**Project Report**

001830082 Lintong Yang

001206171 Haoran Han

**Introduction**

In computer science and operation research, a genetic algorithm is a metaheuristic inspired by the process of nature selection that belongs to the larger class of evolution algorithm. Genetic algorithm are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection. The goal of the project is to develop a genetic algorithm and to use it to find a good solution to a highly complex problem (a good solution will not necessarily be the best possible). In this report, I will introduce and explain our project.

**Genetic Algorithm Implement**

In GA, we usually using binary encoding to mimic genetic coding. After original population is generated, in accordance with the principle of survival of the fittest and the evolution, each subsequent generation evolution produce better approximate solution. In each generation, according to the individual fitness size to choose appropriate individuals. And by means of natural genetics, genetic operators are combined to crossover and mutation. And finally produced new set of the population to represent new solution set.

**Implement Steps**

(1) The fixed length chromosome is used to represent the problem variable field, and the number of chromosome population is N, the crossover probability is C, and the mutation probability is M.

(2) Define an adaptive function to measure the performance or adaptability of a single chromosome in a problem domain. The adaptive function is the basis for selecting a pair of chromosomes during reproduction.

(3) Generate a population of randomly chromosomes of N.

(4) Calculate the adaptability of each chromosome.

(5) Select a pair of chromosomes in the current population. The probability of parental chromosome selection is related to its adaptability. Highly adaptable chromosomes are selected for higher probability than those with lower fitness.

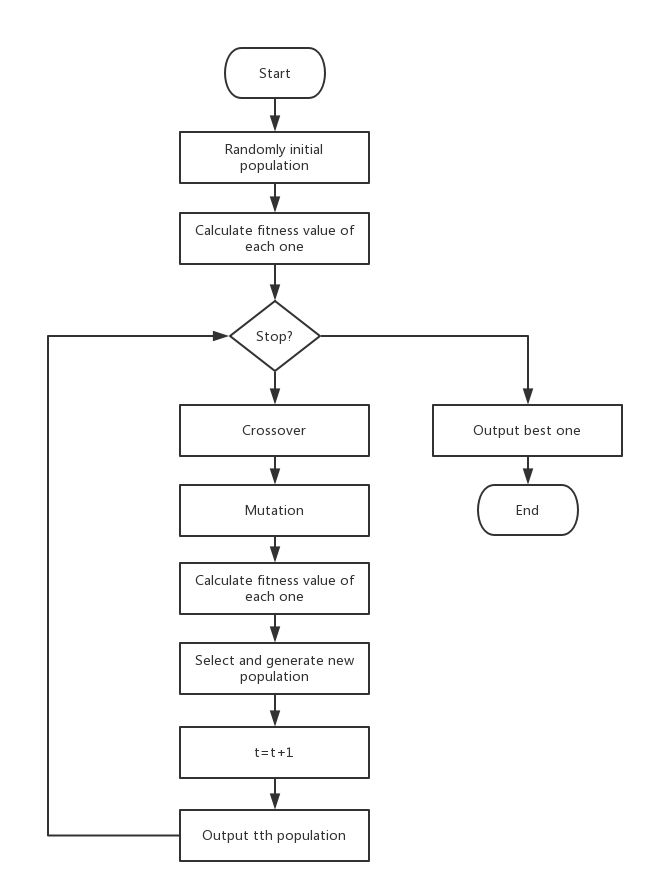
(6) A pair of offspring chromosomes are produced by means of genetic manipulation, crossover and mutation.

(7) Put the offspring chromosomes into the new population.

(8) Repeat step 5 until the new chromosome population size is equal to the initial population size N.

(9) Replace the initial (parent) chromosome population with the new (offspring) chromosome population.

(10) Go back to step 4 and repeat the process until the termination conditions are met.



**Fitness Function**

We use a 10-dimensions trigonometric function as fitness function. We use this function to find the largest value.

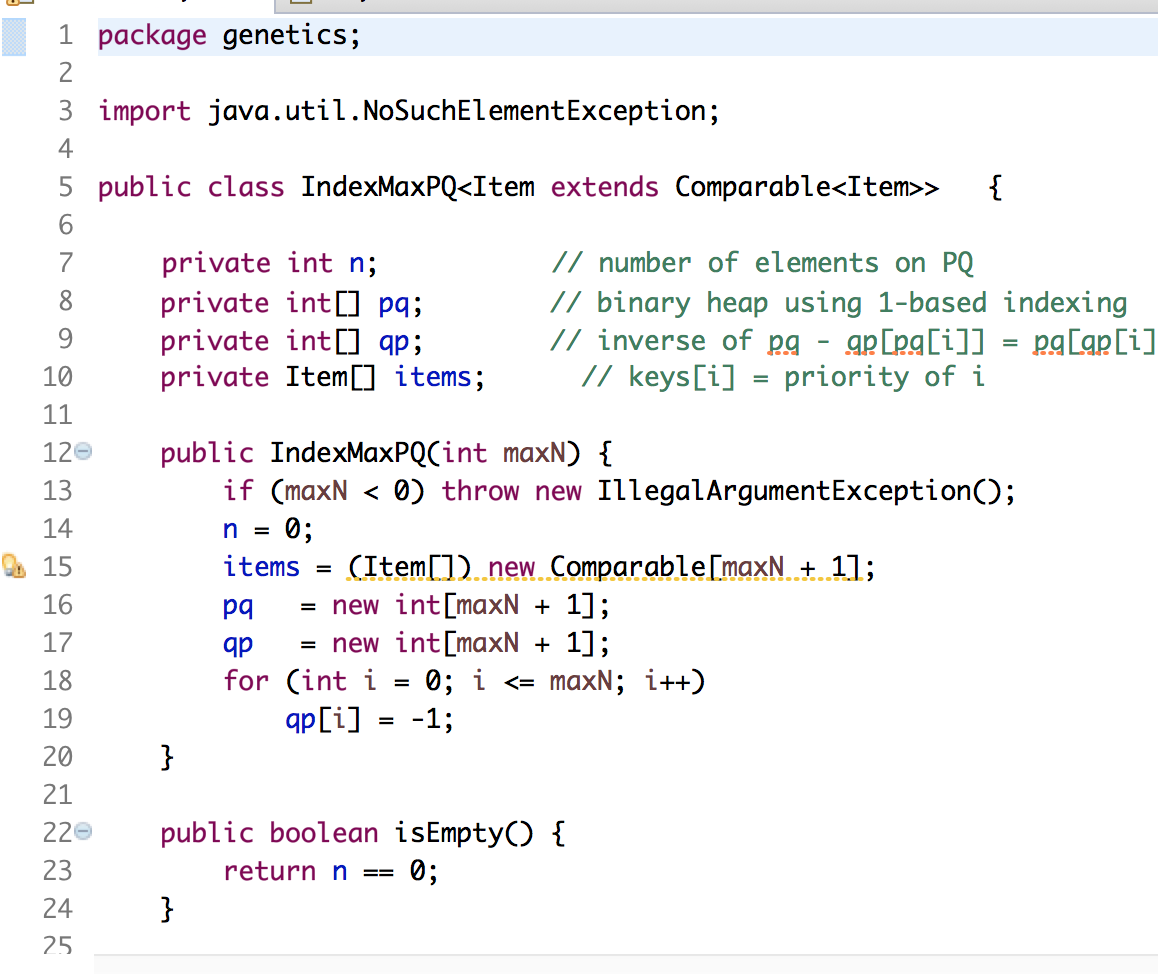
F(x1,x2,x3,x4,x5,x6,x7,x8,x9,x10) = 3-sin2x1+sin2x2-sin2x3+sin2x4-sin2x5+sin2x6-sin2x7+sin2x8-sin2x9+

sin2x10

**Code Implement(Genetic Code and Gene Expression)**

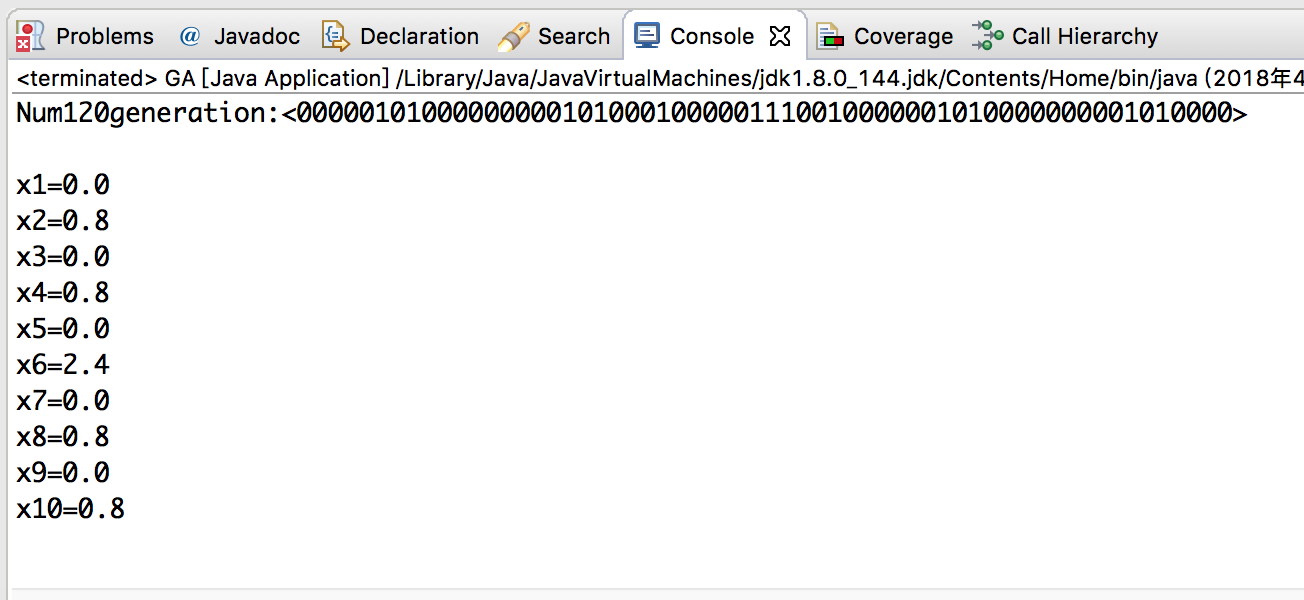
In this project, we choose the n-dimensions function as fitness function to solve. At first, we use the 60 binary codes to represent each chromosome. Each binary code represents a gene and the whole string represent an unit. We set some attributes of each chromosome and we set the method “calculatefitnessvalue”to calculate each unit`s fitness value. In this method, we transfer our binary string to decimal number. After the method, we can do the crossover and mutation processing.

We set the beginning population as 1000. After the crossover, we calculate each chromosome`s fitness value and do the select function. We use index priority queue to implement the select function. We use priority queue to store the set of all chromosomes and sort them by fitness value. And we select the better 500 chromosomes to replace the worse 500 chromosomes in the old 1000 set of chromosomes. For the crossover function, a pair of two chromosomes generate two new chromosomes and we can keep the population as 1000. And we repeat iteration to output the best value of the population and the best fitness value of fitness function. This photo shows the index priority queue code.



**Experiment Result**

After we do the test of our code, we find the solution of our GA



**UniTest**

We also implement the unitest of our code. We write the test code base on every method we used in our GA. We test each part of our GA and these part pass the test.

