IJCSEInternational Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-5, Issue-12 E-ISSN: 2347-2693

Scheduling Time Slots for Class Conduction Using Genetic Algorithm

Vinodh P Vijayan[1],Neena Joseph[2],Neema George[3],Simy Mary Kurian[4]

[1,2,3,4]Department of CSE, MangalamCampus, Ettumanoor, Kottayam, India

[*]*Corresponding Author: vinodhpvijayan@yahoo.com, Tel.: 9961687007*

Available online at: www.ijcseonline.org

Received: 23/Nov/2017, Revised: 09/Dec/2017, Accepted: 15/Dec/2017, Published: 31/Dec/2017

***Abstract*—** Scheduling the class intervals using traditional methods like using on line spreadsheets is complicated and time consuming. The complexity will increase with more than one concern, pupil and teachers because the requirements become extra complex. It is tough to manage topics with multiple instructors or forming student businesses and assigning instructors. In addition, priority of topics or instructors, class hours are not taken into consideration for scheduling. Class scheduling using genetic algorithm, has been advanced to time table class rooms thinking about various resources and parameters. The proposed algorithm accepts diverse parameters like priority values for teachers and topics or elegance hours and give the satisfactory solution. Our new device makes the school room scheduling less difficult and also reduce the time required for scheduling.

***Keywords*—** Scheduling, ComplexityIOT, fuzzylogic, genetic algorithm.

I. INTRODUCTION

Classroom scheduling changed into the maximum tedious and time-consuming process in every faculties and schools. Even though, there are software's to be had for scheduling, it does no longer meet many necessities needed by using faculties and colleges. Existing software program's assist to agenda class room through assigning subjects to instructors and instructors to magnificence room. But it's miles impossible to get information about total hours assigned or to compare and analyze the work hours assigned with in a length.

Therefore, we propose a genetic algorithm for scheduling class rooms which reduces the complexity of the scheduling method. With the help of new device, the user can agenda classes, view and edit current schedule. The genetic set of rules accepts a couple of parameters, practice genetic operators to the parameters to discover the best solution for the scheduling trouble.

The algorithm takes numerous sources as enter parameters and find exceptional score value for each populace taken from input parameters. The population with the excellent rating offers the most suitable solution for scheduling.

II. LITERATURE SURVEY

Genetic algorithms are search algorithms primarily based on the mechanisms of natural choice and natural genetics [1]. Genetic algorithms are used for solving problems because of its ability to search for greatest solutions in large seek area [2]. In GA, it requires the natural set of parameters of optimization hassle to be coded as a finite string length chromosome representation consisting of binary, real-coded, integer and permutation representation. Some operations ought to do in genetic set of rules such as initialization, reproduction, evaluation and selection.

During initialization, some random solution is generated and store in population. The population itself is a collection of solutions and it will be forwarded for the next iteration. Reproduction is producing some other solutions through genetic operations, crossover and mutation. The crossover operation produces multiple child solutions and mutation is modifying the parent solution to become another solution. The algorithm will evaluate all existing solutions using the fitness function after the reproduction process is done. A fitness function is a function that will show how good a solution is. The solutions with the better fitness value are to be processed in the next iteration.

Different approaches are available for solving scheduling problem, however most of the systems are not customizable to provide users special needs [4]. It is important to consider that every organization must structure a schedule allocation system that favors the commitment, development and satisfaction of university professors, at the individual and collection level [3]. The process of scheduling the classroom is considered to be one of the most significant challenging procedures for programmers and teachers [5][6]. The optimization approach to time tables construction aims to optimize sets of general rules and find the best suited solution [3].

Even though, to deal with scheduling, different optimization methods i.e. particle swarm optimization (PSO), genetic algorithm (GA), ant colony system (ACS), fuzzy logic etc. have been used to obtain fast computing abilities, GA is the most suitable approach in the arrangement of scheduling system [4]. Genetic algorithms have the ability to search for optimal solutions in search

space.
Several significant constrained factors such as timing, teacher, lecture, classroom, and students must be considered [5][6]. It requires a lot of time and effort to make a practical and highly efficient classroom scheduling plan [6].

Each course may have one or more lectures and classes depending on the total number of students registered [4]. The scheduling method includes necessary constraints [6]A teacher cannot teach two classes at the same time;

- A student cannot follow two lectures at the same time;
- Some teachers must have at least one day off during the week (Optional);
- All the days of the week should be covered by the time table;
- Subject X must have exactly so-and-so hours each week;

Genetic algorithms are general search and optimization algorithms inspired by processes and normally associated with natural world. Genetic algorithm mimics the process of natural selection and can be used as a technique for solving complex optimization problems which have large spaces [10]. They can be used as techniques for solving complex problems and for searching of large problem spaces. Unlike many heuristic schemes, which have only one optimal solution at any time, Genetic algorithms maintain many individual solutions in the form of population. Individuals (parents) are chosen from the population and are then mated to form a new individual (child). The child is further mutated to introduce diversity into the population [10].Rather than starting from a single point within the search space, GAisinitialized to the population of guesses. These are usually random and will bespread throughout the search space. A typical algorithm then uses three operators, selection, crossover and mutation, to direct the population toward convergence at global optimum.A GA, as shown in figure 1 requires a processof initializing, breeding, mutating, choosing andkilling. It can be said that most methods called GAs have at least the following elements in common: Population of chromosomes, Selection according to fitness, Crossover to produce new offspring, and random mutation of new offspring.

1) Optimization problems

In a genetic algorithm, a population of candidate solutions (called individuals, creatures, organisms, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered; traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.[3]

The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions 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.

A typical genetic algorithm requires:

1. a genetic representation of the solution domain,
2. a fitness function to evaluate the solution domain.

A standard representation of each candidate solution is as an array of bits (also called bit set or bit string).[3] Arrays of other types and structures can be used in essentially the same way. The main property that makes these genetic representations convenient is that their parts are easily aligned due to their fixed size, which facilitates simple crossover operations. Variable length representations may also be used, but crossover implementation is more complex in this case. Tree-like representations are explored in genetic programming and graph-form representations are explored in evolutionary programming; a mix of both linear chromosomes and trees is explored in gene expression programming.

Once the genetic representation and the fitness function are defined, a GA proceeds to initialize a population of solutions and then to improve it through repetitive application of the mutation, crossover, inversion and selection operators.

a) Initialization

The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Often, the initial population is generated randomly, allowing the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

b) Selection

During each successive generation, a portion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as the former process may be very time-consuming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem

dependent. For instance, in the knapsack problem one wants to maximize the total value of objects that can be put in a knapsack of some fixed capacity. A representation of a solution might be an array of bits, where each bit represents a different object, and the value of the bit (0 or 1) represents whether or not the object is in the knapsack. Not every such representation is valid, as the size of objects may exceed the capacity of the knapsack. The fitness of the solution is the sum of values of all objects in the knapsack if the representation is valid, or 0 otherwise.

In some problems, it is hard or even impossible to define the fitness expression; in these cases, a simulation may be used to determine the fitness function value of a phenotype (e.g. computational fluid dynamics is used to determine the air resistance of a vehicle whose shape is encoded as the phenotype), or even interactive genetic algorithms are used.

c) Genetic operators

The next step is to generate a second generation population of solutions from those selected, through a combination of genetic operators: crossover (also called recombination), and mutation.

For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each new child, and the process continues until a new population of solutions of appropriate size is generated. Although reproduction methods that are based on the use of two parents are more "biology inspired", some research[4][5] suggests that more than two "parents" generate higher quality chromosomes.

These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally, the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions. These less fit solutions ensure genetic diversity within the genetic pool of the parents and therefore ensure the genetic diversity of the subsequent generation of children.

Opinion is divided over the importance of crossover versus mutation. There are many references in Fogel (2006) that support the importance of mutation-based search.

Although crossover and mutation are known as the main genetic operators, it is possible to use other operators such as regrouping, colonization-extinction, or migration in genetic algorithms.[citation needed]

It is worth tuning parameters such as the mutation probability, crossover probability and population size to find reasonable settings for the problem class being worked on. A very small mutation rate may lead to genetic drift (which is non-ergodic in nature). A recombination rate that is too high may lead to premature convergence of the genetic algorithm. A mutation rate that is too high may lead to loss of good solutions, unless elitist selection is employed. An adequate population size ensures sufficient genetic diversity for the problem at hand, but can lead to a waste of computational resources if set to a value larger than required.

d) Heuristics

In addition to the main operators above, other heuristics may be employed to make the calculation faster or more robust. The speciation heuristic penalizes crossover between candidate solutions that are too similar; this encourages population diversity and helps prevent premature convergence to a less optimal solution.[6][7]

e) Termination

This generational process is repeated until a termination condition has been reached. Common terminating conditions are:

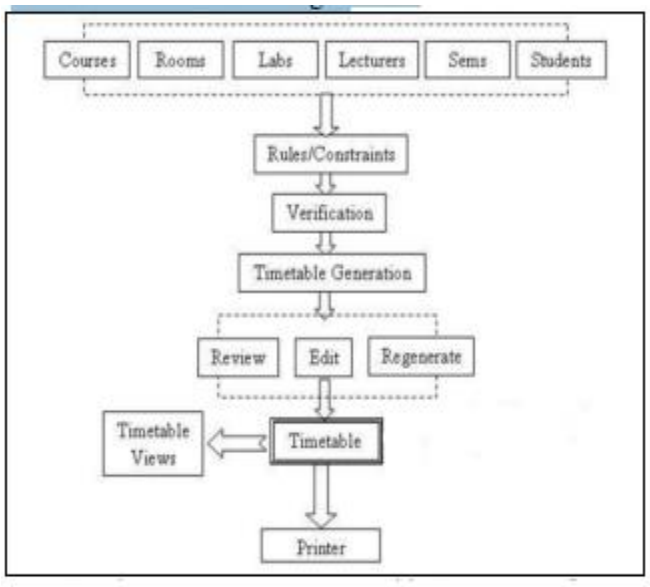
- A solution is found that satisfies minimum criteria
- Fixed number of generations reached
- Allocated budget (computation time/money) reached
- The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results
- Manual inspection

III. PROPOSED SYSTEM

Our new system uses Genetic Algorithm for optimizing and finding best solution for class room scheduling. The algorithm accepts multiple parameters such as Class hours, Subject priority or priority values assigned to teachers. The result with best score is used for scheduling the class room

We start with the requirement analysis and input from all stakeholders, where we represent inputs as chromosomes. We apply genetic operator's crossover and mutation to the input set and fitness is calculated. All chromosomes which does not satisfy the condition fitness greater than threshold are removed. We perform the above steps until number of iterations became greater than threshold.

Course and lectures will be scheduled in accordance with all possible constraints and given inputs and thus a timetable will be generated.



Structure of time table generator consists of input data, relation between the input data, system constraints and application of genetics algorithm.

A. Input Data The input data contains:

1) Professor: Data describes the name of lecturers along with their identification number.
2) Subject: Data describes the name of courses in the current term.
3) Room: Data describes the room number and their capacity.
4) Time intervals: It indicates starting time along with duration of a lecture.

B. System Constraints System constraints are divided into 2 categories:

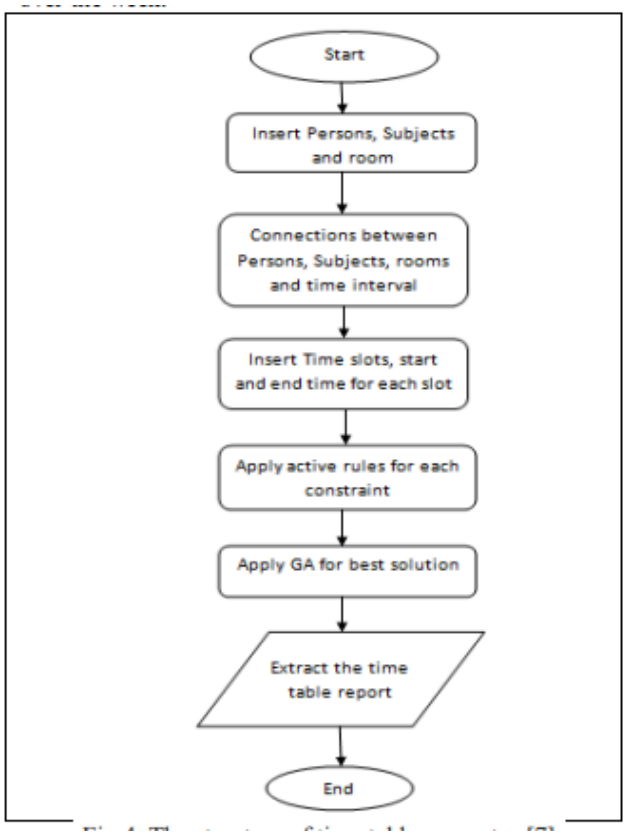
1) Hard Constraints:The timetable is subjected to the following four types of hard constraints, whichmust be satisfied by a solution to be considered as a valid one:

a. A student should have only one class at a Time.
b. A Teacher should have only one class at a time.
c. A room should be booked only for one class at a time.
d. Some classes require classes to have particular equipment.

For example, audio visual equipment, projectors etc.

2) Soft Constraints: These are the constraints that are of no great concern but are still taken into contemplation. They don't need to be satisfied but the solutions are generally considered to be good if they are satisfied.

a. Courses must be eventually distributed.
b. Students should not have any free time between two classes on a day.
c. Scheduling of teachers should be well spread over the week.



IV. EXPERIMENTS AND RESULTS

The scheduling application is advanced the usage of Asp.Net Core, Angular, Angular CLI, Bootstrap, C# and EntityFrameworkCore. MySQL is used as the lower back-quit database. EntityFrameworkCore is used for connecting front cease with the database.

Angular CLI is used for the front-stop improvement and .Net Core is used for the API improvement.

We can locate that the convergence time decreases with the boom in mutation probability. See the graph depicted underneath.

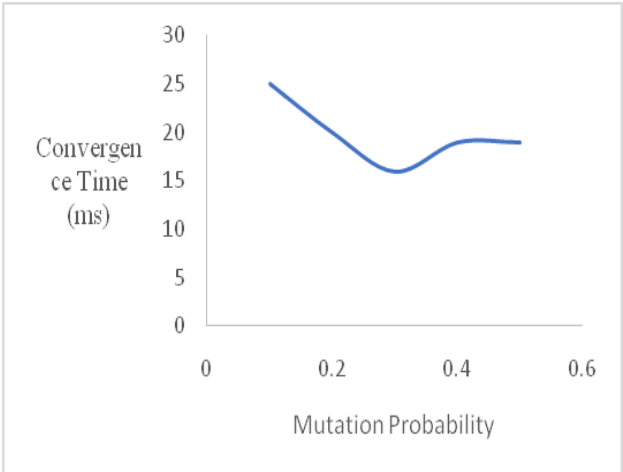


Fig 1: Mutation Probability and Convergence Time result

We can consider any number of parameters for the genetic algorithm. The experimental result shows that the user satisfaction is higher when we use large of parameters when compared to satisfaction with fewer number of parameters.

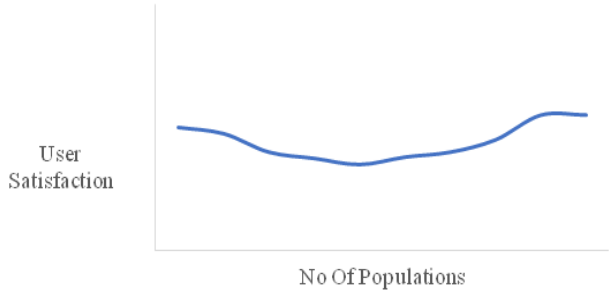


Fig 2: User satisfaction level with no of populations

V. CONCLUSION

The real time problems like elegance scheduling in instructional establishments are multi-targets troubles where multiple parameters like instructors input, students input, usual consolation etc want to be considered. The optimization of ordinary efficiency of the gadget using conventional set of rules are time eating in addition to end up with neighborhood foremost. The application of evolutionary algorithm like genetic set of rules will be able to perform in a dynamic scenario and convey global answer. The device gives stepped forward user pleasure price along with discount in convergence time.

To generate timetable for the institute which will be less time consuming and free of human errors along with high level of efficiency and precision. Moreover improve the overall process of timetable generation with help of genetics algorithm along with the assistance of technology.

REFERENCES

[1]. R. Lewis and J. Thompson, "Analysing the effects of solution space connectivity with an effective metaheuristic for the course timetabling problem," European. Journal of Operation Research, vol. 240, no. 3, pp. 637–648, 2014.
[2]. A. El Amraoui, M.-A. Manier, A. El Moudni, and M. Benrejeb, "A genetic algorithm approach for a single hoist scheduling problem with time windows constraints," Engineering Application of Artificial Intelligence, vol. 26, no. 7, pp. 1761–1771, 2013.
[3]. Rohit. P.S, S.M., "A Probability Based Object-Oriented Expert System for Generating Time-Table" International Journal of Research in Computer Applications & Information Technology, 2013. Volume 1(Issue 1): p. pp. 52-58
[4]. Mohammad, K.H., J, Problem-solving capacities of spiritual intelligence for artificial intelligence. Social and Behavioral Sciences, 2012: p. 170 – 175.
[5]. Isaai, M.T., Kanani, A., Tootoonchi, M., & Afzali, H. R. , Intelligent timetable evaluation using fuzzy AHP Expert Systems with Applications,, 2011. 38(4): p. 3718-3723.
[6]. David E Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning, 1st ed. Addison-Wesley Publishing Company, Inc, 1989, Ch. 1, pp. 1-2..

Authors Profile

Mr. Vinodh P Vijayan, Head of Department, Computer Science, Mangalam College of Engineering, Kottayam, India has completed Bachelors Degree in Electronics and Communication Engineering, Post graduation in Computer Science Engineering His area of interest includes Soft Computing, Robotics, Bio-inspired computing, Fuzzy systems.

Ms.Neena Joseph working as an Assistant Professor in Mangalam College of Engg,Kottayam,Kerala.Masterof Engineering in Computer science and Engineering (M.E CSE) from:M S University.Bachelor of Technology in Computer Science and Engineering (B.Tech-CSE) from MG University, kerala.Her Area of interest Cloud Computing,Theory of Computation.

Ms.Neema George working as an Assistant Professor in Mangalam College of Engg,Kottayam,Kerala.Masterof Engineering in Computer science and Engineering (M.E CSE) from:Anna University Chennai.Bachelor of Technology in Computer Science and Engineering (B.Tech-CSE) from MG University, kerala.Her Area of interest Cloud Computing,Machine learning,Artificial intelligence .

Ms.Simy Mary Kurian working as an Assistant Professor in Mangalam College of Engg,Kottayam,Kerala.Masterof Engineering in Computer science and Engineering (M.E CSE) from:Karunya University Chennai.Bachelor of Technology in Computer Science and Engineering (B.Tech-CSE) from MG University, kerala.Her Area of interest Image Processing,Machine learning,Artificial intelligence.

