Final Project

---Haoran Li, Jun Yang, Xinyuan Jin

**Introduction**

Game of life is a Cellular Automaton which is an individual within a grid of colored cells which evolve according to basic rules relating which neighboring cells change states. The main problem is how can we get the fittest start pattern that is generated by random process. And we choose genetic algorithms as our method to fix this.

**Implementation**

We decide that our GA mutation follows the rules as below:

we decide the mutation rate of individual is 0.3, and mutation rate of each gene of specific individual is 0.2. Then we choose random double n from 0-1, if n<0.3, then the chromosome mutates; if n>0.3, the copy the whole chromosome.

Once the chromosome needs to mutate, we choose random double m from 0-1 to decide each of gene of this chromosome need to change or not, if m<0.2 then this gene mutate, else the gene stay the same. In this way, we can make sure that genotype cannot be copied from parent to child 100% perfectly, but perhaps fidelity is 99.99999%.

Fitness: how well the organism is suited to the environment.

We define the two rules to define the fitness.

|  |  |
| --- | --- |
| The contribution of each survival cell to fitness | CellFitness=8 |
| The contribution of max generation to fitness | generationFitness=20 |

So, the fitness would be counted as cell\*cellFitness+generation\*genetationFitness.

Evolution: in this part, we apply a selection function to choose the survival individuals. For each generation group, the best 10% fittest individuals will survive to the next generation. And the other 90% individuals need to compete with the new 10% immigrants to gain the last 90% survive chance or will die, which means our immigrant replacement rate varied between 0-0.1.

Genotype and Phenotype: We use two-dimension grid to show the phenotype of our chromosome, and we define an array to show the phenotype of the chromosome.

**Result:**

After we apply GA with our population, we successfully saved our best seed to be the starting pattern. And each time we run our project we can get a best seed that show out the best fitness. Besides that, we successfully design our UI to show our result.

**UI Design:**

A group of black text

Description automatically generated

**Screenshot of unit Tests**

Mutation Test:A screenshot of a computer

Description automatically generated

Fitness Test:

A screenshot of a computer

Description automatically generated

Selection Test:A screenshot of a computer

Description automatically generated

Expression Test:

A screenshot of a computer

Description automatically generated