Genetic Algorithm: The Location Selection Problem

Kun Du, Wenyin Zhao

**I: Problem**

We want to set up a vegetable transfer station between vegetable base A and cities: B, C, D, E, F, G, H, I, G, K. The transfer station should meet the following requirements as much as possible:

（1）In order to save transportation costs, the distance between the transfer station and the vegetable base A and various locations should be as small as possible

（2）In order to save the cost of construction fee, the transfer station should be as closer to the vegetable base as possible, because the closer the transfer station is to the city, the higher will construction cost rise.

**II: Genetic Algorithm**

1. *Genetic code*

Each transfer station consists of a coordinate *(x, y).* In our project, we use binary coding to express to convert *x* and *y*. The code length is 20, the first 10 bits are *x,* and the last 10 bits are *y*. For example, if there is a transfer station's coordinate is (20, 50), its gene would be:

0000010100|0001100010

2. *Fitness function (Evaluation function)*

The fitness of each chromosome (transfer station) is composed of two parts. The first part is the distance sum between the transfer station and the vegetable base A. In order to save transportation costs, the total distance should be as small as possible; the second part is the transfer station. The second part is the distance of the transfer station from the vegetable base A. And we want the transfer station should be as close to vegetable base A as possible, because the closer to cities, the rental cost and construction fee would be higher.

We assigned each part a corresponding weight *w.* The first part's proportion is *0.7* and the second part is *0.3.* So the optimal transfer station construction site should have the minimum sum of fitness from two parts. We used the following function to calculate the fitness. 

For analysis convenience, we divide the fitness of each chromosome by 100



3. *Natural Selection*

When selecting the next generation's parent chromosomes, we use *ProrityQueue* to sort the chromosomes according to the fitness of each chromosome, select the best chromosomes, and copy them directly to the next generation without crossing over and mutation.

我们会根据fitness计算每一代每个染色体的累积概率，剩下的染色体根据染色体的累积概率，采用转盘的方式选择父母染色体，这样fitness越小的越容易被选中，符合自然选择；

*4.Crossing Over:*

After we selected parent chromosomes, we randomly exchange fragments in each chromosome to cross over.选出的父母染色体，俩俩杂交，再满足杂交概率的前提下，生成两个后代，杂交的方式是随机交换每个染色体中相应的片段，For example:

Parent c1：00|1101|10010011001100

Parent c2：10|1001|01000000100101

Swap gene fragment index 2-5:

Child 1：00|1001|10010011001100

Child 2：10|1101|01000000100101

Copy chromosomes to the next generation and repeat the process for Parent2.

5. *Mutation:*

如果每个染色体符合变异的概率，chromosomes will mutate. The mutation is a random change on one gene of a chromosomal，and mutate to 0(if the gene was 1) or 1(if the gene was 0).

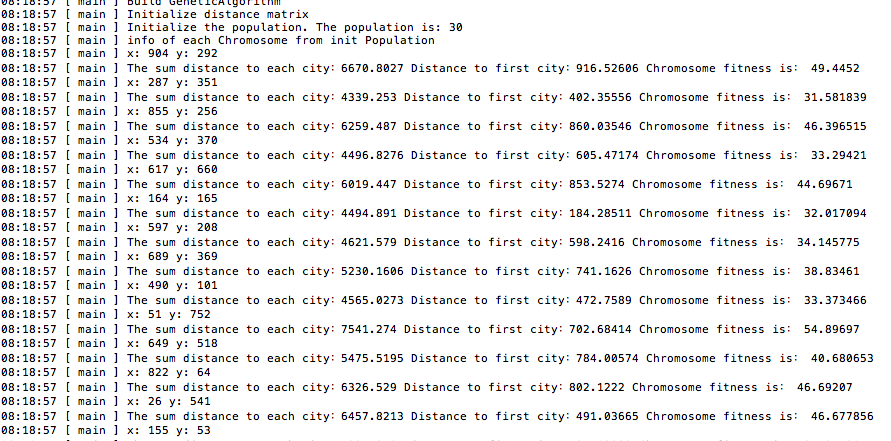
For example,:

Chromosome c gene is：00110110010011001100

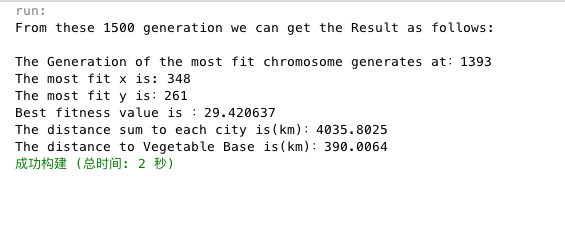
The mutation randomly mutated gene on index 4, and the gene after mutation is：00111110010011001100

*6:Records of evolution process*

In the process of evolution, we will record the best chromosomes, the generation when the best chromosomes shows and the best chromosomes of each generation. Specific evolutionary process is recorded in the log file named log.txt. The following screenshot is some of the records.



III: Results



Unit Tests passed screenshot

