Artificial Intelligence: Assignment 1

Part A: Report

One-Max Problem

For the first part of the assignment, my interpretation of what is being asked of us is to generate a binary string in an optimised state, i.e., maximising the amount of ones in the string.

The one-max problem was used as the base for all code. Here the basic methods were initialised with their functions. The most important of these methods are those relating to the fitness function, selection, crossover and mutation.

Other methods and operations include the initialisation of a random string and the printing and plotting of all the final results.

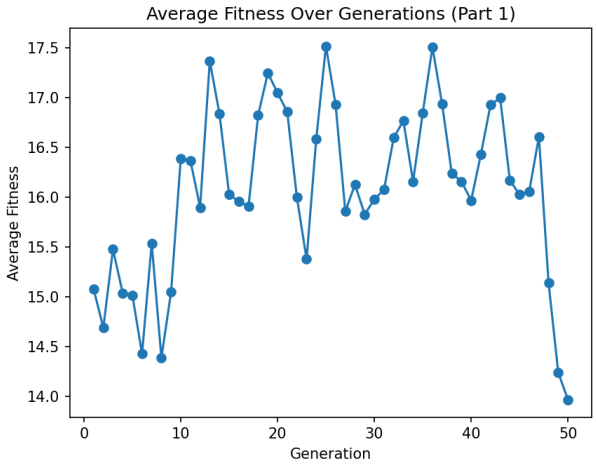
Fitness Function: The fitness function of a genetic algorithm is used to find the closeness of a given solution to the optimum solution[1]. The optimum solution in this problem, is the string with the largest number of 1s. To find this, the function within the code, computes the sum of all the individual inputs of the string. This will tell us how many 1s are in the string, and so the level of the solution.

Selection: This occurs within the genetic_algorithm method, with the variable parents. It chooses the parents based on the fitness scores. It takes the population of the strings and applies the fitness scores as weights. This is to select the solutions to be used in the next generation.[2]

Crossover: This happens using a random crossover point between two parent individuals. This is to combine two solutions to formulate a new solution for the algorithm to search.[2]

Mutation: The function of the mutation method is to flip random bits within the string. Mutation also helps in the exploration of the genetic algorithm, to explore all possibilities. In these methods, if an random number generated is less than a mutation rate (set at the beginning), a bit is flipped to the opposite value. This is done for each individual bit.

Graph:



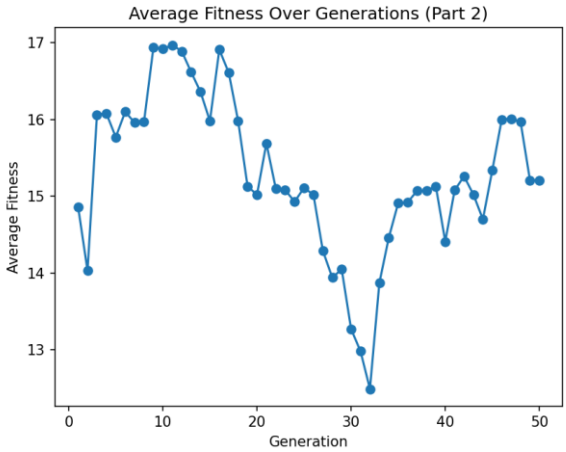
Interpretation of Results: I believe the sharp decline in fitness scores in the larger generations is due to the algorithm not being capable of handling large amounts of data. As well as this the fitness scores are constantly changing, while staying in the same graph region. The highest score seems to be 17.5, which in a string of 30 is roughly 60% of the optimal solution. It seems the most valuable solutions happen between the 10th and 40th generation.

Evolving to a target string

The difference between the first and second section of part A is the introduction of a target string. This target string is treated as the optimum solution. This then changes the fitness function calculation. In this specific algorithm the fitness function is computed by comparing the similarities between the target string and a randomly generated string in the create_individual method.

The crossover, selection and mutation functions remains the same.

Graph:



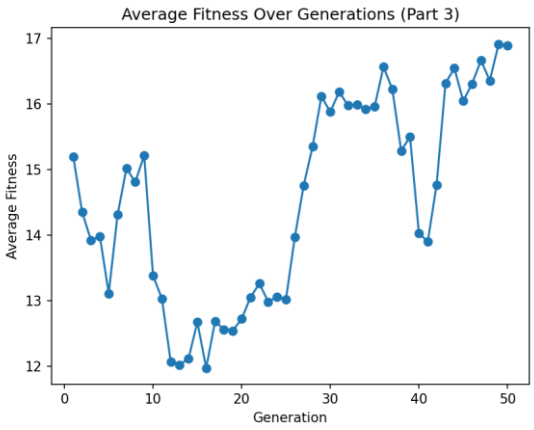
Interpretation of Results: The fitness rapidly declines from roughly the 25th generation and starts to restore after the 30th. Compared to the first graph the values happen on a much wider plane, suggesting larger error margins.

Deceptive Landscape

The difference here again lies in the computation of the fitness scores. A conditional is added in this algorithm, after the computation of the fitness score. If the score is greater than 0 (the string includes 1s), the fitness function is computed and returned, otherwise the length of the string is doubled and returned.

The mutation and crossover methods remained the same as the 'one-max problem' algorithm.

Graph:



Interpretation of Results: Despite a sharp decline between the 10th and 20th generations, when glancing at this graph, it seems to be constantly rising with each generation. With the highest fitness score appearing in the final generation.

Overall Comment on Results

As each algorithm is run separately, and each file generates its own random strings, it was hard to analyse deeply how each algorithm worked. However, the graphs displayed helped to further visualise this.

References

1. Mallawaarachchi, V. (2017) *How to define a Fitness Function in a Genetic Algorithm?*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/how-to-define-a-fitness-function-in-a-genetic-algorithm-be572b9ea3b4#:~:text=What%20is%20a%20Fitness%20Function,how%20fit%20a%20solution%20is. (Accessed: 6 February 2024).

2. *What is mutation (genetic algorithm)?: Autoblocks Glossary* (no date) *Autoblocks*. Available at: https://www.autoblocks.ai/glossary/mutation (Accessed: 6 February 2024).