A Multi-objective Genetic Algorithm for University Timetabling Problem

The Core of Termustat AI Project

Arman Jafarnezhad
Department of Computer Science
Kharazmi University
std_arman_jafarnezhad@khu.ac.ir

Keivan Borna
Assistant Professor
Kharazmi University
borna@khu.ac.ir

Arash Ahadi
Assistant Professor
Kharazmi University
arashahadi@yahoo.com

Developing timetables for educational institutes is a complex problem. It is hard and time consuming to solve due to many constraints. Depending on whether the constraints are necessary or desirable, they are categorized as hard and soft, respectively.

Two types of timetables are designed for educational institutes: course and examination. A feasible course timetable could be described as a plan for the movement of students and teachers from one classroom to another, without conflicts. This task is a NP-complete problem. Many attempts have been made using computational methods to achieve optimal solutions to this timetabling problem. Genetic algorithms, based on the theory of evolution is one such method. The purpose of this study and the aim of developing Termustat AI is to optimize a general university course scheduling process based on genetic algorithms using pre-defined hard and soft constraints.

1 Introduction

Timetabling, planning or scheduling is the process of allocating time-slots for planned activities in an orderly manner to bring about an acceptable result, with all constraints satisfied. In an educational institution for instance, two types of schedules can be defined: course and the examination timetables. Courses must be placed into specific time-slots for five or six working days of the week.

A practical timetable in an educational institute is thus a description of the movement of students and staff from one classroom to the next one and the time-slots. In developing a timetable an institute has to face many constraints which could be described as hard and soft depending on whether they are necessary or desirable.

1.1 Hard Constraints

The definition of hard constraints is the limitations that are impossible. Such as a teacher or student being in two different places at the same time. Also, two teachers are not allowed to teach two courses to the same group of students in a time-slot. Thus, neither the staff and teachers nor the students can be in more than one place at a given time-slot. Besides, all the necessary resources should be available for each time-slot. All the courses must be entered in the timetable.

1.2 Soft Constraints

The soft constraints are limitations that are better to consider. Such as the assignment of math courses in time-slots in the morning sessions when students are able to pay more attention to such subjects. Some teachers may not be available in all days of the week. Thus, another useful soft constraint is staff preferences, such as teaching at times and classrooms of choice.

1.3 Complexity

Although manual timetabling is time consuming and difficult, small universities are adapted to generate their own schedules. As the complexity of university increases it to become essential to adopt computer methods to ease the task of scheduling.

When the student population with diverse interests and requirements increases and the teaching programs get complicated, the number of constraints grows significantly, resulting in an exponential rise in the computational time, making it an NP-complete task.

2 Genetic Algorithm

GA is a metaheuristic method, inspired by the process of natural selection used to solve problems which require large search areas for possible solutions. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation. It often depends on adaptive systems to perform well in changing environments. In scheduling, a self-adaptive method is desired to increase the level of generality.

2.1 Fundamentals of GA

A genetic algorithm (GA) is a powerful problem-solving programming method. GA is in a category of evolutionary algorithms which is a subset of artificial intelligence. It was developed in 1960 by Prof. John Holland of the University of Michigan. This technique was inspired by the Darwinian theory of natural evolution, which states that the organisms in the world multiply in geometric proportions leading to a struggle for existence due mainly to limited space, food and other resources. In this struggle the fittest will survive. The fittest are those organisms with favorable variations, the accumulation of which lead to the evolution of species. The chances for the survival of organisms with injurious variations are rather slim. Thus, evolution is a process of natural selection.

Each cell of an organism of a population has a definite number of genetic structures called chromosomes, consisting of linearly arranged single units called genes (blocks of DNA) carrying genetic information for one or many characters.

The full set of genetic material, namely all the chromosomes is called genome. Men for example, have 23 pairs of chromosomes. A group of genes in a genome is called the genotype. The expression of a genotype is referred as the phenotype. Each gene has a definite position in the chromosome called the locus. A given gene can be in several states called alleles which express characters such as the eye color of the organism. When organisms reproduce, the resulting offspring will have half the genes from each parent. Due to cross-over and mutations of genes offspring may get new features, favorable or unfavorable for survival and reproduction.

In GAs, a chromosome represents a data structure. As in nature, new chromosomes are generated by mixing genetic material by cross-over and mutation. Cross-over is equivalent to mating in the biological process. New information is introduced to the population by way of mutation. Thus, it is the chromosome that changes by changing the order and the make-up of the genes.

2.2 How GA Works

Genotype is referred as the phenotype. Each gene has a definite position in the chromosome called the locus. A given gene can be in several states called alleles which express characters such as the eye color of the organism. When organisms reproduce, the resulting offspring will have half the genes from each parent. Due to cross-over and mutation

2.2.1 Initialization

The process begins with initialization, wherein an initial population of candidate solutions is generated. There are many different methods of initializing populations, but with Genetic Algorithms the most popular method of initialization is simply to create a population of randomly initialized strings. Once the initial population has been created, the evolutionary generational cycle begins.

2.2.2 Selection

At each generational step, a pool of parents is chosen from the parent population, based on the fitness values of each individual using a selection mechanism, such that the fittest individuals will have a higher chance of passing on genetic material to subsequent generations (children). This selected population (known as the parent population) then forms the basis for all subsequent optimizations during the generational step.

2.2.3 Variation

As soon as the parent population is fully populated via the selection process, a child population is created which will form the basis of the next generation. This child population is generated by variation operators, which are performed on individuals from the parent population. The most prominent such methods are crossover and mutation.

2.2.4 Crossover

The crossover method takes pairs of parents from the parent population (usually random selection with replacement, such that the same parent can be selected multiple times to produce children). These parents then have some of their genetic information swapped between them. Crossover of two parent chromosomes is usually done in a single-point manner, where a single crossover point on both chromosomes is randomly selected and

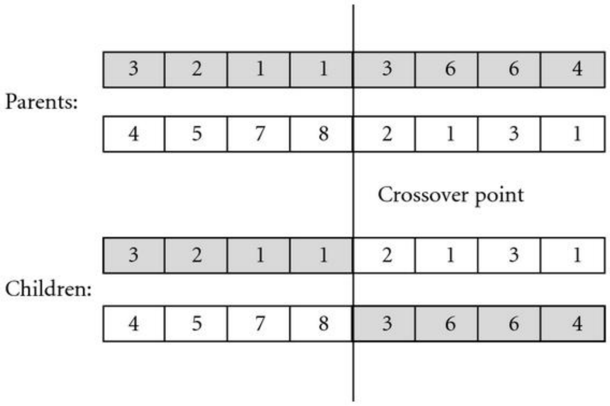

Fig. 1. Single-point crossover of two GA chromosomes

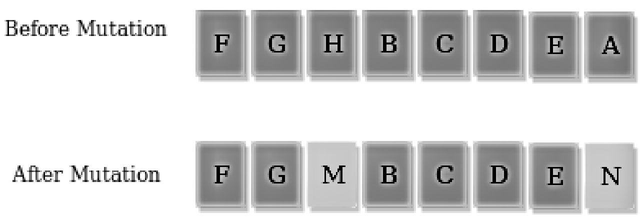

Fig. 2. Bit-flip mutation of GA chromosome

then all subsequent genetic information is swapped between individuals, as shown in Fig. 1.

Crossover is typically performed on randomly selected pairs of parents from the parent population until the new child population reaches the same size as the original parent population.

2.2.5 Mutation

While crossover simply swaps pre-existing information between pairs of candidate solutions, mutation in Evolutionary Algorithms is typically the standard method of introducing new genetic material into the population. Once the child population has been created via crossover, mutation in canonical Genetic Algorithms then occurs on all children in a bit-flip fashion by randomly changing codons on the chromosome between 0 and 1 (or letters, as shown in Fig. 2).

2.2.6 Evaluation

Once the child population has been created, all children then need to be evaluated in order to assign a fitness value by which they can be judged against their peers. This fitness is used to sort/rank a population and to impose probabilities for both selection and replacement phases of the search process, such that the fittest members of the population have a higher probability of passing on genetic material.

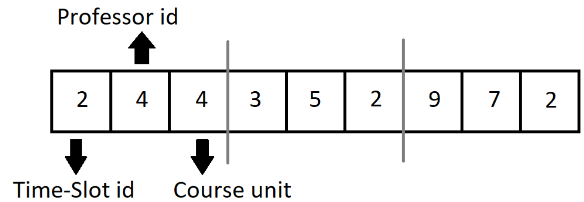

Fig. 3. Array is split into three to retrieve information for each course

2.2.7 Replacement

The final act of the generational step is to replace the old "N" population with the new "N+1" child population. While many methods exist, the most typical replacement method is Generational Replacement, wherein the entire previous generation is replaced with the newly generated child population.

2.2.8 Termination

A Genetic Algorithm will typically terminate after a predefined number of generations, or if some stopping criterion has been met (e.g. fitness is above some threshold, error rate is below some threshold, etc). The fittest solution in the population is then returned as the overall best solution.

3 Using GA for Timetabling

Scheduling classes for a university timetable is often met with hard and soft constraints due to diversity as compared to a school timetable where the requirements are highly limited. The class-scheduling problem will be based on the available professors, time-slots and student groups.

Each course will be assigned a time-slot, a professor and a student group by the class scheduler. The total number of classes that needs to be scheduled can be calculated by summing the number of student groups multiplied by the number of modules each student group is enrolled in.

The encoding must be able to encode all the class properties required. The class properties are the time-slot the course is scheduled for, the professor teaching the course and the unit for the course (In Iran's higher education system, courses have different weights. For instance, general math has a unit of 4 whereas general management has a unit of 2).

A numerical string can be allocated to each timeslot, professor, and the unit. A chromosome can then be used that encodes an array of integers to represent each class.

Timeslot represents the day of the week and time that a class takes place. It contains the time-slot-id and the time-slot details.

Professor accepts a professor-id and professor name properties. It also makes an allowance to retrieve this information as well.

Course unit is an integer between 1 (the least important courses) and 4 (the hardest courses).

4 Conclusion

With the power of genetic algorithms, Termustat AI scheduler finds the best timetable possible that satisfies a number of constraints. Also, the mutation method used guarantees that the mutated chromosomes remain valid. This was achieved by creating a known valid random individual, swapping genes with it similar to uniform cross-over. Use of uniform cross-over and tournament selection completed the genetic algorithm. The factors such as population size, mutation rate, cross-over rate, elite individuals, and tournament size were considered for the course scheduler. The optimum values for these factors were obtained through several runs of the system. Also, the effect of mutation rate and the population size were studied for the course scheduler.

Acknowledgements

Termustat AI and this paper are still on development. The most recent verion of Termustat AI demo is available at http://termustat.ir/ai.