Assignment 1: Genetic algorithms

Rory Murray 17395111

Part A

OneMax

For the one max problem I followed the format for a basic GA with one point crossover and mutation followed by an evaluation at every generation. For the one point crossover function I first sorted the population based on fitness so the fittest parents combined to create children. Children only replaced parents if they were fitter than their parents meaning crossover never led to a worse population. For mutation I simply picked one bit in each chromosome at random and flipped it. For the fitness function I calculated the number of ones in each chromosome as specified. I evaluated the population by calculating the average fitness of the population.

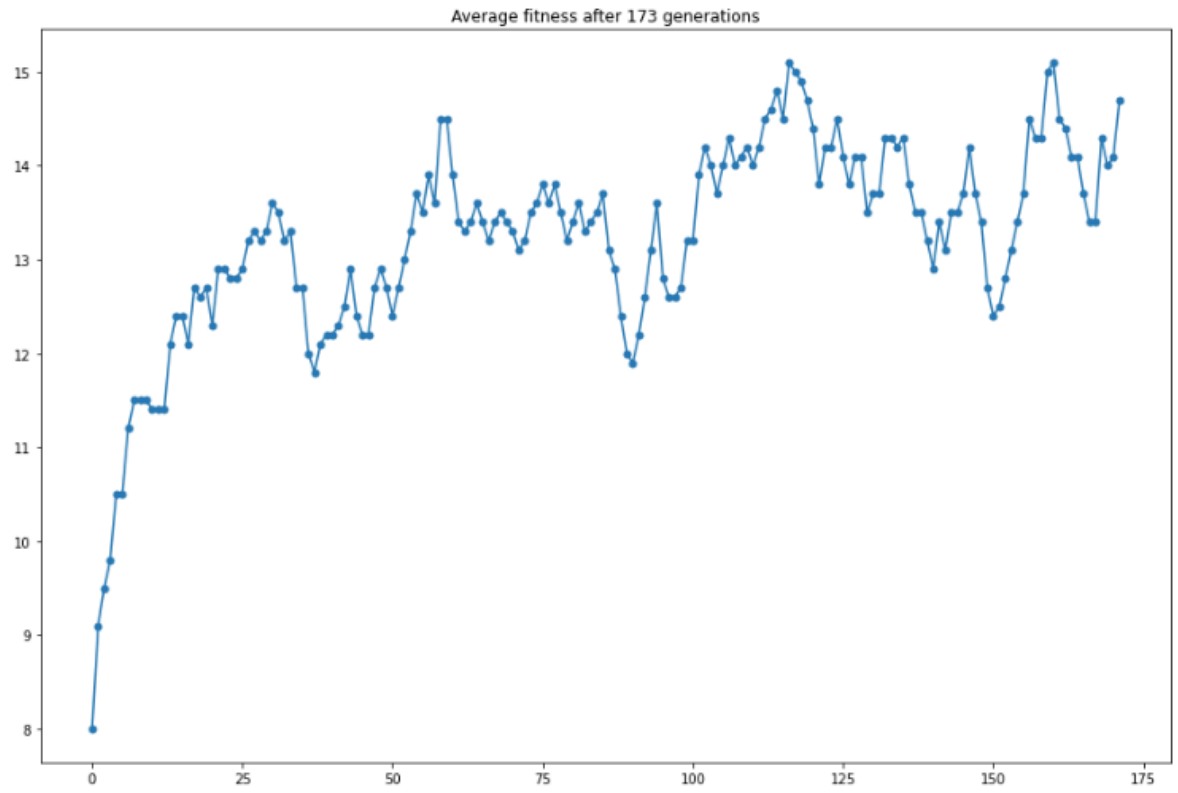

Fig 1. Plot of Onemax average fitness

Results

From Fig 1 we can see that the population fitness rose quite quickly and then began to fluctuate between ~11.5-15. It is clear the mutation function occasionally had a negative effect on the fitness. I had the GA set to run for 400 generations however it found the best chromosome after 173 generations.

Evolving to a target string

For this problem all I kept all aspect of the genetic algorithm the same except for the fitness function. I altered the fitness function to compare the chromosome to the target string and assign it a score based on how many matching bits they share.

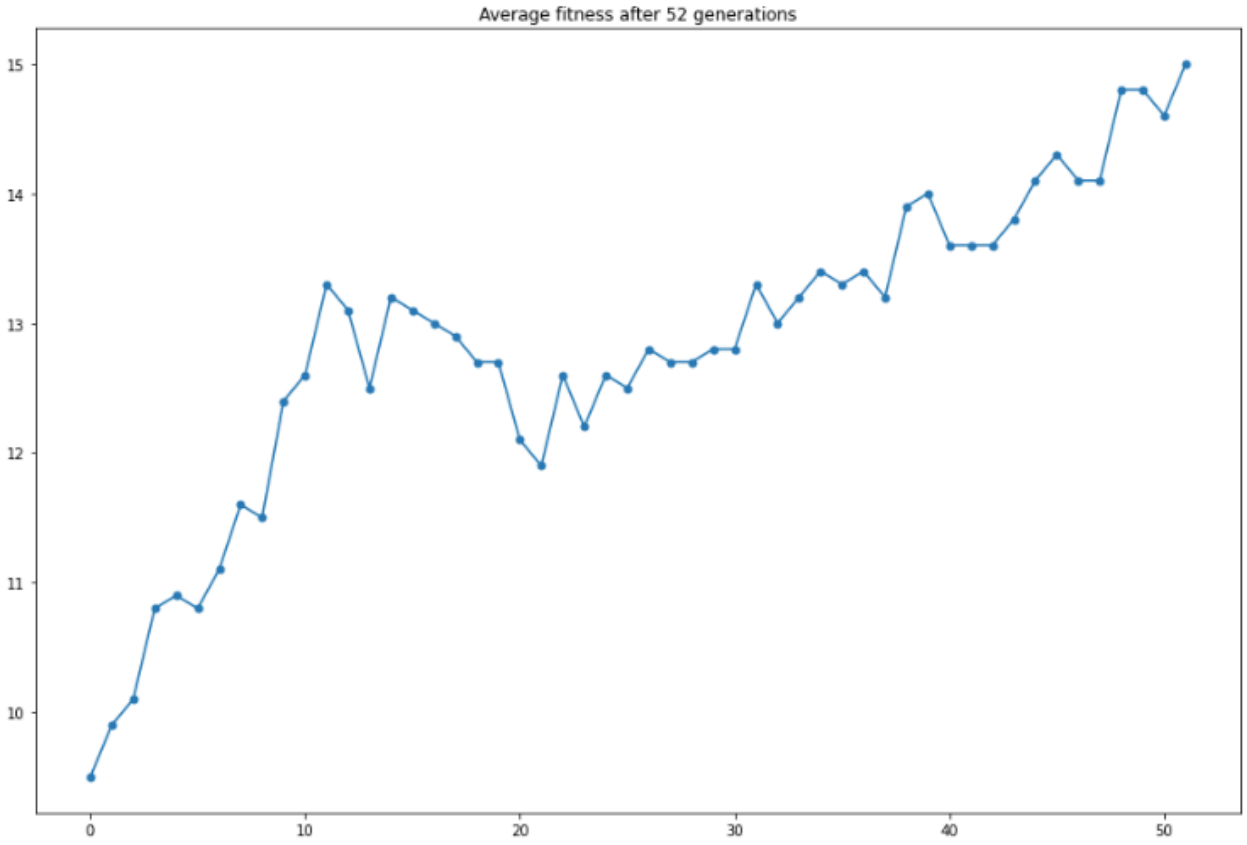

Figure 2. Plot of average fitness of the Evolve to string algorithm

Results

We can see that average fitness grew gradually until it reached the target string after 52 generations. We can also see that the algorithm reached a local Max at around 13.5 but managed to overcome it and keep rising above that level.

Deceptive Landscape

For this problem I altered the fitness function to check if the string featured all zeros, if so a fitness of 40 was assign to the string.

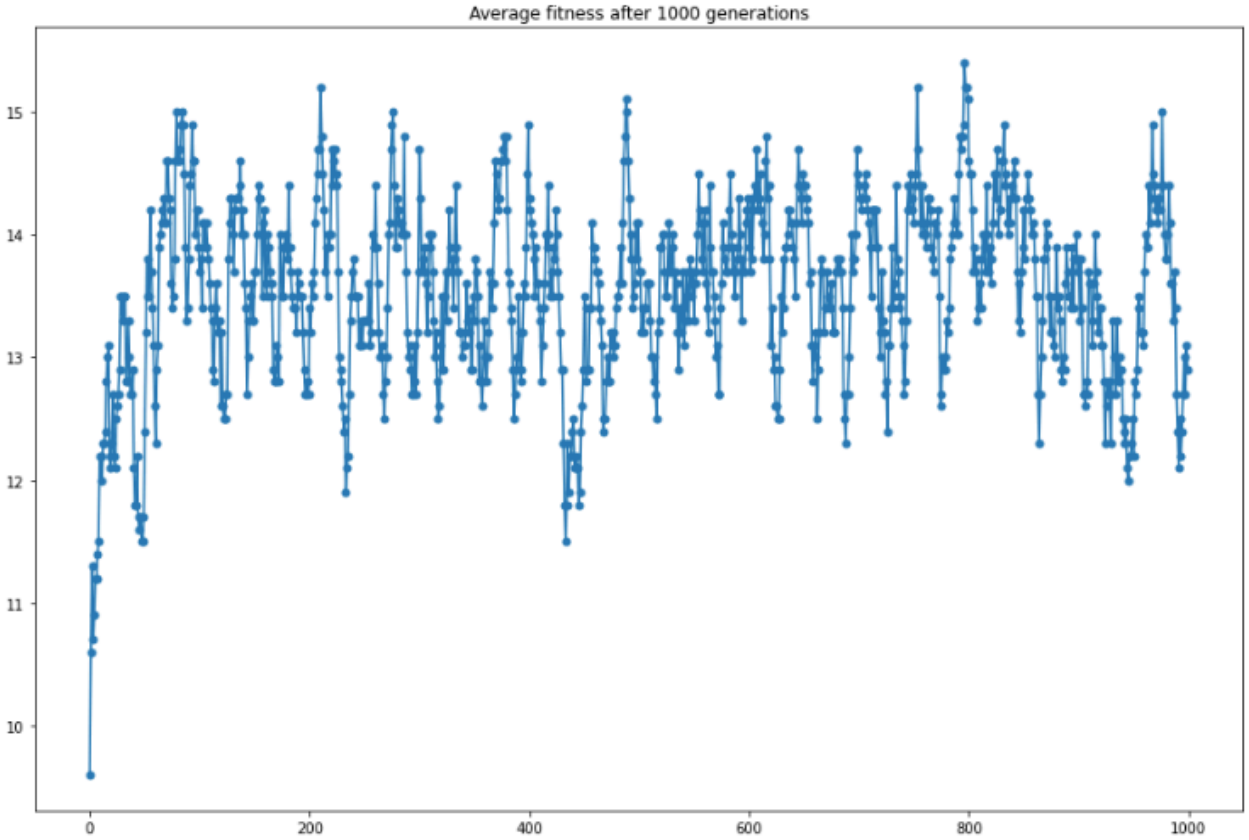

Figure 3. Plot of average fitness of Deceptive Landscape algorithm

Results

After 1000 generations the algorithm was unable to find the optimal string with all zeros. This is down to fact that the algorithm is mostly encouraging to add ones to each chromosome rather than add zeros so the optimal string is never generated.

Part B

Knapsack Problem

To represent this I problem I created strings containing 10 bits each 1 within the string represented including an item and every 0 represented excluding an item. With this representation I could use the previous functions I built for this problem.

For the fitness I simply added up the value of every item that has been included in this ‘bag’ and returned the value. At the same time I added up the weights of each item, if the total weight exceeded the max weight I assigned that string a value of 0. This meant it punished string which had exceeded the weight threshold as these solution are no use to us.

The genetic algorithm for this problem followed the same format as before with mutation, crossover and evaluation executed at every generation. Due to the fact that I could represent this problem as a bit string these functions were effective at gradually improving the fitness of the string in search of the optimal solution. Within each generation the best solution from the population is stored.

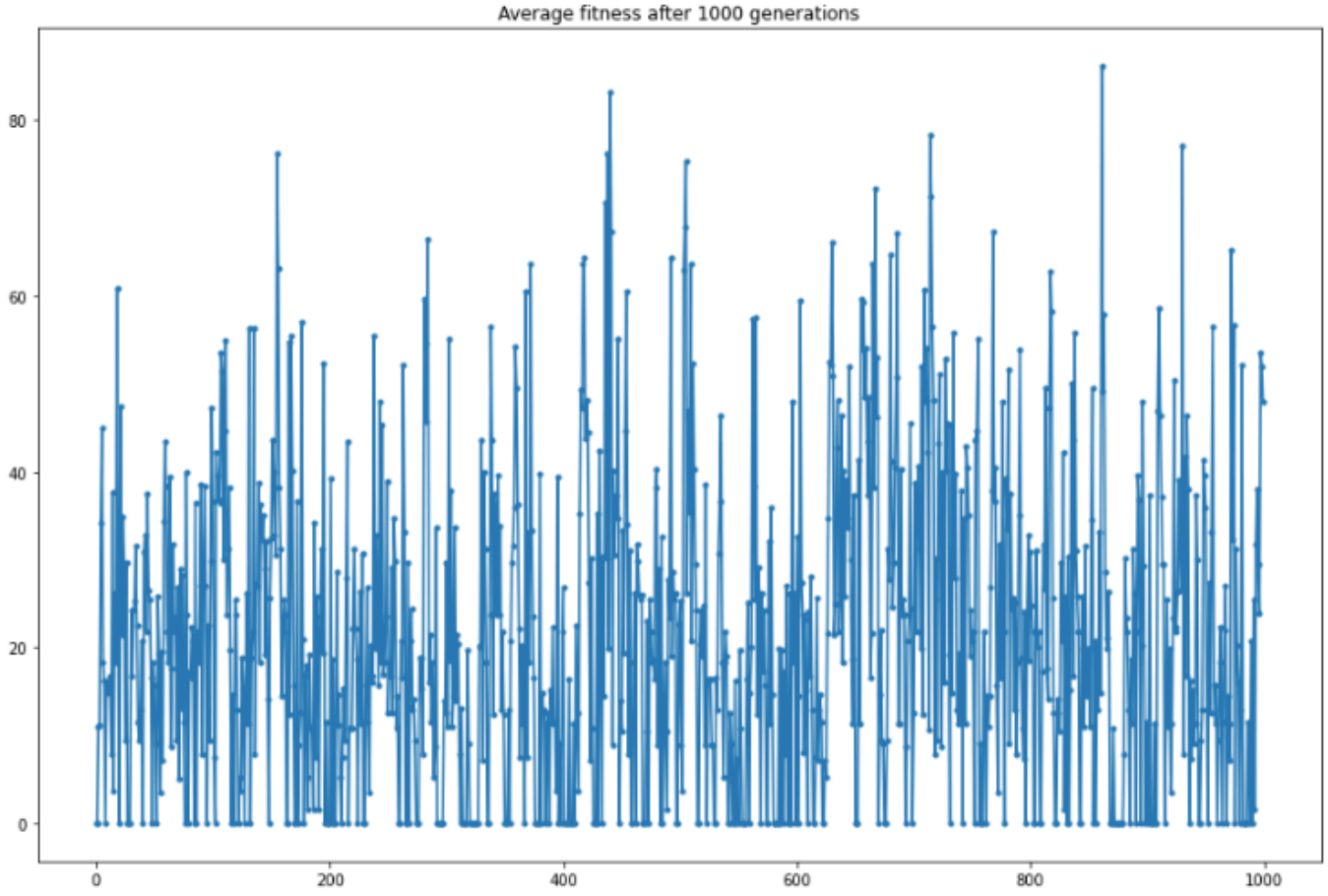

Figure 4. Plot of Average fitness with max weight 103

Due to the strict fitness function quite a lot of generations resulted in having a fitness of 0 as they all exceeded the weight capacity. As an experiment I made an adjustment to the mutation function which only changed a bit from a zero to a one if the number of ones in the given string was less than 4. This meant that strings with more than 5 items were not created as they were most likely exceeding the weight capacity. It resulted in the following graph.

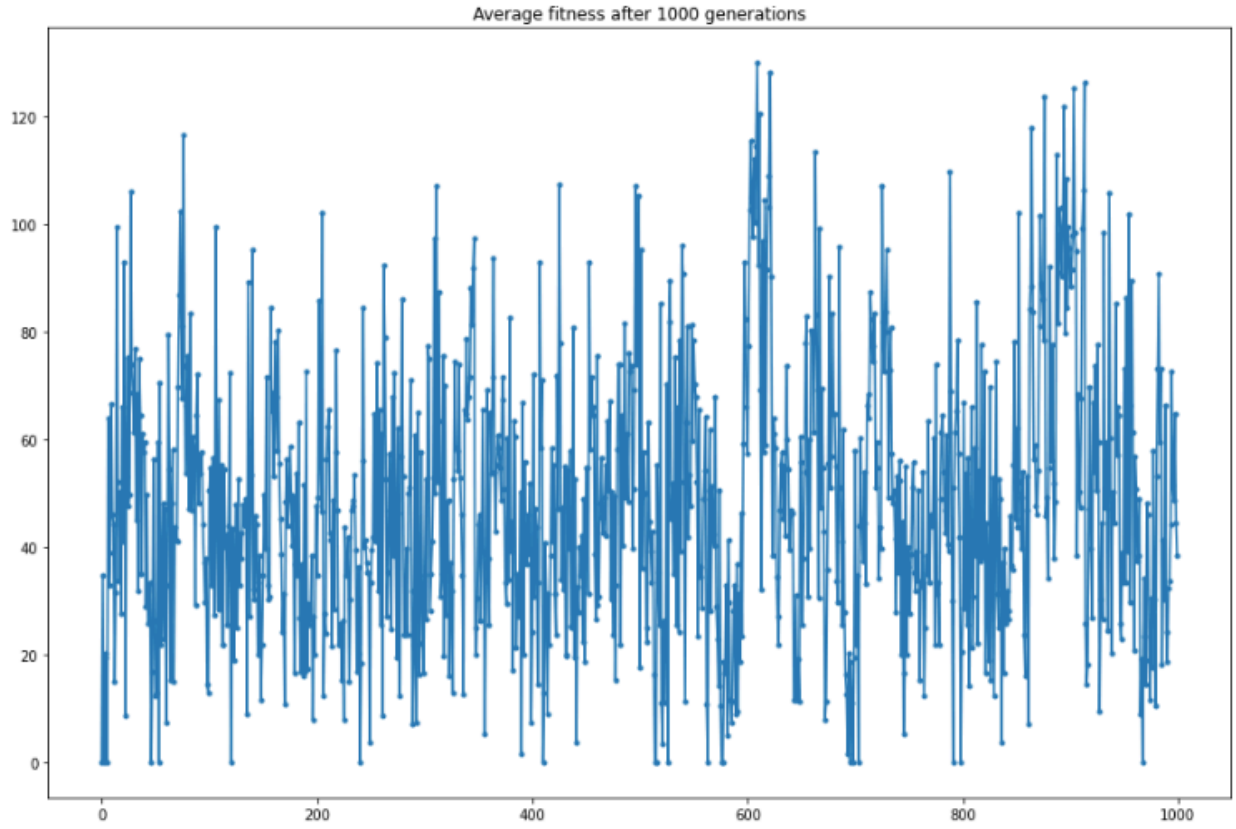

Figure 5. Plot of Average fitness with altered mutation function.

As we can see after this adjustment there were far less cases of 0 averages. With a knapsack size 103 the highest value achieved was 261 after 1000 generations which included 3 items.

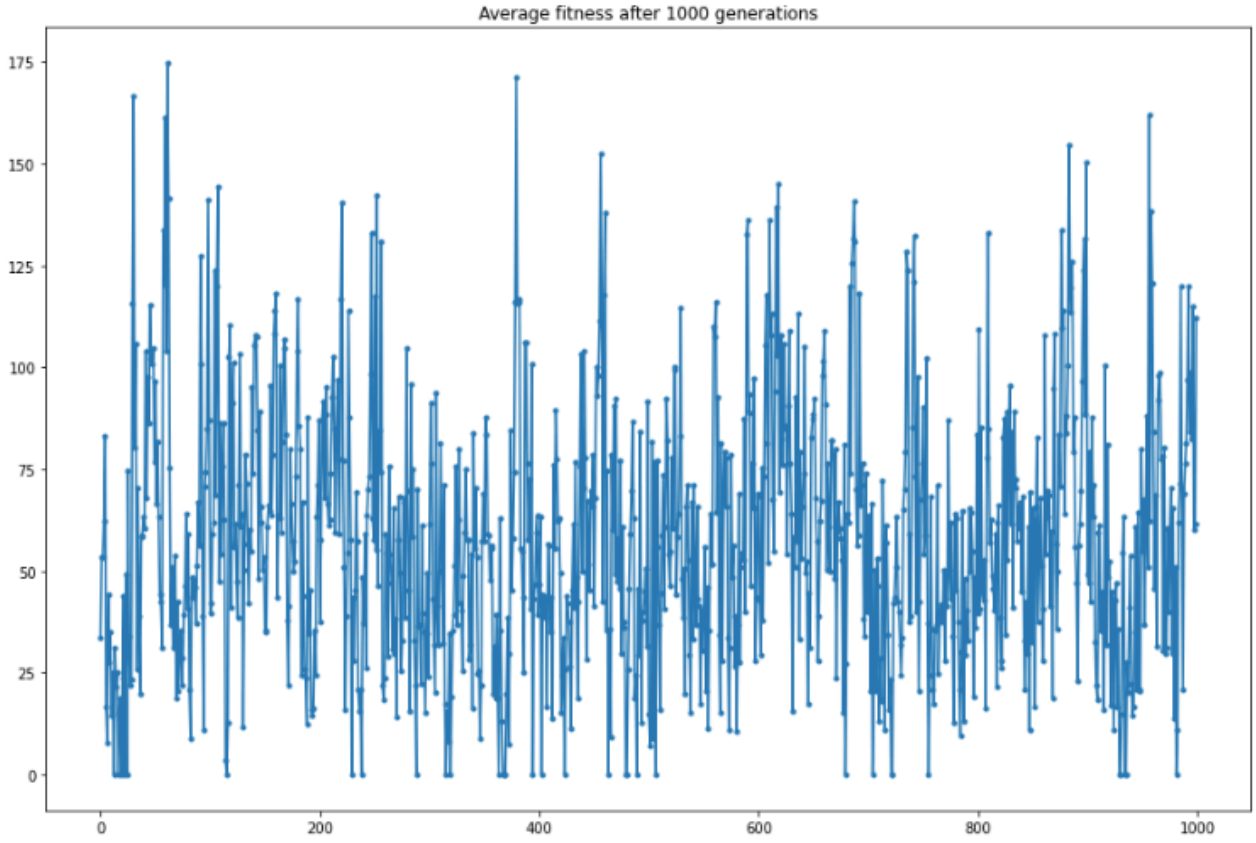

Figure 6. Plot of Average fitness with max weight 153

When the max weight was set to 153 a value of 332 was achieved after 1000 generations which included 5 items.