Travelling Sales Man Genetic Algorithm Analysis

Tyvon Factor-Gaymmon

Abstract—Evolution is a fascinating concept, first proposed by Charles Darwin, that many to this day still cannot completely grasp [2]. It is a concept that was put fourth to explain why certain organisms are the way they are and why others die out. It is responsible for all life on Earth and can explain why the biosphere is composed of the life it supports now.

Index Terms—Evolution, Natural Selection, Survival of the Fittest, Genetic Algorithm, Gene, Chromosome, Euclidean Distance.

1 Introduction

Evolution can provide solutions that enable organisms to survive. In order to understand evolution, we must first talk about Natural Selection and the concept of Survival of the Fittest. It is the process in which organisms adapt and change in response to their environment. Also, the fittest individuals, chromosomes, are those that are able to survive and reproduce. This ensures that the fittest genes are passed to the next generation. Computer scientists have been inspired to create algorithms that are nature based. One such algorithm is the genetic algorithm (GA), developed by John Holland [2]. This type of algorithm is modeled after biological evolution and it involves the process of Natural Selection [2]. Since the process of natural selection allows the fittest organisms to survive, it is a great process to implement within algorithms. That being said, natural selection and survival of the fittest, provides a means to evolve a solution given a population. The idea of evolving solutions can be used for all sorts of problems. One such problem is the classic travelling salesman problem (TSP), which is a problem highly studied in computational mathematics. It is a minimization problem in which there is a salesman that is supposed to take the shortest path to visit all the cities and return back to his/her starting point. This report focuses on solving the TSP using a GA in order to demonstrate how useful GAs are.

2 Background

As briefly mentioned earlier, evolution encompasses a variety of concepts. Although there is much to discuss when speaking about evolution, only a few of these concepts will be covered. They are: survival of the fittest (SOTF), crossover, and mutation. SOTF, is based around the idea that there is a positive correlation between organism fitness and passing on genes to the next generation [1]. An example in history is that the long-necked girrafes outcompeted the short-necked giraffes. This means the fitness of the long-necked giraffes were better suited to their environment than their short-necked counterparts. One species could reach the tall trees while the other could not. Therefore the species that survived got to pass on their genes and that is why the short-necked ones went extinct. Crossover simply means that offsping will inherit their genes from there parents. In nature, everything is subjected to the possibility of mutation. Genes are able to mutate which can be detrimental to an individual's fitness or beneficial to it. Returning back to the TSP, each city can be treated as a gene and a chromosome composes all the genes (cities) that needs to be visited. Now that the basics of a GA has been reviewed, the following pseudocode will be easier to understand.

Algorithm 1 Genetic Algorithm

```
process_city_data()   Initiate & Evaluate chromosome
for g in MAX_GENERATION _SPAN do
    Get Elites
    Tournament Selection
    Perform Crossover
    Perform Mutation
    Transfer Elites
    Compute Fittest and Avg Fitness
```

2.1 process_city_data()

As can be seen in Algorithm 1, the first thing that must be done is the processing of the city data. This data is read from a text file. The function that does this reads the city number, and x and y coordinates into a list. This list is then used to create a dictionary out of the the list so that each city can act as a unique key, with x and y coordinates.

Example 2.1. city_dict['1']: {'x': 38.24, 'y': 20.42}.

Remember, cities are represented by numbers. As can be seen in Example 2.1, city '1' has an x coordinate of 38.24 and a y coordinate of 20.42. Next, since each key represents a unique city, a list will be made up of all the keys. This list will then be subjected to random permutations until there the number of chromosomes is equal to the population size. Afterwards, the chromosomes, now made up of a random permutation of cities, are evaluated based on their total euclidean distance. The lower the total distance, the fitter the individual.

2.2 Elitism Function

Elitism is the process whereby the most fit chromosomes are preserved throughout the generations. This is implemented so that the algorithm can maintain the fittest chromosomes in the population or produce better chromosomes. This function sorts a dictionary of chromosomes based off their fitness values and returns a list of the most fit chromosomes.

Example 2.2.

Chromosome_1: total distance: 97
Chromosome_2: total distance: 102
Chromosome_3: total distance: 110
Chromosome_1 will be selected as an Elite and will later be transferred to the new population without any alterations

2.3 Tournament Selection Function

Tournament selection is a technique used to select the most fit individuals and put them in a mating pool. This way this pool of individuals can mate and produce more fit offspring. The function implemented here takes in the initial population, the permuted population, and subjects them to this tournament selection. K individuals are randomly chosen to compete and the most fittest one will move on to the mating pool. The mating pool will be composed of pairs of chromosomes, known as parents, that have been selected one after the other.

Example 2.3.

Parent_1:{
[22, 11, 9, 1, 17, 21, 4, 6, 13, 14, 18, 3, 10, 5, 7, 19, 15, 20, 16, 12, 2, 8], 'total_distance': 171.748},
Parent_2:
{[19, 21, 12, 16, 8, 18, 4, 9, 6, 3, 2, 1, 14, 7, 10, 13, 17, 11, 22, 20, 15, 5], 'total_distance': 151.313}
Please note that the listed numbers are strings and that each Parent, is actually a 'chromosome'.

As can be seen from Example 2.2, the dictionary if filled with the mating population in pairs. These pairs have won the tournament.

2.4 Crossover Function

Now that a mating pool has been generated, it can be subjected to the crossover function. This function takes in the mating pool, a parent pair dictionary, and produces children chromosomes. Children are produces via a crossover between the genes the parents have. These children will directly inherit chromosomes from either parent and then the rest of to genes will come from the parent they didn't directly receive them from. For the purposes of this study, an ordered crossover operation will be used. This means that a random start and end value will be used so that the children can inherit directly from their parents, starting from one gene all the way to another By using the parents in Example 2.3, the following children will be produced:

Example 2.4.

child_1:{
[9, 2, 1, 17, 11, 22, 4, 6, 13, 14, 18, 3, 10, 5, 7, 19, 15, 20, 16, 12, 21, 8], 'total_distance': 170.523},
child_2:{
[5, 19, 15, 16, 12, 8, 4, 9, 6, 3, 2, 1, 14, 7, 10, 13, 17, 11, 22, 20, 21, 18], 'total_distance': 171.558}
start value: 6, end value:19
All the genes between and including genes 6-19 are directly inherited from their parents. Note that counting genes start at 0.

The two children chromosomes, in Example 2.4, will replace their parents in the current population. These two chromosomes clearly have less fitness values and so they reduce the over fitness in the population. Notice that after each crossover, the children are evaluated right after they receive all their genes.

2.5 Mutation Function

As stated earlier, mutations within a chromosome happen from time to time which could either benefit the organism or negatively effect it. By subjecting a population to mutation, the chromosomes within the population can improve or reduce the over fitness average for the population. This GA will implement the inversion mutation operator in which a segment of genes is randomly chosen within a chromosome and it is inverted. This function will take in the current population of sibling pairs and subject them to mutation if R ¿ Mutation Rate. By mutating the first child chromosome from Example 2.4, a possible outcome is as follows:

Example 2.5.

Chromosome_1:{
[9, 2, 1, 6, 4, 22, 11, 17, 13, 14, 18, 3, 10, 5, 7, 19, 15, 20, 16, 12, 21, 8], 'total_distance': 173.523}
start:3, end: 7
Child_1's chromosome has been mutated starting from and including genes 3-7.

According to the previous example, this mutation was detrimental to the chromosome and so it's fitness value has been reduced. The mutation could have inverted the genes within this chromosome to improve its fitness but it is all up to chance. Notice that after each mutation, the chromosomes are re-evaluated.

2.6 Transfer Elites

Now that a new population has successfully been established, the elites can be transferred over to this population. This function will take in the elites and the mutated population, includes those who have not been selected to mutate. Depending on if the max population size is even or odd, this function will correct the population size. If the max population size is 101 then we'd have 102 children and so the worst one will be removed. Also, the worst chromosomes within the population will be replaced with the elites.

2.7 Compute Fitness and Average Fitness Function

This function is designed to take in a dictionary, composed of the new population, and also the list of elites, generated earlier. It will return the average population distance/fitness, and also the fittest individual within the population.

3 Experimental Setup

The experiment was designed to be user friendly so that the parameters can be changed easily. The parameters used in these experiments are: POPULATION_SIZE, CROSSOVER_RATE, MUTATION_RATE, SELECTION_K, ELITISM_RATE, MAX_GENERATION_SPAN,

MAX_RUNS, and ORDERED. The POPULATION_SIZE refers to the number of chromosomes that will be used in the experiment. The CROSSOVER_RATE is the probability that crossover will occur between two parent. The crossover operators used in this experiment are ordered, and uniform crossover. The MUTATION_RATE is the probability that an individual's genes will be mutated and modified. The mutation operator used in this experiment is inversion mutation.SELECTION_K represents the amount of individuals that will participate in a selection tournament. ORDERED is a parameter that is a Boolean value that lets the user choose whether to perform an ordered crossover or a uniform crossover within the GA.

4 Results

TABLE 1

Experiment 1A		
Statistics	Average pop Fitness	Average Fittest Chromosome
Mean	797.155	679.955
Median	739.242	620.197
Standard Deviation	172.556	154.935
Min	609.316	512.248
Max	1315.944	1130.038

TABLE 2

Experiment 2A		
Statistics	Average pop Fitness	Average Fittest Chromosome
Mean	828.425	703.579
Median	782.163	645.872
Standard Deviation	162.859	152.012
Min	647.869	537.841
Max	1316.405	1130.038

TABLE 3

Experiment 3A		
Statistics	Average pop Fitness	Average Fittest Chromosome
Mean	784.044	682.318
Median	717.872	633.154
Standard Deviation	180.576	158.753
Min	598.486	519.163
Max	1315.578	1117.128

TABLE 4

Experiment 4A		
Statistics	Average pop Fitness	Average Fittest Chromosome
Mean	792.075	681.480
Median	742.058	637.036
Standard Deviation	164.580	144.532
Min	627.513	523.512
Max	1315.204	1117.128

TABLE 5

Experiment 5A		
Statistics	Average pop Fitness	Average Fittest Chromosome
Mean	921.849	743.652
Median	904.311	689.073
Standard Deviation	98.164	111.510
Min	815.462	646.364
Max	1313.844	1105.482

TABLE 6

T-Test			
	Test Statistic	P-value	Degrees of Freedom
Exp 1 & 2	-1.31788	0.18907	198.0
Exp 1 & 5	-6.28107	2.09116	198.0

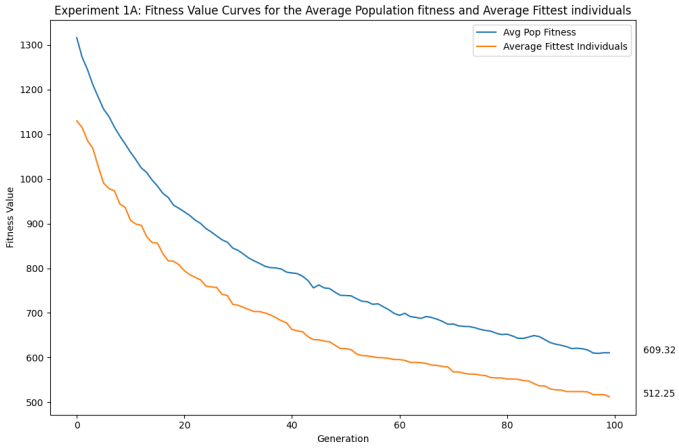

Fig. 1. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was ordered. For this experiment, the crossover rate was 100% and the mutation rate was 0%.

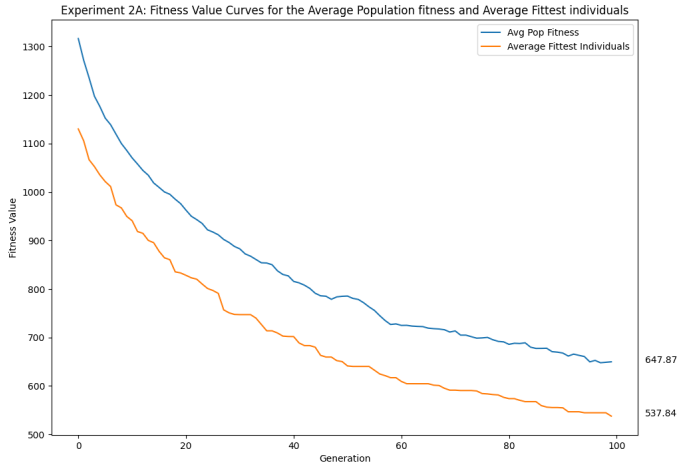

Fig. 2. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was ordered.For this experiment, the crossover rate was 100% and the mutation rate was 10%.

Table 1 shows us that experiment 1A has yielded the best results with a crossover rate of 100%. As can be seen from Fig. 3, experiment 3A had the second best results by the end of 100 generations. As can be seen by looking at the tables, the configuration used in experiment 5A had the worst set up. The mean of the average fittest chromosomes, by generation 100, of all the experiments is calculated by taking the sum of these averages and dividing them by the number

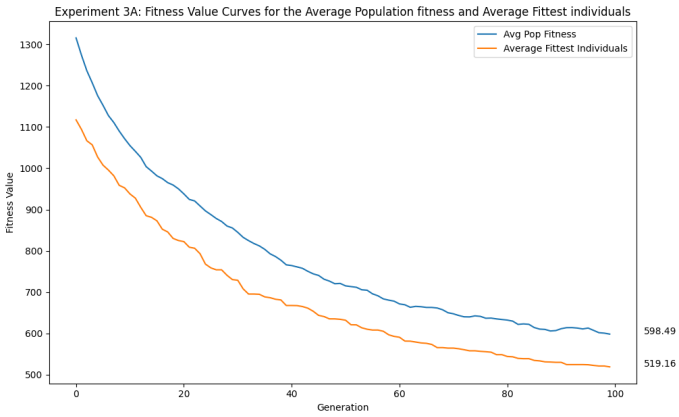

Fig. 3. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was ordered.For this experiment, the crossover rate was 90% and the mutation rate was 0%.

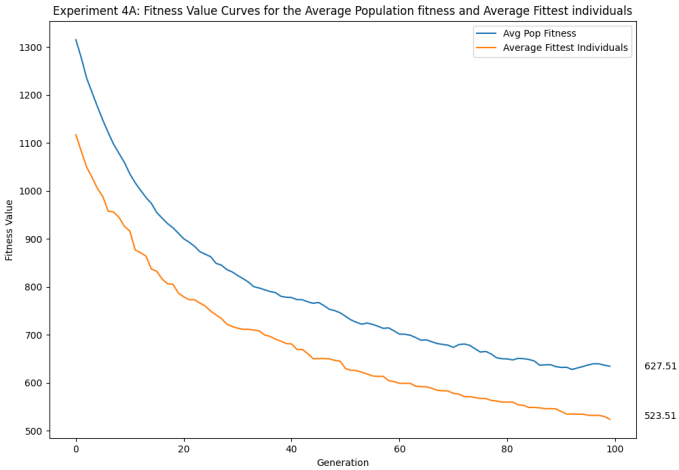

Fig. 4. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was ordered.For this experiment, the crossover rate was 90% and the mutation rate was 10%.

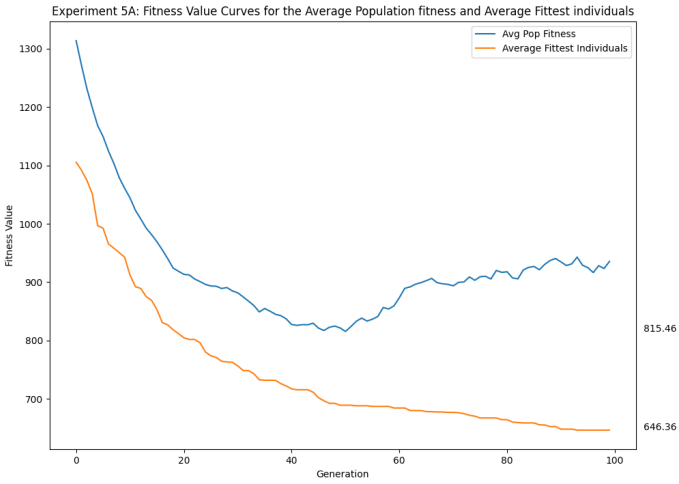

Fig. 5. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was ordered.For this experiment, the crossover rate was 80% and the mutation rate was 20%. The average population fitness is not necessarily improving.

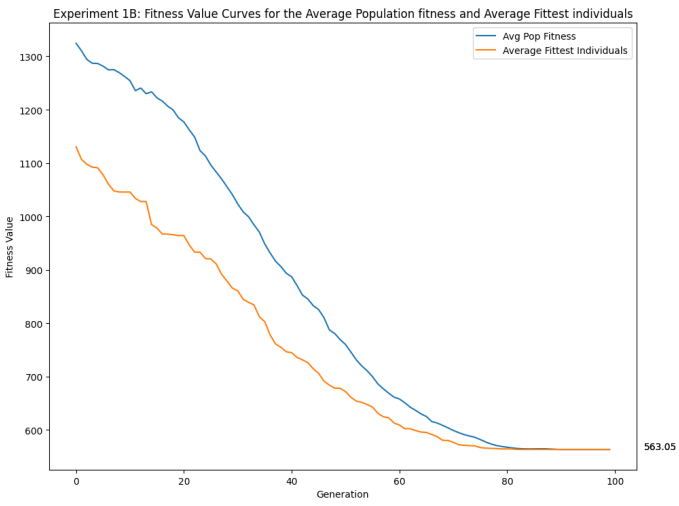

Fig. 6. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was uniform. For this experiment, the crossover rate was 100% and the mutation rate was 0%.

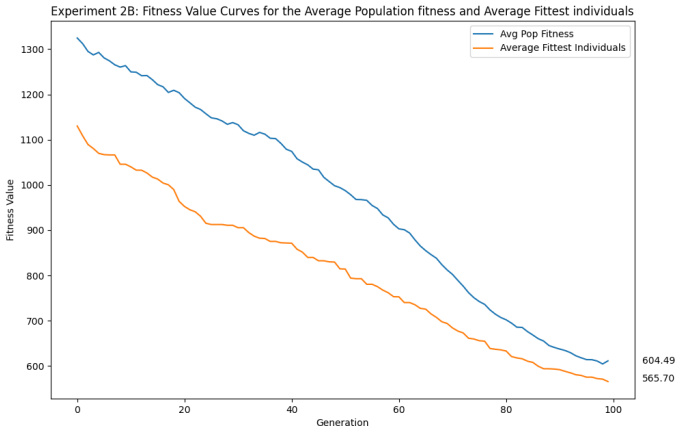

Fig. 7. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was uniform.For this experiment, the crossover rate was 100% and the mutation rate was 10%.

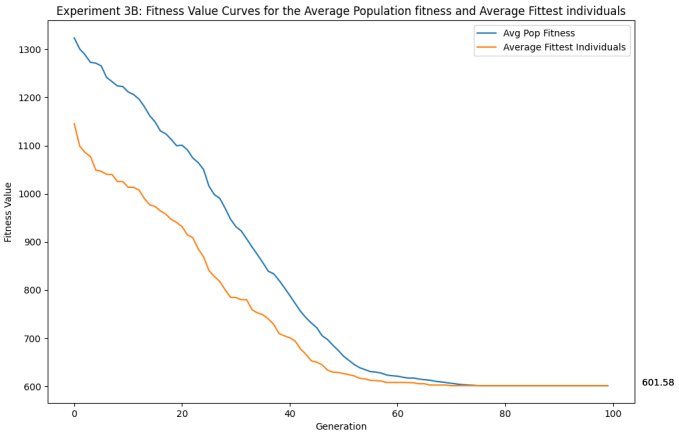

Fig. 8. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was uniform.For this experiment, the crossover rate was 90% and the mutation rate was 0%.

Fig. 9. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was uniform.For this experiment, the crossover rate was 90% and the mutation rate was 10%.

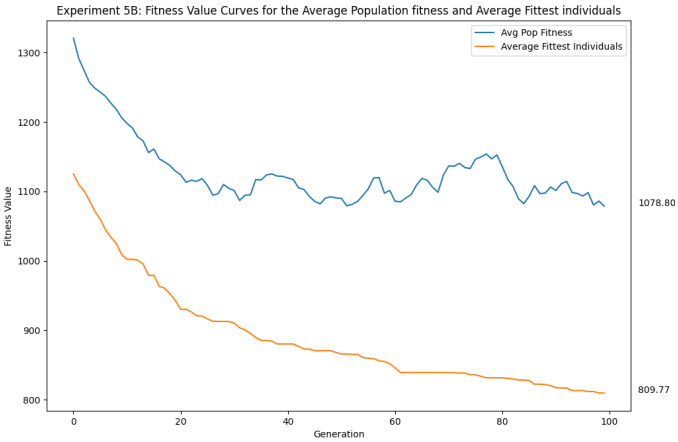

Fig. 10. This graph shows convergence curves for the average population fitness and the average fittest individuals. The Crossover Operator used here was uniform.For this experiment, the crossover rate was 80% and the mutation rate was 20%. The average population fitness is not necessarily improving.

of experiments. This overall mean for the 5 experiments is a fitness of 547.82. A T-test has been performed between the best, average and worst experiments. Table 6 shows that the t tests between the best and average experiments and the best and worst experiments. For the first t-test, it can be said that since the t statistic is greater than the value of

$$t_{1-\alpha}, \tag{1}$$

the null hypothesis can be rejected. The null hypothesis states that there is not difference between the two experiments. Clearly there is a difference between experiment 1A and experiment 2A. Looking at the second comparison between experiment 1 and 5, it is evident that the data occurred due to chance. The p value is so high that the experiment is most probably due to chance. This makes sense as the mutation rate was pretty high at 20% for experiment 5A.

5 Discussions and Conclusions

It is evident by looking at both Figures 1 and 6, and Figures 3 and 8 that the crossover operator plays a significant role when applied to a GA. By observing Figures 6-10, one can assume that the crossover used here, uniform crossover, yields a less diverse population than that of its counterpart, ordered crossover. This means that the gene pool is lacking and that it will be difficult to try and evolve a solution better than whats already present in the population. The first experiment using ordered crossover shows that the average fitness decreased gradually without any major perturbations. This suggests that performing an ordered crossover may be superior than that of performing a uniform crossover. Also, as shown through Figures 5 and 10, regardless of the crossover operator used, there exists the possibility that as the crossover rate decreases and the mutation rate increases, the average population fitness doesn't improve after a specific amount of generations. The best results were achieved with a 100% crossover rate and a mutation rate of 0%. Further research and fine tweaking of the parameters is needed to suggest what parameters are optimal when applying a GA to the TSP.

References

[1] ." u*x*l complete life science resource. . encyclopedia.com. 25 oct. 2021 ., Nov 2021.

[2] Xin-She Yang. *Chapter 6: Genetic Algorithms*, page 91–100. Academic Press, 2021.