INFO6205
Section1 108
Fangqing Wu 001305642
Tingyu Liu 001087903

Report for Final Project

Purpose

1.)Be able to try to optimize the specific biological engineering.

2.)Conducting Genetic Algorithm for Conway Life Game by using Jenetics in Java.

2.)Observing the patterns see how it change the generation rate and growth rate.

4.)To implement different algorithm increase the diversity of gene.

Implementation Details

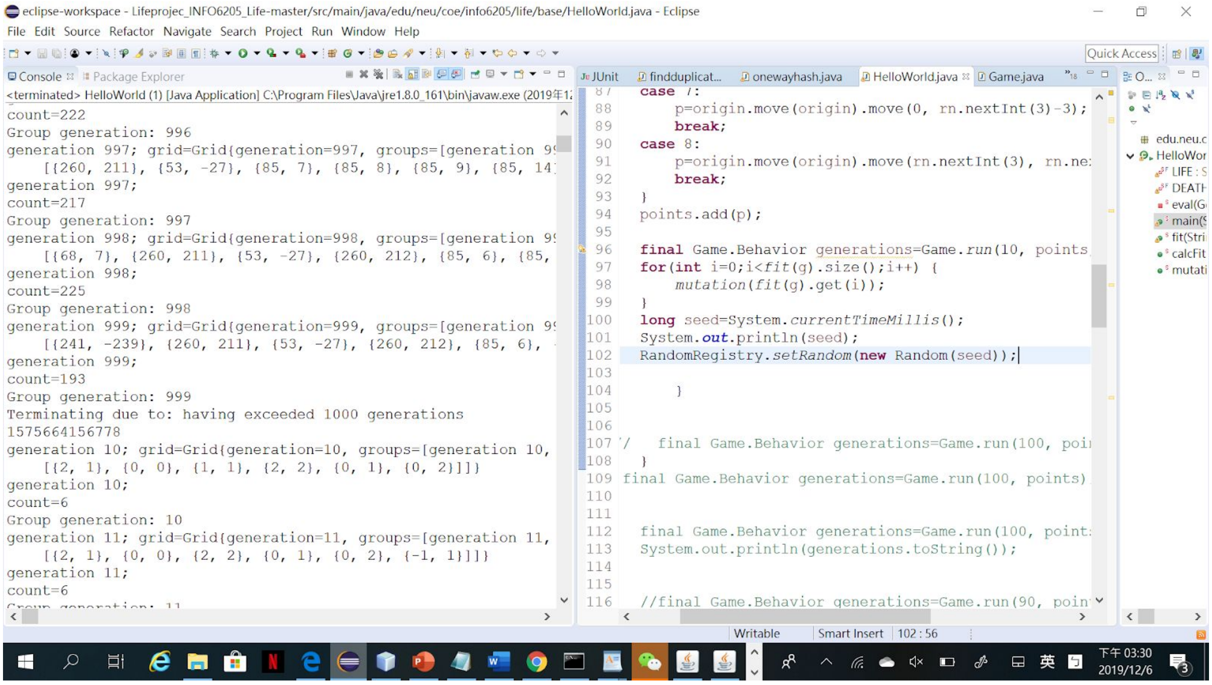
1.) Firstly generate the chromosome using genotype. (We decided to use three chromosomes at a time to, 3 group to increase the possibility of productive offsprings)

2.) Designing our fitting algorithm, we use calFitness() function to calculate bit 1 in each gene, we pick the gene with most 1 as our gene to pass down to next generation. Fit() function we choose half of our best gene to be passed down as we regard that will be sufficient to increase the variety of next generation better.

3.) We use first 3 bits to decide the directions and distance of movement of each gene, turns out pretty well.(They go far at the end)

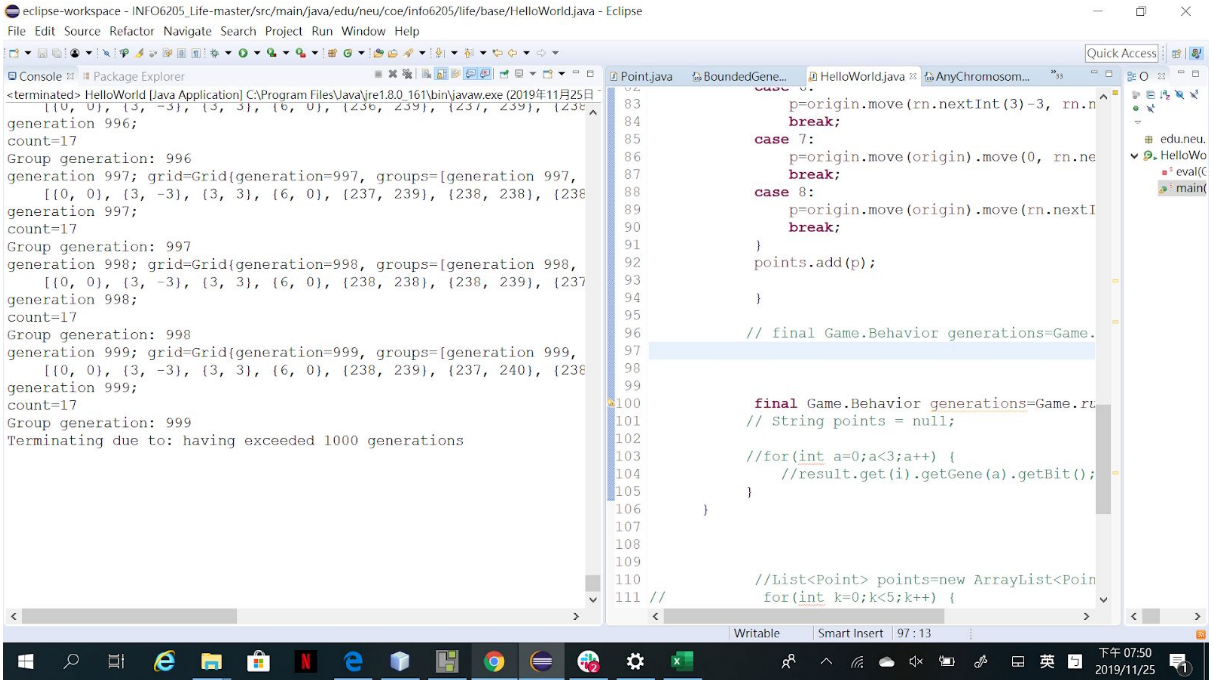
4.) After the first generation is produced and put to run() function, we design a mutation() function to mutate the immediately and mutate the whole chromosome using our Fit() function.

5.) It turns out to work well. However, i also observe that running to above 1000 or 10,000 are not always the cases. We also have some results that stop generate before it get to 1000 generation.



```
<terminated> HelloWorld (1) [Java Application] C:\Program Files\Java\jre1.8.0_161\bin\javaw.exe (2019年1
count=222
Group generation: 996
generation 997; grid=Grid{generation=997, groups=[generation 9
    [{260, 211}, {53, -27}, {85, 7}, {85, 8}, {85, 9}, {85, 14
generation 997;
count=217
Group generation: 997
generation 998; grid=Grid{generation=998, groups=[generation 9
    [{68, 7}, {260, 211}, {53, -27}, {260, 212}, {85, 6}, {85,
generation 998;
count=225
Group generation: 998
generation 999; grid=Grid{generation=999, groups=[generation 9
    [{241, -239}, {260, 211}, {53, -27}, {260, 212}, {85, 6},
generation 999;
count=193
Group generation: 999
Terminating due to: having exceeded 1000 generations
1575664156778
generation 10; grid=Grid{generation=10, groups=[generation 10,
    [{2, 1}, {0, 0}, {1, 1}, {2, 2}, {0, 1}, {0, 2}]]}
generation 10;
count=6
Group generation: 10
generation 11; grid=Grid{generation=11, groups=[generation 11,
    [{2, 1}, {0, 0}, {2, 2}, {0, 1}, {0, 2}, {-1, 1}]]}
generation 11;
count=6
Group generation: 11
```

```java
case 7:
    p=origin.move(origin).move(0, rn.nextInt(3)-3);
    break;
case 8:
    p=origin.move(origin).move(rn.nextInt(3), rn.ne
    break;
}
points.add(p);

final Game.Behavior generations=Game.run(10, points
for(int i=0;i<fit(g).size();i++) {
    mutation(fit(g).get(i));
}
long seed=System.currentTimeMillis();
System.out.println(seed);
RandomRegistry.setRandom(new Random(seed));

        }


'/    final Game.Behavior generations=Game.run(100, poi
    }
final Game.Behavior generations=Game.run(100, points)


    final Game.Behavior generations=Game.run(100, point
    System.out.println(generations.toString());


    //final Game.Behavior generations=Game.run(90, poin
```


eclipse-workspace - INFO6205_Life-master/src/main/java/edu/neu/coe/info6205/life/base/HelloWorld.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Console
Package Explorer
Point.java
BoundedGene...
HelloWorld.java
AnyChromosom...
edu.neu.
HelloWo
eval(
main(
Writable
Smart Insert
97 : 13
下午 07:50
2019/11/25

```
<terminated> HelloWorld [Java Application] C:\Program Files\Java\jre1.8.0_161\bin\javaw.exe (2019年11月25日
    [{0, 0}, {3, -3}, {3, 3}, {6, 0}, {236, 239}, {237, 239}, {238
generation 996;
count=17
Group generation: 996
generation 997; grid=Grid{generation=997, groups=[generation 997,
    [{0, 0}, {3, -3}, {3, 3}, {6, 0}, {237, 239}, {238, 238}, {238
generation 997;
count=17
Group generation: 997
generation 998; grid=Grid{generation=998, groups=[generation 998,
    [{0, 0}, {3, -3}, {3, 3}, {6, 0}, {238, 238}, {238, 239}, {237
generation 998;
count=17
Group generation: 998
generation 999; grid=Grid{generation=999, groups=[generation 999,
    [{0, 0}, {3, -3}, {3, 3}, {6, 0}, {238, 239}, {237, 240}, {238
generation 999;
count=17
Group generation: 999
Terminating due to: having exceeded 1000 generations
```

```java
        case 6:
            p=origin.move(rn.nextInt(3)-3, rn.n
            break;
        case 7:
            p=origin.move(origin).move(0, rn.ne
            break;
        case 8:
            p=origin.move(origin).move(rn.nextI
            break;
        }
        points.add(p);

        }

       // final Game.Behavior generations=Game.



        final Game.Behavior generations=Game.ru
       // String points = null;

       //for(int a=0;a<3;a++) {
           //result.get(i).getGene(a).getBit();
       }
    }



        //List<Point> points=new ArrayList<Poin
//       for(int k=0;k<5;k++) {
```

Problem

Our numbers of counts maintained at about hundreds. We assumed it would be higher.

Also, we tried to optimize the code to allow the numbers keep alter, we tried not to let our gene move in a stable pattern, but we failed to achieve that. About 2000, the generation will stay stable and the points on matrix rotate in a same pattern.

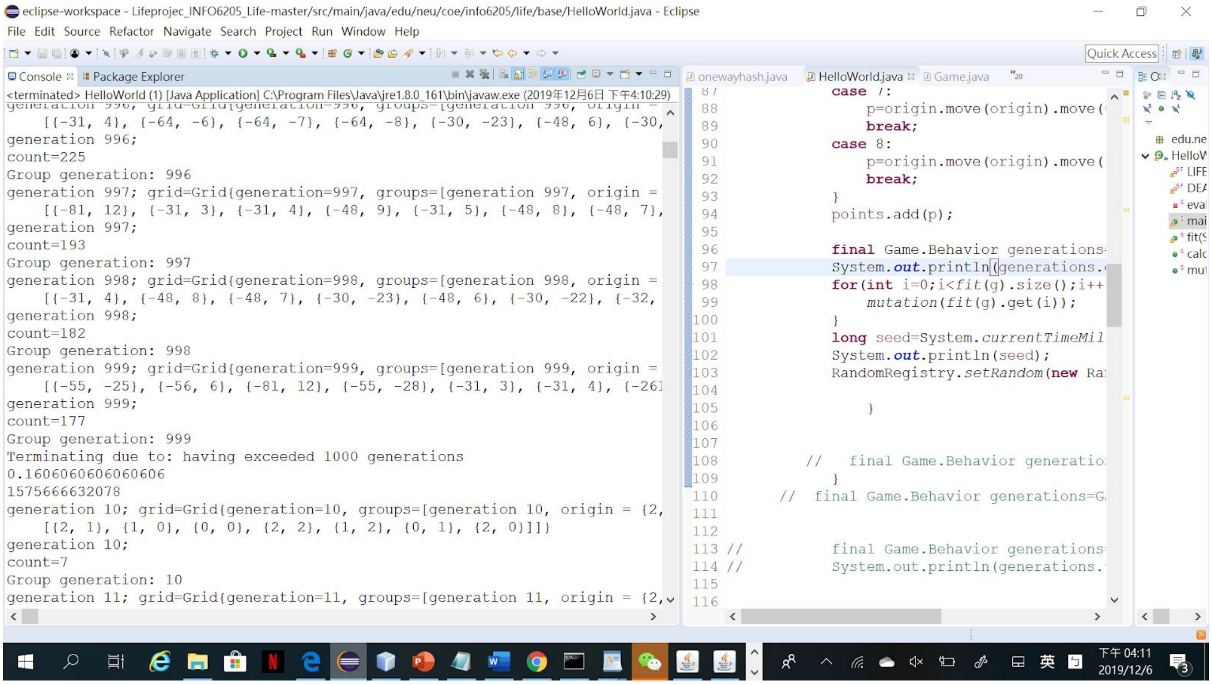
Findings

We did a customized mutation function instead of using bits in genes itself, and it turns out works. We found out that if we choose to pass down half of chromosome instead of just first one or two or three. The diversity of points will be higher, telling from the count of points on about 1000 generations.

Results

Our GA code do have chances successfully generated 1000 above generations, and had a roughly 0.233 growth rate which is in expectation range. In the case that we increase our max generation to 10,000, our count on generations will keep at 100~200 on average and have an average growth rate between 0.1~0.2, usually if we hit 1000, it should maintain above 1.5. if we hit 10,000 then growth rate maintain above 1.0.

Conclusions


eclipse-workspace - Lifeprojec_INFO6205_Life-master/src/main/java/edu/neu/coe/info6205/life/base/HelloWorld.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Console
Package Explorer
onewayhash.java
HelloWorld.java
Game.java
»20
edu.ne
HelloW
LIFE
DEA
eval
mai
fit(S
calc
mut
下午 04:11
2019/12/6
英

```
<terminated> HelloWorld (1) [Java Application] C:\Program Files\Java\jre1.8.0_161\bin\javaw.exe (2019年12月6日 下午4:10:29)
generation 996; grid=Grid{generation=996, groups=[generation 996, origin =
    [{-31, 4}, {-64, -6}, {-64, -7}, {-64, -8}, {-30, -23}, {-48, 6}, {-30,
generation 996;
count=225
Group generation: 996
generation 997; grid=Grid{generation=997, groups=[generation 997, origin =
    [{-81, 12}, {-31, 3}, {-31, 4}, {-48, 9}, {-31, 5}, {-48, 8}, {-48, 7},
generation 997;
count=193
Group generation: 997
generation 998; grid=Grid{generation=998, groups=[generation 998, origin =
    [{-31, 4}, {-48, 8}, {-48, 7}, {-30, -23}, {-48, 6}, {-30, -22}, {-32,
generation 998;
count=182
Group generation: 998
generation 999; grid=Grid{generation=999, groups=[generation 999, origin =
    [{-55, -25}, {-56, 6}, {-81, 12}, {-55, -28}, {-31, 3}, {-31, 4}, {-261
generation 999;
count=177
Group generation: 999
Terminating due to: having exceeded 1000 generations
0.16060606060606
1575666632078
generation 10; grid=Grid{generation=10, groups=[generation 10, origin = {2,
    [{2, 1}, {1, 0}, {0, 0}, {2, 2}, {1, 2}, {0, 1}, {2, 0}]]}
generation 10;
count=7
Group generation: 10
generation 11; grid=Grid{generation=11, groups=[generation 11, origin = {2,
```

```java
            case 7:
                p=origin.move(origin).move(
                break;
            case 8:
                p=origin.move(origin).move(
                break;
            }
            points.add(p);

            final Game.Behavior generations
            System.out.println(generations.
            for(int i=0;i<fit(g).size();i++
                mutation(fit(g).get(i));
            }
            long seed=System.currentTimeMil
            System.out.println(seed);
            RandomRegistry.setRandom(new Ra

                }


        //   final Game.Behavior generatio
            }
      //  final Game.Behavior generations=G.


//          final Game.Behavior generations
//          System.out.println(generations.

```

It has been a great chance to observe our own algorithm in a direct way. To ge gene and change them is not the most difficult, but to design a best algorithm to select best gene and best way to mutate it to increase the possibility of surviving the the gene. It is not easy but we also got to fix and optimize our thoughts and cod in a big way.