Maze Game

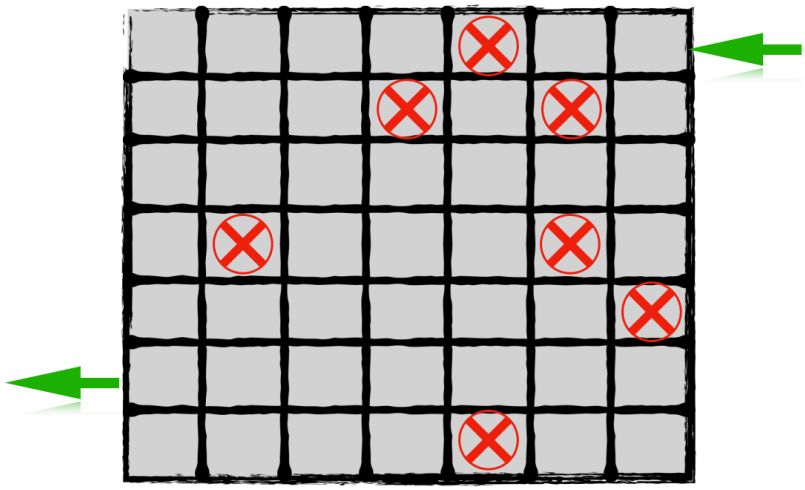
Application of Genetic Algorithm in the Maze Game

Xiao Liu & Jianwei Zhang

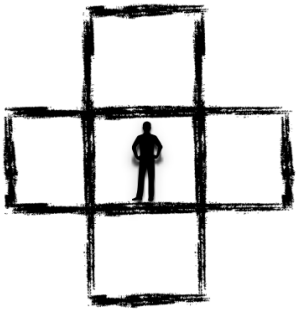
2018 Fall

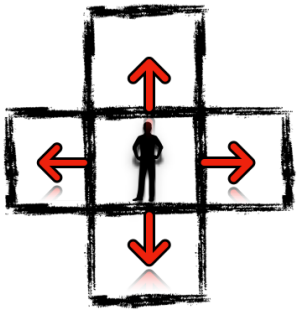
1. Problem Description

Genetic algorithm is commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspire operators such as mutation, crossover and selection. A person in the maze game(7x7) tries various paths to reach the destination with the given start point and the maze. In the process of reaching the endpoint, the next step will be decided by mutation, crossover and selection until the optimal path is found.



There are five options to select when deciding the next direction. The work pattern is based on the different choices, which is written in each gene. Before he makes a single movement each time, he will read the gene inside his chromosome, and he must follow the gene to make action completely.





2. Model Detail Design

2.1 Maze design

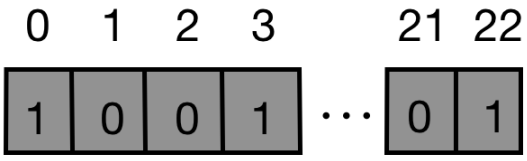
There are four kinds of grids, which barricades are represented as -1, paths are represented as 0, the starting point is represented as 1 and the endpoint is represented as 2. And a person is only able to move within this maze.

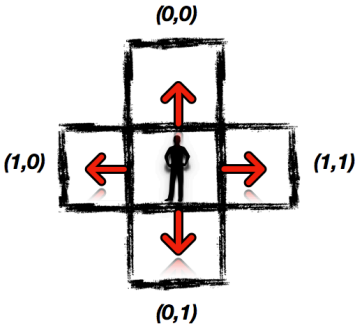
2.2 Genotype & Phenotype

A standard representation of each gene is as an array of bits. The best property that makes these genetic representations convenient is that their parts are easily aligned sue to their fixed size, which facilitates the crossover operation. And the optimal solution is 11 steps, therefore, each chromosome has 11 genes, which represent 11 steps. Each movement is represented by 2 bits. It has 2 * StepNums bits, like the following array. For example, $(0,0)$ represents as **up**, $(0,1)$ represents as **down**, $(1,0)$ represents as **left**, $(1,1)$ represents as **right**.

$$\{1,0,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,1,1,0,0,1\}$$

{left, down, left, left, down, left, left, down, down, down, left, down}





2.3 Fitness Design

Individual movements are selected through a fitness-based process, Certain selection methods rate the fitness of each movement and preferentiality select the best solutions. It is important to select a proper fitness function, since it affects the quality of generations. According to the Genotype and Phenotype for each generation, the distance between the current position and the endpoint (5,0) is acted as fitness value to rate

the movement. The closer they are away from the endpoint, the higher fitness value they are. We could use following function for fitness:

$$fitness = \frac{1}{|x - x_{end}| + |y - y_{end}| + 1}$$

The value range of fitness is $(0,1]$, where the 1 is the maximum value we want get when the point is moving out from the Maze.

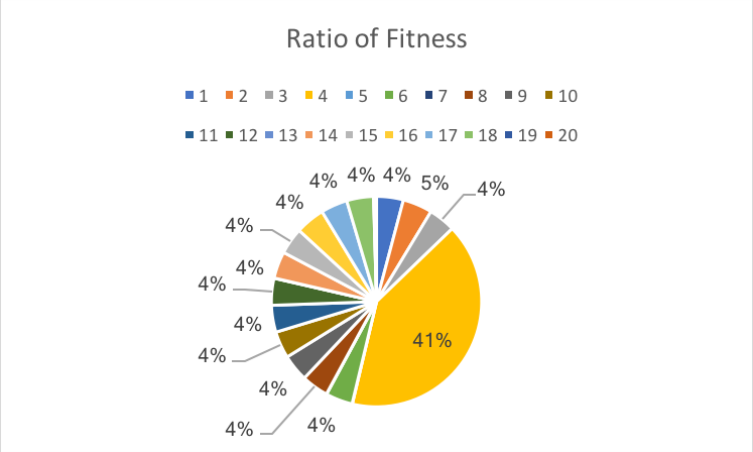
3. Evolution Process

3.1 Initialization

The initial population is 40, which includes the all ranges of possible solutions. And each has genes of 22 bits, including 0 and 1. Each action is represented as 2 bits. The maze is generated by reading from data file, the starting end and the endpoint is invariable, which are (0,6) and (5,0).

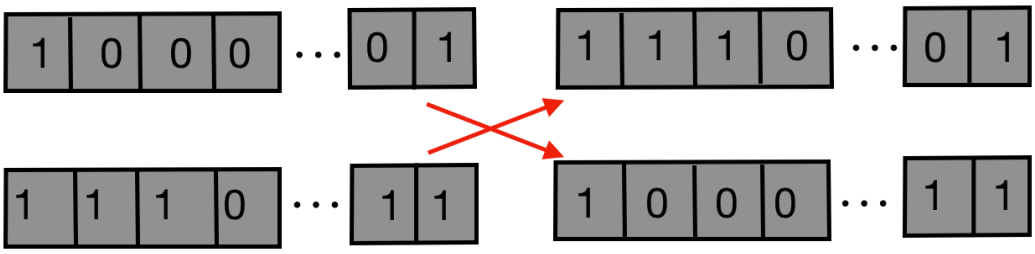
3.2 Fitness Selection

The ratio of area they have in the pie is decided by the score percentage among total points of the generation.



3.3 Crossover

When generating the next generation, a random bit will be selected as a pivot. A part of one chromosome before the pivot is exchanged to another chromosome. Meanwhile, the gene of another chromosome with the same position is delivered to former chromosome.



3.4 Mutation

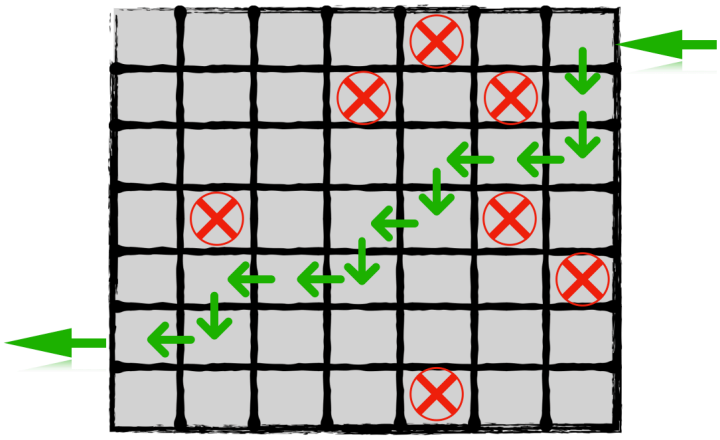
When generating the next generation, a random gene will be changed from 0 to 1, or 1 to 0, which means two bits are affected.

3.5 Optimal Path

In the example of the code, the shortest steps are 11, as showing below:

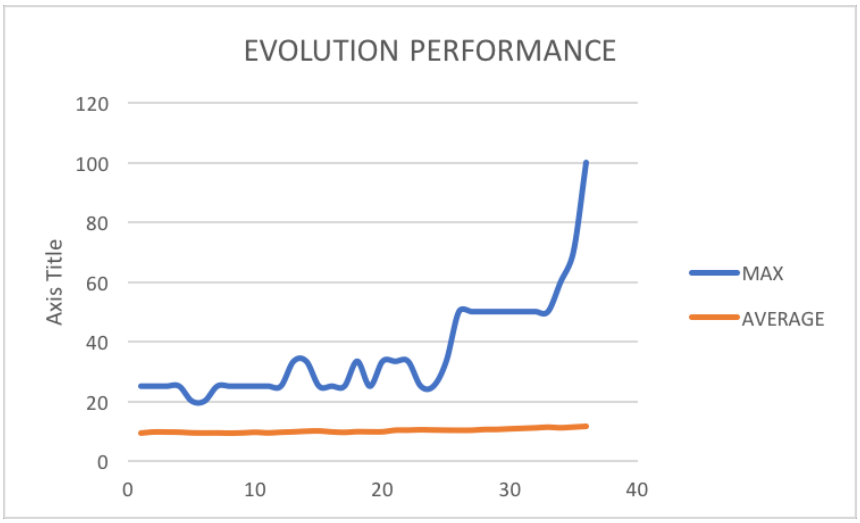
{0,1,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,1,1,0,1,0}

which means that a person could get out of the maze successfully after the above steps.



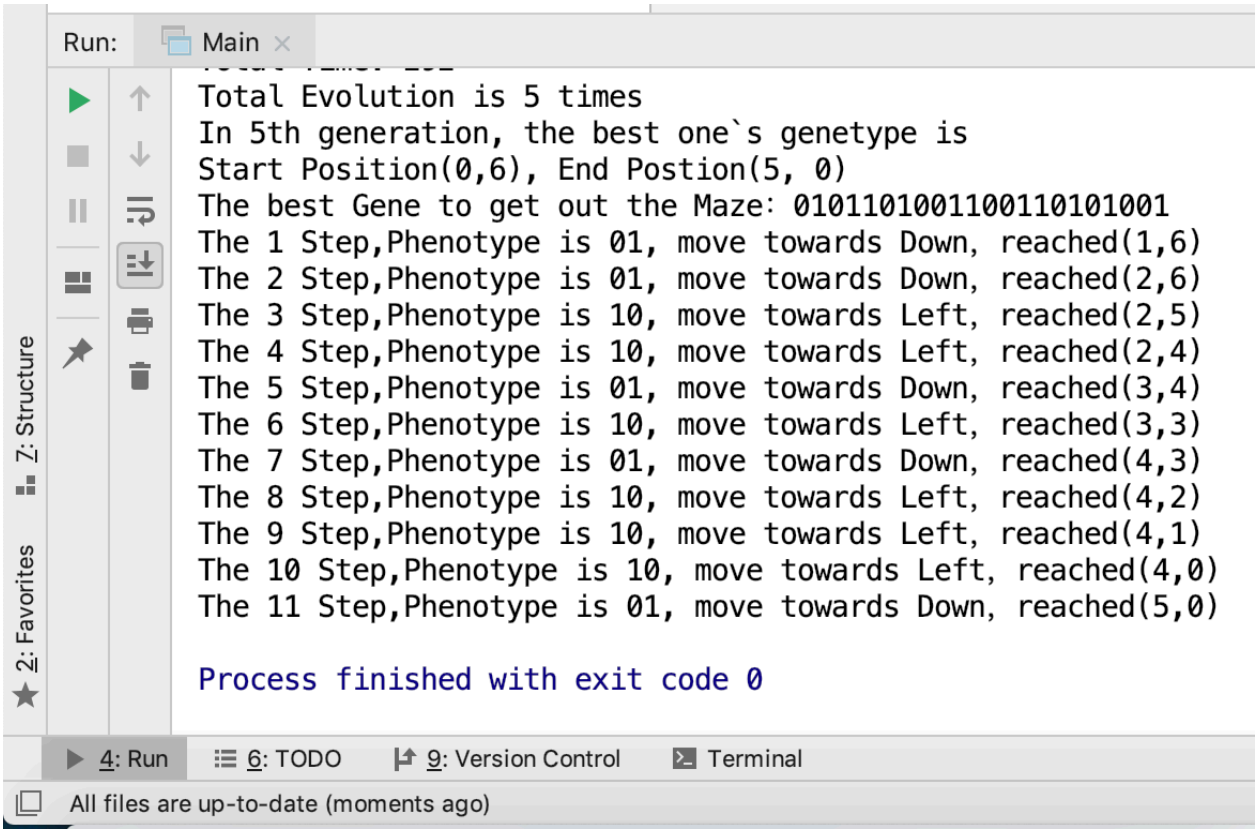
4. Results & Conclusions

As the number of evolutions increases, the fitness will improve continuously, which means the person is approaching the endpoint. As the image shows below, the blue line shows the optimal movement each step, and the orange line shows the average performance.

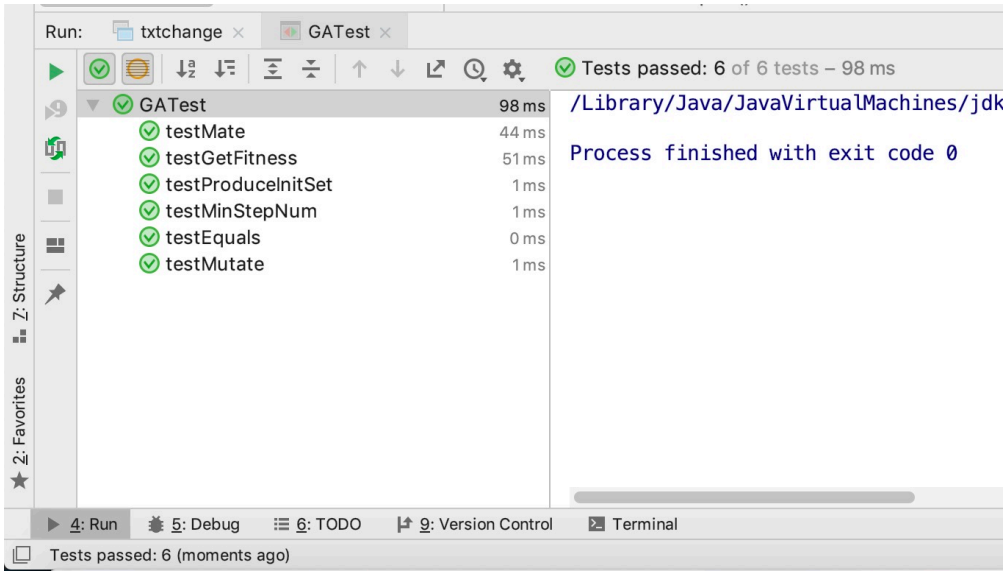


5. Evidence of Running

A person makes 11 movements and reach the endpoint finally, including (1,6), (2,6), (2,5), (2,4), (3,4), (3,3), (4,3), (4,2), (4,1), (4,0), (5,0).



6. Unit Tests



Implementation Details:

```
90      /**
91       * 1. The genetic code and a random generator of such codes;
92       * 2. Gene Expression is inside
93       * 5. Genetic Driver: Produce the first and later generation
94       * @param
95       *
96       * @return
97       */
98      public void produceInitSet() {...}
119     //3. The fitness function
120     public double calFitness(int[] code) {...}
131     /**
132      * 4.The Survival Function - select the survivors according to fintness;
133      *
134      * @param initCodes
135      * @return
136      */
137     public ArrayList<int[]> selectOperate(ArrayList<int[]> initCodes) {...}
191
192 @   private String valueOf(int[] input) {...}
199
376
377         //6. Login function to keep track of the progress of the evolution
378         private void printFoundRoute(int[] code) {...}
412
413         public int getStepNum(){
414             return stepNum;
415         }
416
```