1. Introduction

In this assignment, genetic algorithm is implemented to solve travelling salesman problem. Genetic algorithm are evolutionary techniques used for optimizing survival of the fittest idea, genetic algorithms do not usually give the best solution but a close approximation. It is expected that the next generation is fitter than the previous generation because of the fittest selection algorithms. Genetic algorithms often used for NP-hard problems such as travelling salesman problem. Genetic algorithm has 5 methods

1. Evolve
2. Crossover
3. Mutate
4. Tournament

The travelling salesman problem can be described as the minimization of total distance traveled by visiting each city once and returning to the original city. For n cities there are (n - 1)! combinations. In the implementation of the travelling salesman problem such an approach can be used

1. Gene: City
2. Chromosome (Individual): A single route
3. Population: List of feasible routes
4. Parents: two routes combined to create a new route
5. Mating pool: List of parents that are used to create a new population
6. Fitness: Minimum distance
7. Mutation: Randomly swapping two cities in a route
8. Elitism: A way to carry the best individuals into the next generation

2. Algorithms

1. Evolve

Select the minimum distance routes and make them parent. Create a new child using these cities. This selection is made by using tournament method.

2. Crossover

Using parents' crossover is going to create the next generation. I used a random crossover point. Because in travelling salesman problem each location must be visited once randomly select a subset from first parent and fill the rest from the second parent in order without duplicating any cities from the first parent. Using elitism to retain the best possible routes from the current population.

3. Mutate

Mutation allows the genetic algorithm the avoid local maxima problem by allowing the algorithm to explore other parts of the solution space

4. Tournament

A set of number of individuals are randomly selected and highest fitness in the group chosen as the first parent, second one is also chosen this way.

3. Implementation Details

1. Pseudocode

START

Generate the initial
population Compute fitness

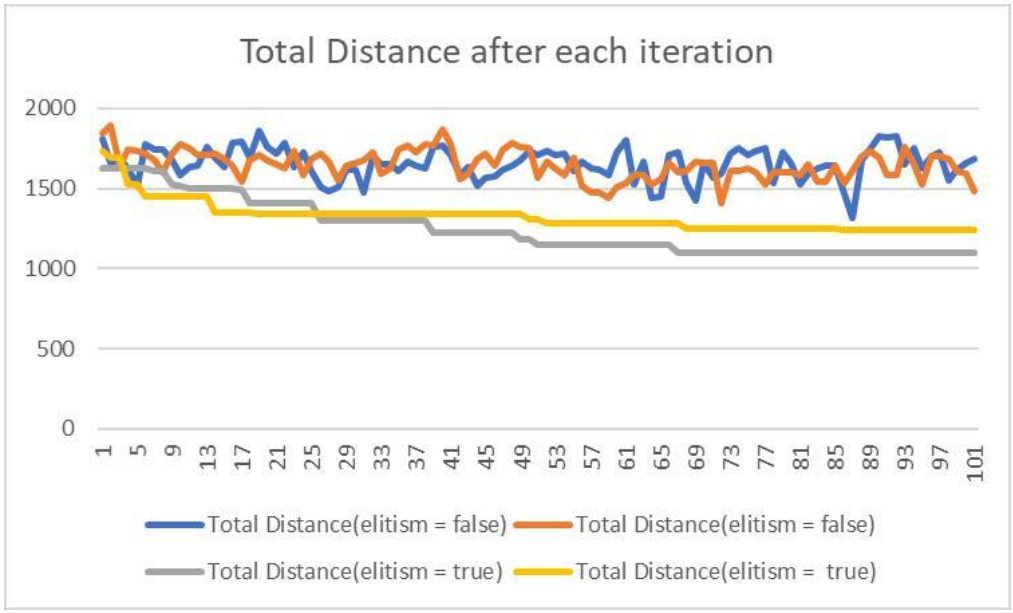
REPEAT Selection
  Crossover
  Mutation
  Compute
  fitness

UNTIL population has converged

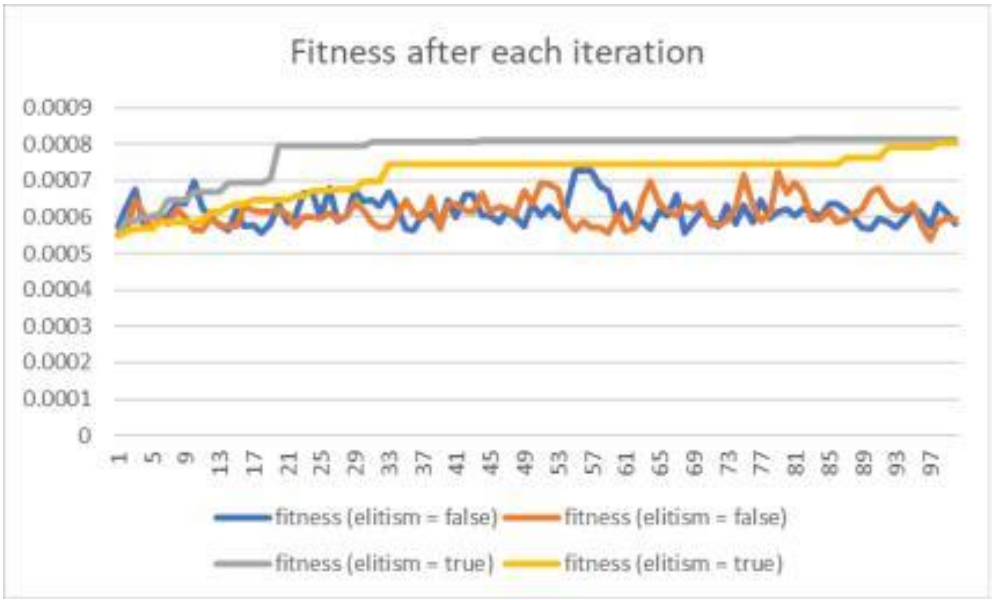
STOP

4. Results

1. Total Distance



2. Fitness



5. Discussion

With the randomly generated route we were able to shorten the distance with both elitism values. With elitism true the algorithm was both able to find the better fitness value and shorter total distance. The result in both cases was better than the initial values but exact optimal solution was not reached in all cases because of the mechanics of the generic algorithm (randomness).