A Hybrid Genetic Algorithm Approach to a Departmental Class Timetabling Problem Using Efficient Data Structures

Arvind.S.Babu
Protechsoft Technologies
Pvt Ltd, Chennai- 600 029.

R.Chockalingam
Syncfusion Software Pvt. Ltd
Chennai- 600 040

S.Kavitha
SSN College of Engineering
Kalavakkam-603 110

ABSTRACT
The application of Genetic Algorithm with a local search operation performed within its loop has provided very accurate results, but the algorithm take a lot of time to arrive at an optimal solution. This paper describes the use of a Hybrid Genetic Algorithm using efficient data structures to automate the construction of a departmental class timetable. This problem is concerned with the allocation of faculty resources to concerned studentgroups and their corresponding timeslots. The quality of the solution is determined in terms of a penalty value which determines the degree to which various constraints are satisfied. This algorithm is tested over established datasets and the performance of the algorithm over different datasets. The result has confirmed that this algorithm in conjuncture with efficient data structures is able to produce high quality solutions for a departmental class timetable with short span of time. It is thus concluded that organization of data plays a major role in the performance of the Hybrid Genetic Algorithm to produce high quality solutions.

Keywords
Hash, Hard and soft constraints, Hybrid GA, Local search.

1. INTRODUCTION
In Timetabling literature, there has been numerous works done in automating the construction of a University Timetables. Over time various methods have been employed in constructing a timetable. One such successful approach to scheduling and timetabling problems [1, 2] is represented by the combination of Genetic algorithm with local search algorithm (this type of algorithm also referred to as Memetic Algorithm). This paper is organized as follows: The next section gives a slight overview of a departmental class timetabling problem and all of the constraints involved in it. Section 3 gives a detailed description of the working of the Hybrid Genetic Algorithm. Section 4 describes the data structures used and the representation of the data. Section 5 showcases the experimental results. The performance of the algorithm for varied input sizes and varied Genetic algorithm parameters and some brief concluding comments are presented in Section 6.

2. THE DEPARTMENTAL CLASS TIMETABLING PROBLEM

This problem involves assignment of faculty members to Studentgroups and their corresponding timeslots. In this domain students are registered to various courses and are grouped to classes based on the similarity of these courses. When a faculty is assigned to a timeslot of a particular Studentgroup, a lot of hard constraints are needed to be satisfied first. When a timetable satisfies all the hard constraints it is a feasible solution. In this paper, the Hybrid Genetic Algorithm is tested with real time constraints provided by the Department of Computer Science and Engineering, SSN College of Engineering. The constraints are classified into two types, Hard and Soft constraints.

The following represent the hard constraints,

- No faculty should take more than one subject at a particular timeslot
- No faculty should be allocated different Studentgroups on the same timeslot
- No more than two double periods(continuous two timeslots having the same course) will be allocated in a day
- The sum of all hours allocated for all subjects in a week should not exceed the total number of timeslots available
- Certain timeslots must be reserved for special activities
- A subject should not be scheduled for a class more than the maximum timeslots to be allocated for a week
- No faculty must be overloaded with more than four timeslots in a day

The following represents the soft constraints taken into account,

- Faculty from other departments should be given topmost priority when allocating timeslots

- Other department faculties should not be allocated a timeslot that should come in the 1st timeslot of the day
- The core courses listed out for a Studentgroup (except other department subjects) will be evenly spread out in the 1st timeslot
- Subjects handled by other department lecturers will be allocated based upon the timeslots requested/given by that lecturer.
- Continuous periods of same course on same day should be split such a way that there exists some timeslots between the two timeslots
- Classes that require special features such as lab must also be taken into account
- Allocation of lab classes to first two hours of a day should be avoided.

The objective of this problem is to avoid the violation of the above stated hard constraints and minimize the violation of soft constraints.

In recent years, several University course timetabling papers have come in literature. In 2002, Socha *et.al* [3] applied an ant based approach to the various datasets. A fuzzy approach to the problem was introduced by Asmuni *et. al.* 2005 [4]. Rossi Doria *et. al*. [5] considered the same datasets and presented a comparison of a number of meta heuristic methods. In [6] Burke *et al* employed a tabu search within a graph based hyper-heuristic benchmark datasets. The aims in all these papers were to raise the level of generality by operating on different problem domains .In 2007, Salwani Abdullah. Edmund burke and Barry Mc Collum presented a paper [7] to solve University Course timetabling problem with hybrid evolution approach. In [8], a randomized iterative method for a local search operation was discussed. The results produced were very accurate. This paper employs the concepts of Hybrid Evolutionary algorithm in conjuncture with efficient data structures for a departmental class timetable to produce accurate results in a very short time.

3. HYBRID GENETIC ALGORITHM

A Hybrid Genetic Algorithm is a combination of Genetic Algorithm and local search operation performed within the loop of the Genetic algorithm. Though Genetic Algorithm is a search technique used in computing to find exact or approximate solutions, the results are often not the best optimal solution but generally "acceptably good" solutions. Hence a local search operation is employed within the loop of the Genetic Algorithm. A local search algorithm [9] moves from solution to solution in a space of candidate solutions until an optimal solution is found. But the result is not always the optimal solution, but local search tend to climb the hill of a search space and produce optimality very quickly. The method described in [1] employed a Memetic algorithm for university examination timetabling where two evolutionary operations were used in initial phase followed by a hill climbing algorithm. A set of Genetic algorithm parameters define the working condition of the algorithm. The Hybrid Genetic Algorithm is applied to a Departmental class timetabling problem. The schematic overview of the algorithm is given below in Figure.1

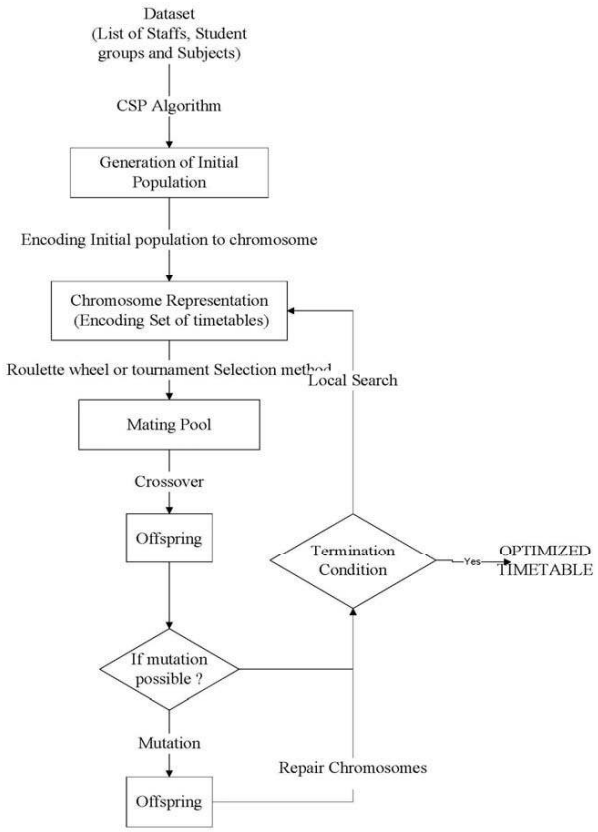

Figure.1 Hybrid Genetic Algorithm for a departmental class timetable

The Hybrid Genetic Algorithm first generates a set of chromosomes (abstract representation of the department timetable). This set of chromosomes generated is collectively called as Initial population. Once the Initial population is generated the chromosomes are encoded into a suitable data structure such as a hash, described later in Section 4. A fitness value is calculated for every chromosome. The algorithm then employs a selection method which chooses a pair of chromosomes and depending on the elitism rate decides to perform crossover and mutation operations or just carry over the chromosome to the next generation. The rate is a probability at which the chromosome will be selected for crossover or reproduction and is random. After crossover and mutation operations are performed, a local search is applied to bring the best possible candidate solution to the top. The local search applied here is a simple sort function due to use of data structures like hash and arrays. These steps are repeated till and end criterion is satisfied. Once it is reached the chromosome with best fitness value is chosen as the optimized timetable.

3.1. Genetic Algorithm operations

The Genetic algorithm operations are Selection, Crossover and Mutation

3.1.1. Selection

Roulette wheel selection [10] is the most preferred selection method which can be imagined as a biased wheel as shown in Figure.2. The Chromosomes in the selection pool are arranged in the roulette wheel based upon their fitness value. Thus when the wheel is rotated to the number of chromosomes present in the population pool the chromosome with the highest fitness has the highest probability of getting selected for crossover or mutation.

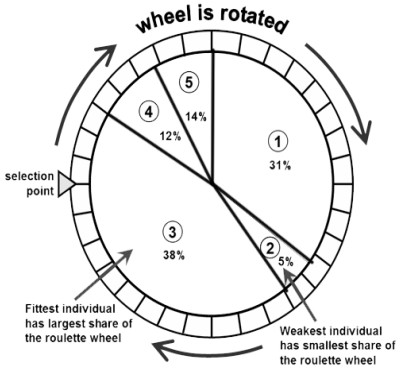

Figure.2 Roulette wheel selection

3.1.2. Crossover

Crossover is a genetic operator used to vary the programming of a chromosome or chromosomes from one generation to the next. One-point crossover randomly selects a point in two chromosomes. Everything after the point is swapped between the parent organisms, rendering two child organisms as shown in Figure.3. Since the population size is fixed, the child will be replacing the first parent. Crossover will not be applied to the best chromosomes. The major fact to be noticed is that when applying crossover operation it may result in staff clashes and renders the timetable infeasible; to prevent this repair strategy is employed to find a feasible timeslot and swap the alleles inside the chromosome.

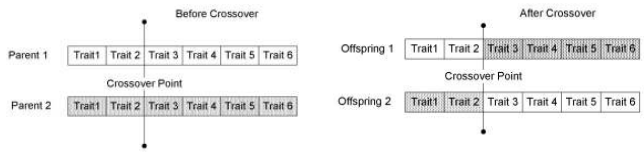

Figure.3 Crossover Genetic operation

3.1.3. Mutation

Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next. The purpose of mutation in GAs is to allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution of chromosomes to a better fitness. In this paper a flip mutation was established where two alleles are flipped changing the composition of the timetable as shown in Figure.4.

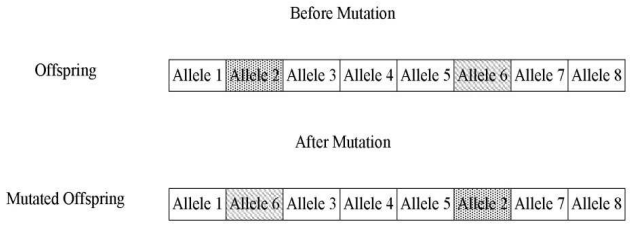

Figure.4 Mutation Genetic operation

4. DATA STRUCTURES USED AND THEIR SIGNIFICANCE IN THE ALGORITHMS

As far as a departmental class timetable is concerned, it involves the automation of the construction of the timetable for individual Studentgroups. The collection of all the individual Studentgroup timetables makes up the entire departmental class timetable. In terms of Hybrid Genetic Algorithm, all data are abstractly represented known as a chromosome. The chromosome is organized in such a way that it can be broken down into traits and further into alleles. The entire departmental class timetables form the chromosome for the Hybrid Genetic Algorithm to work on. The individual class timetable represents the traits of a chromosome. Since the Hybrid Genetic Algorithm assigns faculties such a way that there are no hard constraint violations when traits are exchanged, this allows the application of crossover operation where primarily the traits are being exchanged between the chromosomes. The timeslots under each timetable forms the alleles of the chromosome. The different combination of alleles gives the chromosome its distinct identity shown in Figure5.

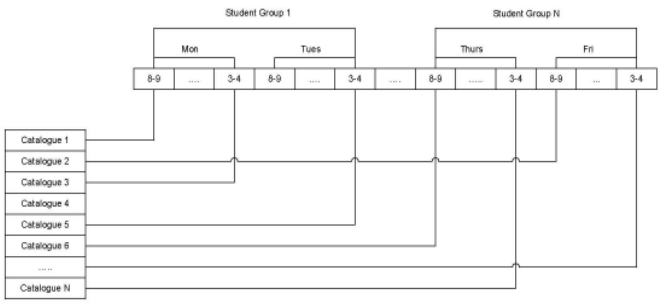

Figure.5 Chromosome representation of a departmental class timetable added to a hash

Due to the organization of data into a hash structure the Hybrid Genetic Algorithm performs operations in such a way that the results are produced very quickly. Crossover genetic operation involves exchange of traits between chromosomes. The data is organized in a manner that individual Studentgroup timetables are distinct and are switched at the crossover point which is determined by a crossover rate. The timetables in Initial population are generated with no hard constraint violations, but it is not the case in crossover operation. Therefore a repair strategy is employed, which looks for a possible timeslot to which course can be exchanged and the hard constraint violation be eliminated. If too many violations persist that particular chromosome is discarded. Since the entire timetable is stored in a hash function, the retrieval of a individual class timetable or individual timeslot be performed by manipulation of array index. This enables a direct addressing and exchange of traits during crossover operation. The use of pointers also helps in a quick exchange of traits.

5. EXPERIMENTS

The approach described in Section 4 was implemented in Visual C++ 2008 under Windows Vista Ultimate in an AMD Dual Core 4000+. The algorithm is run for 20000 iterations till an optimized result is obtained. If there is no improvement in fitness of the best chromosome the algorithm stops execution and presents with the optimal results. The results presented here are best out of 5 run for each dataset. It has been found that time taken to find the accurate solution increases with increase in number of Studentgroups and the variation in number of faculty members or the number of rooms does not affect the time taken to compute and arrive at the optimized result. The algorithm is run under different parameters are presented in Table 1.

Table 1- Parameters values for departmental class timetabling problem

Category	Small	Medium	Large
Chromosome population size	100	100	200
No of Studentgroups	2	6	8
No of Courses	15	54	70
No of Faculty	12	54	58
No of Rooms	2	6	8

The comparison in our approach with these parameters is shown in Table 2. The algorithm is run for 20000 iterations till a stop criterion is encountered.

Table 2- Comparison results with different parameters

Category	Small	Medium	Large
Execution time	15 sec	1 min 32 sec	4 min
Initial population fitness (Best chromosomes fitness)	10	82	134
Optimized timetable fitness	0	4	7
Final generation (Iteration at which algorithm came to halt)	90	200	264

Legend: min : minutes
sec : seconds

From Table 2 we can observe that the use of efficient data structures like hash and arrays has made the Hybrid Genetic Algorithm to work efficiently and produce accurate results. The Fitness value represents the degree to which the solution satisfies the soft constraints. The algorithm has managed to reduce the high fitness in Initial population chromosomes to a minimal number of soft constraint violations. Also it has been observed that the algorithm reaches optimality in very few generations and does not run all 20000 iterations. It can also be noted that the time taken increases exponentially for increase in number of Studentgroups.

6. CONCLUSIONS

From the results obtained we can justify that the Hybrid Evolutionary Algorithm produces one of the most significant and accurate results. The initial fitness values and final fitness values shows the efficiency of the algorithm in optimizing timetable. Also the use of such data structures does not limit the addition of new Studentgroups into the dataset. It can be extended to a University level Class timetabling. Also the final fitness value shows that it is less than the total number of Studentgroups which indicates there might be a violation of one soft constraint per Studentgroup. The performance of the algorithm is improved by the use of efficient data structures and data organization which results in a faster result.

REFERENCES

[1] E. K. Burke, J.P. Newall, R.F Weare, "A Memetic Algorithm for University Exam timetabling", The Practice and Theory of Automated Timetabling I, Springer Lecture Notes in Computer Science Vol 1153, Springer Verlag, 241-256, 1996

[2] B. Patcher, A. Cunning. M G Norman, H. Luchian, "Extension to a Memetic Timetabling System", The Practice and Theory of Automated Timetabling I, Springer Lecture Notes in Computer Science Vol 1153, Springer Verlag, 251-265, 1996

[3] Socha, J Knowles, M. Samples, "A Max Min Ant system for the University course timetabling Problem". In the proceedings of 3rd International Workshop on Ant algorithms, ANTS2002, Springer Lecture Notes in Computer Science Vol. 2463 Springer Verlag. 1-13,2002

[4] B. Patcher available at http:///www.dcs.napier.ac.uk/~benp

[5] O. Rossi Doria, M. Samples, M. Bittari, M. Chiarandini, M. Dorigo, Paetche. L . Paquete and T. Suzzle, " A comparison of performance of different meta heuristics on the timetabling problem", Practice and Theory of Automated Timetabling V, Springer Lecture Notes in Computer Science, Vol 2740, Springer Verlag, 241-256, 1996

[6] E. K. Burke, G. Kendall, E. Soubeiga, "A Tabu –Search Hyperheuristic for Timetabling and Rostering". Journal of heuristics Volume 9, No:6,451-470.2003

[7] Salwani Abdullah. Edmund Burke and Barry Mc Collum, "A Hybrid Evolutionary Approach to University Course Timetabling Problem", proceedings of IEEE, 2007.

[8] S. Abdullah, E. K. Burke, B. Mc Collum, "Using a randomized iterative Improvement Algorithm with composite Neighbourhood structures for the University Course timetabling problem", Metaheuristics International Conference MIC 2005 Vienna, Austria 22nd-26th June, 2007

[9] W.E .Hart. N. Krasnogor, JE Smith, "Memetic Evoultionary Algorithms", Recent Advances in Memetic algorithms: Studies in Fuzziness and soft computing. Springer-Verlag 3-27, 2004

[10] Selection Methods from "Genetic Algorithms in Search, Optimization, and Machine Learning". ISBN 978-0201157673 by David .E. Goldberg.