International Journal of Innovation, Management and Technology, Vol. 1, No. 3, August 2010
ISSN: 2010-0248

Formulation of Genetic Algorithm to Generate Good Quality Course Timetable

Ashish Jain[1], Dr. Suresh Jain[2] and Dr. P.K. Chande[3]

***Abstract*—Course timetabling is the NP-hard problem, Scheduling course time tables for large modular courses is a complex problem which often has to be solved in university departments. This is usually done 'by hand', taking several days or week of iterative repair and after feedback from students complaining that the time table is unfair to them in some way. Which we in this research tried solving through conventional method (Result shows that the problem of conventional method) but at a certain level we recognized that solving timetabling problem with increased constraints is very difficult by conventional methods. A highly constrained combinatorial problem, timetabling can also be solved by evolutionary techniques. In this research we are showing evolutionary based genetic algorithm approach as an effective solution to solve course timetabling problem.**

***Index Terms*—Genetic Algorithms, Meta-heuristic, Course Timetabling .**

I. Introduction

A timetable is a set of meetings in time. A meeting is combination of resources (rooms, people etc), some of which is specified by problem and some of which may be allocated as part of solution. Our aim is to schedule teachers and classes to fully utilize available resources. This is done by assigning correct time and place to appropriate event hence timetable must be designed to satisfy all user requirements or else it would be a waste of time and requirements.

In course timetabling the hard constraint will be that at a time lecturer cannot take lecture in two different classes and after each lecture at least one lecture relaxation should be given to the lecturer. Timetabling constraints are many and varied. In this research, genetic algorithm approach is applied for solving university course timetabling problem. GA is a way of addressing hard search and optimization problems which provides a good solution although it requires large execution time.

In section 2 we represent the formulation of the lecture timetable problem, experiments which we have done represented in section 3 and in section 4 formulation of genetic algorithm approach for solving course timetabling problem is discussed in detail. Section 5 presents the result of several experiments conducted using conventional methods, followed by conclusion in section 6.

[1]Reader, Department of Computer Science, Truba college of Engineering & Tech., Indore, India
[2]Director, KCB Technical Academy, Indore
[3]Group Director, Truba group of institutes, Indore
Email:ashishjn.research@gmail.com,suresh.jain@rediffmail.com,pkchandein@yahoo.co.in

II. Problem Description

The timetables to be treated in this paper are constructed for the two year courses like M.B.A. for Devi Ahilya University, Indore. The course is divided into semesters (for fall semester: semester 1 & semester 3 and for spring semester: semester 2 & semester 4). There are 7 time periods within a day and each time period consists of a 50 minutes. In every semester the course contains 8 theoretical subjects one of which contains laboratory work also. Each subject should be allocated 4 time periods in a week along with 2 time periods for laboratory work.

A. Problem Definition

Given the participants consisting of lecture timetable:
k lecturers $l_1, l_2, l_3, \ldots l_k$ and
m classes $c_1, c_2, c_3, \ldots c_m$, with
r courses $s_1, s_2, s_3, \ldots s_r$
q rooms $r_1, r_2, r_3 \ldots r_q$ and
n periods $p_1, p_2, p_3, \ldots p_n$
Where an assignment is a 5 tuple<l, c, s, r, p> [1].

B. Constraints Involved

The constraints that we treat are classified as hard and soft. Hard constraints are those to which a time table has to adhere in order to be satisfied.

Hard constraints involved are:

1) No participant (lecturer or class) can be in more than two rooms at the same period.
2) No room should be double booked.
3) The room capacity should be large enough to hold each class.

Violating the above constraints will cause the time table to be unfeasible. In addition, we would also like to satisfy as many soft constraints as possible in order to produce a good quality timetable. Soft constraints for this constrained optimization problem are actually the students and lectures preferences which can be as follows.

1) The second time for each subject should not be in the same day.
2) No subject should be allocated to a time period that heads of department don't demand because of other work.
   - The subject should not be allocated to a time period inconvenient for a lecturer.

C. Approaches to Automate Timetabling

Problem specific heuristic methods can produce good timetable, but the size and complexities of modern university

timetabling has provoked a trend which was confirmed by the presentation at the first international conference on the practice and theory of automated timetabling towards more general problem solving algorithm, or meta-heuristic such as genetic algorithms, simulated annealing and tabu search [2] and hybrid approach such as memetic algorithms. A recent approach to the timetable problem is to use the genetic algorithms as a powerful method of solving difficult timetabling problems [1].

1) Genetic Algortihms:

A population of feasible timetables is maintained. The fittest timetables are selected to form the basis of next iteration or generation. Basic operators such as selection, mutation and crossover are applied to get the best results.

A GA optimizer tool for university timetable has been generated with specific reference to a particular department of University of Malaya, Lembah Pantai, Kuala Lumpur [1].

In section 4 formulation of genetic algorithm approach for solving our course timetabling problem is discussed.

2) Simulated Annealing:

Simulated annealing is a search strategy which keeps a track of one feasible timetable. On each iteration, a neighbor is generated another feasible timetable, slightly altered at random from the current one. This neighbor is accepted as the current timetable if it has a lower penalty.

It is successful applied to timetabling problem in Swansea's TISSUE examination scheduling [3, 4].

3) Tabu Search:

Tabu search is a metaheuristic algorithm that can be used for solving combinatorial optimization problems. Tabu search enhances the performance of a local search method by using memory structures: once a potential timetable solution has been determined, it is marked as "tabu" so that the algorithm does not visit that possibility repeatedly.

Tabu Search has been successfully applied by Boufflet and Negre to generate examinations timetables at University of Compiegne [5].

4) Memetic Algorithms:

The concept of a memetic algorithm was first introduced by Moscato and Norman to describe evolutionary algorithms in which local search is used to a large extent. Each solution in the population is represented as a number of memes. Each meme contains event (i.e. course, class, subject & room no.) to be taken by lecturer in a particular period. A further meme is used to hold lecturer and period associated with him / her. A meme differs from a gene in that as it is passed between individuals, each individual adapts the meme as it sees best whereas genes are passed unaltered.

The University of Nottingham is currently developing memetic algorithm for examination scheduling [6].

III. Experiment Done

To solve the course timetabling problem of two year courses we performed some experiments. In our experiments we used conventional techniques. .NET 2.0 is chosen as the front end and SQL server 2000 as the backend with C# as the programming language. Our experiments used real data from the M.B.A department of Truba College of Engineering & Technology, Devi Ahilya University, Indore for the year 2009-2010. 9 Lecturers were available, conventional methods produced satisfying results but as the constraints were increased it became difficult to produce good quality timetable using conventional techniques. Thus, we recognize that for solving this problem we need a metaheuristic approach that fits our requirements which is possible by using fitness function and other operators provided by genetic algorithm. Formulation of chromosome, evaluation criteria and how other operators will be used is as follows.

IV. Timetable Formulation Using Genetic Approach

[7] The genetic algorithm works as follows:

1) Initialization of parent population
2) Evaluation (fitness function)
3) Selection
4) Crossover/recombination
5) Mutation
6) Evaluate child and Go to step 3 until termination criteria satisfies.

Initial population is generated randomly.

A. Chromosome Representation

The chromosome chosen to represent a timetable solution is a two dimensional array A_{ij} i=1,……k, j=1……..n such that each element of array represents a 3 tuple < c, s, r >; where:

c represents class (students taking the same course),
s represents course taken,
r represents classroom,
n is the number of periods in a week,
k is the number of lecturers

A typical solution timetable is shown in figure1.

This representation is chosen because:

1) The row constraints are always satisfied by its own representations i.e. each lecturer cannot be assigned more than one lecture for each period since there cannot be more than one element in each cell of array.
2) A_{ij} can be easily mapped to the weekly lecture timetable where the rows represent the classes instead of lecturers.

Period PE / Lecturers	PE(0)	PE(1)	PE(m)	PE(2m)	PE(n-1)
Lecturer 1	Event 1				Event 7
Lecturer 2			Event 3		
Lecturer k					Event n

Figure 1: A solution timetable.

B. Evaluation

Fitness in biological sense is a quality value which is a measure of the reproductive efficiency of chromosomes [7]. In genetic algorithm, fitness is used to allocate reproductive traits to the individuals in the population and thus act as some measure of goodness to be maximized.

The fitness function of each chromosome is evalulated by examining the soft constraints .Each soft constraints is

assigned a penalty value which contributes to the fitness function for each constraint voilated.

If, Φj denotes the penalty value associated with the *jth* violation detected for constraint *I* then the summation of penalty values P(i) for constraint *I* is given by

$$P(i) = \sum_{j=1}^{\beta} \phi j$$

Thus

$$\text{Fitness value} = \sum_{i=1}^{\alpha} P(i)$$

Where:

β is the number of violation for constraints *I*, and

α is the number of constraints

C. Genetic Operators

The genetic operators are as follows:

1) Selection

Reproduction (or selection) is an operator that makes more copies of better strings in a new population. Reproduction is usually the first operator applied on a population. During each successive generation, a proportion of existing population is selected to breed a new generation. Individual solutions are selected through fitness-based process, where fitter solutions are typically more likely to be selected [7]. Selection methods rate the fitness of each individual and preferentially select the best solution.

Common Methods of Selection are:

1) Roulette wheel method
2) Tournament selection
3) Stochastic remainder selection

We are using Roulette wheel for our research work.

Roulette wheel Selection:

The i^{th} string in the population is selected with a probability proportional to f_i. Since the population size is usually kept fixed in a simple GA, thus the sum of the probability of each string being selected for the mating pools must be one. Therefore, the probability for selecting the string is

$$P_i = f_i \Big/ \sum_{i=1}^{n} f_i$$

Selection provides an advantage over conventional method because through selection we can select the best timetable from the available feasible timetables. Fitness function is used as a measure for the selection of chromosomes (timetable).

2) Crossover

A crossover operator is used to recombine two strings to get a better string. The next step after selection is crossover which generates a second generation population of solutions from those selected through selection.

Some Common types of Crossover:

3) One site crossover
4) Two site crossover
5) Trade of Uniform Crossover(TOUC)

Currently, in our research work we are using one site crossover. Crossover is advantageous over the conventional method because through crossover we can easily exchange the information in the timetable which makes it optimal and effective as per the requirements.

3) Mutation

Mutation adds new information in a random way to the genetic search process and ultimately helps to avoid getting trapped at local optima. It is an operator that introduces diversity in the population whenever the population tends to become homogeneous due to repeated use of reproduction and crossover operators.

V. Results

The results of our experiments are shown in figure2 and figure 3.The figure2 shows the timetable for semester1 and figure3 shows the timetable for semester 3.

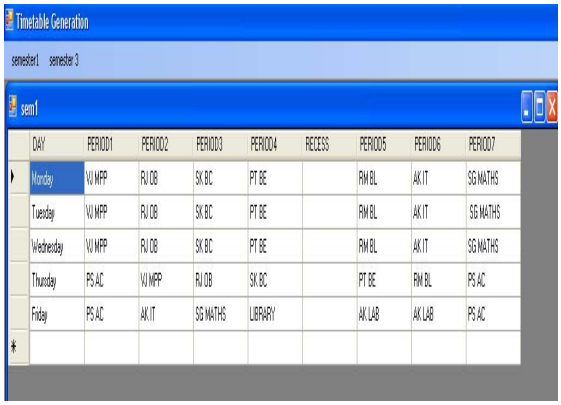

Figure 2: Time Table M.B.A. I Sem

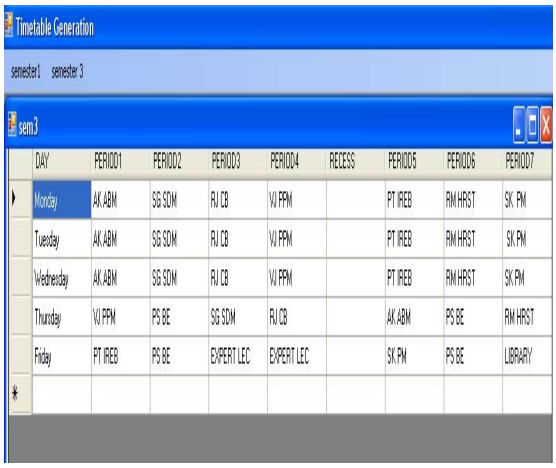

Figure 3: Time Table M.B.A. III Sem

VI. Conclusion

We have tried to show that genetic algorithm is a powerful method for solving timetabling problem, which we tried solving through conventional methods and through results we want to shows that time period is fixed for faculty & subject, which is very difficult to change and as the constraints were increased it became very complex to produce qualitative results. These problems of conventional methods can be eliminated by making use of fitness function provided by genetic algorithm. The various genetic operators such as selection, mutation and crossover enhance the results in various aspects such as through selection we can select the

best chromosomes on the basis of fitness function from the pool of chromosomes and similarly through crossover we can exchange the information of the timetable as per our requirements. Our future work will attempt to use this genetic approach technique for solving our real-world course timetabling problems.

REFERENCES

[1] Omar el Mahdi, R.N. Ainon and Roziati Zainuddin, "Using a Genetic algorithm optimizer tool to generate good quality time tables", Proceedings of the 2003 10th IEEE International conference, 14-17 Dec 2003 Page(s):1300-1303 Vol. 3.

[2] E.K.Burke, K. Jackson, J. H. Kingston and R.E.Weare, "Automated university timetabling: The state of the art", The computer journal, vol. 40, no.9, 1997.

[3] B. Paechter, A. Cumming, M.G. Norman and H. Luchian (1996), "Extension to a Memetic Timetabling System", In Burke, E.K. and Ross, P. (eds), The Practice and Theory of Automated Timetabling, pp. 251-265. Springer-Verlag, Berlin.

[4] J. Thompson and K.A Dowsland (1996) "Variants of simulated annealing for examination and timetabling problem". Ann. operations Research" 63,105-128.

[5] J.P. Boufflet, and S. Negre (1996), "Three method used to solve an examination timetable problem", In Burke, E.K. and Rose, P. (eds), The practice and theory of automated timetabling, pp. 327-344, Springer-Verlag, Berlin.

[6] E.K. Burke, J.P. Newall, and R.F. Weare (1996), "A Memetic Algorithm for University Exam Timetabling", In Burke, E.K. and Ross, P.(eds), The practice and Theory Of Automated Timetabling, pp.241-250.Springer-Verlag,Berlin.

[7] David E. Goldberg, 1989, "Genetic Algorithms in search, optimization and machine learning".