University Timetabling using Genetic Algorithm

Anandaraj Soundarya Raja Murugan

2009

Master Thesis

Computer Engineering

Nr: E3729D
DEGREE PROJECT
Computer Engineering

Program
Master in Science in Computer Engineering

Reg. Number
Nr: E3729D

Extent
15 ECTS

Name
Anandaraj soundarya Raja Murugan

Year Month Day
2009-03-04

Supervisor
Examiner

Dr. Pascal Rebreyend
Prof. Mark Dougherty

Company/Department
Department of Computer Engineering, Dalarna University

Title
University Time Tabling using Genetic Algorithm
Note:

This thesis was started by two students, and later our supervisor changed their directions, therefore content similarities can be considered as a combine work.
Abstract

The field of automated timetabling and scheduling meeting all the requirements that we call constraints is always difficult task and already proved as NP Complete. The idea behind my research is to implement Genetic Algorithm on general scheduling problem under predefined constraints and check the validity of results, and then I will explain the possible usage of other approaches like expert systems, direct heuristics, network flows, simulated annealing and some other approaches. It is observed that Genetic Algorithm is good solution technique for solving such problems. The program written in C++ and analysis is done with using various tools explained in details later.
Acknowledgement

I would like to acknowledge the endless support of Dr. Pascal Rebreyed in formulating the idea and shaping it into practical form which ultimately helped at each stage to better understand the topic. He provided the best supervision in entire work. He was always available to discuss and clear doubts any time. He is a man of patience.

Also I cannot forget the motivation provided by Dr. Siril Yella for his endless support and valuable comments. His unique way of explaining makes the complex things simple to understand. He is the key of Motivation. I feel pleasure to thank him.

I am also extremely grateful to Prof. Mark Dougherty and Dr. Hassan Fleyeh for their support towards my work. A special thanks to my Dear Parents, class members and my friends for their guidance, patience and valuable advices.
Table of Contents

Abstract .............................................................................................................. 4
Acknowledgements ........................................................................................... 5
Table of contents .............................................................................................. 6
1: Introduction ................................................................................................. 8
2: Scheduling ................................................................................................. 9
3: Genetic Algorithm ...................................................................................... 11
   Overview of Genetic Algorithm:
   Population .................................................................................................. 14
   Chromosome Encoding ............................................................................. 15
   Phenotype and Genotype .......................................................................... 15
   Chromosome evaluation ........................................................................... 15
   Crossover operator .................................................................................... 16
   Mutation operator ..................................................................................... 18
   Inversion operator .................................................................................... 19
   Other GA Operators .................................................................................. 21
   Memetic operator ..................................................................................... 21
   Lamarckian operator ................................................................................ 21
   Final stage in GA ...................................................................................... 21
   Merits of Genetic Algorithm ..................................................................... 22
4: Benchmarks ............................................................................................... 23
   Benchmark-1 ............................................................................................ 23
   Benchmark-2 ............................................................................................ 25
5: Constraints in automated timetabling ......................................................... 28
   Critical constraints ...................................................................................... 28
   Soft constraints ............................................................................................ 29
   Other timetabling applications .................................................................... 30

6: Review of other approaches using Timetabling ........................................... 31
   Tabu search .................................................................................................. 31
   Constraint Logic Programming approach ................................................... 32
   Simulated Annealing .................................................................................... 33
   Particle Swarm Optimization ......................................................................... 34

7: Test Analysis .............................................................................................. 35
   T1-Class Cluster Analysis ........................................................................... 35
   T2-Class Cluster Error Analysis .................................................................. 36
   T3-Overbook Error Analysis ......................................................................... 38

8: Implementation Strategy ............................................................................ 39

9: Motivation of Genetic Algorithm .............................................................. 40

10: Conclusion and Future Work ................................................................. 41

11: Reference lists .......................................................................................... 42
1: Introduction

Manual timetable management is a setup of inexpensive and free advertising for concerned authorities. Today science and technology pays a vital role in every operation switching towards automation. Reality can be classified into two categories. Some are deterministic and many are non-deterministic solutions. There are many problems fall in second category which has no solution for all possibilities. It occurs due to wide search spaces and handling such search spaces with respect to certain circumstances.

It is quite agreed to say that no generic solution is possible till now. University time tabling is one of these kinds, because of different constraints varying in classes, session and so on. Here, the usage of Genetic Algorithm in University timetabling is done to get the possible optimal solution. Even though it is proven that such problems are NP-Complete problems [1], still we can get a better result.

This project consists of three steps. Firstly, I will explain the method of Genetic Algorithm, its operator usage with respect to their effect. Secondly is to bring the Genetic Algorithm in application and generating different schedules with their respective effectiveness. Finally, we will give some highlights of using different approaches to deal the same problem. It pays the new way for the upcoming students to work with different approaches and concepts to get better results. The problems of Timetabling were discussed in details by many researchers; details can be found [2].
2: Scheduling

“Scheduling is a key concept in multitasking and multiprocessing operating system design and in real-time operating system design. It refers to the way, the processes are assigned priorities in a priority queue which is the process of deciding how to commit resources between varieties of possible tasks.

Scheduling Timetabling is quite complex and is mainly based on the constraints and the instances. But in case of limited resources, tight time and cost restrictions it is very hard to formulate schedules.

In university timetable scheduling, the constraints are classified on the basis of certain rules. There are two different types of constraints involved. They are Hard (critical) and soft constraints. The proven NP-completeness property leads the solution to be satisfactory. Our ultimate goal is to satisfy hard constraints and then soft constraints. Soft constraints can be seen as preferential constraint whose satisfaction is not required but can be preferred [8]. It is extremely hard to handle soft constraints [5], despite the fact that we limit the scope of soft constraints but their violation affects the efficiency of the solution.

We have taken Högskolan Dalarna as our case study. It has two campuses and in both the campuses the timetabling are managed manually and students through web application name “Time Edit” is done.
Automated timetabling is special class of scheduling because no general solution can be provided. Different approaches are available to sort out this problem like Genetic Algorithm [4], Tabu Search, Evolutionary Algorithm and Artificial Intelligent [9] and Particle Swarm Optimization and simulated annealing.

Although there are lot of software’s available for scheduling [5] in market still there exists a problem in the generality to make these tools flexible to the demands of different universities and institutions. The universities and companies are not getting the appropriate algorithms for their needs. This is one of the biggest problems in different concerns.
3: Genetic Algorithm

Genetic algorithms were introduced as computational analogy of adaptive systems. It is used for problem solving and for modeling. They can learn high performance knowledge structures. Before 40 years the concepts of evolutionary algorithms were evolved and founder of these approaches is J.H. Holland [HOL75], USA described in his book “Adaptation in natural and artificial systems” and De Jong’s “Adaptation of the behavior of a class of genetic adaptive systems” published in 1975. In fact, there are many ways to view genetic algorithms and most of the users are looking for a problem solver, but it is a restrictive view [10].

Genetic Algorithm is a particular class of Evolutionary algorithms that uses techniques of evolutionary biology such as mutation, crossover, selection and inheritance. In Holland’s method he classified objects, selected breeding with these objects to produce new one.

This methodology is same as modeling Darwinian natural system following the simple natural pattern of growth and Reproduction. A genetic evolutionary growth can be described as follows:
Create a population of objects (creatures)  //Initialization
Evaluate the fitness of each object //analyzing
While the population is not fit enough //Fitness check
{
    Delete all unfit objects //Removing unfit objects
    While population size < max: //size check
    {
        Select two best populations
        Create new objects
        Random mutations
        Evaluate and place in population //breeding
    }
}

**Figure-1 Evolutionary growth**

The overall view of the concept is same as the above. The genetic algorithm acts upon genes chromosome. Every object or creature has its own chromosome. Chromosome is a set of parameters which define a proposed solution to the problem that genetic algorithm is trying to solve.
The process of GA like Initializing, breeding, mutating, choosing and removing and unfit creatures are clearly shown in the below figure. Although the improvements in GA has changed its way as compared to recommended by Holland but basic idea is still the same.

Figure-2  The basic Genetic Algorithm [6]
Population:

The population is initiated as random assignment but should be legal one that it is satisfying the constraints and limitations of the problem under discussion. Unfortunately, there is no particular methodology to decide about the size of population but one thing is agreed that, it is all dependent upon the problem. Search spaces are also important phenomena to decide about the size of the population but according to some researchers for a regular search space population of 40 to 100 and for larger more complex problems and irregular search space larger populations of 400 or more than that are recommended.

The other factor in deciding the size is convergence time, which is for the small size convergence, might be quicker and vice versa, but it all depends on payoff that decides between convergence time and exploration of search space.

Initializing can be done using one of two techniques. One is already established benchmarks with stored data and loading from secondary storage. Then generating random solution and this method is used by most GA. [15]

Other approach is gene by gene comparison by single offspring called overcrowding strategy with continuous replacement of parents [11]

The approaches are almost same as recommended by Holland concept that the healthiest should be allowed to breed so that more required creatures can be produced [11]. The reason of having same traits in all of the approaches is the
main objective which is to obtain highest possible fitness value chromosomes [11].

**Chromosomes Encoding:**

In genetic algorithm we can encode chromosome as a bit string and use string operators to perform operations like cross over and mutation. To provide simple, elegant and effective flow of Gas, Holland used string of binary digits to encode chromosomes. But it does not mean that no other ways are available to encode chromosomes, rather many other schemes are recommended along with their respective advantages [15].

**Phenotype and Genotype:**

In GA, the encoded objects are called genotype and actual object is called phenotype [11], anything can be encoded because different approaches are available in literature. As the physical structures begin to act and interact with one another they produce large and more phenomena such as metabolism. The only difference is storage of these genes, that in GA genes are not stored in pairs while in natural process are pairs representing their parent’s participation [15].

**Chromosome Evaluation:**

The randomly generated population are extremely unfit most of times[15].Therefore evaluating the fitness is most critical part of Genetic Algorithm or other approaches following same fashion like Particle Swarm Optimization[7]. This evaluation is done on the basis of some predefined criteria.
and for that purpose we need to have information about the environment. Multiple fitness evaluation is also possible, although we are not using this approach in our work but some multiple fitness criteria can be introduced [20].

The fitness evaluation will provide us with the fit (according to our predefined criteria) population. The most discussed criteria are the cost. Cost is not the money here but according to Gen and Cheng cost is a unit of efficiency [11]. Both types of cost either minimum or maximum according to the requirement of the problem can be set.

**Crossover Operator:**

Crossover operation is a genetic operator that mates two chromosomes to produce a new chromosome. The idea is the new chromosome may be better than both of the parents if it takes the best characteristics from each of the parents. Once the best parents are selected then breeding process will take place to improve the average fitness of whole population. In fact these best individuals will mate and form the next generation having more good traits than their parents. Crossover occurs during the evolution according to user definable crossover probability [12].

The genetics theory states that, if the parents are stronger in terms of fitness then the offspring will be fitter. So, this theory can lead us a better solution. To maintain randomness, I generated the random number and then follow the process according to that random number. Cross over emulates the exchange of chromosomes to generate more fitter new generation.
After selecting parents, we generate a random number between 0 and 1 and compared with the cross over probability (usually it is taken 0.7). If the random generated value is less than crossover probability, the crossover takes place otherwise parents are simply placed into the next generation. This process can be seen in following figure.

**Figure-3: Crossover [6]**

There are certain methods available in GA library for cross over operation. They are one point, two point, uniform, arithmetic and heuristic [11]. Heitkoetter recommended to use unity order based crossover in which each gene has equal probability of coming up from either parent, so everything is equally likely.[13]
Mutation:

Mutation is a genetic operator that alters one or more gene values in a chromosome from its initial stage. It results in a new gene value added to the gene pool. With the help of this value, the genetic algorithm will get a better Solution than previous one. The importance of mutation is that it won’t inactivate the population at any local optima.

The major purpose is to introduce or inject noise and new alleles in the population. According to Gen and Cheng it is useful in escaping local minima as it helps to explore new regions of the multi dimensional solution space. [11]. Mutation is not an integral part of Genetic Algorithm but there are some systems in which mutation is not used, rather they prefer to use some noisy random populations at initial stage, which make effective search [14].

There are many types of mutation available. They are Flip bit, Boundary, Non uniform and Gaussian. [14]. these methods vary with binary and non binary representation. In binary representation, the conventional mutation of 0 to 1 or vice versa can be helpful but for non binary.

The result of mutation operator cannot guarantee that it will provide more strong solution, but it is believed that mutation is doing something new by changing some part of chromosome. But if it does not happened then it cannot cause negative to the problem because only fitter wins to stay otherwise it will die.
The following figure can represent the process of mutation:

![Figure-4 Mutation](image)

**Inversion:**

In inversion, the process is simple that detaching the portion of chromosomes and after changing the direction attaching it again with the chromosome. Generally due to its difficulty level at implementation level, this method is not used while using GA [15]. But according to many known researchers in the field of GA this can play an important role in future [15][11]. The below figure represents the process of inversion.
Chapter 4: Benchmarks and programming logic

Chromosome

Before Inversion:

001001011010100101011010010100101010001010

During Inversion: 001001011010100101011010010100101010001010
00101010001010
01010110100101

One portion inverts: becomes
(Order is reversed) 10100101101010

Recombination: 001001011010101001011010010100101010001010
00101010001010

After Inversion:

001001011010101010010110101000101010001010
Other GA operators:

Gen and Cheng also suggested some other operators like Lamarckian operator and Memetic Algorithm and Repair strategies.

Memetic Algorithm:
Memetic algorithm is also known as hybrid genetic algorithm. It is a new method where local search is applied during the evolutionary cycle. In some problem fields, they are found more efficient than traditional evolutionary algorithms. The idea of this algorithm comes from memes [16].

Lamarckian Algorithm:
In Lamarckian concept of evolution, the characteristics that are acquired by an individual in the course of its life time can be passed on its offspring.

Final Stage in Genetic Algorithm:
Once the stages of genetic algorithm gets over, the fitness is checked like as did in first step of initialization.

But still no deterministic solution is there that how many times GA should run for. Simple problems may converge on good solutions after only 20 or 30 generations. More complex problems may need more. It is not unusual to run a GA for 400 generations for more complex problems such as job shops. The above figure suggests 100 generations. The most reliable method of deciding on
this is trial and error, although numbers of authors have suggested methods for determining how long a solution should live. [3]

**Merits of Genetic Algorithm:**

It has been shown that the genetic algorithm perform better in finding areas of interest even in a complex real world scene. Although GA is entitled for many listed applications but it can be used for any function involving minimizing or maximizing function [15], but GA proved good for pipe network optimization problems [21]. A Genetic algorithm approach to multiple Fault diagnosis by Gunar. E Liepens and W.D potter also provides more credits.

A hybrid Technique for engineering Design optimization by Gilbert Syswerda, Transportations problems [11], conformational analysis of DNA [15], image processing and machine learning [23] and at last but not least, the problem which we are working on scheduling [4], [23].
4: Benchmarks

Guide lines for Benchmarks:

- There are three benchmarks.
  1. Rooms
  2. Time Slots.
- When inputs are given to the benchmarks, it produces benchmarks in the form of text files.
- The data’s are stored in the corresponding benchmark.
- The algorithm for Genetic Algorithm loads the data while making the schedule, after comparing with constraints to avoid any type of violation.

Benchmark-1:

Input:
Number of rooms.

Output:
Benchmark texts file in following format (sample format)
We have to set the capacity for each room using following standard:

<table>
<thead>
<tr>
<th>Percentage of rooms</th>
<th>Actual size</th>
<th>Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>N</td>
<td>30</td>
</tr>
<tr>
<td>20%</td>
<td>N</td>
<td>40</td>
</tr>
<tr>
<td>10%</td>
<td>N</td>
<td>50</td>
</tr>
<tr>
<td>10%</td>
<td>N</td>
<td>60</td>
</tr>
<tr>
<td>10%</td>
<td>N</td>
<td>60+</td>
</tr>
</tbody>
</table>

Where N is input of total number of rooms to be scheduled.

**Figure-7 Room capacity criteria**
Explanation:

1. The above criterion is set only for having schedule, for future purpose slight changes in parameters setting, according to the requirements.

2. Assign the room capacities randomly to each room using following standard:
   - 60% of n should be lecture rooms including all type of capacities.
   - 30% of n should be computer labs.
   - 10% of n should be other labs. (Like Electrical or mechanical labs etc)

3. For Room ID: Room ID reflects very important information, as suggested in this work; it should be 4 places ID. First place should be alphabet and rest of the places should be integers.

   For example if we have T 355, it means it is lab on third floor and number is 55, hope it is giving fair idea why we want to keep id like this.

Benchmark-2:

Input:

1. Number of Groups
2. How many time slots a group need in a week
3. How many courses can a professor can teach.
Output:
A well designed benchmark with filled data in following format:

<table>
<thead>
<tr>
<th>Time slot</th>
<th>Teacher</th>
<th>Group of student</th>
<th>Room Requested</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>T1</td>
<td>A</td>
<td>LC</td>
</tr>
<tr>
<td>2</td>
<td>T2</td>
<td>B</td>
<td>LBC</td>
</tr>
<tr>
<td>3</td>
<td>T3</td>
<td>DK</td>
<td>LBC</td>
</tr>
<tr>
<td>4</td>
<td>T4</td>
<td>HK</td>
<td>LC</td>
</tr>
<tr>
<td>5</td>
<td>T5</td>
<td>KL</td>
<td>LBO</td>
</tr>
</tbody>
</table>

... ... F LC
... ... ... LC
... ... ... ... ....
... ... ... ... ....

Figure-8 Sample benchmark for time slots

Explanations:
1. Dimensions of above matrix will be set after taking input which is how many groups we have to schedule, and how many time slots a group need.
2. Students group means, BBA-1, MSC-2, B.S Physic and etc.
3. Time slots means, there are 4 time slots are there in a day, as (8-10, 10-12) AM and (1-3, 3-5) PM. and we are designing schedule for weekly basis, so there are (5*4)=20 time slots are available in a week for each room.
**Example:** Suppose there are 5 groups of student are available and each group needs 6 time slots per week, it means we have to schedule 30 time slots in a way that no constraint is violated (Constraints are explained in next section).

**Pseudo Code:**

- Take input of number of groups and number of slots each group need.

- Assign each time slot to groups in a way they should not increase their maximum required time slots.

- Assign teacher to group, but in a way that each professor should not exceed their maximum limit specified as input.

- Out of slots reserved for a group, two third of the slots should be in lecture room (LC), and rest in laboratories. For example if a group needs 6 time slots the 4 should be lecture and 2 should in labs.

- End
4. Constraints in Automated Timetabling

The schedule which we are looking for should be essentially consistent with the constraints set and required for the schedule. There are some universal constraints set by the people dealing with that schedule [4]. According to Burke and Ross, it can be divided into two categories: Soft and hard constraints.

**Critical Constraints:**

Critical constraints (CC) are the backbone of the system devised. Any violation in CC can fail the schedule. For example, one professor cannot appear in two time slots [20].

<table>
<thead>
<tr>
<th>Critical Constraints:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC1: No professor appears more than once in one time slot.</td>
</tr>
<tr>
<td>CC2: No room appears more than once in one time slot.</td>
</tr>
<tr>
<td>CC3: Room allocation should be done after evaluating the size* and Kind** of class.</td>
</tr>
<tr>
<td>CC4: No professor can appear in two simultaneous time slots.</td>
</tr>
<tr>
<td>CC5: One course cannot appear in two time slots in the same day.</td>
</tr>
<tr>
<td>CC6: Multiple class subjects should be allocated to the same time slot.</td>
</tr>
</tbody>
</table>

*Size: Size means seating capacity, it should be more than registered students to avoid the problem in case of any guest student(s).

**Kind: Kind of class means whether it is lectures type or laboratory.
Soft Constraints:

Most of the times soft constraints are set according to the preferences of the professor and sometimes with respect to the class preference. Different researchers presented different types of soft constraints and it is according to their respective research domains.

```
Soft Constraints:

SC1: A professor may not prefer to teach morning or evening session.
SC2: A professor may not prefer to teach more than certain number of Credit hours.
SC3: A professor can refuse to teach certain class.
SC4: Lectures should evenly be distributed during week.
```

Figure-7 Soft Constraints

Timings can be divided into two categories:

First two time slots (8-10, 10-12) AM------------------ Morning session
Last two time slots (1-3, 3-5) PM--------------------- Evening session.
Other Timetabling applications:

Burke and Ross used the same approach for examination timetable, explained in this work and it is believed having almost same nature as timetabling has [4]. Minor changes are required to make this approach available for exam timetabling like avoiding two exams in one day and two exams on two consecutive day [17].

Exam schedule facilitates in some areas are number of students, timing constraints and bit complicated in a sense that we have to consider each and every student on individual basis [17] because students study courses are according to their own preferences, so this is the trickiest part of exams scheduling.

Another application is scheduling for transportation system in large corporation, where optimization concerns are minimizing shipping time and cost, while maximizing client’s satisfaction and improving services. Multi-level job-scheduling of Operating Systems can be tackled with the same approach and intricate digital circuits [15].
6: Review of other approaches used for Timetabling

1. Tabu search:

A Meta heuristics approach was developed considering strong approach to solve large and difficult combinatorial optimization problems (Ferland et al. 2000, Gendreau et al. 1994). As timetabling is difficult combinatorial optimization problem, this approach applied extensively to solve and make schedules. Although having its own merits and demerits [22], “Tabu search is an intelligent search technique that uses a memory function in order to avoid being trapped at a local minimum and hierarchically canalizes one or more local search procedures in order to search quickly the local optimality”.

To improve the efficiency of the exploration process, some historical information related to the evolution of the search is kept. Such information will be used to guide the movement from one solution to the next one avoiding cycling.

Merits:

The Application of Tabu Search requires the definition of the following.

1. Feasible solution

It is not defined based on the hard constraints because it does not guarantee that the corresponding search space is connected with respect to neighbor relation.
2. The neighbor relation

The neighborhoods of a solution consist of all timetables that can be obtained by assigning a lecture to a different period.

3. The objective function to minimize

It is the weighted sum of the number of teacher and students conflicts for each period.

Demerits:

The work in Tabu search contains the optimization problem with the grouping option, whereas extends (Hertz, 1992) the approach to a more complex case which takes into account also lectures of different length.

2. Constraint Logic programming approach

CLP is widely and successfully used for finite domains and solving large combinatorial problem. A timetabling is same kind of problem having number of resources (Rooms and Teachers) with certain constraints to Schedule in a way to get the optimum or near to optimum solution by making maximum utilization of resources possible. Declarative handling of constraints (both hard and soft) is real beauty of CLP. As Timetabling is proved to be NP complete, so although number of software are available in market still due to the variety of constraints
in certain situations make it inflexible, and this ultimately make it challenging field of research.

**Merits:**

As mentioned above the declarative representation of all the constraints is the beauty of this approach.

**Demerits:**

The full back tracking capability of the prolog machine is overridden by a heuristics that allows only for a limited attempt to reschedule assignments that create conflicts.

3. **Simulated annealing:**

This approach is used for examination timetabling [18]. The presentation can be repeated more than once if it is necessary. Here, the number of rooms is fixed and all the rooms are used for all the periods.

**Merits:**

It has an ability to provide good solution for many combinatorial problems and relative case of implementation. It is relatively easy to code even for complex problems. It can also deal with arbitrary and cost function.

**Demerits:**

The main drawback of this method is that it cannot tell whether it has found an optimal solution. It requires other methods like branch and bound to do this.
4. Particle Swarm Optimization:

Particle Swarm optimization is a population based stochastic optimization technique by social behavior of bird flocking or fish schooling. It has no operators like cross over and mutation. Here, each particle keeps track of its Coordinates in the problem spaces that are associated to the best solution. This value is called pbest.

Merits:
It can be used for wide range of applications focused on specific requirement. There are only few parameters to adjust. It is faster and cheap way compared with other methods.

Demerits:
Once Particle swarm optimization gets into the local optimization it is very hard to jump out from local optimization.
7. TEST ANALYSIS:

The program is implemented to check the efficiency of the Genetic Algorithm with different constraints at different status. The tests which we tried are

- T1- Class Cluster Analysis.
- T2- Class Cluster Error Analysis.
- T3- Overbook Error.

T1-Class Cluster Analysis:
In Test 1, the colloidal of the classes are mainly focused because in allocation the time slots to the groups, finding suitable time slot is really difficult task particularly when you have to satisfy hard constraints. In this test, the related class cluster errors are removed so after adjustments in the algorithm the mechanism was established to reduce or remove this problem, and after number of generations, my program reached this target. After each generation, the class cluster errors are recorded one by one.

Finally, the average numbers of generations are calculated from random generations. Here, we can notice that when the performance of the generations becomes more, the average number of related class clusters becomes less.
Therefore, the errors are reduced only when the generations are frequently repeated.

**T-2 Class Cluster Error Analysis:**
In Test 2, the class cluster errors are mainly focused. Here, the errors which occurred when the class gets clashed are recorded. It is then generated again and again to get the average better result. Here, when the number of generations increases, then the average number of related class cluster error decreases. The average value is taken from the repeated generations to get the better result. The possible objection on this test would be the increase in both time and space.
complexity, but to achieve good result and reach the required target, it was necessary to increase generation to tune the results.

Class Cluster Error analysis

![Graph showing the decrease in average number of class cluster errors over generations.](image-url)
T-3 Overbook Errors:

In test 3, this was the most important test in my analysis, in which number of students and the seating capacity was observed, while allocation a certain room to some group. Another view of this test indicates the repetition of some certain room again and again, because of less room’s availability with respect to the certain size limitation. The rooms which are booked repeatedly are mainly focused. When it is generated it removes the rooms which are booked twice. The generation is repeated more to get good results. Here, when the generations are increased, the average numbers of staff over booked errors are decreased. Finally the average value is selected from the random generations.
8. Implementation Strategy

The design and the development of different products occur in every field regularly. But replacing the current manual system with the automated system values a lot. There are different approaches and schemes devised for this purpose (UPID MIS Study V1.0). But I would like to be very specific with respect to the proposed automated timetabling solution.

The main activities for implementation any newly devised automated system can be listed as follows:

- Planning the activities.
- Task assignments, time required parallel or independent implementation.
- Planning the activities and organizing the personnel.
- Acquiring the facilities required for implementation.
- Developing step by step procedures proper implementation regarding installation and testing.
- Developing the training program for operating personnel
- Because the administrative staff cannot handle the technical stuff, so they need some training to better understand the system and its functionality.
- Proposed timetabling solution.
- The solution of the thesis proposed is not complete to implement as it needs two more things.
9. Motivation for Genetic Algorithm

- The credit of the genetic algorithm is that it provides multiple solutions after generating initial solution on the basis of population with the restricted fitness value. It provides continuous improvement of the solution.

- When it comes to highly complex problems like timetabling or job shop, Genetic Algorithm provides robust search techniques.

- The Fitness Value decides the fit solution for the analysis.

- Genetic Algorithm provides an optimal solution, but time taken to implement this not very fast.
10: Conclusion and Future Work

In this paper, we have seen the concepts and analysis of the genetic algorithm based scheduling that takes important design decisions to obtain the efficient implementations. The main advantage of this approach is that it allows choosing an appropriate implementation from a set of functions and convenient data distributions.

In future work, we can continue the process of establishing Robotic system through evolution and it is also mandatory to know that how multiple behaviors are required for higher level task interact. The solution of the thesis can be developed with the Examination scheduling.

It can open some project for students who can build Robotic system and design examination scheduling system making an expert system to solve complete scheduling problem This work will provide the best feature and will be a useful work for the university.
11. References


[8] Roman Barták, Charles University in Prague, Faculty of Mathematics and Physics .Malostranské námestí 2/25, 118 00, Praha 1, Czech Republic.


Available from the Sante Fe Institute on the WWW at http://alife.santafe.edu/~joke/encore/www/


Ferland, J.A., Berrada, I., Nabli, I., Ahiod, B. Michelon, P., Gascon,