COSC 343 Assignment 2 Report.

Daniel Thomson

ID: 5040702

Simulation:

The parameter I used for my simulation are mostly the same as the default, with a grid size of 50, 100 turns per simulation and 500 generations. I increased the size of the grid in order to have the simulation start off with a large population, meaning it would have a higher chance of generating some creatures with good “genes” in their chromosome. I left the turn per simulation at the default of 100 turns. I tried using both 50 and 200 turns. With 50 turns too many creatures with not so great chromosomes were surviving to the end, and because my fitness function heavily favours survivors, these not so great chromosomes spread through the generations. With 200 turns I didn’t notice much of a difference relative to 100 turns, so I just left it at the default. I left the number of generations at the default of 500, because as your will see in the graph most of growth in average fitness occurs in the first fifty generation so there was no need to increase it and I didn’t decrease it just show how consistent the fitness is after 50 generations.

Model:

The percept format I used for my model was number 2. I choose format 2 because it allows the creature to detect multiply entity of the same type at once and can tell the difference between red and green food all the time. With my model, the creatures consider how to react to each different type of entity one at a time, the entity types being monsters, creatures, red food and green food. The way that this consideration process works is that the creature first determines how it will react to this entity type, whether it will run away or towards this kind of entity. It then uses the information it gets from its percept’s to detect if any entity of that type is near it. If any entity of that type is nearby, it then loop through the percept’s and for any detected entity it set the action that corresponds to move away or towards the entity (depending on the reaction value determined at the start) to a value defined in the creature chromosome. This process is repeated for all 4 entity types. If this process tries to set an action that already has a value in it, it set that action to the highest value of the two. The process is slightly different for the food entities as it also adds a value to the eat action if it is sitting on food. After this process is completed for each entity type, the model then considers how to do an exploration move if no nearby entities were detected. The model considers two option for exploration, either a random moves or moving straight in a direction defined in the chromosome. The values that are in the actions array at the end are all float values between 0 and 1.

Chromosome:

The chromosome I defined for the creatures is an array of 48 float values between 0 and 1 plus an integer value between 0 and 3 or 5 and 8. The values in the array can be broken down in to groups of 11 or 12. Each group of 11 is used in the model to determine how the creature reacts to each entity type. The first two value in each group determine how the creature will react to that entity type, either run away or towards, the higher of those two value being the action taken. The other 9 values are weights for how likely a creature is to move in a direction given an entity of that type is nearby, the 9 values corresponding to the 8 direction plus staying put. The groups for the two food entity type have an extra value, that is a weight for how likely the creature is to eat the food given its on top of it. The last two values in the array are to determine what kind of exploratory action the creature will take, either a random move or off in a straight line. The integer value that is defined corresponds to the direction the creature will take if it uses the straight line exploratory move.

Genetic Algorithm:

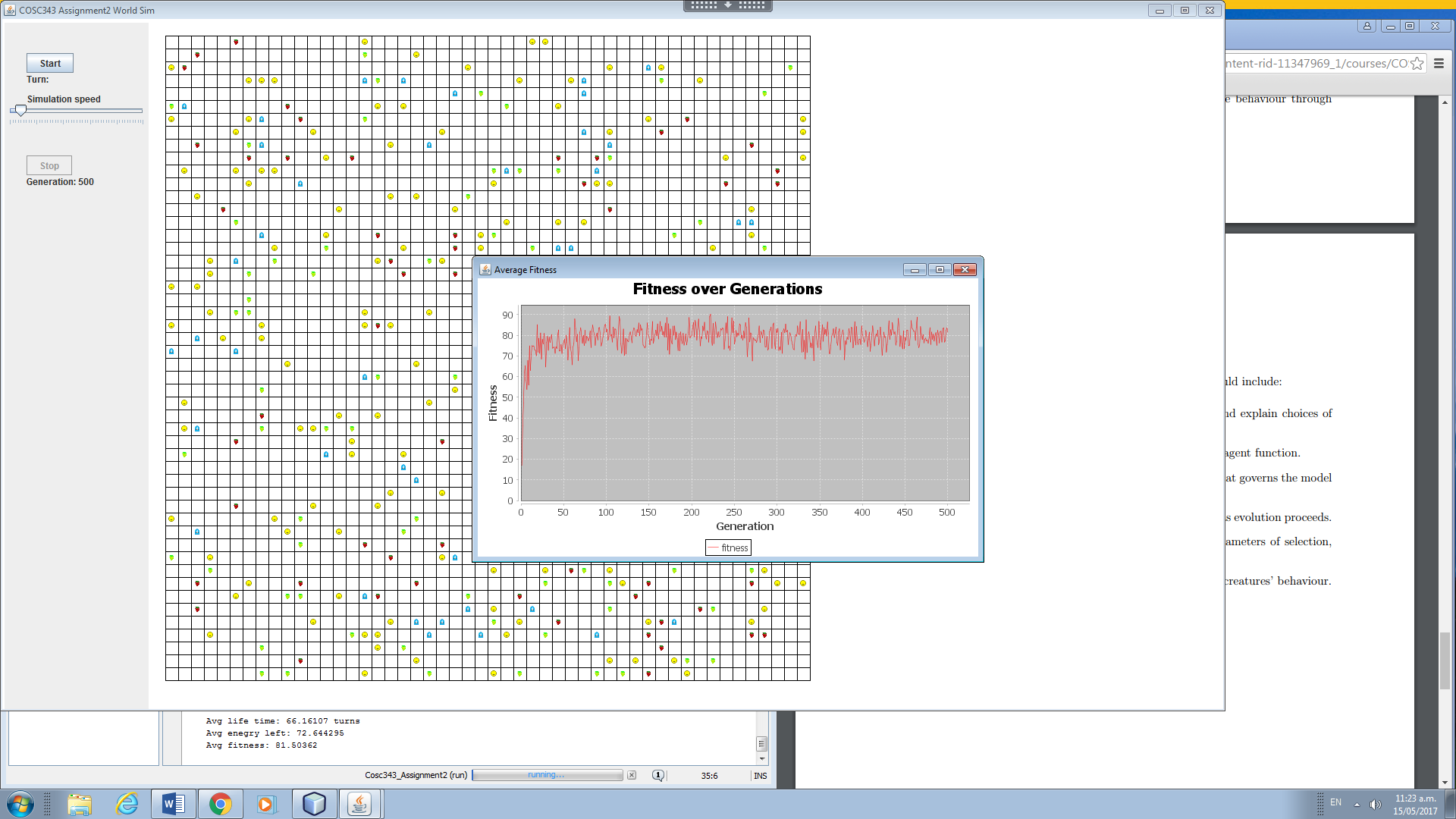
Fitness Function: The fitness function I used for my genetic algorithm is turns alive \* ((energy left+1/100) + 1 if survived till the end of simulation). This function means survivor will always have higher fitness than creatures that die, and generally have much large fitness than those who die as well. This also means survivors are ordered by how much energy they have left. There is plus 1 to energy left just so creature who stave to death don’t have a fitness of 0, but they do have a relatively small fitness.

Parent Selection: For the parent selection the genetic algorithm uses roulette wheel selection. I choose that method over tournament selection because the fitness of creature that survive is generally a lot higher than the fitness of other creatures, so the roulette wheel selection favours the survivors a lot. Roulette wheel selection also allows all the other creature a chance to be selected, which tournament doesn’t, even though it’s a small chance.

Crossover: Originally I just used a simple crossover method where the first halve of the new creature’s chromosome was the first halve of the first parents chromosome and vice versa with the second halve and second parent. However I noticed that after some generations the chromosomes of creatures in each generation ended up the exact same (this was before I implemented mutations). In an attempt to reduce this I modified the crossover method so that for each chromosome value in the new creature it would randomly pick between the corresponding chromosome values from the two parents (meaning you could end up with a new creature that is an exact copy of one of its parents). This seemed to reduce the issue of exact same chromosome across a generation, as even if multiply new creatures had the same parents (which is likely) it is much less likely that they would end up with the same chromosome.

Mutation: For mutation I implemented it so that each value in a new creatures chromosome had a 0.1% chance to change by +/- 0.2 \* a random float between 0 and 1. This change is only applied if the resulting value is between 0 and 1. This means that roughly one chromosome value is changed per twenty creatures.

Graph of Average Fitness over Generations:



Results:

As can see from the graph the average fitness of the creature grows rapidly during the first 50 generations before it flattens out and become roughly consistent for the rest of the iterations. I suspect the rapid growth is due to a combination of the fitness function heavily favouring the survivors (the creatures who most likely have good “genes”) and the uses of roulette wheel selection, which is highly likely to select these survivors as parents and thus spread these good “genes”. Thus the good “genes” are spread out among the population, quickly increase the average fitness of creatures. It then gets to a point where the creatures all roughly have similar chromosomes, and the graph flattens out with no major growth for the rest of the evolution. In terms of behaviour the creature have developed after 500 generation, somethings I have noticed are that creatures don’t run away food or run towards monster, this most likely arising from creatures that starve to death (run away from food) or die early (run towards monster) having very low fitness. I’ve also noticed that some creature will sit atop of a green strawberry and wait until it turns reds (or a monster/creature scares it away). During the first simulation run I notice that creature would cluster together and just sit there, which they stop doing after the evolution. Once again I think this due to creature that starve (which is how most of the cluster creatures die) having low fitness, so the behaviour of go towards creatures (which causes the clustering) dies out. For the exploratory move behaviour most creature seem to use the straight line explore method over random move after evolution. I think this happens because the random move method has a chance that the creature will move to the same tile over and over again, which results in the creature finding less food and thus having less energy/fitness. The model does limit the go away from monster behaviour that develop, as in a situation where a creature has a monster on opposite sides of it, it’s guaranteed to run in one or the other monsters.