Institute Timetable Scheduler

Bhaven Gore[1], Disha Shirdhankar[2], Giriraj Belanekar[3]

[1]Bhaven Gore, Student of Rajiv Gandhi Institute of Technology, Mumbai, Maharashtra
[2]Disha Shirdhankar, Student of Rajiv Gandhi Institute of Technology, Mumbai, Maharashtra
[3]Giriaj Belanekar, Student of Rajiv Gandhi Institute of Technology, Mumbai, Maharashtra

---

Abstract - *This paper presents a better approach to prepare the timetable of the university rather than using the manual system. The manual system of creating a timetable for the college with a large number of students is very time consuming and usually ends up with various classes clashing either at the same room or with the same teacher having more than one class at a time. To overcome these hectic problems we propose to make an automated scheduler. The system we will develop will take various inputs like details of students, subjects and classrooms depending upon these inputs it will create or generate a possible timetable. Making best utilization of all resources in a way that will best opt any of constraints or college rules. This can be achieved by using Genetic Algorithm.*

Key Words - Scheduler, Timetable, Genetic Algorithm.

1. INTRODUCTION

A manual lecture time-table scheduling demands considerable time and efforts. Although, most of the work done in the administrative field has been digitised, the lecture time table scheduling is still done manually across universities due to its inherent difficulties.

This scheduling problem is based on constraint satisfaction; in which we intend to find a solution which satisfies the given set of constraints. This problem draws its roots to the 'NP-Hard' class of problems in which the computational time required for scheduling tends to grow exponentially as the number of variables increases. Through our project, we reckon a realistic time-table algorithm which is capable of handling hard as well as soft constraints.

2. LIMITATIONS OF EXISTING SYSTEMS

All the Existing methods used were either manual, local search or using algorithms whose time complexity reaches polynomial time.

As mentioned when Timetable generation is done, it should consider the maximum and minimum workload that is in a college. In those cases, timetable generation will become more complex. Also, it is a time consuming process.

We know all institutions or organizations have their own timetable; managing and maintaining these will not be difficult. Considering workload with this scheduling will make it more complex.

3. PROPOSED SYSTEM

Our paper entitled "Institute Timetable scheduler" is meant to generate timetable scheduling processes in colleges or in any other institutions which could minimize the human effort and maximize the efficiency and the timetable is stored in excel sheets.

Main Objectives are :-

- The final system should be able to generate time tables in a completely automated way which will save a lot of time and effort of institute administration.
- To make the timetable system generic so that it can work equally well for different schools, colleges and universities.
- User defined constraints handling.
- Ease of use for users of the system so that he/she can make automatic timetable..
- Focus on optimization of resources i.e teachers, labs and rooms etc
- Provide a facility for everyone to view the timetable.
- Generate multiple useful views from timetable.

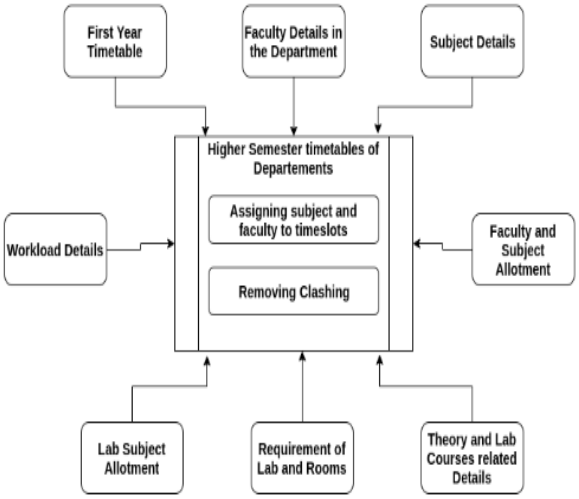


Figure 1: Detailed Design

4. METHODOLOGY / PROCEDURES

Genetic Algorithm -

Genetic Algorithm (GA) is an adaptive heuristic search algorithm that belongs to the larger part of evolutionary algorithms. Genetic algorithm is based on the ideas of natural selection and genetics. It is intelligent in exploitation of random search provided with historical data to direct the search into the region of better performance in solution space. It is commonly used to generate high-quality solutions for optimization problems and search problems.

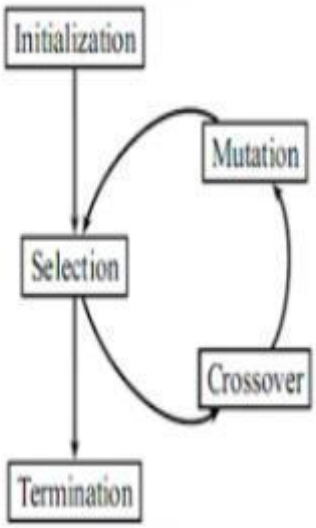


Figure 2: Flow of Genetic Algorithm

Procedure

Algorithm for Genetic:

Step 1: Initialize the Data Set

Example :-

```
rooms = ["R1","R2","R3","R4",]

meeting_times = 35

instructors = [["I1","Dr James Web"],
               ["I2","Mr Mike Brown"],
               ["I3","Dr Steve Day"],
               ["I4","Mrs Jane Doe"],
               ["I5","Mr Bhaven Gore"],
               ["I6","Mrs Bhavika Gore"]]

courses = [["C1","325K",[instructors[0]]],
           ["C2","319K",[instructors[1]]],
           ["C3","462k",[instructors[2]]],
           ["C4","464K",[instructors[4],instructors[5]]],
           ["C5","360C",[instructors[3]]],
           ["C6","303K",[instructors[4],instructors[0]]],
           ["C7","303L",[instructors[5]]]]

depts = [["SEA", [courses[0], courses[2]]],
         ["SEB" , [courses[1], courses[3], courses[4]]],
         ["BE", [courses[5], courses[6]]]]
```

Figure 3: Initializing Data

Step 2: Create Population

This will take these dataset and make a Schedule and a group of 9 Schedule is termed as a Population.

Example :- Here we take 5 sets of schedules for your simplicity.

[[Schedule 1],

[Schedule 2],

[Schedule 3],

[Schedule 4],

[Schedule 5]]

Each Schedule Consists of 70 Classes

[Schedule] >> [[Class 1],[Class 2],[Class 3]..... .....[Class 70]]

Each Class Consists of 3 parameters

[Class] >> [Course Name , Room Number , Instructor List]

Step 3: Calculate Fitness of the population

We then take this whole population and Calculate fitness by taking one Schedule at a time.

Fitness Calculation

Fitness =1 / ((1.0*Number of conflict + 1))

The Conflicts are :-

– No Two Classes should have the same Teacher at the same time.

– No Two Lectures should take place at the same room at the same time.

– Number of hours of a particular lec should be maintained.

If any of the above conflicts occur then the number of conflict parameters increases.

After Calculating fitness we sort it

schedule #	fitness	# of conflicts
0	0.333	2
1	0.25	3
2	0.25	3
3	0.25	3
4	0.167	5
5	0.167	5
6	0.167	5
7	0.143	6
8	0.143	6

Figure 4: Fitness

If Fitness is Equal to 1 i.e Number of Conflicts is Equal to 0

Then,

Print the Table Generated

Else

– 3.1 Crossover the Population

In Crossover we take 2 schedules at a time from the population and perform crossover operation.

Example :-

Schedule 1 > [[Class 1],[Class 2],[Class 3] … …..[Class 70]]

Schedule 2 > [[Class 1],[Class 2],[Class 3]…… …..[Class 70]]

We then set crossover rate at **0.5**

Using this we form a new schedule

We generate a random number and compare it with the crossover rate.

For all 70 Classes

If rand > rate then,

new Schedule of that class > Schedule 1 of that class

Else

new Schedule of that class > Schedule 2 of that class.

– 3.2 Mutate the Crossover Population

In mutation we take the crossover Schedules one at a time and perform Mutation

Example :-

First we Initialize a new Schedule then we take the crossover Schedule

Crossover Schedule 1 > [[Class1] , [Class2] , [Class3] . . . . . [Class 70]]

We set Mutation rate at **0.01**

We generate a random number and compare it with the crossover rate.

For all 70 classes

If rand > 0.01 then,

Crossover Schedule of that class > new Schedule of that class

– 3.3 Thus after Mutation we got the new Evolved Population.

- **Step 4: Go to step 3**
- **Step 5: END**

5. SIMULATIONS AND EXPERIMENTAL RESULTS

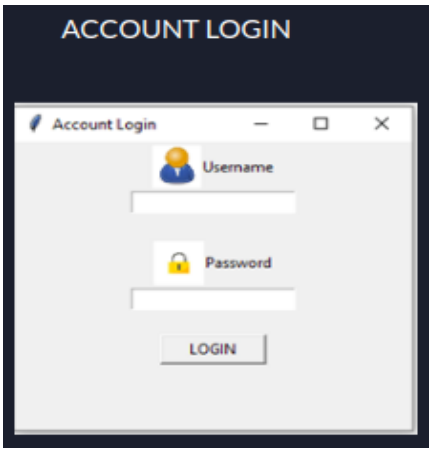


Figure 5: Account Login

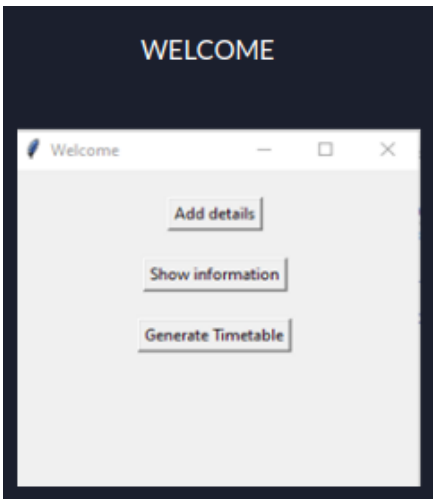


Figure 6: Welcome Page

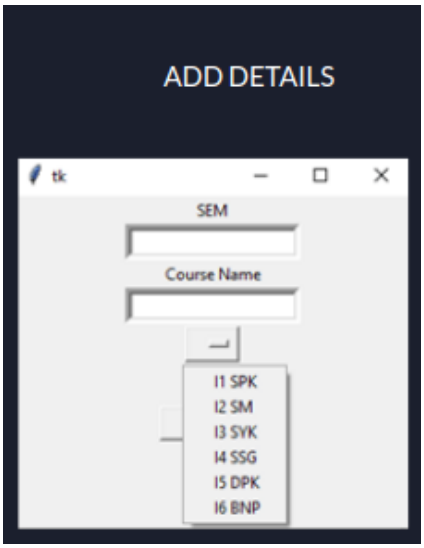


Figure 7: Add Details Page

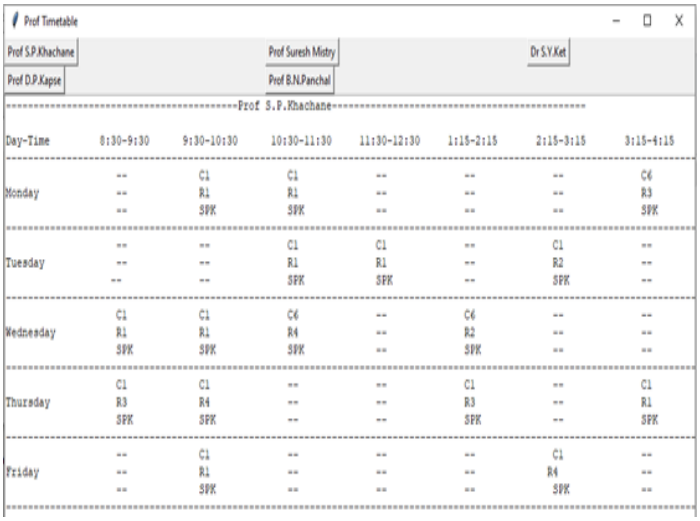


Figure 8: Faculty Timetable

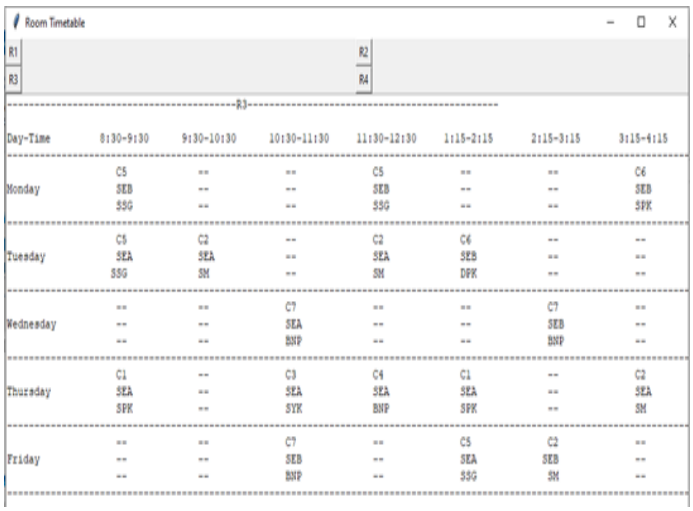
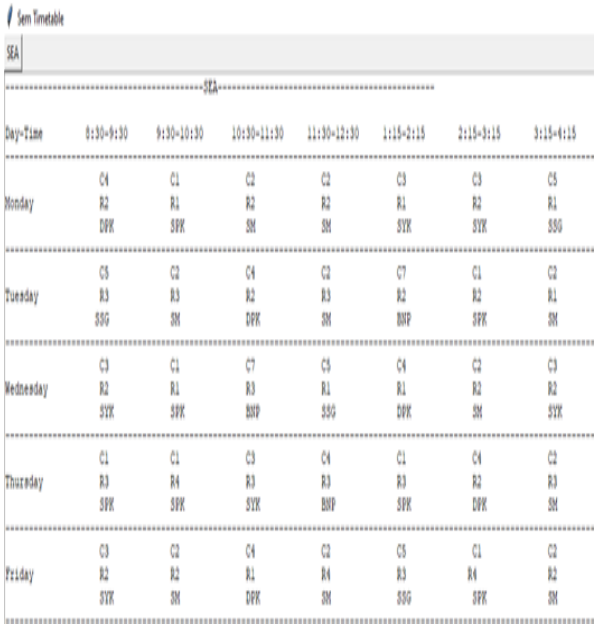


Figure 9: Room Timetable


Sem Timetable

SEA

SEA

Day-Time	8:30-9:30	9:30-10:30	10:30-11:30	11:30-12:30	1:15-2:15	2:15-3:15	3:15-4:15
Monday	C4 R2 DPK	C1 R1 SPK	C2 R2 SM	C2 R2 SM	C3 R1 SYK	C3 R2 SYK	C5 R1 SSG
Tuesday	C5 R3 SSG	C2 R3 SM	C4 R2 DPK	C2 R3 SM	C7 R2 BNP	C1 R2 SPK	C2 R1 SM
Wednesday	C3 R2 SYK	C1 R1 SPK	C7 R3 BNP	C5 R1 SSG	C4 R1 DPK	C2 R2 SM	C3 R2 SYK
Thursday	C1 R3 SPK	C1 R4 SPK	C3 R3 SYK	C4 R3 BNP	C1 R3 SPK	C4 R2 DPK	C2 R3 SM
Friday	C3 R2 SYK	C2 R2 SM	C4 R1 DPK	C2 R4 SM	C5 R3 SSG	C1 R4 SPK	C2 R2 SM

Figure 10: Class Timetable

CONCLUSION

It is an intricate task to organize and manage the schedule of the entire faculty throughout the university and also assign specific subjects to them as per their expertise which fulfills the requirement of the students. Our proposed system will help to overcome the impediment faced because of physical means. Our system has the potential to generate a time-table for any number of courses across departments/sections in a university In the broader aspect, this can also be put to use by various commercial organizations too.

REFERENCES

[1] Dipti Srinivasan, Tian Hou Seow, Jian Xin Xu,Automated timetable generation using multiple context reasoning for university models, 2002 IEEE conference.

[2] Anuja Chowdhary, Priyanka Kakde, Shruti Dhoke, Sonali Ingle, Rupal Rushiya, Dinesh Gawande TIMETABLE GENERATION SYSTEM, IJCSMC Vol. 3, Issue. 2, February 2014.

[3] Saritha M, Pranav Kiran Vaze, Pradeep, Mahesh N R Automatic Time Table Generator, Volume 7, Issue 5, May2017 ISSN: 2277 128X International Journal of Advanced Research in ComputerScience and Software Engineering.

[4] S. Abdullah, E. K. Burke and B. McCollum,'A Hybrid Evolutionary Approach to the University Course Timetabling Problem,Proceedings of the IEEE Congress Evolutionary Computation, Singapore,(2007).

[5] Lahoti, Y., Aaditya Punekar, H. P. (2015),'Automated Timetable Generator, International Journal of Science and Research, 4.

[6] https://www.researchgate.net/publication/326265336 A STUDY ON AUTO-MATIC TIMETABLE GENERATOR.