Project Documentation

Date: 2017/11/30

Topic: Genetic Algorithm Course Scheduling

Task we did today: Discussed about the crossover method, and what constraints we can add to the initial problem, and what the fitness score is composed of. We also had a plan on the object model and data structures we would use to represent each essential genetic algorithm form.

Task tomorrow: Discuss about the format of input and output. Start coding the elementary classes, including course class and classroom.

**Objects:**

Course Class: Each course class is attended by a certain number of students. It contains a class ID, the duration in hours, and the number of the students.

Classroom: A classroom is a place where a course class takes place. It contains a room ID, and the total seats of the room.

**Chromosome:**

It is a list which contains all the time slots (splitted down to hours) of all the classrooms sequentially. Each element of the list contains one class and one classroom. Assume that courses are only arranged between 8 am and 9 pm, from Monday to Friday.

A hashmap is used to record each course class in a schedule. The key is class ID, and the value is the first time slot that this course class begins. It is used to calculate the fitness score.

A chromosome would contain the number of the crossover points, which is used in the crossover process.

A chromosome also contains a number that indicates the mutation probability. A mutation is randomly occurred after the offspring is generated driven by the mutation probability.

**Fitness:**

A Fitness Method gives every schedule candidate a score according to the constraints we add to the problem, in order to find the candidates with highest scores. In each generation, only 10% percent of candidates will be remained as the parents of the next generation.

Each course class has a fitness point from 0 to 2:

* If a course class uses a spare classroom, we add 1.
* If a course class uses a classroom with enough seats, we add 1.

Then we accumulate the class points of the schedule candidate.

* Fitness score = schedule score / the highest possible score

**Crossover:**

In each generation, only 10% percent of candidates with highest fitness scores will be remained as the parents of the next generation.

We already know that each chromosome represents a schedule candidate. Each chromosome has a number of crossover points, which split the chromosome into number of crossover points plus one parts. Every time we generate the next generation of number N, we random the position of the crossover points (in other words the length of chromosome parts), and an offspring is generated by alternatively copying the chromosome parts of the father and mother. And we do this random for N times. Here we need to emphasize that we use the hashmap of a schedule to crossover.

**Mutation:**

After each offspring is generated by crossover, we randomly mutate the offspring by randomly pick a couple of course classes in its schedule and put it to another random time slot of a random classroom. The number of the classes we pick is defined by another parameter called mutation size.

**Reference:**

<https://www.codeproject.com/Articles/23111/Making-a-Class-Schedule-Using-a-Genetic-Algorithm#Introduction0>