Brandon Piotrowski

CS461

Genetic Algorithms Program Report

**SAMPLE OUTPUTS:**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**I )** The major challenges I ran into while writing the program were with the natural selection and crossover method. This was largely due to my inexperience with Python as a language, as this is the first class I have attempted to use it. Initially when the program would run it would go through 100 generations, but I noticed that the fitness scores of the schedules were exceeding what was possible within the function (scores like -502 or 140). I figured out that this was due to how I had implemented the crossover method. When assigning elements from an existing list to a new list (newList[0] = oldList[0]), Python does not assign the elements of the new list by value. Instead, it assigns the reference of the element in the old list to the element in the new list. This meant that whenever a new generation was having its fitness evaluated if a child shared the same parent as multiple other children, all of their values would be changed. For example, when a sibling was evaluated in the fitness function, after a previous sibling had been evaluated, its starting fitness score would not equal 0 and any other operations performed on the fitness score would also reflect in the child that had already undergone the fitness evaluation. So, to counteract this issue I began using the deepcopy() method to create a true copy of the elements. This slowed down my program immensely, having each generation take over ten seconds to complete. Eventually, I had to refactor how the crossover method was implemented again by removing loops and also changing the way parents were selected for the mating pool.

**II )** When a final schedule is produced after running the program the final fitness score typically ranges from 10.5 – 12.9. The only odd thing that is ever produced is one activity having a fitness score of >2 while all other activities range between 0 and 1.75.

**III )** If I were to improve my program in any way, I think I would choose a different method of selection. Currently, my program uses roulette wheel selection with a slight modification to prevent an excessive amount of looping. I would maybe try elitist selection if I were to redo the program.

**IV )** I enjoyed this assignment a lot, although I feel like I learned a lot more about Python than genetic algorithms. I also noticed when researching genetic algorithms that there isn’t necessarily a best way to do them, so when I was struggling to figure out different implementations it was difficult to actually find an answer.