COSC343 Assignment 2 Report

Yanxin Hu
2856476

Simulation:

Grid Size: 24
Turns: 100
Generation: 500
Percept format: 2

My model is based on the default settings, there isn't a specific reason to why I stayed with the default values. After evolving 500 generations, a graph will pop up which displays how the model has performed based on the average fitness of each generation, the model will also print the "average life time" of all 500 generations (average turns survived). The graph function is based on a pre-built java graph tool called "jFreeChart", and it requires two external "jar" files ("jcommon" and "jfreechart") in order to run.

Note: "Average life time" is the same variable from your framework, "Average fitness" is the variable I created for measuring each creature's fitness.

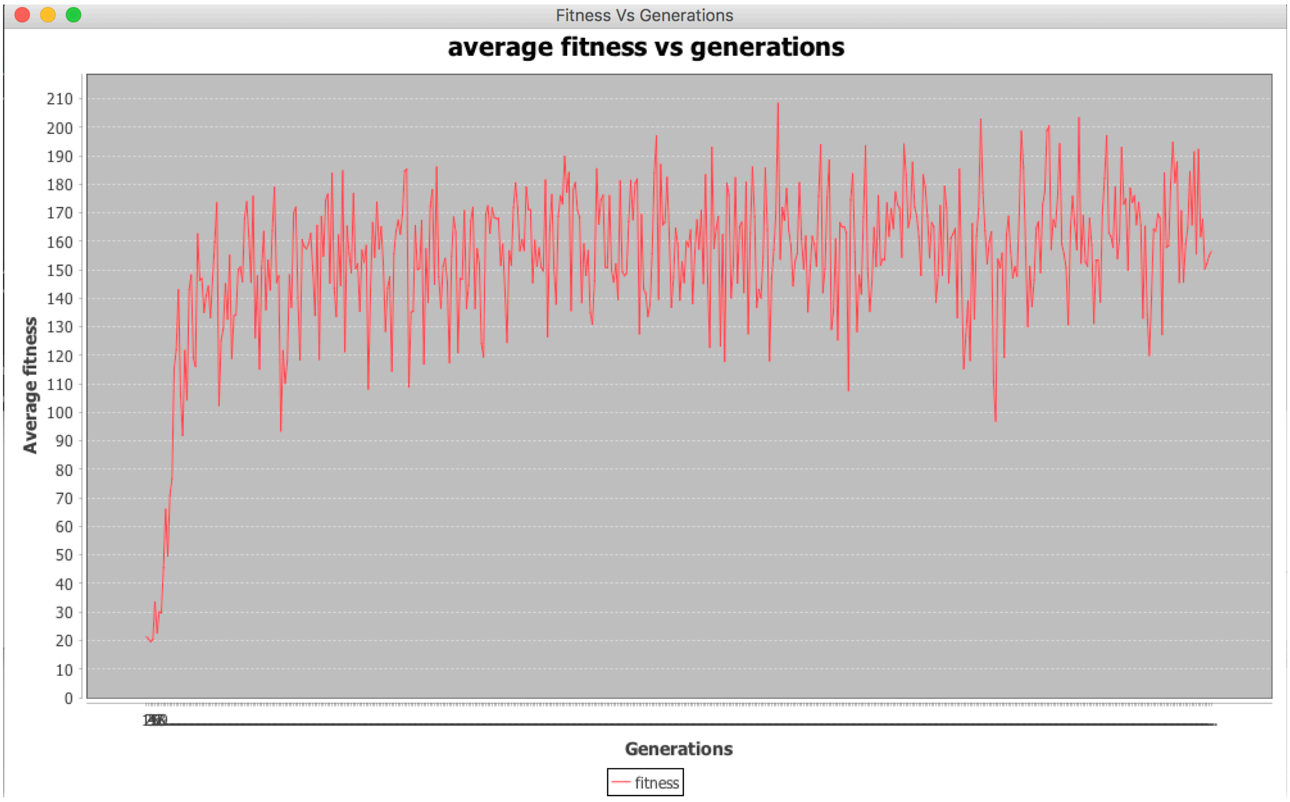
Chromosome:

The chromosome includes 9 genes, which represents the 9 tiles. The reason why I picked 9 is because percept format 2 has 27 percepts, and every 9 percepts represent the location of either a monster, a creature or a strawberry. That way my model can accomplish the "move closer" and "run away" behaviour based on what's occupying each tile. Each gene can be represented by a number ranging from 0 to 10, which represents the 11 possible actions. And this is how the agent function connects with the chromosome. For example, say the first gene is "7", then when my agent function is called, it first checks whether tile X1 has a monster on it, if there is, it checks the corresponding gene's value, in this case, it is "7", so the agent function adds more weight to the action with index "7" (whatever that action is). This is a very simple "run away" and "move towards" model, I chose this one because of the simplicity and it is relatively fast for creatures to learn, I found that sometimes a simpler model often yields better results.

Agent functions:

I've already explained a bit about the agent function, here' some more details. By checking all 27 percepts, if there's a monster present, I add more weight to certain action based on the creature's chromosome. So if I assume this action is "run the opposite direction" (if the chromosome is really good), then bad chromosome will probably run straight towards the monster, this way bad chromosomes should perform a lot worse. My weight model has 4 weight values, ranging from 1 to 4 which represents the priorities. For example, if the creature senses a monster, the agent function will add 3 values to the corresponding action, and only 1 value if the creature senses another creature in the vicinity so that the creature will run away from a monster more often than running away from another creature. If there's a strawberry nearby, the agent function will add more weight to the opposite direction (like you said in the email), basically the agent function just checks if the corresponding gene of that chromosome is less than 8, if it is, then add more weight to that action since I know it will be a "move" action. If the received percepts are all 0s, which could happen if there's nothing in the vicinity, then add weight to a random movement. There's also a boolean value called "monsterNear", it checks if there's any monster in the area, if there is, add less weight to "eating", especially if the strawberry is green. With this model, creatures will eat less when there are multiple monsters/creatures around since each monster/creature detected will add more weight to the “run away” action, and if there are multiple strawberries and only one monster around, the creature might take a chance and choose to eat the strawberry.

Graph:



This model learns fast but stops improving very quickly (often after generation 40 or so).

Genetic algorithm:

How the average fitness is calculated for each creature is very simple, the model first checks if this creature is dead, if so it checks whether the creature died before or after 50 turns (since creatures that don’t know how to eat will die after 50 turns), if the creature lasted more than 50 turns, then the fitness of the creature is “energy + timeOfDeath”, if the creature died before eating any food, then the fitness is “timeOfDeath * 0.5”, and the “0.5” is the punish/reward coefficient. However, if the creature survived, the model checks whether or not it also survived the last generation, if it did survive 2 generations in a row, then the fitness is “(energy + _numTurns)*3 and it will participate in the next generation as well, so I reward heavily if a creature has good chromosome, and if it did not survive the last generation, then the fitness is “(energy + _numTurns)*2” and with a 30% chance it will participate in the next generation (100% chance if less than 2 survivors currently). And I used “Roulette Wheel Selection” as my parent selection algorithm, this way good chromosomes can be easily picked. I used “Uniform Crossover” for my crossover algorithm so that there’s some room for improvement on the already decent genes. For mutation, my method is that a random gene gets set to a random value ranging from 0 to 10, the chance for mutation increases from 20% to even higher if the current generation did not perform well, if the current generation performed well then the mutation chance is a constant 5%. With this mutation method, learning should be faster at the beginning, and once the creatures start to get smarter, there’s still that 5% chance for more improvements (hopefully). Out of all 34 creatures of one generation, only 2 creatures are initialised randomly, the rest are all children/survivors from the last generation. With this model, good chromosomes are rewarded heavily and there is still some randomness/diversity for further improvements.

Results:

The model performs fairly well, it learns very fast if there's a good chromosome early on. I think this model is also quite balanced in terms of eating and avoiding. The first generation is awful which is expected, creatures behave randomly and they rarely eat food, after several generations some starts to eat, some starts to avoid and some does both, finally most of them show intelligent behaviours with some "elites" that eat several strawberries in a row which is rather interesting because that's how the real world is going to be in a zombie apocalypse.