INF3490 - Assignment 1

Torgeir Hoffmann / twhoffma@uio.no

September 25, 2017

How to run

In assigment folder:
    python3 oblig1.py

Exhaustive Search

How long time did it take for 10 cities

This took just shy of 18.4s to run for 10 cities.

What is the shortest route among 10 cities (all starting with B,C,D,H and I)?

The shortest route is 7486.31 km. Visiting the cities in order: Hamburg, Brussels, Dublin, Barcelona, Belgrade, Istanbul, Bucharest, Budapest, Berlin, Copenhagen (then return).

How long time would the 24 cities take?

Given that the number of solutions to try is 24!, one can assume that since the search algorithm is not in any way optimized with Djikstra or anything else and it is O(n) at best so rough estimated time would be (in years):

$$\frac{24!}{10!} \times \frac{18.4}{60 \times 60 \times 24 \times 365} \approx 9,75e + 10$$

Given the vast amount of solutions to attempt, nothing would actually help all that much in terms of the vast number of potential solutions to search.

Hill Climbing

It is no doubt a lot faster, but it seems that the initial routes used is a very big factor in how close it will get to the best route determined by exhausive search. For 20 initial runs, I tried combinations of 100, 1000 and 10000 switches. As expected, the run times increased linearly.

10 cities

Samples	Best	Worst	Avg	Stdev	Runtime (s)
100	7729.01	11266.64	9846.98	918.97	0.0193
1000	8393.26	11389.30	10006.95	834.94	0.1791
10000	8512.48	11140.54	9785.05	729.13	1.7359

24 cities

Samples	Best	Worst	Avg	Stdev	Runtime(s)
100	25442.11	32191.43	28109.39	1794.33	0.0266
1000	22668.25	30851.69	27097.98	2326.06	0.2519
10000	23063.84	30377.06	27445.59	2006.18	2.5174

Genetic Algorithm

It was hard to say what was the right parameters to be passed here. What I was most concened about in the selection process was premature convergence, but given the problem it would be hard to argue one way or another that two solutions with similar fitness (lowest distance), could (or couldn't) be wastly different.

Setup

Population	10,100,1000
Selection	Rank-based, top 50%. Parents are paired on descending rank (higher fitness)
Crossover	One point mirrored from other parent by switching. Probability 100%.
Mutation	Random switch of two cities. Mutation probability 100%.
Termination	No improvements in 10 generations (stdev(avg(pop)) = 0) or 5 * population children

Results: 10 Cities

Population	Best	Worst	Avg	Stdev	Run time (s)
10	7486.31	8436.39	7896.68	330.38	0.3202
100	7486.31	7726.36	7541.64	83.25	1.0180
1000	7486.31	7503.10	7487.15	3.75	9.3708

Results: 24 cities

Population	Best	Worst	Avg	Stdev	Run time(s)
10	18339.71	23965.66	21232.53	1357.17	0.3305
100	17145.50	20094.82	17993.62	735.79	1.1423
1000	15753.45	18228.10	16963.78	619.88	10.6494

Find and plot the best fit individual in each generation

... TBA ... Should include all three population sizes across generations run.

Did it find the solution in the 10 cities case? Did it come close?

I found it surprising that the genetric algorithm could run so well on 10 cities and it the mark even with only minimal cross over and mutation.

Which is best in terms of tour length and evolutions of evolution time.

The pruning in GA seems to be superior, but in terms of time, it is surprisingly slow.

How did running time compare to exhaustive search?

A lot faster on the lower populations, but rapidly increasing with population size, probably going to be slower for a large population size.

How many tours were inspected?

To be added.