ISSN : 2394-2975 (Online)
ISSN : 2394-6814 (Print)
Vol. 3, Issue 2 (April - June 2016)
International Journal of Advanced Research in Education & Technology (IJARET)

Comparitive Study of Algorithms for Solving Time Table Generation Problem

[I]Anuja Deshmukh, [II]Mayuri Kandalkar, [III]Prof. Rashmi. P. Bijwe
[I,II]BE Final year CSE,HVPM COET Amravati, Maharashtra, India
[III]Associate Prof, HVPM COET Amravati, Maharashtra, India

Abstract
The aim of this document is to understand the timetable generation problem and all the challenges that may occur while generating a clash free and optimal timetable. Planning timetable is one of the most complex and error prone application. There are still serious problems like generation of high cost of time while scheduling and these problems are repeating frequently. Therefore there is a great requirement for an application distributing the course evenly and without collisions. In this paper we are going to study and compare some popular algorithms that are used for solving this problem. Genetic algorithm , meme algorithm and differential evolution algorithms are the main algorithms that are going to be considered, besides we will also try to understand the heuristic approach to resolve this problem.

Keywords
Genetic Algorithm, fitness function, Timetable Generation, MEME algorithm, Differential evolution.

I. Introduction
Many problems in industrial engineering nowadays concern themselves with the goal of an "optimal" solution. Various optimization methods have therefore emerged, being researched and applied extensively to different optimization problems. For small scale problems, exact solution methods such as linear programming can be used effectively. When the problems are large and complex, however, heuristic methods have to be called into play due to the exponential growth of the search space and the time required to find optimal or near optimal solution. Over the past few decades, researchers have proposed many novel nature inspired heuristic methods such as the evolutionary algorithms (EA) for optimization design based on specific domain knowledge. Two well-developed methods belong to the class of EA are Genetic Algorithms (GA) and Evolutionary Programming (EP). Also the meme algorithm approach which has evolved from genetic algorithm . A simulation of 'natural selection' in meme algorithm takes place by first evaluating the quality of each solution and then selecting the fittest ones to survive to the next generation and pass on meme as a unit of information that reproduces itself as people exchange ideas .

II. Basic Structure of Time Table Generator
The structure of time table generator consist Input Date Module, relation between the input data module, time interval, time slots module, applying active rules and GA module then extract the reports.

a) Input Data
The input data module is described by a type of data from the database. The data contains:
1) Person: Data describe the name of lecturers.
2) Subject: Data describe the name of the courses in the class.
3) Room: Data describe the name of the classes and capacity of each it.
4) Time Interval: It is a time slot with a starting time and duration.

b) Constraints
Constraints can be divided in to three parts:

1)Validity violation constraints: There are the constraints which are needed to be incorporated necessarily otherwise there is no guarantee of valid time tables generated. They are included as a part of initial generation of population, as they cannot be violated.
» Lab lectures are constrained to appear together. If scheduled lab lectures are odd in number, then last three lectures appear together.
» There are certain lectures that may or may not appear at the same time in more than one class.
» The most trivial violation constraint is that a teacher must not clash in two different tables of a Time table.

2) Hard constraints: Hard constraints are the ones which need to be fulfilled necessarily.
» Classrooms must not be double booked.
» Every class must be scheduled exactly once.
» Classes of students must not have two bookings simultaneously.
» A classroom must be large enough to hold each class booked to it.
» Lecturers must not be double booked.
» A lecturer must not be booked when he/she is unavailable.

3) Soft constraints: These are constraints that are not that obvious but still demanding. They not to be really satisfied but the solutions are generally considered good if large numbers of them are taken care.
» No consecutive lectures of the same teacher in a class.
» No consecutive lectures of the same teacher in a class
» Preference to first lecture.
» Courses must be evenly distributed
» Same teacher must not have consecutive periods unless specified.

c) Active Rules
Active Rules are based upon an Event-Condition-Action architecture. The meaning of an ECA rule is: "when an event occurs check the condition and if it is true execute the action". There is an event language for defining events and for specifying composite events from a set of primitive ones. The condition part of an ECA rule formulates in which state the database has to be,

in order for the action to be executed. The action part of an ECA rule usually starts a new transaction which when executed may trigger new ECA rules. The system will then select the rule with the highest priority to fire, or will arbitrarily select a rule to fire if thereis more than one with the same priority.

For proper working of Active Rules they must designed by considering every possibility to avoid erroneous conditions such as :

I. Faculty must not be double booked
II. Class room must not be double booked
III. Batch must not be double booked
IV. Classroom must be large enough to hold each batch booked to it

III. Genetic Algorithm

Genetic algorithms are methods of solving problems based upon an abstraction of the process of Natural Selection. They attempt to mimic nature by evolving solutions to problems rather than designing them. Genetic algorithms work by analogy with Natural Selection as follows. First, a population pool of chromosomes is maintained. The chromosomes are strings of symbols or numbers. There is good precedence for this since humans are defined in DNA using a four-symbol alphabet. The chromosomes are also called the genotype (the coding of the solution), as opposed to the phenotype (the solution itself). In the Genetic algorithm, a pool of chromosomes is maintained, which are strings. These chromosomes must be evaluated for fitness. Poor solutions are purged and small changes are made to existing solutions and then allow "natural selection" to take its course, evolving the gene pool so that steadily better solutions are discovered.
The basic outline of a Genetic Algorithm is as follows:
Initialize pool randomly For each generation
{
Select good solutions to breed new population Create new solutions from parents Evaluate new solutions for fitness Replace old population with new ones
}

An outline of the algorithm is given below:

1) Start: Randomly generate a population of N chromosomes.
2)Fitness: Calculate the fitness of all chromosomes.
3)Create a new population:
» Selection: According to the selection method implemented, select 2 chromosomes from the population.
» Crossover: Perform crossover on the 2 chromosomes selected.
» Mutation: Perform mutation on the chromosomes obtained.
4)Replace: Replace the current population with the new population.
5)Test: Test whether the termination condition is satisfied. If so, stop. If not, return the best solution in current population and go to Step 2.

IV. Memetic Algorithm

Genetic algorithm is a general purpose optimization tool based on Darwin's theory of evolution. It has the capability of producing optimized solutions even when the dimensions of the problem increase and for this reason it has been successfully applied to a wide variety of problems. Genetic algorithm operates on a population of solutions represented by some coding. Each member of the population consists of a number of genes, each of which is a unit of information. New solutions are obtained by combining genes from different population members (crossover) to produce offspring or by altering existing members of the population (mutation). A simulation of 'natural selection' then takes place by first evaluating the quality of each solution and then selecting the fittest ones to survive to the next generation Memetic algorithm (MA) is motivated by Dawkins's notion of a meme as a unit of information that reproduces itself as people exchange ideas [2]. A key difference exists between genes and memes. Before a meme is passed on, it is typically adapted by the person who transmits it as that person thinks, understands and processes the meme, whereas genes get passed on whole. Moscato and Norman linked this thinking to local refinement, and therefore promoted the term mimetic algorithm to describe genetic algorithms that use local search heavily [3]. Radcliffe and Surry gave a formal description of mimetic algorithms [4], which provided a homogeneous formal framework for considering mimetic and GA. According to Radcliffe and Surry, if a local optimizer is added to a GA and applied to every child before it is inserted into the population, then a mimetic algorithm can be thought of simply as a special kind of genetic search over the subspace of local optima. Recombination and mutation will usually produce solutions that are outside this space of local optima, but a local optimizer can then repair such solutions to produce final children that lie within this subspace.

An outline of mimetic algorithm is given below:

1) Start: Randomly generate a population of N chromosomes.
2) Fitness: Calculate the fitness of all chromosomes.
» Create a new population:
» Selection: According to the selection method, select 2 chromosomes from the population.
» Local search: search for the best chromosomes
» Crossover: Perform crossover on the 2 algorithm is better suited to handling the NP hard chromosomes selected.
» Local search: search for the best chromosomes
» Mutation: Perform mutation on the chromosomes obtained with small probability.
3) Replace: Replace the current population with the new population.
4) Test: Test whether the termination condition is satisfied. If so, stop. If not, return the best solution in current population and go to Step 2.

V. Differential Evolution Algorithm

The method of differential evolution can be applied to real-valued problems over a continuous space with much more ease than a genetic algorithm. The idea behind the method of differential evolution is that the difference between two vectors yields a difference vector which can be used with a scaling factor to traverse the search space. As in the genetic algorithm's beginning a random population is chosen, equally over the problem space, and to create the next generation an equal number of donor vectors are created through means of

$$i \quad n : D_i = X_a + F(X_b - X_c) \text{ where } i, a, b, c \text{ are distinct} . (1)$$

where Xb and Xc are randomly chosen and Xa is chosen either randomly or as one of the best members of the population (depending on individual encodings). A trial vector Ti,j is created by choosing between the donor vector and the previous generation for each element (j) according to the crossover rate CR[0–1], for each element in the vector we choose either the corresponding element from the previous generation vector or from the donor vector such that

i, j : if (random < CRkj = Jrand) then Ti,j = Di,j otherwise Ti,j = Xi,j (2)
where Jrand is randomly chosen for each iteration through i and ensures that no Ti is exactly the same as the corresponding Xi. Then the trial vector's fitness is evaluated, and for each member of the new generation, X′i , we choose the better performing of the previous generation, Xi, or the trial vector, Ti.

```
Begin
   Generate randomly an initial population of solutions.
   Calculate the fitness of the initial population.
   Repeat
      For each parent, select three solutions at random.
      Create one offspring using the DE operators.
      Do this a number of times equal to the population size.
      For each member of the next generation
         If offspring(x) is more fit than parent(x)
            Parent(x) is replaced.
   Until a stop condition is satisfied.
End.
```

VI. Comparing The Algorithms

The strength of a genetic algorithm lies in its ability to find a good solution to a problem where the iterative solution is too prohibitive in time and the mathematical solution is not attainable. The way that the Genetic Algorithm works allows it to find this solution in a fast manner. Another value of the genetic algorithm's approach is that there can be many different constraints to the problem based on the specifics of the solution for which one is searching. One of the most noteworthy facets of the genetic algorithm is the Schema Theorem which explains the relationship between groups of similar 'chromosomes' and their fitness. The theorem shows that a group with above average fitness should continue to increase it's fitness over successive generations. Another strength of genetic algorithm's approach is that there exists a proof of convergence for an elitist version of the genetic algorithm .While the concept of the genetic algorithm is not overly complicated, the individual parameters and implementation of the genetic algorithm usually require a large amount of tuning. In genetic algorithm every individual are characterized by a fitness function. After analysis if there is higher fitness then it means better solution and then after based on their fitness, parents are selected to reproduce offspring for a new generation where fitter individuals have more chance to reproduce. The objective of the work is to create a model used to generate the acceptable schedule using probabilistic operators.

The strength of differential evolution's approach is that it often displays better results than a genetic algorithm and other evolutionary algorithms and can be easily applied to a wide variety of real valued problems despite noisy, multi-modal, multi-dimensional spaces, which usually make the problems very difficult for optimization. Another impressive trait of differential evolution is that the parameters CR and F do not require the same fine tuning which is necessary in many other evolutionary algorithms. Differential evolution has been used effectively in many applications on various domains such as neural network learning, digital signal processing, and image processing.

The two algorithms (genetic and mimetic) converge to the optimal results with 100% accuracy when the population size is about 2.5 times the number of items available in the knapsack. When the population size is greater than 2.5 times the number of items, the accuracy of the optimal results obtained becomes stable at the global optimum but this requires more iterations since the size of the search space will increase for both algorithms. Also, a major difference between the algorithms is that the time it takes to complete iteration in mimetic algorithm is much more than the time it takes genetic algorithm to complete the same iteration. This is because of the local search algorithm that is embedded in mimetic algorithm. However, it is safe to state that mimetic algorithm is more efficient and better than genetic algorithm since it is guaranteed to converge to an optimal result within a reasonable amount of iterations regardless of the initial population size. After the development, analyzes the efficiency of a range query over the data that is encrypted by EOB where the proposed OB is used. The main focus was to analyze the searching efficiency in terms of the false positive rate. To do this, the probability distribution of the rate of the width of a bucket to the size of the plaintext space was first analyzed to show that the width of a bucket is not skewed to be extremely large or small. This even-bucket-width property gives the proposed scheme a good querying performance on average. In the proposed OB, the p _ 1 points are randomly uniformly sampled in the plaintext space [0,|M|-1].The width of the ith bucket was determined by the selected points of (i-1)th order and ith order because the width is the difference of these two points. Therefore, to analyze the width of a bucket, it is important to analyze the probability distribution of the position of the selected points

VII. Conclusion

All the three algorithms that were discussed in the paper are capable of solving the timetable generation problem. The three algorithms performance are compared on three factors speed , time required for tuning and calculation and simplicity. Genetic algorithm provides the most simple or easy to understand solution to the problem it is even considerably faster than both the differential evolution and meme algorithm . But unlike the differential evolution it cannot give better results when provided with noisy data and Differential evolution can also be applied to wide variety of problems. Where as the meme algorithm seems to be slow and requiring large amount of iterations due to the local search algorithm embedded in it but provides more efficient output when compared with the Differential evolution and the genetic algorithm.

Refrences

[1]. *P.B Shola (2003), Logic and Program Verification and Development, pp 82.*

[2]. *B´elanger, N., Desaulniers, G., Soumis, F. & Desrosiers, J. (2006). Periodic airline fleet assignment with time windows, spacing constraints, and time dependent revenues, European Journal of Operational Research, 175(3), 1754-1766.*

[3]. *B´elanger, N., Desaulniers, G., Soumis, F., Desrosiers, J. & Lavigne, J. (2006). Weekly airline fleet assignment with homogeneity, Transportation Research Part B: Methodological, 40(4), 306-318.*

[4]. *Stojkovic, M., & Soumis, F. (2001). An OptimizationModel for the Simultaneous Operational Flight and Pilot Scheduling Problem, Management Science, 47(9), 1290-1305*

[5]. *Darrell, W, A Genetic Algorithm Tutorial, Computer Science Department, Colorado State University.*

[6]. *Deitel P.J. and H.M. (2007), JAVA How to Program, Seventh*

Edition. Pearson International Edition.

[7]. Edemenang E. and Ahmed M. (1994), "Algorithm Design Techniques: Case Study the Knapsack Problem". The Journal of Computer Science and its Applications (JCSA). 5(1):57-65.

[8]. Francis A. (2007), Charles Darwin and the Origin of Species, pp 2-94.

[9]. Hristakeva M. and Shrestha D., Solving the 0-1 Knapsack Problem with Genetic Algorithms Computer Science Department, Simpson College

[10]. Kim, K. H. & Kim, H. (1998). The optimal determination of the space requirement & the

[11]. J. J. Grefenstette, editor. Proceedings of the First International Conference on Genetic Algorithms and their Applications.Practice and Theory of Automated Timetabling VI Proceedings of The 6th International Conference on the Practice and Theory of Automa.

[12]. J. J. Grefenstette, editor. Proceedings of the Second International Conference on Genetic Algorithms and their Applications. Practice and Theory of Automated Timetabling VI Proceedings of the 6th International Conference on the Practice and Theory of Auto.

[13]. N. R. Jennings. Coordination Techniques for Distributed Artificial Intelligence. University of London Mile End Rd.London E1 4NS UK, 1995. Om Prakash Shukla, Amit Bahekar, Jaya Vijayvergiya, "Effective Fault Diagnosis and Maintenance Optimization by Genetic Algorithm" Available : http://researchjournals.in/documents/published/2204.pdf

[14]. Optimization by Genetic Algorithm" Available : http://researchjournals.in/documents/published/2204.pdf

[15]. Leon Bambrick Supervisor Dr B Lovell "Lecture Timetabling Using Genetic Algorithms" Available : http://secretgeek.net/content/bambrilg.pdf

[16]. Alberto Colorni, Marco Dorigo "A Genetic Algorithm to solve the time table problem" Available :http://citeseerx.ist.psu.edu

[17]. Sanjay R. Sutar, Rajan S. Bichkar "University Timetabling based on Hard Constraints using Genetic Algorithm" Available : http://research.ijcaonline.org/ volume42/ number15/ pxc3877964.pdf

[18]. Eng.Ahmed Hamdi Abu ABSA, Dr Sana'a Wafa Al-Sayegh," Elearning Timetable Generator Using Genetic Algorithms" Available:https://uqu.edu.sa/files2/tiny_mce/plugins / filemanager/files/30/papers/f18 9.pdf

[19]. Leon Bambrick, "Lecture Timetabling Using Genetic Algorithms" Available: http://secretgeek.net /content/ bambrilg.pdf.

[20]. Evaggelos Nonas, Alexandra Poulovassilis,"optimisation of active rule agents using a genetic algorithm approach pdf" Available : http://ebookbrowse.com/optimisation-of-active-rule-agentsusing-a-genetic-algorithm-approach-pdf-d381872402

[21]. Cite Seerx , Optimisation of Active Rule Agents using a Genetic Algorithm approach,Available: http://citeseerx.ist.psu.edu/ viewdoc/summary ? doi= 10.1.1.56.2599

[22]. Wikipedia, Selection (genetic algorithm), Available: http://en.wikipedia.org /wiki/Genetic_algorithm

[23]. Genetic Algorithms, Conclusion and Future Work Available: http://www.doc.ic.ac.uk/~nd / surprise_96/journal/vol4/tcw2/report.html

[24]. Wikipedia,Mutation (genetic algorithm) Available: http://en.wikipedia.org/wiki/Mutation%28 genetic_algorithm%29