GENETIC ALGORITHM AND UNIVERSITY TIMETABLE PROBLEM

B. Midodashvili[1], L. Midodashvili[2], P. Midodashvili[3]
1-I. Javakhishvili Tbilisi State University
2-Gori University, Georgia.
3- Ilia State University, Tbilisi

Summary

Scheduling university timetable is a complex NP-hard problem, which is usually done "by hand", taking several days of routine work. There are known many attempts to solve this problem using classical methods, such as integer programming and graph theory algorithms. These methods are inconvenient to algorithmize the process of solution. We offer a solution to this problem using appropriately configured genetic algorithm. The program presented by authors, using real university data stored in a SQL database, successfully solves the university timetable problem.

Keywords: genetic algorithm. Timetable problem. Optimization. Computer program.

1. Introduction

Timetable problems occur in cases when certain resources must be specifically allocated to time period slots (airport terminal and railroad station scheduling, conference timetabling, job scheduling, etc.). University timetable problem is a problem of assigning courses to time periods and to rooms, satisfying various constraints and objectives. In general, the constraints may be hard constraints, i.e. conditions which must be necessarily fulfilled, or soft constraints, i.e. conditions which satisfaction can be assumed as expectations.

Timetable problems belong to a class of combinatorial type NP-hard constrained optimization problems, which main goal is to satisfy all problem constraints, rather than optimizing a certain objective [1]. Automated algorithm of the solution of timetable problems is of great importance, as it can save a lot of work to institutions and companies.

Many authors have proposed different methods for solving timetable problems. These methods come from such scientific disciplines as Operations Research, Artificial Intelligence and Computations and they can be divided into four categories [2]:

- Sequential Methods: graph problem method of ordering of events using domain-specific heuristics without constraint violations.
- Cluster Methods: constructing event sets satisfying the hard constraints as well as soft constraints which are then assigned to real time slots.
- Constraint Based Methods: modeling the problem into a set of variables (events) to which values (resources such as teachers and rooms) are assigned by satisfying a number of constraints.
- Meta-heuristic methods: nature-inspired processes such as genetic algorithms (GAs), simulated annealing, tabu search, and other heuristic approaches that are applied to solutions or populations of solutions, in order to evolve them towards optimality.

As the authors of work [3] report, while various methods like tabu search, simulated annealing, network flow, graph coloring, etc, have been on the play, genetic algorithms prove more effective in solving timetable optimization problems.

2. Genetic algorithm

Genetic Algorithm (GA), developed by John Holland [4], is a robust and efficient search and optimization techniques inspired by the Darwin's theory of natural evolution; the original goal for the research was to explain the adaptation of natural systems and to design artificial system that retains mechanism of natural systems.

The evolution process in the genetic algorithm is done with a population of individuals represented by chromosomes, parameters encoded to the string, bits or other data representation.

Since the first population does not have the final or "good enough" solution, there is a need for keeping an artificial diversity in the population. Diversity can be maintained by using the crossover and mutation operations.

The crossover in the natural evolutionary process means that child will inherit its properties (genes) from its parents. In genetic algorithms, the crossover operation is needed to mix and inherit good gene combinations from the current population to the new population.

The mutation is performed by applying a random change to the individual's chromosomes. A mutation usually affects only few genes.

Usually the genetic algorithm performs with the following cycle:

1. Evaluate the fitness value for all the individuals in current population.

2. Create new population by using crossover; mutation and reproduction operations.
3. Discard the old population and continue iteration.

3. Implementation of the genetic algorithm for university timetable problem

The objects of the offered genetic algorithm consist of the following:

- Professor - with ID and the name.
- Students Group - with ID and the number of students (size of group)
- Classroom - with ID and the name of the classroom, as well as the number of seats and information about equipment (computers).
- Course - with ID and the name of the course.
- Subject - with ID and the name of the subject.
- Class - with a reference to the subject and course to which the class belongs, a reference to the professor who teaches, and a list of student groups that attend the class. It also stores how many seats (sum of student groups' sizes) are needed in the classroom, if the class requires computers in the classroom, and the duration of the class (in hours).

We consider only hard constraints consisting of the following:

- A class can be assigned to a free room.
- No professor or student group can have more than one class at a time.
- A room must have enough seats for all students.
- The room must have equipment (computers) if the class requires it.

If the total number of classes is L, then a chromosome for a class schedule is represented by hash-table with L items, where for each item the key is the Class ID (integer from 1 to L) and the value is the number of the time-space slot in the vector, to which belongs the first hour of this class (integer from 1 to working_days * number_of_rooms * classes_per_day).

The fitness of the chromosome is calculated as follows:

- Each class can have 0 to 8 points.
- If a class uses a free room, we add 1 to its score.
- If a class requires equipment and the room assigned to is equipped, we increment the score of the class.
- If a class is located in a room with sufficient number of seats, we increment its score.
- If a professor has no other classes at the time, we increment the class's score.
- If any of the student groups that attend the class has no other class at the time, we increment the score of the class.
- If duration of the class is one hour or all time-space slots of the class with duration exceeding one hour are allocated in one working day, we increment the score of the class.
- If preferred room of the class is not pointed or if it coincides with preferred room, we increment the score of the class.
- If preferred University Building of the class is not pointed or if it coincides with preferred University Building, we increment the score of the class.
- The total score of a class schedule is the sum of points of all classes.
- The fitness value is calculated as schedule_score/maximum_score, where maximum_score is total_number_of_classes * 8.

The fitness value is in the range 0 to 1.

A crossover operation combines data in the hash-tables of two parents, and then it creates a new hash-table. A crossover 'splits' hash-tables of both parents in parts of random size. The number of parts is defined by the number of crossover points (plus one) and in our case it is 2. Then, it alternately copies parts from parents to the new chromosome.

A mutation operation takes a class randomly and moves it to another randomly chosen slot. The number of classes which are going to be moved in a single operation is defined by the mutation size and in our case it is 2.

For each generation, consisting of N=100 chromosomes, the algorithm performs the next operations:

1. Selects N/5 best chromosomes of the population.
2. Randomly selects N/5 pairs of parents from the current population, produces N/5 new chromosomes by performing a crossover operation on the pair of parents and replaces randomly selected N/5 not best chromosomes in population.

3. N/5 times randomly selects 2 pairs of parents from the best chromosomes of the population, produces new chromosome by performing a crossover operation on the pair of parents and replaces randomly selected not best chromosome in population
4. Randomly selects N/5 chromosomes from remaining chromosomes, performs mutation with mutation size=2 and replaces randomly selected N/5 not best chromosomes in population.

The algorithm is repeated until the best chromosome reaches a fitness value equal to 1.

4. Conclusions

The algorithm described above has been applied to the program written in VB.NET using university data stored in a MS SQL 2008 database and successfully tested with different parameter values. In contrast to known to us works, together with best chromosomes in each generation we necessarily retained their offspring (operation 3 of the algorithm). This circumstance is a certain analog of natural evolution of natural systems. As experiments showed this approach sharply decreases a number of generations needed for receiving a solution.

Genetic algorithms appear to find a good solution for university timetable problem, however the rate of the algorithm highly depends on the way the problem is encoded and which crossover and mutation methods are used.

References:

1. Brailsford S.C., Potts C.N., Smith B.M. Constraint satisfaction problems: Algorithms and applications. European Journal of Operational Research, vol. 119, 1999. pp. 557–581

2. Kazarlis S., Petridis V., Fragkou P. Solving University Timetabling Problems Using Advanced Genetic Algorithms. Proc. 5th Int. Conf. on Technology and Automation, Thessaloniki, Greece, 2005, pp. 131-136

3. Nandhini M., Kanmani S., Gilbert S., Theepan S., Venkatesan K. Automated Course Timetabling Using Gam-6. Intern. Conf. On Information Science And Applications, *ICISA* 2010, India, 2010

4. Holland J. Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor. 1975.

გენეტიკური ალგორითმი და უნივერსიტეტის სასწავლო ცხრილის პრობლემა

ბიძინა მიდოდაშვილი[1], ლევან მიდოდაშვილი[2], პ. მიდოდაშვილი[3]
1-ივ. ჯავახიშვილის სახ. თბილისის სახელმწიფო უნივერსიტეტი,
2-გორის უნივერსიტეტი, 3-ილიას სახელმწიფო უნივერსიტეტი

რეზიუმე

უნივერსიტეტის სასწავლო ცხრილის შედგენა წარმოადგენს NP-რთულ პრობლემას და მისი გადაჭრა, ჩვეულებრივ, ხორციელდება „შეუიარაღებელი ხელით", რაც მოითხოვს არცთუ ხანმოკლე დაძაბულ შრომას. ცნობილია აღნიშნული პრობლემის კლასიკური მეთოდების გამოყენებით გადაწყვეტის არაერთი მცდელობა, მაგრამ ასეთი მეთოდები არაა მოსახერხებელი ამოხსნის მიღების პროცესის ალგორითმიზაციისათვის. ავტორების მიერ წარმოდგენილია პროგრამა, რომელიც იყენებს რეალური უნივერსიტეტის Ms SQL მონაცემთა ბაზაში შენახულ მონაცემებს და წარმატებით წყვეტს უნივერსიტეტის სასწავლო ცხრილის პრობლემას.

ГЕНЕТИЧЕСКИЙ АЛГОРИТМ И ЗАДАЧА СОСТАВЛЕНИЯ РАСПИСАНИЯ ЗАНЯТИЙ В УНИВЕРСИТЕТЕ

Мидодашвили Б.[1], Мидодашвили Л.[2], Мидодашвили П.[3]
1-Тбилисский Государственный Университет им. Ив.Джавахишвили,
2-Горийский Университет, Грузия
3-Государственный Университет им. Ильи, Тбилиси

Резюме

Задача составления расписания занятий университета является NP-сложной задачей, которая обычно выполняется "вручную", занимая несколько дней рутинной работы. Известны многие попытки разрешить эту задачу с использованием классических методов, таких как целочисленное программирование или теория графов. Эти методы неудобны с точки зрения алгоритмизации процесса решения. Мы предлагаем решение этой задачи с использованием генетического алгоритма конфигурированного соответствующим образом. Программа, представленная авторами, используя реальные данные университета, которые хранятся в базе данных Ms SQL, успешно решает задачу расписания занятий университета.