BAU CMP3004 2020 Spring Term Project

Introduction

The aim of this project was to come up with different methods to solve a classic Computer Science problem known as the Travelling Salesman Problem.

In this project we have used one bad algorithm (exhaustive) just as a base-point. The other algorithms come close to the shortest path, and they do it in a short amount of time. Except for the genetic algorithm, which will be explained further in its section.

Currently the programs have the default 48 cities in their source files, but this can easily be changed to be read from an external file.

Exhaustive Search (Onur Orkun Kader)

This is a factorial time algorithm, O(n!), from some benchmarks it would probably take 10^{57} years to go through all 48! permutations.

What it does is basically generate every single possible city route, and tries to find the shortest distance from them.

Unfortunately because the exhaustive search takes a long time we cannot get a short distance, the best distance as of running for 15 minutes was 150604, which is around five times as long as the optimal solution.

Nearest Neighbour Algorithm (Onur Orkun Kader)

This algorithm starts at the first city then finds the closest city to it by checking all the other cities, resulting in an $O(n^2)$ check for the cities. It iterates through the remaining slots and filling them with the next closest city.

The best distance starting at the first city is 40527, which takes 293μs (microseconds).

There is also a repetitive version of this algorithm which repeats NN with different starting cities, there is no point in repeating more than 47 times so that is the default.

And the best distance for the repetitive version is 40527 which is the same as the original version. RNN takes 11ms (milliseconds) to complete.

Greedy Algorithm (Eren Özdemir)

This algorithm achieves to find the best results it can find with the least amount of operations. Each individual step of the algorithm takes O(n), and for every n element the total runtime results in $O(n^2)$.

The program finds the edges between the vertices, cities, calculates the distance and puts everything into a list and sorts them.

The shortest distance it finds is 97397, and it does it in 292μs

Divide & Conquer (Oktay Mert Aküzüm)

This algorithm in essence is just dividing the problem into smaller parts and solving the smaller parts first and then continues recursively to find a good solution.

In the Travelling Salesman Problem we can first divide the route into smaller routes and first find the optimal sorting for the smaller ones, and then combining them together.

The shortest distance is 37008, and it takes 153ms to find the route.

Genetic Algorithm (Onur Orkun Kader)

This algorithm starts with a population of a given size filled with random orderings of the cities. Then it calculates the fitness values for all of them according to how small the total distance of that array is. Afterwards the program selects two parents using their fitnesses to apply a crossover maneuver, creating a child from them with a chance of mutation on the DNA of the child. The mutation is just iterating over the array and swapping a value with the next one if the probability matches.

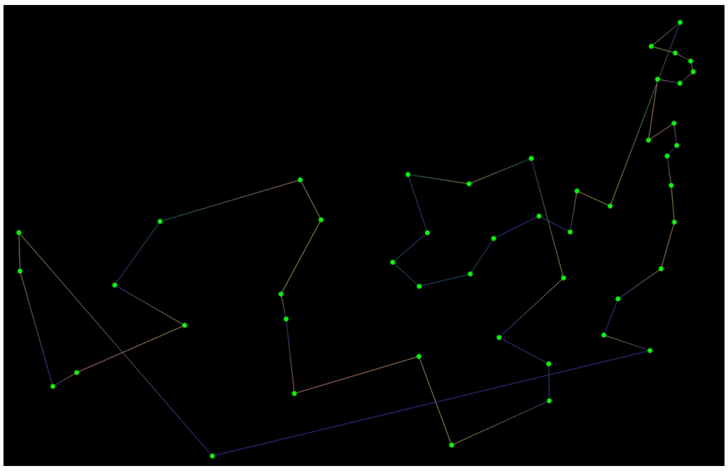
The children make up the new generation, and the cycle continues. Because the more **fit** parents have higher chances of being picked, the newer generations will have better city orders, genes, in them resulting in the population getting fitter and approaching the optimal solution.

Unfortunately this specific algorithm doesn't work that well, the most likely cause is the fitness function and the way the crossover happens, and the major contributor to its slowness is the randomness side of it. A proper uniform distribution is important for this algorithm but C++'s default uniform distributor is quite slow.

But after running the program for around 18 minutes gives the distance of 71057 after 155591 generations.

Visualizations / Outputs

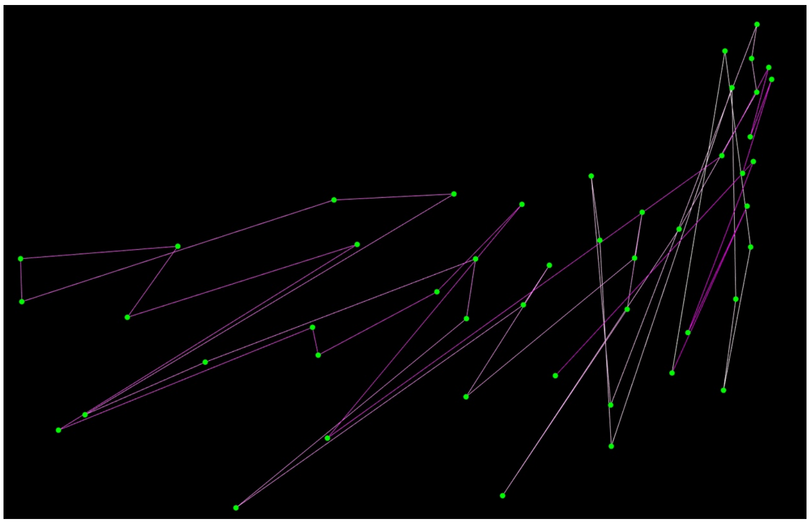
Nearest Neighbour:



[0,8,37,30,43,17,6,27,35,29,5,36,18,26,42,16,45,32,14,11,10,22,13,24,12,20,46,19,39,2,21,15,40,33,28,4,47,38,31,23,9,41,25,3,34,44,1,7]
Distance: 40527
Time: 293µs

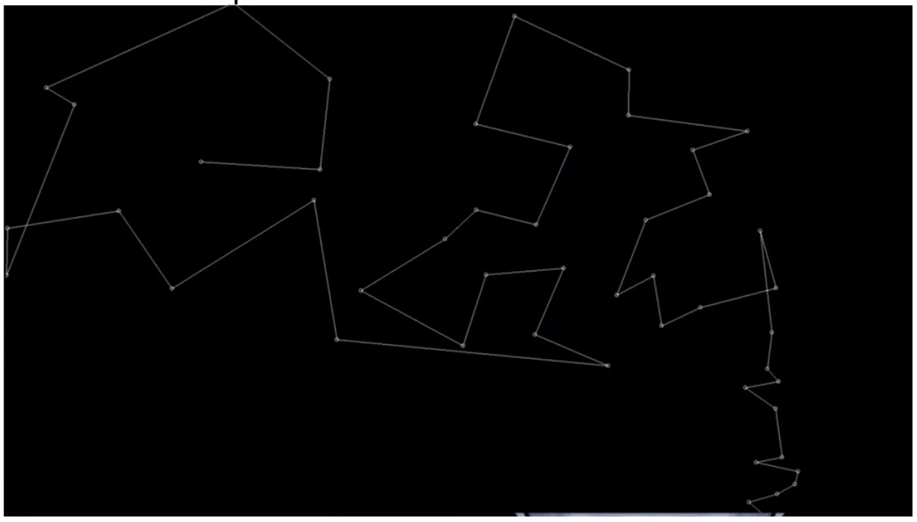
Repetitive Nearest Neighbour:
[0,8,37,30,43,17,6,27,35,29,5,36,18,26,42,16,45,32,14,11,10,22,13,24,12,20,46,19,39,2,21,15,40,33,28,4,47,38,31,23,9,41,25,3,34,44,1,7]
Distance: 40527
Time: 12ms
RNN finds the same result as NN.

Greedy:



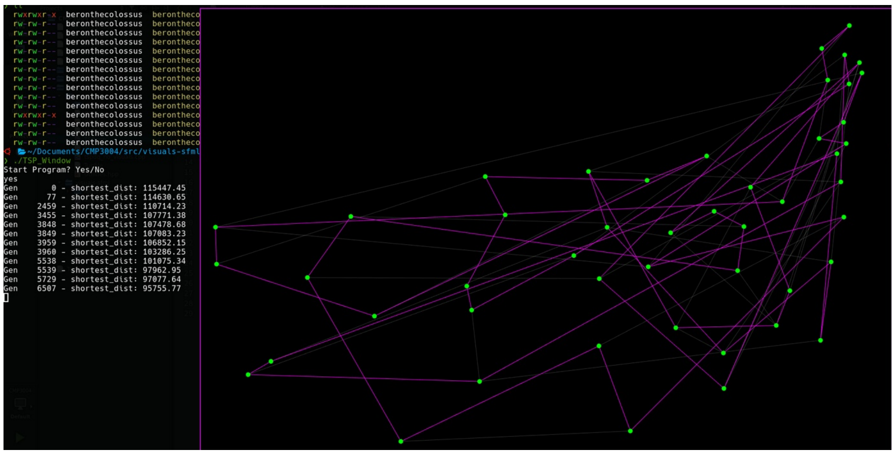
[0,42,30,7,37,29,15,11,19,21,16,26,5,45,40,39,32,14,33,10,22,1,13,12,41,25,20,31,34,44,
23,9,38,3,47,4,24,46,28,35,18,27,36,17,2,6,8,43]
Distance: 97397
Time: 292μs

Divide and Conquer:



[42,5,29,2,4,26,45,35,10,24,48,32,20,47,11,13,21,39,25,14,23,3,34,41,16,22,8,1,9,40,12,1
5,33,46,31,38,44,18,7,36,28,6,30,37,19,27,43,17]
Distance: 37008
Time: 153ms

Genetic:



```
  rwxrwxr-x  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rwxrwxr-x  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  rw-rw-r--  beronthecolossus  beronthecο
  ~/Documents/CMP3004/src/visuals-sfml
> ./TSP_Window
Start Program? Yes/No
yes
Gen       0 - shortest_dist: 115447.45
Gen      77 - shortest_dist: 114630.65
Gen    2459 - shortest_dist: 110714.23
Gen    3455 - shortest_dist: 107771.38
Gen    3848 - shortest_dist: 107478.68
Gen    3849 - shortest_dist: 107083.23
Gen    3959 - shortest_dist: 106852.15
Gen    3960 - shortest_dist: 103286.25
Gen    5538 - shortest_dist: 101075.34
Gen    5539 - shortest_dist: 97962.95
Gen    5729 - shortest_dist: 97077.64
Gen    6507 - shortest_dist: 95755.77
```

Unfortunately the shortest path wasn’t saved so here’s another one:
[38,6,37,43,17,39,12,1,14,36,5,18,16,42,35,0,21,33,28,9,20,31,40,47,4,41,23,19,32,7,3,25
,34,44,27,26,10,46,2,22,24,13,15,11,29,45,30,8]
Best Distance: 71057 (155591st Gen.)
Approximate Time: 18 minutes

Some links for the visuals (YouTube):
https://www.youtube.com/watch?v=3loWZlE4NTY
https://www.youtube.com/watch?v=8j6pbNzBQhg
https://www.youtube.com/watch?v=266Fqjr-5YQ
https://www.youtube.com/watch?v=Qq63476TXUc