timetable Problem by Genetic Algorithm and Heuristic Search Case Study: Universitas Pelita Harapan Timetable	Samuel Lukas, Arnold Aribowo and Milyandreana Muchri	Steps involved in GA are thoroughly explained in this paper. The operations performed in each step, i.e. Selection, Crossover and Mutation are given in detail. Input format is presented well in the paper. The target matrix is the data structure used for input.	Soft Constraints are not included in the scope along with constraints of classrooms.
2	Electronic Lecture Time-table Scheduler using Genetic Algorithm	Jumoke Soyemi, John Akinode and Samson Oloruntoba	Compares MTTS with GA, and shows flowchart of how GA is implemented	Preference of the University to use Automatic Scheduling with GA. However Soft constraints are not taken care of.
3	Solving Timetable Scheduling Problem Using Genetic	Branimir Sigl, Marin Golub and Vedran Mornar	Shows formation of Chromosomes and Genes which are used as the basis for GA	Alternate approach to the data structure being used to optimised.

	Algorithms			
4	Solving Timetabling Problem Using Genetic and Heuristic Algorithms	Nguyen Duc Thanh	Incorporates the use of a hybrid algorithm consisting of Genetic and Heuristic Algorithm. Using this Algorithm Timetabling problem is converted into a way of optimally solving a 2D Matrix.	Crossover and Mutation operations of the given system will be used.
5	Genetic Algorithms vs. Simulated Annealing: A Comparison of Approaches for Solving the Circuit Partitioning Problem	Theodore W. Manikas and James T. Cain	A simplified model of the placement problem circuit partitioning was tested on three circuits with both a genetic algorithm and a simulated annealing algorithm When compared with simulated annealing the genetic algorithm was found to produce similar results for one circuit and better results for the other two circuits	The study concluded its findings based on a very small dataset rather than considering a large data set to give an approximate percentage as on how much is genetic algorithm more efficient than simulated annealing

Table 2.1 Summary and Conclusions of Surveyed Papers

Summary and Conclusions of Surveyed Papers:

Detailed Summary of the Surveyed Papers:

1. Samuel Lukas, Arnold Aribowo and Milyandreana Muchri, "*Solving Timetable Problem by Genetic Algorithm and Heuristic Search Case Study: Universitas Pelita Harapan Timetable*":
    a. Methodology Employed:

        The paper describes in detail the Steps involved in GA. The operations performed in each step that is Selection, Crossover and Mutation are given in detail. The Architecture Design of the System makes use of a Target Matrix. Six sets are applied to this Matrix. The description of each set are as follows:

        i. Course Code set **M** = $\{M_1, M_2, \ldots\}$
        ii. Type Course Class set **T** = $\{T_0, T_1, \ldots\}$
        iii. Lecturer Code set **L** = $\{L_1, L_2, \ldots\}$
        iv. Class Name set **C** = $\{C_1, C_2, \ldots\}$
        v. Day set **D** = $\{D_1, D_2, \ldots\}$
        vi. Hour set **H** = $\{H_1, H_2, \ldots\}$

        The columns of this matrix are made up by the close relationship of L_i, M_j and T_k which indicate Lecturer 'i' teaches Course 'j' of the type 'k'.

        The rows are made of the relationship between D_x, H_y and C_z which indicate Slot on Day 'x' at Hour 'y' with Class 'z'

        Each combination between the row and column indicates a unit of the timetable, i.e., for Target Matrix 'V', V_{ij} being the element on i^{th} row and j^{th} column gives us d, h, c, l, m, t which is on day 'd' at hour 'h' for class 'c', Lecturer 'l' teaches course 'm' of type 't'.

Genetic Algorithm steps as mentioned above are then applied to this Target Matrix.

b. Result Achieved:

An experimental setup for 7 courses and 10 lecturers with 2 type course classes (lecture and lab), 2 class names (A and B), 5 days lecture a week and 10 hours lecture a day. The system ran with 10 chromosomes and 10 generations are set in the experiment. An error free Schedule was achieved.

c. Scope for Future Improvement:

Room set has not been explicitly defined in the target matrix; however, the system is still able to manage and allot rooms. The author/s note that if room plays a key role in the scheduling, then it should be included in the target matrix.

d. Platform Dependency and Experimental Setup Requirement:

The proposed system has no platform dependency and the setup requirement would be based on the technology used.

2. Jumoke Soyemi, John Akinode and Samson Oloruntoba, *"Electronic Lecture Time-table Scheduler using Genetic Algorithm":*

a. Methodology Employed:

The given paper describes timetable generation to be an NP hard problem (a problem where there isn't any specific approach or algorithm that can be used to solve it).The problem of timetable generation was defined as a problem of assigning the various resources to the meetings in a consistent manner. Timetable involves a set of incident E = {e1, e2,…, en} meeting a set of time to the said incident T = {t1, t2, …, ts}. The set of places where the incident occurred P = {p1, p2, …, pm} and a set of agents to conduct

the incident A = {a1, a2, ….an} for example, lecturers. The paper describes the system design for an automatic timetable generation system using genetic algorithm. All the steps of genetic algorithm were clearly explained in the paper.

b. Result Achieved:

A study was conducted to adopt the electronic timetable generation system against the manual system that was used in most of the Nigeria Tertiary Institutions of learning. Around 215 questionnaires were distributed, and their response was recorded. Opinion of respondents towards their perception on the two systems based on time consumption, efficiency, convenience and overall performance were analysed on a Likert scale of three thereby resulting in the test of degree of relationship and formulation of hypothesis. Based on the empirical analysis of the study it was concluded that electronic system was more appropriate and would be the best method to adopt in tackling the lapses of its manual process.

c. Scope of Future Improvement:

The given system designed was approved by 93% of the people involved in the study. However, the system doesn't provide the system specifications and modifications to handle soft constraints appropriately.

d. Platform Dependency and Experimental Setup Requirements:

Java was the programming language used for the development of the front end and Microsoft SQL Server was employed for designing the back end. Window 7 operating system is the minimum operating system required to run the application. Also, Adobe PDF reader or any other PDF reading software was used.

3. Branimir Sigl, Marin Golub and Vedran Mornar, *"Solving Timetable Scheduling Problem Using Genetic Algorithms":*

   a. Methodology Employed:

      The papers approach to solving the Timetable problem is using a 3D structure. The dimension of this structure is:

      i. Days on the x axis
      ii. Time Slots on the y axis
      iii. Rooms on the z axis

      The scheduling process is placing those cubes into a timetable such that conflicts are not there. Conflicts in a timetable are a room being scheduled to two different classes at the same time, a lecturer teaching two or more classes at the same time or a class having two classes at one time.

   b. Result Achieved:

      The proposed system was tested on both large and small instances of problem. The large problem consisted of almost 770 classes with 41 rooms and 114 groups and the smaller problem was 70% less than this with 227 classes, 27 rooms and 55 groups. The smaller problem was solved with no conflicts. The larger problem with 95 conflicts which was later brought down to 20 conflicts using intelligent operators.

   c. Scope for Future Improvement:

      The algorithm can be improved in certain ways. One way is when generating constraints, sign each one so that no constraint is checked twice. Individuals should be generated in such a way that classes which are difficult to schedule occupy the front genes and the easier ones behind.

d. Platform Dependency and Experimental Setup Requirement:

The proposed system has no platform dependency and the setup requirement would be based on the technology used.

4. Nguyen Duc Thanh, *"Solving Timetabling Problem Using Genetic and Heuristic Algorithms":*

a. Methodology Employed:

Like the *Samuel Lukas, Arnold Aribowo and Milyandreana Muchri, "Solving Timetable Problem by Genetic Algorithm and Heuristic Search Case Study: Universitas Pelita Harapan Timetable*", this paper too makes use of Target Matrix with the same sets as described in the summary of the paper. Along with this a detailed set of rules for filling the target matrix is included. The Gene representation employed is using the column id which represents (c, o, i) where c is classes, o is courses and i is instructors. The flowchart of the algorithm is explained in detail. The initial population is generated randomly. Crossover is implemented using a two-point crossover and mutation is randomly changing the Gene.

b. Result Achieved:

In the experiment, the size of population, and the ratio of choosing parents and children for the next population are 100, 0.4 for parents and 0.6 for children. The probabilities of crossover, mutation and reversion are set to 0.8, 0.05, and 0.05 respectively. These values were chosen experimentally through testing. In all cases an error free timetable was achieved.

c. Scope for Future Improvement:

In order to solve the general scheduling problem, further research needs to be done. For example, solving timetabling problems that allow one

combination of classes, courses, and instructors takes more than one time per week.

d. Platform Dependency and Experimental Setup Requirement:

The proposed system has no platform dependency and the setup requirement would be based on the technology used.

5. Theodore W. Manikas and James T. Cain, *"Genetic Algorithms vs. Simulated Annealing: A Comparison of Approaches for Solving the Circuit Partitioning Problem":*

a. Methodology Employed:

The paper compares genetic algorithm and simulated annealing for its implementation in circuit design for placement where components are assigned to physical locations on a chip. A popular contemporary method for placement is the use of simulated annealing. While this approach has been shown to produce good placement solutions recent work in genetic algorithms had produced promising results. Genetic algorithm and simulated annealing algorithms were explained and implemented for the above problem and the results were compared and studied. A genetic algorithm is an iterative procedure that maintains a population of individuals these individuals are candidate solutions to the problem that had to be solved. Each iteration of the algorithm is called a generation. During each generation the individuals of the current population are rated for their effectiveness as solutions Based on these ratings a new population of candidate solutions is formed specific genetic operators. Simulated annealing is an iterative procedure that continuously updates one candidate solution until a termination condition is reached.

b. Result Achieved:

Both a genetic algorithm and simulated annealing approach were tested on a set of circuits Three circuits were selected for data sets. For the genetic

algorithm the population size P and swing value W were varied during testing for simulated annealing the starting temperature T cooling factor number of move state M and stopping value ts were varied during testing Each set of parameter combinations forms a treatment. There were approximately 20 trials per treatment. Genetic algorithm produced a smaller average cut size than simulated annealing for two circuits and while no significant difference was found between the methods for the third circuit. Based on the results of the study the genetic algorithm was shown to produce solutions equal to or better than simulated annealing when applied to the circuit partitioning problem.

c. Scope of Future Improvement:

The study concluded its findings based on a very small dataset rather than considering a large data set to give an approximate percentage as on how much is genetic algorithm more efficient than simulated annealing. However, the paper explains both the algorithms in detail with the procedure and finding for all the test cases considered.

2.2 Literature Survey Summarization:

In two of the Surveyed papers, namely, Samuel Lukas, Arnold Aribowo and Milyandreana Muchri, "*Solving Timetable Problem by Genetic Algorithm and Heuristic Search Case Study: Universitas Pelita Harapan Timetable*" and Nguyen Duc Thanh, "Solving Timetabling Problem Using Genetic and Heuristic Algorithms" the system makes use of a target matrix on which Genetic Algorithm is applied to find the most suitable/optimised positions of slots in the matrix. Also, in both these papers Encoding of information into chromosome and Selection and Mutation Operators described are of the same type.

One gap in the Research is that none of the papers surveyed consider soft constraints. The papers only consider that classrooms, lectures, student groups and labs should not clash, i.e. the hard constraints.

The approach to solving the timetable problem using Genetic Algorithm is straight forward, but one aspect that is consistent in all the research papers is that each system implements certain optimization techniques to allow the solution to reach a global optimum. Those operations may include heuristic algorithms along with Genetic Algorithms or even modified Genetic Algorithm Stages.

2.3 Problem Statement:

The timetable generator aims to create the best optimised timetable for any given institution, consider various constraints while generating a timetable and apply the genetic algorithm approach to produce the best optimal timetable based on the organization's preferences and specified constraints.

2.4 Objectives:

- To generate an optimized timetable.
- To use genetic algorithm to generate the timetable
- To satisfy all the hard constraints.
- To satisfy soft constraints as much as possible.
- Final timetable is the timetable with the best fitness.

2.5 Scope of the Project:

Project Goals:

- Make a comprehensive "Automatic Timetable Generation" Software that Schedules lectures, lab, tutorials, workshop, etc. efficiently.
- Make use of Genetic Algorithm for its implementation.
- Software handles all the relevant constraints.

Project Deliverables:

- Timetable generated automatically using Genetic Algorithm
- Exporting Timetable for each lecture, lab, tutorials, workshop into suitable formats.

CHAPTER 3: PROPOSED SYSTEM

3.1 Architecture Framework of Proposed System:

Timetable is a very common problem when it comes to Scheduling and that is time consuming, energy sapping and leads to waste of resources. Almost all Universities have issues when it comes to lecture scheduling due to all the constraints such as lecturer's availability, classroom and lab availability, etc.

An efficient way for scheduling this is using Genetic Algorithm. Genetic Algorithms are adaptive heuristic search/optimization algorithms that belong to the family of evolutionary algorithms. They are commonly used to generate high-quality solutions for optimization problems and search problems.

Genetic algorithms simulate the process of natural selection. It is based on the concept of survival of the fittest.

The algorithm can be summarised as:

i. A population p is randomly initialised
ii. The fitness of the population is determined
iii. Until the stopping criterion is met, do:
    a) **Select** parents from population
    b) **Crossover** and generate new population
    c) Perform **mutation** on new population
    d) Calculate fitness of new population

The main operators of GA are:

i. **Selection**: It is the process of selecting a part of the population to breed a new generation. Figure 3.1 shows Roulette Wheel Selection which is a selection procedure in which the possibility of selection of a chromosome is directly proportional to its fitness.

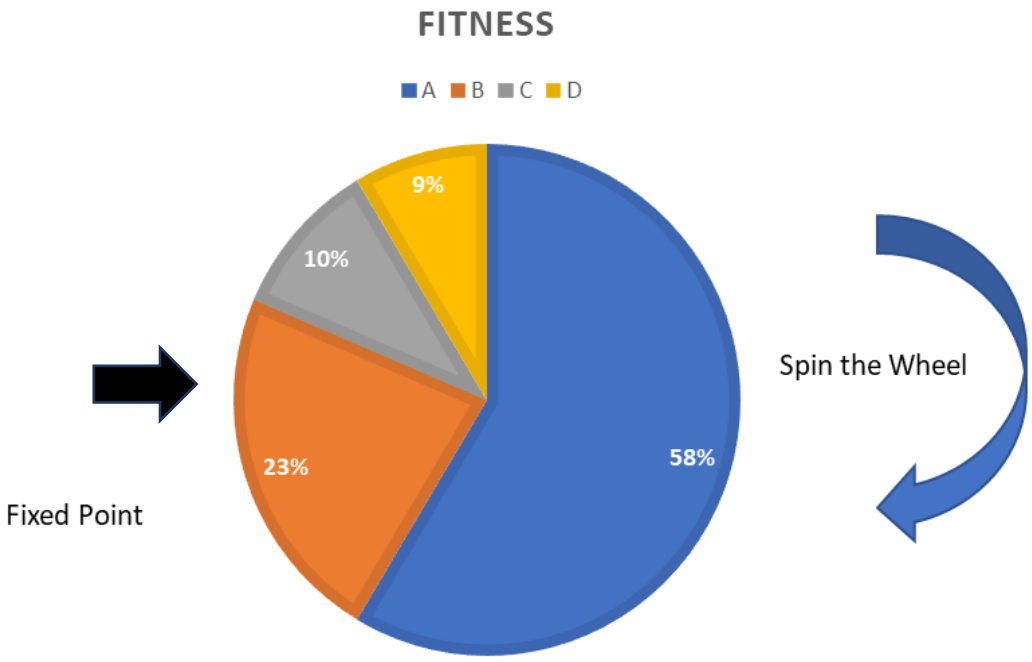


Figure 3.1 Roulette Wheel Selection

ii. **Crossover**: Also known as recombination, it makes use of two parents' properties to generate a child. Figure 3.2 shows Two Point Crossover

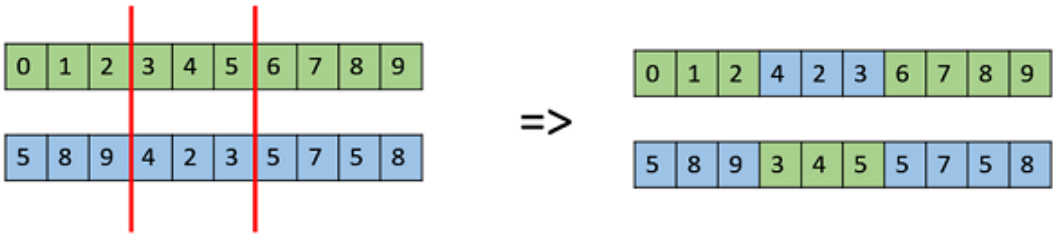


Figure 3.2 Two Point Crossover

iii. **Mutation**: The key idea is to insert random genes in offspring to maintain the diversity in population to avoid the premature convergence

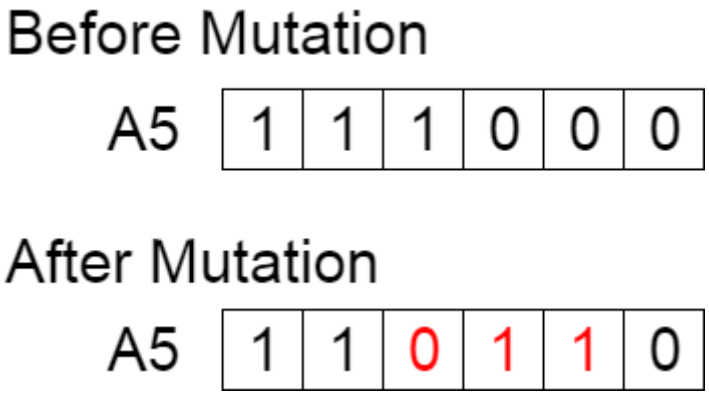


Figure 3.3 Mutation

Now that we have discussed the main steps in Genetic Algorithm, let us outline the framework of the proposed system.

We will make use of a 2D matrix called a 'Target Matrix'. Along with this matrix, six sets which are pertinent to timetable scheduling will be defined. These six include:

i. Course Code set **M** = {M1, M2, M3, …}
ii. Lecturer Code set **L** = {L1, L2, L3, …}
iii. Class Code set **C** = {C1, C2, C3, …}
iv. Room Code set **R** = {R1, R2, R3, …}
v. Day set **D** = {D1, D2, D3, …}
vi. Hour set **H** = {H1, H2, H3, …}

Each column in the matrix will represent a set of <**M, L, C**> which represents Lecturer '**L**' teaching Course '**M**' for Class '**C**'.

Each row in the matrix will represent a set of <**R, D, H**> which represents Room '**R**' on Day '**D**' at Hour '**H**'.

For set 'X', N(X) is defined as number of elements of X.

$$N(rows) = N(R) \times N(D) \times N(H)$$

Number of columns however would be the number of valid combinations of <M, L, C>.

Target Matrix:

	m1, l1, c1	m1, l2, c1	m2, l1, c2	…
r1, d1, h1	1	-1	-1	…
r1, d1, h2	1	-1	-1	…
r1, d2, h1	-1	1	-1	…
r1, d2, h2	-1	1	-1	…
r2, d1, h1	1	-1	-1	…
r2, d1, h2	1	-1	-1	…
r2, d2, h1	0	0	0	…
r2, d2, h2	-1	-1	1	…
…	…	…	…	…
No. of units Scheduled (th) $\sum_{i=1}^{N(Rows)} Mij,\ \forall j \in Columns, Mij > 0)$	4	2	1	

Table 3.1 Target Matrix

From the target matrix for an element M_{ij}, if it is set to 1, it indicates that Lecturer 'L' teaching Course 'M' for Class 'C' is scheduled to Room 'R' on Day 'D' at Hour 'H'.

Each Cell value can have the following values

i. $M_{ij} = 0$, indicates that the slot is available to be scheduled
ii. $M_{ij} = 1$, indicates that the slot has been scheduled
iii. $M_{ij} = -1$, indicates that the slot cannot be scheduled

Also, for each element M_{ij} a set of 6 functions are defined namely f_m, f_l, f_c, f_r, f_d and f_h which return information of attributes course, lecturer, class, room, day and hour respectively.

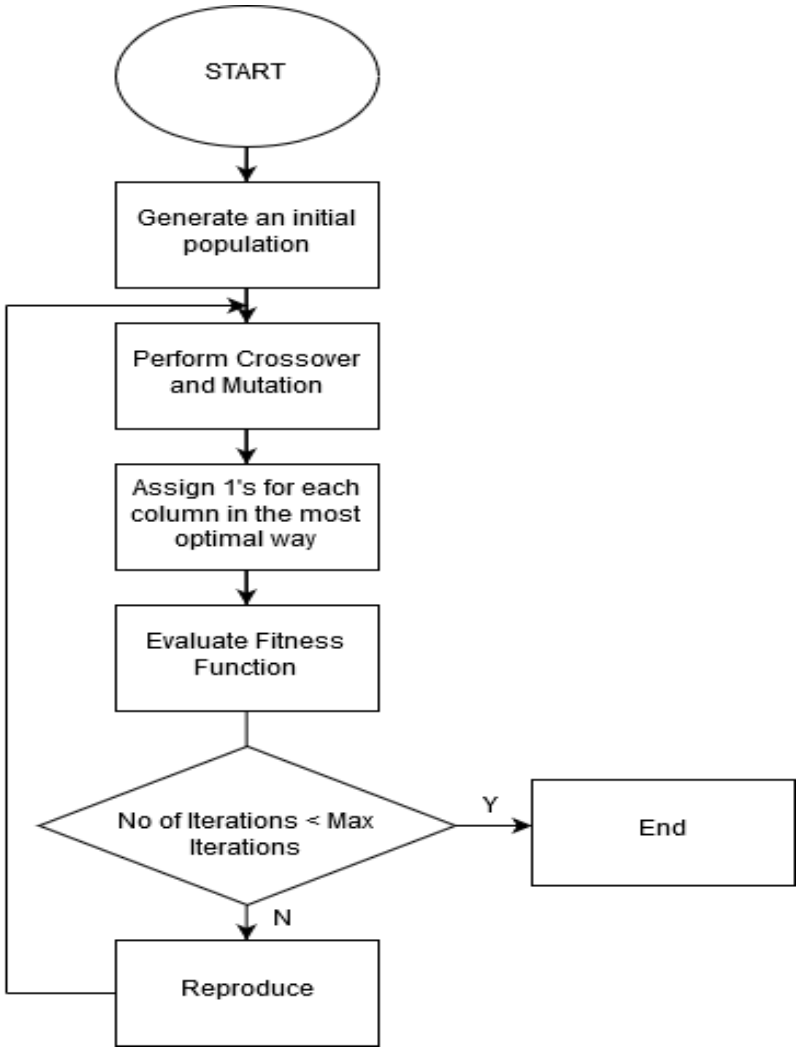
Apart from the above, certain rules must be followed:

i. $\sum_{j=1}^{N(Columns)} Mij \leq 1, \ \forall\, i \, where \, Mij \geq 0$, i.e. sum of all non-negative numbers in a row should be less than or equal to one.

ii. $\sum_{i=1}^{N(Rows)} Mij = th\,, \forall\, j \, where \, Mij \geq 0$, i.e. the number of hours that a combination of <M, L, C> is scheduled should be the number of hours required for it.

iii. The combination <M, L, C> should only be scheduled in combination <R, D, H> if it can be accommodated in R.

The Flowchart 3.1 indicates the proposed Algorithm



Flowchart 3.1 Proposed Algorithm

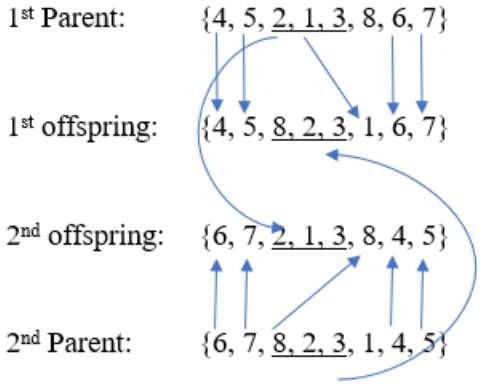
Genetic Algorithm:

i. Gene Representation: The chromosome will be represented as a string of numbers made up of the column ids. For e.g. an 8 Gene representation can be {4, 5, 2, 1, 3, 8, 6, 7}. In this case we shall arrange the 4^{th} column first then the 5^{th} and so on.

ii. Initial Population: An Initial Population will be randomly generated. In the above example we looked at one chromosome, a population would be a

collection of chromosomes that are mostly unique. A population example is C1 = {4, 5, 2, 1, 3, 8, 6, 7}, C2 = {4, 7, 8, 2, 3, 1, 6, 5}, …, Cn.

iii. Crossover: A two-point Crossover will be used wherein parents are divided into three portions cut by two randomly generated points. Matching sections of two individuals are swapped to create offspring. Example



iv. Mutation: Two random Genes are swapped while inversion.

v. Fitness Evaluation: Let *s(k)* be number of units scheduled in k^{th} column and p is the maximum column in target matrix, *u(j)* means unit number of j^{th} course to be scheduled for each course and t is total classes. Fitness is given by:

$$Fitness = \frac{\sum_{k=1}^{p} s(k)}{n(c).\sum_{j=1}^{t} u(j)}$$

3.2 Details of Hardware and Software:

Technologies that will be used for the implementation:

- Backend Development
  - PHP
  - Java 8
- Database
  - MySQL
- Frontend Development
  - React JS

- Logical Implementation
  - J2EE Technologies/Spring Framework

3.3 Design Details:

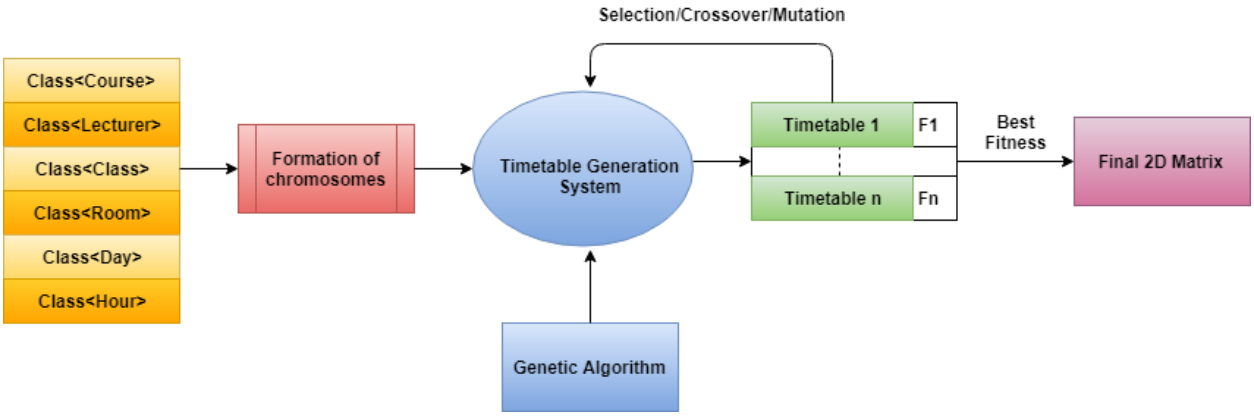


Figure 3.4 Proposed Architecture

As Discussed in the Framework, the Target Matrix makes use of six sets. Each Element in a set will be an Object which will be identified uniquely by an ID. The following are the description of each Object:

1. The **Class** has an ID, name of student group, number of subjects, array of subject names and hours of study required for each subject per week. It also contains the id of teachers who will teach those subjects.
2. The **Lecturer** is a class to hold the faculty information. It has an id, name of faculty, subject that he/she teaches.
3. The **Course** is a class that hold the information on the Course, the Faculty that teach it, whether it is a lab or regular lecture and is identified by Course ID.
4. The **Room** is class which stores the details of a classroom/lab, its occupancy, its speciality, corresponding courses to be held, venue details and is identified by Room ID.
5. The **Hour** class represents the timeslots available for Scheduling.

6. The **Day** class represents the Days that the Lectures are scheduled on.

3.4 Methodology:

The given problem is an NP hard problem with no specific approach to solve it. We will be using genetic algorithm to solve the problem based on our previous study which highlighted the superiority of genetic algorithm over other algorithms.

The given application has four main steps:

1. Taking input from the user:

- This indicated various constraints that would have to be considered for generating the timetable.
- The information would contain details about the students, teachers, classrooms, labs, days and lecture hours.
- The above information would be provided to the application through a user interface designed to do so.

2. Mapping the input to genetic algorithm:

- The input received from the user end will be stored in the form of objects.
- However, genetic algorithm cannot work on objects it needs the input to be in the form of chromosomes.
- This step will consist of converting all the inputs to chromosomes.

3. Generating the timetable

- The final deliverable of the project is the timetable with the best fitness.
- Genetic algorithm will perform the operations like selection, crossover and mutations iteratively on the chromosomes to generate the timetable.
- At each step the fitness of each timetable generated will be calculated.
- The application will return the timetable which has the highest fitness score.

4. Final timetable

- The final timetable is a 2D matrix which contains the schedule for the given institution which has the maximum fitness.
- The timetable must be sent to the given organization in the format that they would approve. E.g. XLS, text, excel, etc.

So, the final step is delivering the timetable in any suitable format as requested by the given institution.

CHAPTER 4: Implementation

4.1 Implementation plan:

Work Breakdown Structure:

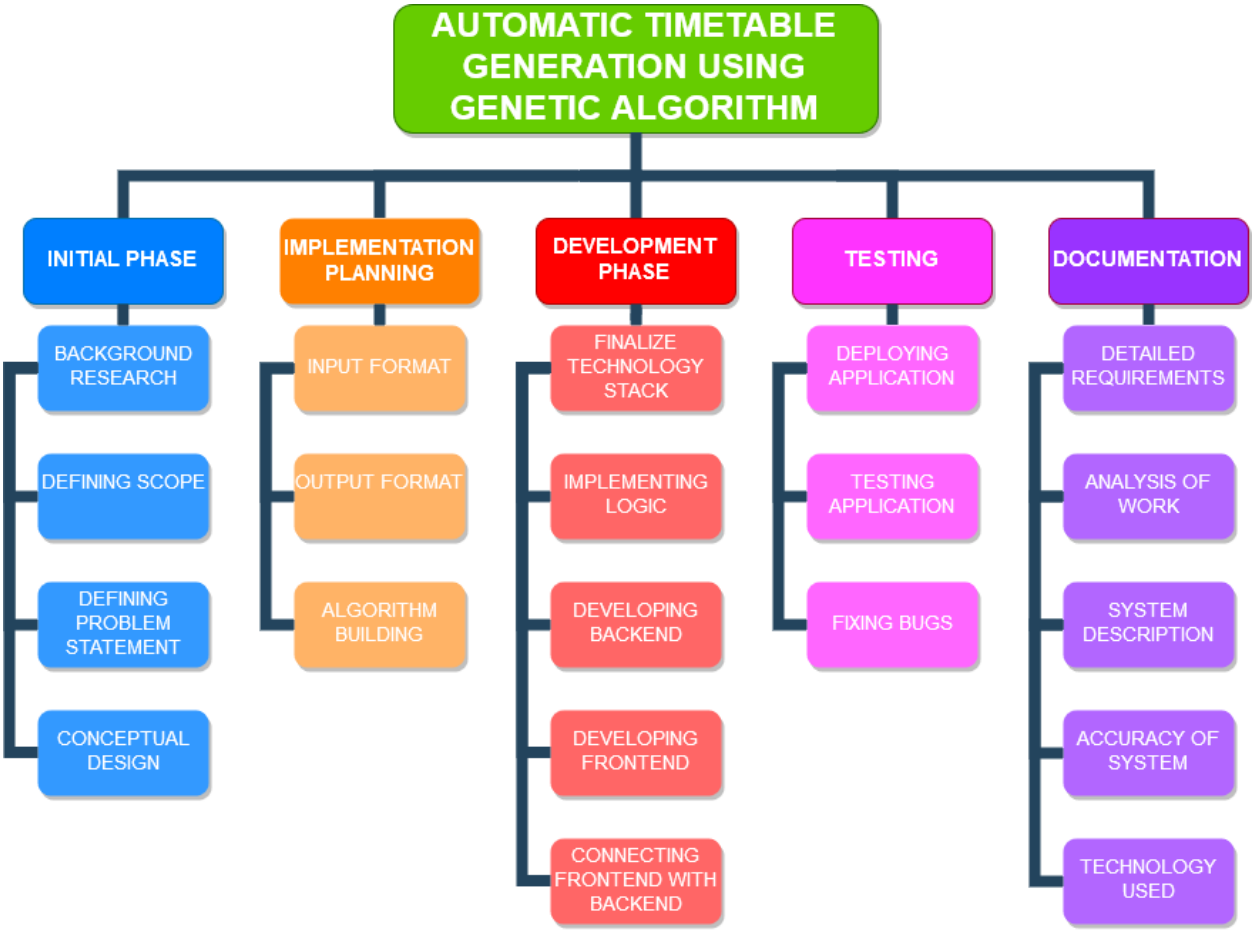


Figure 4.1 Work Breakdown Structure

Timeline chart

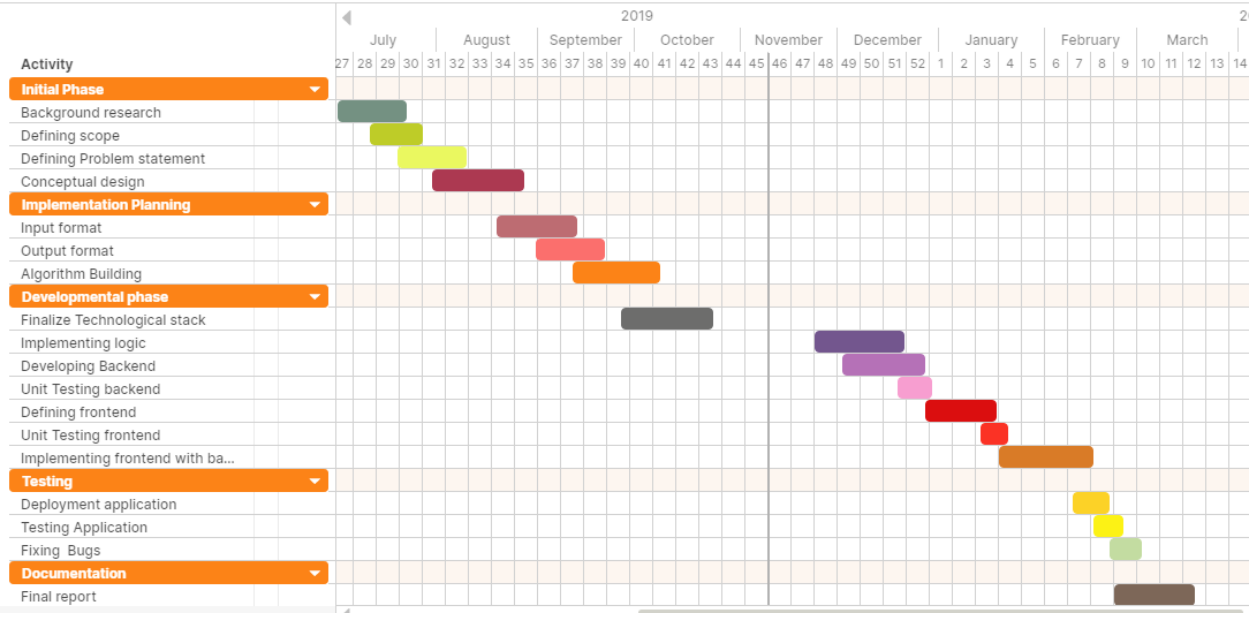


Figure 4.2 Timeline chart

Initial phase

- The background research was one of first task completed in which the project requirements were understood, and various approaches were studied to do the same.
- The scope of the system as planned will be confined to DBIT and is aimed to generate an optimized timetable using genetic algorithm.
- Problem statement was defined before the first presentation and was approved by Dr. Phiroj Shaikh.
- The conceptual design was made considering the hard and soft constraints and the process of chromosome formation.

Implementation Planning

- The input format was defined considering various constraints of the timetable generation problem and their mapping to the inputs that can be processed by genetic algorithm.
- Output format was defined as a 2D matrix which would represent the final timetable.

- Genetic algorithm was studied thoroughly and the steps to be considered to solve the timetable generation problem.

Development phase

- The technological stack was defined by the end of September to using J2EE technologies and React JS.
- The main logical coding of genetic algorithm will be done at the end of November. Java 8 will be used for the logical coding.
- The backend will be developed after the successful implementation of the algorithm using PHP and MYSQL.
- Unit testing will be performed on the back end to test its reliability.
- Frontend will be developed at the end of December after the unit testing of the backend.
- Unit testing will be performed on the front end to test its reliability.
- The front end will relate (connect to) to the backend to develop a perfectly functioning application to perform the process of timetable generation by mid-February.

Testing

- The deployment of the software will be performed after the development phase in February.
- The application will undergo different kinds of testing necessary to ensure the system to be very accurate and efficient.
- Any bugs present will be fixed during the testing phase.

Documentation

- The documentation process will take place from mid-February to March until every detail about the application is noted successfully.

CONCLUSION

The process of Timetable generation will be fully automated with this software. This web app can cater to multiple colleges, universities and schools which can rely on it for their Timetable scheduling which earlier had to be done by hand.

Using Genetics Algorithm, several trade-off solutions, in terms of multiple objectives of the problem, could be obtained very easily.

The user input is mapped to the appropriate input of the Genetic Algorithm. Genetic Algorithm will be designed to effectively perform selection, crossover, mutation and fitness calculation. Once desired fitness is achieved, output of Genetic Algorithm is mapped to appropriate User output. Intermediate results will be stored in database for later implementation.

REFERENCES

[1] Lukas, Samuel & Aribowo, Arnold & Muchri, Milyandreana. (2012). Solving Timetable Problem by Genetic Algorithm and Heuristic Search Case Study: Universitas Pelita Harapan Timetable. 10.1109/ICADIWT.2009.5273979.

[2] Soyemi, Jumoke & John Lekan, Akinode & Oloruntoba, Samson. (2017). Electronic Lecture Time-Table Scheduler Using Genetic Algorithm. 10.1109/DASC-PICom-DataCom-CyberSciTec.2017.124.

[3] B. Sigl, M. Golub and V. Mornar, "Solving timetable scheduling problem using genetic algorithms," Proceedings of the 25th International Conference on Information Technology Interfaces, 2003. ITI 2003., Cavtat, Croatia, 2003, pp. 519-524.
doi: 10.1109/ITI.2003.1225396

[4] N. D. Thanh, "Solving Timetabling Problem Using Genetic and Heuristic Algorithms," Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007), Qingdao, 2007, pp. 472-477.doi: 10.1109/SNPD.2007.464
URL:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4287899&isnumber=4287802

[5] Manikas, Theodore W. and Cain, James T., "Genetic Algorithms vs. Simulated Annealing: A Comparison of Approaches for Solving the Circuit Partitioning Problem" (1996). Computer Science and Engineering Research. 1.
https://scholar.smu.edu/engineering_compsci_research/1

ACKNOWLEDGEMENT

Apart from the efforts of our team, the success of any project depends largely on the encouragement and guidelines of many others. We would like to take this opportunity to express our gratitude to the people who have been instrumental in the successful progress of this project. We would like to show our greatest appreciation to our project guide Dr. Phiroj Shaikh for his useful comments, remarks and engagement through the learning process of this project. We would also like to thank Dr. Amiya Tripathi and Dr. Supratim Biswas for their encouragement and guidance.

We would also like to express our deepest appreciation to the project panel of DBIT for giving us this opportunity to do this project.

Project Team Members:

1.Olivia D'sa B. E. – (18)

2. Denzil D'souza B. E. – (21)

3. Prayuj Pillai B. E. – (49)