TIME TABLE SCHEDULING USING GENITIC ALGORITHM

Apurva T Kanavade[1], Akshata D Kshirsagar[2], Sonali N Kokane[3], Minal S Kulkarni[4]

Dr. A. Kalyan Kumar[5]

[1] *BE Computer, SRES' COE Kopargaon, SPPU, Maharashtra, India*
[2] *BE Computer, SRES' COE Kopargaon, SPPU, Maharashtra, India*
[3] *BE Computer, SRES' COE Kopargaon, SPPU, Maharashtra, India*
[4] *BE Computer, SRES' COE Kopargaon, SPPU, Maharashtra, India*
[5]*P.hd Computer, SRES' COE Kopargaon, SPPU, Maharashtra, India*

ABSTRACT

Scheduling is one of the major matters of concern and research, as to allocate maximum resources efficiently is a rigid task to perform. Time Table Scheduling is a category of Scheduling in which the mission is to generate a formatted schedule for particular organization. This paper deals with time tab le scheduling problem by illustrating genetic algorithm as a solution to the problem. Analysis of genetic algorithm was done on small and large instances of problem. Excellence of algorithm was appreciably enhanced with alteration of basic genetic operator, which pins down foundation of novel conflicts in individual and hence supports for most issue of study.

.

Keyword: *- scheduling timetable, Genetic algorithm, 3dimension illustration*.

1. INTRODUCTION

This Piece of writing explains an accomplishment of genetic algorithm for time table scheduling issue. Ample of educational institutes frequently are concerned about this Objective of algorithm is to diminish the amount of conflicts in time table. Decrease in Encoding of search space was also observed after implementation. The algorithm was experienced on miniature and huge timetable cases at Faculty of Electrical Engineering and Computing (FER) in Zagreb [1]. The program interface is prepared in java. The Center part of genetic algorithm was built in C++ By means of STL (Standard Template Library) support.

2. LITERATURE SURVEY

- **Existing system**
  Successfully implemented first in "Devi Ahilya University, Indore" for one of the MBA college.

- **Hard constraints involved in existing system**
  No participant (lecturer or class) can be in more than two rooms at the same period.
  No room should be double booked.
  The room capacity should be large enough to hold each

3. EXISTING SYSTEM LIMITATION

The existing System has a few drawbacks they are as follows:

1. The system at present does not take care of other constraints like unavailability of lecturers.
2. Small size of rooms.

4. SYSTEM OVERVIEW

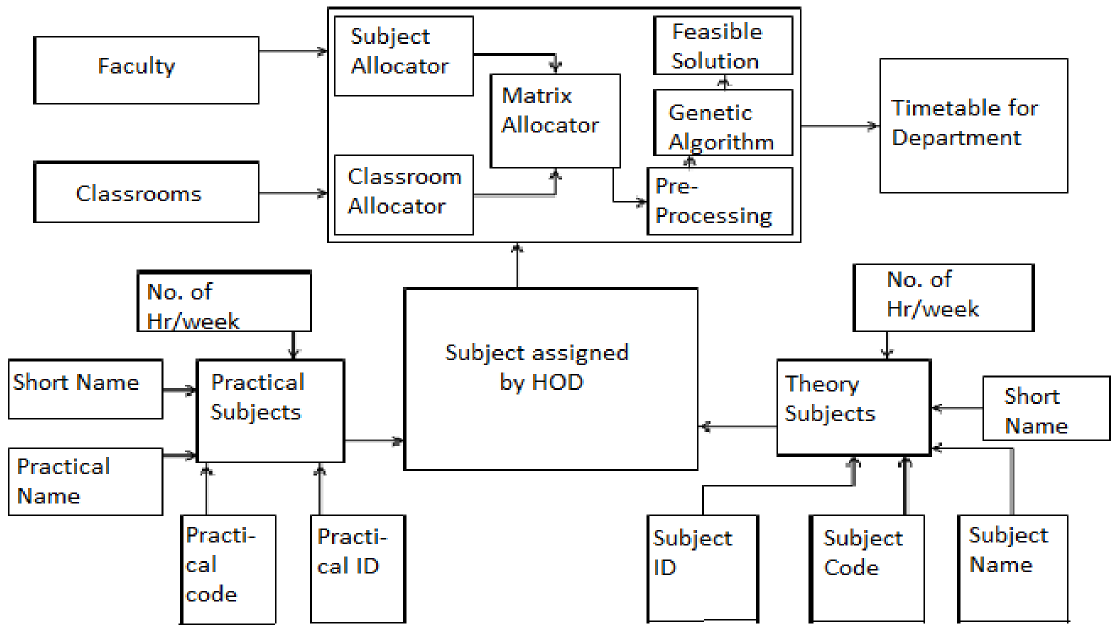

Fig -1 : System Overview

The proposed systemis stated in Fig -1. The input of system will be a record (Triplet) consisting information of already paired class group, subject and subject teacher. This record will go through the matrix allocator which will randomly allocate the record to any position in the matrix. After construction of one complete matrix fitness value is checked for it. To improve output efficiency following operations are performed

A. Selection
B. Crossover
C. Mutation

After every operation fitness value is calculated.

Efficient allocation of resources is the major matter of concern that compact with scheduling problems. Number of constraints have to be worked on throughout scheduling process. Generally due to restricted availability of resources no two jobs should take up particular resource simultaneously. Considering almost all scheduling problems, it has been observed that they are NP-Hard and hence are not solvable in polynomial time by means of deterministic algorithm.
Timetable scheduling problem represents a set of tasks (classes) and a set of resources (rooms, groups, instructors). Each task requests some resources for its implementation and has the fixed length. Those set of time intervals in which classes can be engaged is also obtained. The aim is to allocate those jobs to their resources while fulfilling all of the hard constraints – no resource should owe several tasks at the same interval of time.

I. TIMETABLE ILLUSTRATION WITH 3D ARRANGEMENT
In 3D cutting section, time table scheduling can be considered as an extraordinary case. To represent time table, 3D structure can be widely used. The proportions of 3D timetable are: days (*x*-axis), time interval (*y*-axis) and classrooms (*z*-axis). The classrooms are represented as cubes, which should be positioned in a 3D timetable

representation. Time table scheduling is a procedure of introducing those cubes into a timetable, in the manner such that no contradictory classes (which allocate the same resource, a student group or an instructor) are sited in the same timeslot. The timetable scheduling practice could be officially defined with binary variables x*cdtrgi*, which have the value of 1 if and only if instructor *i* lectures the class *c* on day *d* at time *t*, for group *g* in room *r*.
where, all the variables belong to one of the following set:

1. C = set of all classrooms available for department.
2. T = set of all teachers available in the department.
3. S = set of all theory subjects for the department.
4. P = set of all practical subjects for the department.
5. L = set of all practical labs available for the department.
6. D = set of total number of working days for the department

The timetable should assure the subsequent conditions:
1. No two teachers should be allotted with same classroom in same interval of time.
2. No two teachers should be allotted with same laboratory in same interval of time.
3. Maximum set of teachers allotted to one cell should be less than or equal to total number of groups present in the department.
4. No two classrooms should be allotted with same teacher in same interval of time.
5. No two laboratories should be allotted to same teacher in same interval of time.
6. No two subjects should be allotted to same teacher in same interval of time.
7. No two practical subject should be allotted to same teacher in same interval of time.
8. Maximum number of subjects either theory or practical should be less than or equal to total number of teachers allotted to the department.

A GUI (Graphical User Interface) has been provided to make smooth interaction and facilitate the input of data. For every class, given below can be the set:
Teachers who teach the class.
Classrooms where the classes *could* be engaged;
Number of classrooms utilized by a class simultaneously.
Groups of students that attend the class;
 Days and time slots when the classes *could* be engaged;
.
The hypothetical problem size can now be abridged. As the groups and teachers are fundamental part of a class definition, indices *i* and *g* are eliminated. The information about groups and teacher is not removed, however it is stored with the variable to be used later while invoking conflicts. In addition to this, since many patterns of indices *c,d,t,r* are impracticable, only the possible combinations are generated. As single class can make use of more than one classroom (e.g. a bulky group occupies two labs at the same interval of time), the index *r* of variable x*cdtrgi* actually denotes one of the probably possible combinations of the classroom allocations.
To facilitate the occurrence of conflict, three supplementary 3D structures are formed. Every structure gives a particular category of view on the timetable: from the aspect of classroom, group and teacher. As of from each view new constraints can be recognized. *X* and *Y* axes of each and every of three supplementary structures corresponds to the day and time. The *z*-axis differs in every structure, symbolizing classrooms, groups and teachers, correspondingly. For the period of the constraint generation progression, each variable is to be found, for all achievable day-time match up, at the suitable *z* coordinate, which stands for classrooms, groups or teachers allocated to the respected class. After satisfying all the data, each *x*-*y*-*z* coordinate is checked. If more classes struggle for a particular resource, a new constraint has to be produced. This process is equivalent to decreasing of resource to only one job in one timeslot. The developer is now guaranteed that only one variable will take (for example) a specific classroom in a single day-time arrangement.

II. CONFLICT PRODUCTION AND BOUNDS

The Genetic Algorithm Implementation

Genetic algorithms are evolving methods motivated by natural fruition. They can be utilized as methods for finding solution for complicated problems and for penetrating large problem spaces. Genetic algorithms fit in to guide arbitrary search practice, which attempt to discover the global peak. J.H. Holland has given this perception in initial years of seventies[1]. The supremacy of genetic algorithms and additional comparable practices (evolutionary approaches) resides in the point that they are proficient to discover global optimum in multi-modal spaces (spaces with many local optimums). Conventional ascent approaches will endlessly settle from preliminary position to certain local optimum, which could also be global, but it cannot be initiated for certain algorithms. Genetic

algorithms are engaged with the set of feasible and possible results, which is named as *population*. Each solution item (*individual*) is restrained by *fitness function*. The fitness value characterizes the superiority degree of an individual, so the algorithm can choose individuals with improved genetic substantial for constructing novel individuals and added generations. The reproduction of progression permits existence of improved individuals and disappearance of substandard ones as represented by fig1. Evolution's objective is to catch enhanced individuals in each group. The procedure of evolution is preserved by selection, crossover and mutation. In relation with genetic algorithms those procedures are termed as *genetic operators*. The selection selects grander individuals in every generation and guarantees that substandard individuals are vanished. The crossover operator elects two individuals from existing population (*parents*) and forms a novel individual (*child*) grounded on parents' genetic material. Selection and crossover operators will enlarge respectable characteristics of grander individuals throughout the entire population. They will correspondingly straightly direct the exploration procedure towards a native optimum. The mutation operator alters the significance of certain genes in an individual and profits to explore additional chunks of problem space.
In the algorithm offered at the juncture, each individual in the population signifies one timetable. The algorithm surprises from an infeasible timetable, and attempts to get the practicable one. In a timetable, every single class can be positioned only on one occasion in the 3D timetable assembly. This could be confirmed with generation of a novel constriction check for each class that ought to be planned. These additional constraints would just expand the problem scope and the quantity of constraints that should be crisscrossed. For the reason that every single class can have single variable set to 1, individuals can be spawned in such manner that every gene in an individual signifies one class. The assessment of a gene will be the ordinal of a binary variable which belongs to that class.

III. ENCODING OF INDIVIDUALS

The fitness value of an individual is obtained as :
fitness(Individual) =
[(Total number of constraints satisfied/ Total number of constraints checked)*100]

The amount of conflicts demonstrates the number of constraints that have been dishonored for the existing individual. When an individual approaches to zero number of conflicts, this shows that it denotes a viable timetable and that there are no crashes of classes.
The excellence of the timetable is obtained by earliness of planned classes. Students have improved skill to acquire more knowledge in early hours of the day and after that, the attention for learning is frequently diminishing. That is why try is that the finest eminence value is set to the early hours and poorer values are set for late hours. The genetic algorithm will attempt to plan classes as early in the pre-lunch time as it can, circuitously abate the figure of holes in a student's time table. The concept is right now hypothetical.
The foremost objective of the genetic algorithm represented here is to attain a practicable timetable. That is why the fitness function will largely attempt to diminish the numeral of conflicts in an individual. The eminence of the timetable is of the minor prominence. The program practices eradicating choice, which selects and removes corrupt or bad individuals from the present population, manufacture classroom for new children that will be born from the remaining individuals. The possibility of rejection grows equivalently with the fitness value of the individual. As the left behind individuals are improved than the ordinary population, it is estimated that their offspring will be superior as well. There is certain possibility (however precise) that eradicating selection erases the finest individual. That would collapse the algorithm energies and put its effort back for identical quantity of generations. Thus, security mechanism for finest individuals has to be prepared, so the virtuous genetic material is continued in population. It is called *the elitism*. Our choice is to preserve just the topmost individual. The replica (reproduction) operators establish a highly significant portion of genetic algorithms. Those operators mark usage of good individuals (which persisted in population afterwards selection) and build new, superior individuals and global population. The crossover operator works on individuals (termed *parents*) and mark new, *child* individual commencing their genetic substantial. This operator covers up vacant spaces in population normally that characteristics continued afterwards eradication. If parents are virtuous, it is expected that their children will correspondingly be virtuous. Uniform crossover operator authorizes entire genes of both parents. If parents consume identical standards of a gene, this standard is transcribed towards the offspring. If standards from parent genes fluctuate, formerly the algorithm arbitrarily selects one parent as a leading one and receipts its gene. The possibility of assortment of solitary individual is comparative to its fitness value. Increasing fitness values are utilized for every single individual.

…. SIZE OF THE POPULATION.

An arbitrary numeral r is produced through the algorithm from the interim *(0, D)* and picks out an individual which satisfies the minimum condition.

IV. PERFORMING UNIFORM MUTATION ON INDIVIDUAL

The mutation is likewise a classic operator for the genetic algorithm which works in the following way as shown in fig2. It receipts one or more genes starting from an individual and then alters its standard. The possibility of the mutation is a contribution (input) parameter aimed at genetic algorithm. The offered algorithm repeats over every single gene of each individual in the population. Meant for every gene an arbitrary number form the interim (0, 1) is produced. Uncertainty if the spawned value is lesser than the assumed prospect of the mutation, the gene fluctuates the assessment to a casual (random) value which signifies a dissimilar classroom grouping.

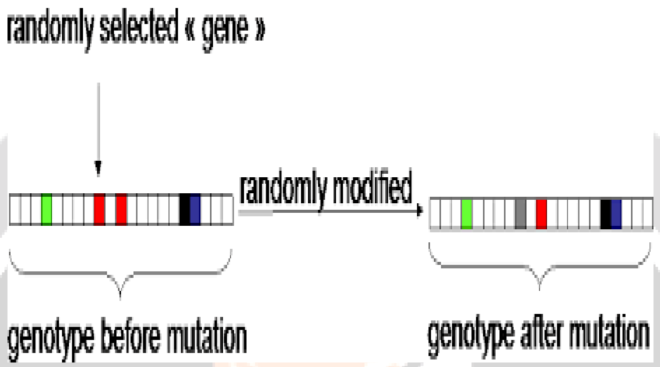

Fig2 Applying Mutation operator

V. APPLYING CROSSOVER OPERATOR ON INDIVIDUALS

Cultivating genetic algorithm performance

The chief thought of algorithm was aimed at enlightening operators and was to ban the starter of novel conflicts. Elementary operators did not consider whether they create individuals with additional or with a fewer conflicts. The consequence of the algorithm applying such elementary operators was extremely deprived. The judgment was prepared to supplement specific programming logic to the operators and to practice various selection algorithm. The major basic footstep of the upgraded crossover operator plainly duplicates equivalent genes from parents to the sibling's individual. No any extra conflicts can be appended by replication of those identical values. Diverse parent's genes are particularly noticed and passed on for advance processing. In the subsequent phase, the crossover operator as shown in fig3, authorizes the collection of equations for every single marked gene in offspring individual and totals(counts) the number of probable conflicts produced for both parent's selections. The gene starting from a parent that causes scarcer conflicts is preferred, so no extra conflicts are raised in accumulation to the conflicts instigated by the parents. This type of an alteration was not presented to the mutation operator. Analogous working of the mutation operator would possibly lead the entire population to the confined optimal. In utmost circumstances the mutation operator creates substandard individuals, but then again furthermost of its genes are virtuous. Mutated genes assist us to discover supplementary fragments of penetrating space and to escape attainment of local optimal value. Further additional conflicts in population will be produced when two identical individuals participate in the breeding procedure as parents. In that situation one of the parents will be mutated and the offspring will be arbitrarily produced. In this technique a poor child individual will be formed, on the other hand it will not make a fuss over the population. Reasonably the converse, it will take along fresh genetic content to the flooded population. The enhanced algorithm uses tournament eliminating selection. This type of assortment guarantees elitism. Accordingly, the lead individual cannot be replaced. The tournament selection permits bigger population scope (size) deprived of decelerating down the algorithm. Improved outcomes were attained with population of considerably superior size, in distinction to the elementary algorithm, where expansion of the population give rise to substantial performance deprivation.

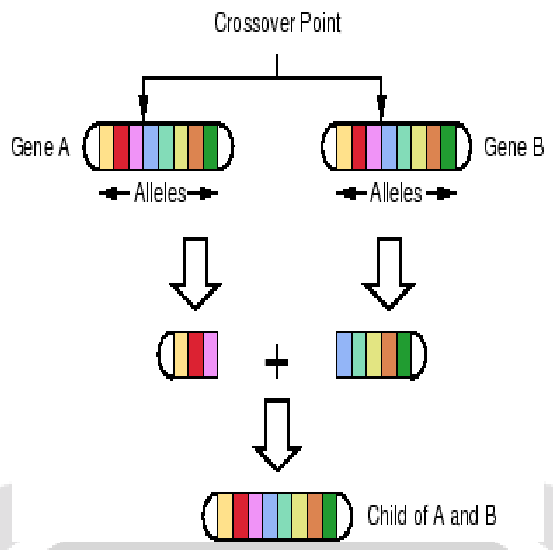

Fig3 Applying crossover operator

4.1 System flow diagram

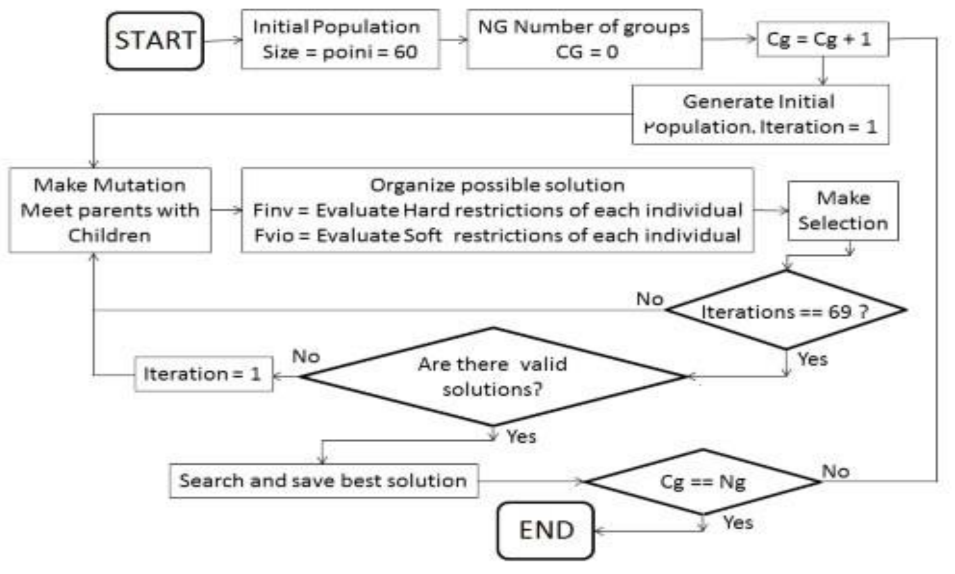
START
Initial Population Size = poini = 60
NG Number of groups CG = 0
Cg = Cg + 1
Generate Initial Population, Iteration = 1
Make Mutation Meet parents with Children
Organize possible solution
Finv = Evaluate Hard restrictions of each individual
Fvio = Evaluate Soft restrictions of each individual
Make Selection
No
Iterations == 69 ?
Yes
Iteration = 1
No
Are there valid solutions?
Yes
Search and save best solution
Cg == Ng
No
END
Yes

Fig-2 System Flow Diagram

Fig -2 shows the system flow diagram. It gives us detailed information about the Process which will lead to Time Table generation using Genetic Algorithm.

5. CONCLUSION

The basic scheduling problem with bulky figure of binary variables has been abridged to the satisfactory extent by eradicating definite proportions of the problem and integrating those proportions into constraints. The alliance of numerous binary variables into solitary gene standard meaningfully condensed the individual size. Currently it is imaginable to attempt to resolve the complete size problem (Time table scheduling problem) with genetic algorithm methodology. Such an illustration of the scheduling problem attains the adequate algorithm speed, thus lesser size problems are resolved in tens of seconds. Important enhancements have been attained by means of intelligent operators. The intelligent algorithm unites much quicker than the elementary algorithm and signifies a virtuous initial point for complete solution of the full scale problem. To entirely crack the full scale problem, additional algorithms enhancements will have to be accomplished. When producing the constraints, it could be beneficial to sign every one, so no constraint will be fixed (and checked) twice. Individuals should be created in such manner that classes which are further problematic to be programed inhabit the façade (front) genes of an individual, while classes are at ease to arrange should inhabit the back genes. This would be convenient for intelligent crossover operator, which sets and authorizes the conflicts from start to end of the individual. Also, a parallel computing attitude or methodology should be experimented, so inspection space of the problem could be broadened. Each thread could begin with diverse initial population and the excellence of solution is predicted to be enhanced.

ACKNOWLEDGEMENT

It is our gratitude to acknowledge respected principal “Dr. D.N.Kyatanwar”, HOD Computer Department “Prof. D.B,Kshirsagar”, Guide for the project “Dr. A. Kalyan Kumar”, Co-Guide for the project “Prof. G. Madhukar” and staff of entire Computer Department of SRES COE ,Kopargaon for their immense help and utmost valuable guidance during the completion of entire project, without them successful implementation of the project would have remained the dream. We would also like to thank our colleagues for their ample support.

REFERENCES

[1] Branimir Sigl, Martin Golub, Vedran Mornar, “Solving Timetable Scheduling Problem by using Genetic Algorithm” Faculty of Electrical Engineering and Computing University of Zagreb Unska 3, 10000 Zagreb, Croatia branimir.sigl@bj.hinet.hr,marin. golub@fer.hr, vedran.mornar@fer.hr