**A Genetic Algorithm that mostly works**

Let’s talk about the elephant in the room, my code doesn’t always work and doesn’t do everything it needs to do. On the other hand, I did use the genetic algorithm as intended and when it does run you can see that the fitness score is improving. The best run I had was 175 generations with the best fitness score of 603.6. In this paper I will talk about why it doesn’t work as intended, go through how I implemented the algorithm, and what I would do differently.

Why doesn’t my code work and what did I not get to complete? If I knew why it didn’t work, I wouldn’t be writing this however the problem is when I run my program an abort window will pop up and it doesn't tell me too much why the abort function was called. Now I could just go a debug it and see what is happening, but I ran out of time. Another problem is the error does not always happen at the same time as sometimes it happens right away at generation 1 or it will last till generation 175. That is all I know so moving on to what I did not complete is having my algorithm go until there was little improvement and outputting to a file. My code will try to run for 1000 generations and then output what the best fitness was. The best fitness is just outputted to the console however outputting to a file is not the hard part as I could easily have it output the average fitness of the last population, the best fitness, and the chromosome to a file using a few for loops. However, I never got time to implement it.

This is my first genetic algorithm and that will probably show but I was able to get a working algorithm that does follow the steps that it should. Since I am using C++, I did use a Member class which is just the chromosome and functions for the chromosome. I then used a vector in main that was made up of all the Members so when I had to work with the population it was in main and when working with a chromosome it was done in Member. I did follow the idea of a genetic algorithm by first creating a population of 1000 members. That was done by fillInitPop that created all the Member(chromosomes) then add it to the population vector. After that I needed start selecting the parents by evaluating their fitness scores and use L2 normalization to pick the best ones. Since the Members have a variable for fitness that is a simple getter function. That is why the next step in my program is L2norm which I based off the last slide on the normalization PowerPoint. I first needed to find the sum, then find the probability of each Member using the sum. Lastly, I needed to add up the probabilities to get the cumulative probability. I decided to give each Member these variables so I could just call them when I wanted to start selecting from the population. Once thing is in the slide it does not look like you sort the cumulative probability, so I did not either. Now that it is time to select, I call the selection function which picks a random number from 0 to 1 and then goes through all the members and finds which one cumulative probability is greater than the random number and select it for a parent. Once it does it, it does it again, so we have two parents it is time to reproduce.

I like to call it reproduce since there are a few steps included in it and it is basically reproducing. The reproduce function will return a new population to replace the old one but first it needs to make the children. So, I call the crossover function that is being passed the selected parents from the selection function. Crossover picks a random number between 1 and 400 to split the chromosome from the parents. Once that number is decided I make a new child vector and set it equal to the parents. That way they have half the “DNA” and I just need to flip the rest of the DNA to the other parent. In my code I go over in more detail on how that is done but I do handle each child individually. Once both child chromosome is complete I mutate both of them by calling the mutation function. This function goes over each selection it sees and has a 1/10000 chance to flip if it otherwise nothing happens. It returns both children now that they are done but I must reset their fitness since they are copied from their parents so that gets set for them. Now we just need to do that 499 more times to get all the children ready to become the new population. All the children get reassigned as the new population and it repeats the process starting with L2 normalization again. This goes until 1000 generations if it doesn’t abort first.

I didn’t finish this project and even looking back now there would be other things I would change. First, I used vectors a lot and really, I just choose them because but I now realize that I use them a lot and I never needed to resize them so I should have at least used an array to save on space. I also had a lot of trouble getting the input data and setting it up to be used by the whole program. I originally had the file open every time the constructor was called, or when I was setting the fitness score. I don’t even know how many times I originally opened a file, so I am glad I changed it, but I don’t think my current solution works well with my logic since I changed it at the end and required a lot of refactoring. Another thing I would change is if I had more time, I would find the reason it is aborting early and fix it since I know the algorithm works there is just something causing it to stop early. While working on this another thought came to my mind and that would be implementing this in Racket. Now we both know I only made a parser in Racket so I still have a lot to learn however when I was working on the reproduction function, I had an idea on how I would do it with racket since a lot of the functions needed a list and returned a list. Everything else would have really puzzling for me but I made the joke in class, and I know realize that it is do able.

Overall, I am happy with my genetic algorithm because even though it doesn’t finish the algorithm, I know it follows the proper steps and the spirit of the algorithm. If I had more time, I know I could turn in a fully working program and clean it up a bit but that also my fault for not managing my time better. I am also happy that I was able to get a fitness score of 603.6 at I think we should have gotten around 700 so if my program didn’t abort, I would have most likely gotten there.