A Genetic Algorithm Approach for a Real-World University Examination Timetabling Problem

Oluwasefunmi T. Arogundade
Institute of System Science, Academy of Mathematics and System Science, Chinese Academy of Science, Beijing, 100190, China.

Adio T. Akinwale
Department of Computer Science, University of Agriculture, Abeokuta, Ogun State – Nigeria.

Omotoyosi M. Aweda
Department of Computer Science, University of Agriculture, Abeokuta, Ogun State – Nigeria.

ABSTRACT
With the introduction of modularity, increasing student numbers and the continued expansion of university departments, space in Nigerian Universities is becoming an increasingly precious commodity. To address this, some institutions have tried to ensure efficient space utilization by employing different proposed solutions to space allocation problems especially during examination period.

A number of approaches have been explored in the casting of examination timetables for academic institutions. The approach to be discussed here applies genetic algorithm using hierarchy of constraints. This hierarchy can incorporate individual requests or organizational requirements by weighing them according to some criteria. In this paper, we present a new real-world examination timetabling dataset at the University of Agriculture, Abeokuta Nigeria that will hopefully be used as a future benchmark problem. In addition, a new objective function that attempts to spread exams throughout the examination period is also introduced. This objective function that taking into account both timeslots and days assigned to each exam, is different from the often used objective function from the literature that only considers timeslot adjacency. Also room capacity for each room is included in the examination datasets specification. This approach has been tested with real data from the university and numerical results is presented and discussed.

General Terms: Scheduling, Algorithms, NP hard

Keywords: Genetic Algorithms, Examination Timetabling, Constraint, NP Complete, Scheduling.

1. INTRODUCTION
Timetabling problems are well known complicated constraint satisfaction problems given to be NP complete. Organizations like Universities, Polytechnics and most educational institutions in general use timetables to schedule classes and lectures, assigning time and places to future events in a way that makes use of the available resources in the most optimal way. Universities in particular increasingly have to deal with a large number of courses and flexible degree structures. Poorly designed timetables are not just inconvenient but also prove costly in terms of time and money.

The difficulty is due to the great complexity of the construction of timetables for exams, due the scheduling size of the examinations and the high number of constraints and criteria of allocation of space.

Timetabling problem has been studied by numerous researchers due to its NP complete nature (Even et.al., 1976). There are varieties of timetabling problem classes on which variety of approaches are used. Most of the research in the area focuses on employee shift timetabling, especially; nurse rostering, course timetabling and examination timetabling [13, 5, 12, 1, 3, and 9].

According to A. Wren, 1996; "Timetabling is the allocation, subject to constraints, of given resources to objects being placed in space and time, in such a way as to satisfy as nearly as possible a set of desirable objectives." Real timetabling problems have many forms like educational timetabling (course and exam), employee timetabling, timetabling of sports events, timetabling of transportation means, etc.

The aim in examination timetabling is to produce the most appropriate schedule for a set of examinations under a given constraint.

This subject has received special attention of the scientific community in the last five decades. This great interest, caused in 1995, the creation of series of conferences PATAT (Practice and Theory of Automated Timetabling) with new editions every two years (PATAT, 1995) and the establishment of EURO (Association of European Operational Research Societies) WATT (Working Group on Automated Timetabling). In 2002 emerged with the support of PATAT, the International Competition of Timetabling (ITC, 2002).

Examination timetabling problems are generating considerable interest from researchers across the fields of Operations Research and Artificial Intelligence. More details about some future directions for timetabling research such as case based reasoning (to solve the problem directly and also to select appropriate heuristics); multi-criteria methods and hyper-heuristics can be seen in [2].

Tabu search approach was explored by [6] using graph coloring based heuristics. Merlot et. al. in his article [7] proposed a hybrid approach for solving the final examination timetabling problem that generates an initial feasible timetable using constraint programming, and then applied simulated annealing with hill climbing to obtain a better solution. Burke et.al. [4] Proposed a general and fast adaptive method that arranges the heuristic to be used for ordering examinations to be scheduled next.

The method of solution to examination timetabling problem in this work is genetic algorithm using hierarchy of constraints. Genetic algorithm is a repetitive process, which consists of constant size population of individual, each one (i.e. individual) being represented by a finite string of symbols, usually equating

to chromosomes in real organisms, encoding a possible solution in a given problem space. This space referred to as the search space encodes all feasible solution to the problem at hand. Generally speaking, the genetic algorithm is applied to spaces which are too large to be exhaustively searched.

The dataset presented here is real data for undergraduate examinations for first semester, at the University of Agriculture Abeokuta (UNAAB), Nigeria 2008. The dataset presented here has been processed which excluded the courses with no exam and modified the original dataset by replacing the appropriate exams accordingly. In this dataset, the total number of examinations is 437 exams with 8,000 students, and the number of exam days is 22 days. The total number of hall used is 42. The capacity of each of the hall is shown in table 1. The examination timetabling dataset presented in this paper has more practical constraints compared to existing examination dataset and the results achieved in real and complex scenarios are satisfactory; hence the exam timetabling meets the imposed regulations.

1.1 The current Examination Timetabling System in University of Agriculture Abeokuta.

The examination schedule is displayed as a grid (usually with days of the week on one axis and time periods on the other). Being a familiar format, it facilitates visual checking for scheduling conflicts, which takes a lot of time. This format allows the person constructing the schedule to input and modify schedule, attempts various options and make changes until the schedule is satisfactory.

Most of the time new schedule is based on the schedule from a previous academic calendar. This kind of unintelligent scheduling system, though having graphical interface is very different from a standard menu or dialog box based system as in the new examination timetabling system implemented in this paper. It requires considerably more creativity in implementation.

This approach may appear simple to implement, it is only usable for very small inputs.

2. PROBLEM STATEMENT

The exam timetabling problem is essentially concerned with scheduling a number of exams into a limited number of timeslots or periods in order to satisfy, as much as possible, a set of specified constraints. These constraints vary from institution to institution. It is often essential that some constraints are completely satisfied. Such constraints are called hard constraints. Usually these constraints relate to operational limitations that cannot be bypassed in the real world, such as the constraint that one person cannot be in two places at once or that there are a maximum number of people that can be accommodated in a particular room. We call a timetable that satisfies all hard constraints a feasible timetable.

Another class of constraints that occur in timetabling problems is those that are deemed desirable, but that are often either difficult or impossible to fully satisfy. This could include providing study time for each student between any two exams, or making more efficient use of rooms. These constraints are usually called soft constraints. Such constraints often determine the quality of a timetable.

In general we would think of a good quality timetable as one that is (firstly) feasible and that (secondly) satisfies the soft constraints to an acceptable level. Of course, the quality of a solution is very much subjective. One institution's idea of a good timetable could very well be a poor timetable for another institution. For example, it may be that one institution insists on having a clear day in between exams for all of its students (i.e. it makes this a hard constraint). Another institution may be more concerned with holding all of the exams as quickly as possible in which case the inclusion of the above constraint would be detrimental to the quality of the timetable.

2.1 Constraints

The constraints considered in this work to schedule examinations are:

1. No student may have two examinations in the same period if examination i and j are scheduled in slot p, the number of students sitting for both examination i and j must be equal to zero i.e.

$$k_{ij} = 0;$$

$$\sum_{i=1}^{L-1} \sum_{j=i+1}^{L} k_{ij} * x(t_i, t_j) = 0 \qquad where\ x(t_i, t_j) = \begin{matrix} 1 & if\ t_i = t_j \\ 0 & otherwise \end{matrix}$$

2. For each timeslot t, the total number of students taking examination must not exceed the maximum seat capacity available for that period. In this case, 3350 seats per slot. Classes ≤ seats for $t\ \varepsilon\ (1, ..., p)$.

The same hall should not be scheduled for two exams at the same time except if the size of the hall is large, and then two different exams can be scheduled.

Exams are spread as evenly as possible throughout the schedule.

All exams must be scheduled

$$\sum_{s=1}^{T} \lambda_{is} = 1 \quad for\ all\ i\varepsilon\ 1, ..., l \quad where\ \lambda_{is} = \begin{matrix} 1 & if\ exam\ i\ is\ assigned\ to \\ 0 & otherwise \end{matrix}$$

In addition, a soft constraint where no student should have to sit for three examinations consecutively is being considered.

In this work, we are mainly concerned with generating examination timetable for the university system using University of Agriculture Abeokuta as a case study. Nevertheless, we tried to formulate the timetabling problem in a general way in order to take into account all the particular requirements of every school in Nigeria. Thus, we have the following data sets:

A set H = $\{h_1, \ldots, h_n\}$ of halls. A hall is a venue of specific capacity.

A set C = $\{c_1, \ldots, c_m\}$ of courses.

A set K = $\{k_1, \ldots, k_k\}$ of classes. A class is a group of students writing the same exam.

A set A = $\{a_1, \ldots, a_l\}$ of exams. An exam is an instance of a list of halls, a list of courses and a list of classes. Each exam has a duration expressed in time slots.

A set T = $\{t_1, ..., t_p\}$ of time slots. Time slots are distributed in d week days and h daily periods.

K = (k_{ij}) AxA is the conflict matrix where each element denoted by Kij, (i,j e$\{1, \ldots, L\}$ is the number of classes taking exams A_i and A_j.

3. METHODOLOGY

The processing algorithm used is the Genetic Algorithm.

3.1 Genetic Algorithm

Genetic Algorithms (GAs) are adaptive methods which may be used to solve search and optimization problems. They are based on the genetic processes of biological organisms. Over many generations, "natural populations evolve according to the principles of natural selection and survival of the fittest", this was first clearly stated by Charles Darwin in "The Origin of Species". By mimicking this process, genetic algorithms are able to evolve solutions to real world problems, if they have been suitably encoded.

A simple genetic algorithm describes the following cycle:

1st Generation of random **n** chromosomes that form the initial population;

2nd Assessment of each individual of the population;

3rd Verification of the termination criteria;

4th It verifies termination criterion - cycle ending;

5th Selection of n/2 pairs of chromosomes for crossover;

6th Reproduction of chromosomes with recombination and mutation;

7th New population of chromosomes called new generation;

8th Go back to step 2.

Genetic algorithm is fairly simple. For each generation it performs two basic operations:

Randomly selects N pairs of parents from current population and produces N new chromosomes by performing crossover operation on pair of parents.

Randomly selects N chromosomes from current population and replaces them with new ones. The algorithm doesn't select chromosomes for replacement if it is among the best chromosomes in population.

And these two operations are repeated until the best chromosome reaches fitness value equals to 1 (meaning that all classes in schedule meet requirement). Genetic algorithm keeps track of M best chromosomes in population and guarantees that they are not going to be replaced while they are among the best chromosomes.

In this work, the population is represented in an array and each chromosome serves as an array index. It is briefly described as follows.

We represent total number of halls as length of array of halls.

We represent total number of courses as length of courses.

Using random generation for crossover, if an array index say 50 is randomly generated in an array of courses, and the number in index is greater than 500, it checks the big hall array for a chromosome less than or equals to 500. If it is equal to 500, it assigns and indicates so that, that hall chromosome will not be assigned again for another course chromosome.

If chromosome is less than 500, there is a check in the big hall array and continues to pick the chromosome hall until it adds up to 500. It then indicates assignment to that hall and courses satisfying all the constraints in section 2.1.

The fitness is generated so as satisfy the new population and it is evaluated as:

```
fitme = (Math.random()+ 0.5);
        if (fitme>1)
      fitme=fitme - 0.3
if (fitme<0.7)
     fitme = fitme + (Math.random()+ 0.2);
```

The fitness has values of 0.5-1.5.

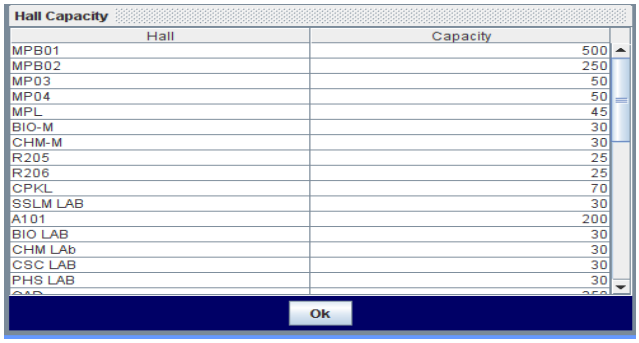
3.2 Experimental Data

In this dataset, the total number of examinations is 437 exams with 8,000 students, and the number of exam days is 22 days. The total number of hall is 42. Hall specifications are shown in table 1 and course specifications are shown in table 2.

3.2 .1 The Hall Capacity Table

This table contain information about the halls and its seat capacity as shown in figure 1.

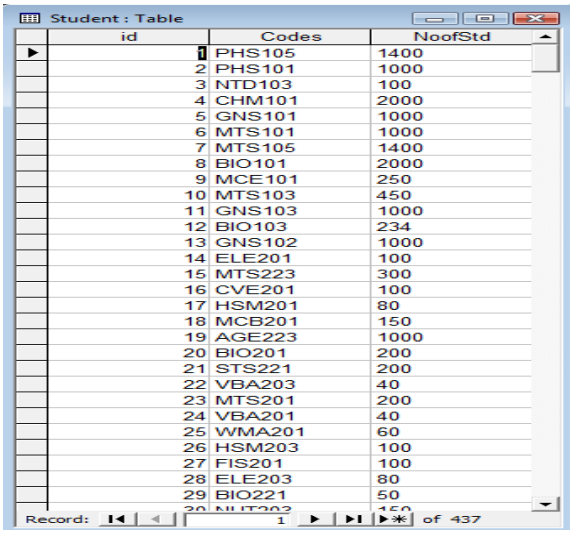
Figure 1: Hall Capacity



3.2.2 The Student Table

This table contains information about courses and the number of student offering each course as illustrated in figure 2.

Figure 2: Student Table



4. IMPLEMENTATION AND RESULTS

The genetic algorithm was converted into java codes using the methodology in section 3.0. The program was tested on IBM PC with hardware configuration of Pentium IV 1GHZ processor, 512Mb Ram and SVGA monitor. The program used hall and course specifications in figure 1 and figure 2 as input. The system output is in six faces because of the length of examination period and the number of courses being offered. The output of this system is quite large, therefore only a phase is shown in figure 3.

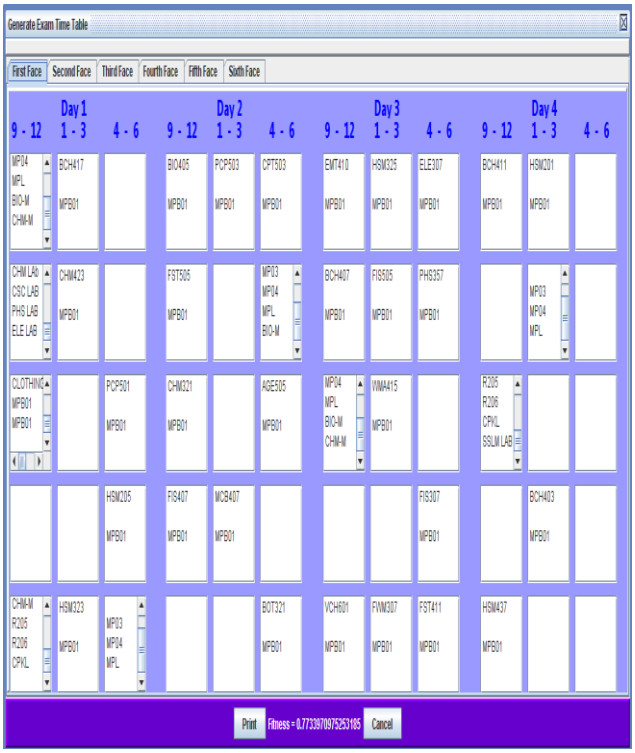


Figure 3: Sample of Examination Timetable Generated

5. CONCLUSION

Many institutions have different ways of computing a timetable that will suit their population size which often has correctable errors. However, genetic algorithm as a tool in artificial intelligence, a branch of computer science, has offered a near to optimal solution to scheduling problems in general and produces little or no error. It has been demonstrated that genetic algorithm is useful in expressing constraint and provides intuitive solution to the constraint with an object oriented language.

In this paper we have introduced a new real-world examination timetabling problem at the University of Agriculture Abeokuta, Nigeria with an objective function that attempts to create gap between examinations for groups of students. To adhere with the practical examination timetabling, room capacity for examination purpose (this is different from sitting capacity for lectures) is also included in examination datasets specification. This exam timetabling system can be used in another institution of higher learning that has the same pattern of time slots. This system also allows large input unlike the system being used which is described in section 1.1. Since the system allows the user to input the resources available in terms of space and courses with the total number of students that will write the exams. This flexibility of domain requirement makes this system ideal for generating exam timetables because it will save time and money.

6. REFERENCES

[1]. Alkan, A. and Ozcan, E. "Genetic Algorithms for Timetabling', Proc. of 2003 IEEE congress on Evolutionary Computation, 2003 pp. 1796-1802.

[2]. Burke and Petrovic, , Recent Research Directions in Automated Timetabling, European Journal on Operational Research-EJOR, vol. 140 (2),2002 266-280.

[3]. Burke E.K and Newall J.P, , "Solving Examination Timetabling Problems through Adaptations of Heuristic Orderings: Models and Algorithms for planning and Scheduling Problems" Annals of Operation Research, vol. 129, no. 1-4, 2004 pp. 107-134(28).

[4]. Burke, E.K., and Bykov, Y. An adaptive fle-deluge approach to university exam timetabling submitted to INFORMS journal of computing 2008.

[5]. Fang, H.L. "Genetic Algorithms in Timetabling and Scheduling', PhD thesis, department of Artificial Intelligence, University of Edinburgh, Scotland. 1994.

[6]. Gaspero, L. Di and Schaerf, A. " Tabu search techniques for examination timetabling", LCNS archive selected papers from the third international conference on practice and theory of automated time tabling, 2000 pp. 104-117.

[7]. Merlot, L., Boland, N.; Hughes, B. and Stuckey P. " Ahybrid algorithm for the examination timetabling problem" Lecture Notes in Computer Science, vol. 2740, Gent, Belgium, Springer-verlag, 2003 pp.207-231.

[8]. Mitchell, Melanie, *An Introduction to Genetic Algorithms.* MIT Press 1996.

[9]. Ozcan, E. "Genetic Algorithms for Nurse Rostering", The 20th international symposium on computer and information sciences 2005.

[10]. Ozcan, E. Final Exam Scheduler (FES), Proc. of 2005 IEEE Congress on Evolutionary Computation, Vol.2, 2005. 1356-1363.

[11]. Petrovic, S., Yang, Y., Dror, M. "Case-based initialization of metaheuristics for examination Timetabling", pp. 137-154; proc. of 1st multidisciplinary int'l conference on scheduling: Theory and Application (MISTA 2003) 2003 pp. 137-154.

[12]. Schaerf, A. " A survey of automated timetabling" artificial Intelligence Review, 13(2):87-127 1999.

[13]. Werra, D. De, "An introduction to timetabling" European Journal of Operation Research, 19 1985 :151-162.