Homework 3 Report

Bernard Pirlea

January 9, 2020

1 Introduction

1.1 Motivation

The Travelling salesman problem is a problem with real applications in the transport industry. To minimize the cost of transport, the consumption of gas and others. Given that the problem has a lot of solution going through all of them takes way to much time. We want to see if running a GA and Hill Climbing algorithm over this problem can help us find a good solution in less time.

1.2 Description

The purpose of this paper and of the experiment is to see how a Genetic Algorithm and Hill Climbing Best Improvement with different parameters over different data test find a better solution over the Travelling Salesman Problem.

2 Algorithms

The distance between 2 point in every case is the Euclidian Distance 2D. And the test cases have been chosen accordingly.

2.1 Genetic Algorithm

Basic Terminology of a Genetic Algorithm and its implementation:

Population - To represent the population I used a 2 dimensional vector of integers.

Chromosomes The chromosome is represented in my algorithm as a vector integers, representing the order in which the salesperson should travel.

Gene A gene is one element of a chromosome and is and integer between 0 and the number of cities-1.

Fitness Function - The function we used to determine the fitness is:

$$1.1 * max - lengthOfRoute(chromosome)$$

Selection As for the this I used the Wheel of Fortune in which every individual can become a parent with a probability which is proportional to its fitness.

Operator:

Mutation is implemented here using a swaping function that swaps 2 genes in a chromosome. The probability of mutation is given by the pm parameter. The number of genes that are mutated is around

$$pm * chromosomelength * popsize$$

Crossover Combines the genes of 2 chromosomes. For the implementation, I take the first part of a chromosome and complete the rest with the elements that are not already in the chromosome so we don't have the same city twice, taking them from the other chromosome in the order found. The probability of crossover is given by the pc parameter. The number of genes that are mutated is around:

$$pc * popsize$$

2.2 Hill Climbing

Implemented Hill Climbing Best Improvement with a restart of 20, and a function that defines a neighbor of a solution as a swap between 2 position in the solution.

3 Test Data

I choose the following data from the TSPLIB to test the algorithm:
Minimum for each data set:
Berlin52 : 7526
Bier127: 118282
Ch130: 6110
Pr76: 108159

Figure 1: Berlin52 plot

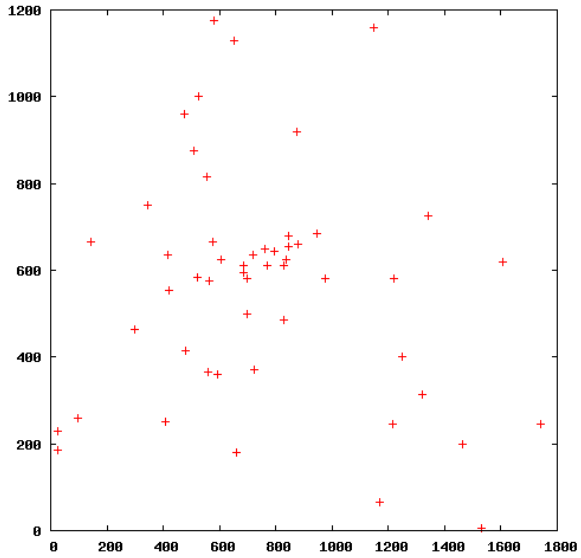
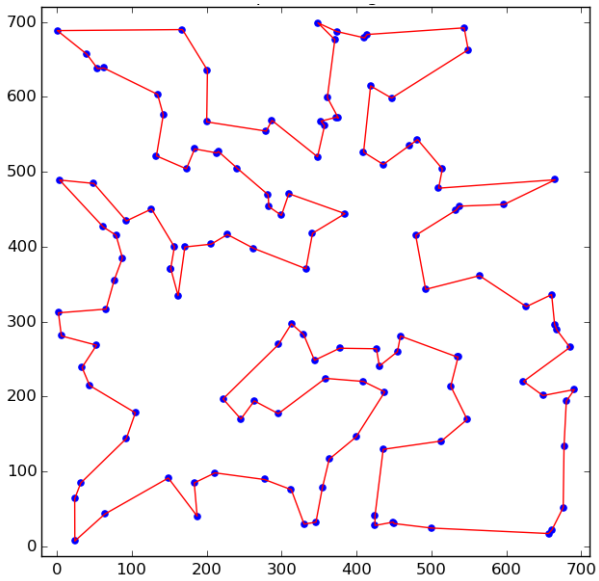


Figure 2: CH130



Other data sets:
bier127
pr76

4 Experiment

As for the experiment, I executed algorithms 30 times, on a population of 100 individuals, over 1000 generations. The pm = 0.02 and pc = 0.7. As I saw that mutation and crossover weren't enough to find the solution, I added a new function that executes 2-otp-search on the best solution given by the GA. I wrote down the solutions given by the algorithm. From these solutions I took the minimum, the maximum, the average, the median, calculated the standard deviation.

5 Results

5.1 Genetic algorithm

alg	min	max	avg	median	σ	time of ex
GA	7774.0	18494.0	8692.56667	8330.5	1834.33517	397.81s

Figure 3: GA on Berlin52

alg	min	max	media	median	σ	time of ex
GA	127078.0	135582.0	129822.33333	129631.5	2204.35274	1521.96s

Figure 4: GA on Bier127

alg	min	max	media	median	σ	time of ex
GA	76334.0	6918.0	6598.5	6610.5	145.97277	1604.88s

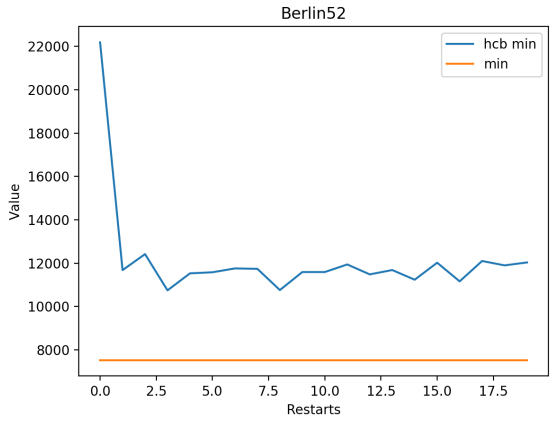
Figure 5: GA on Ch130

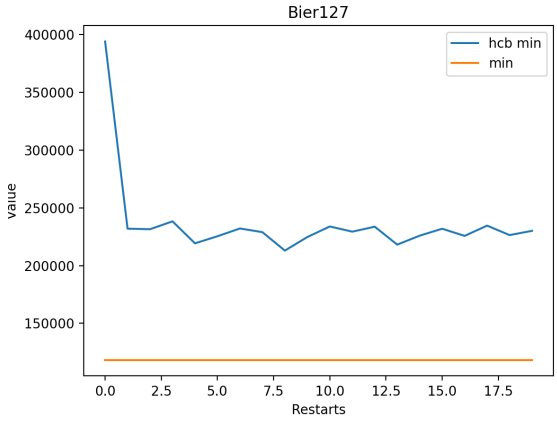
alg	min	max	media	median	σ	time of ex
GA	110816.0	122159.0	115342.23333	115075.0	2923.54377	651.702s

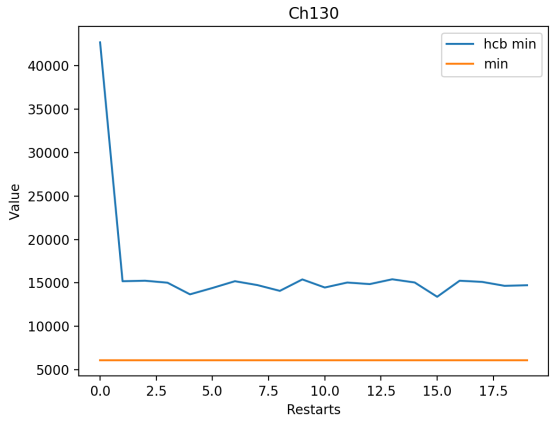
Figure 6: GA on Pr76

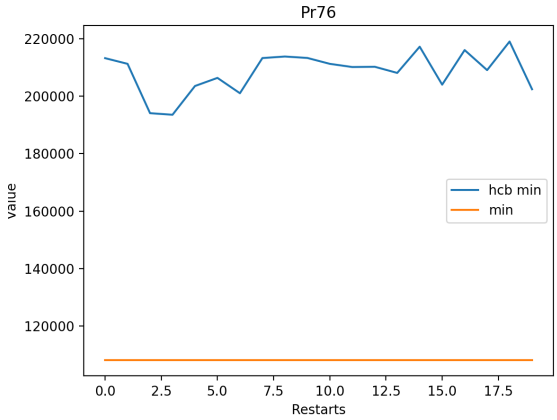
These are the results after running the GA for 30 times, and applying 2opt search for the best solution found by by the GA each time. (population size = 100, pc = 0.7, pm = 0.02, 1000 generations)

5.2 Hill Climbing



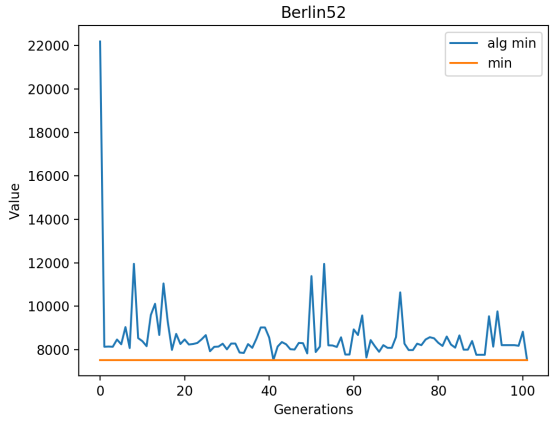


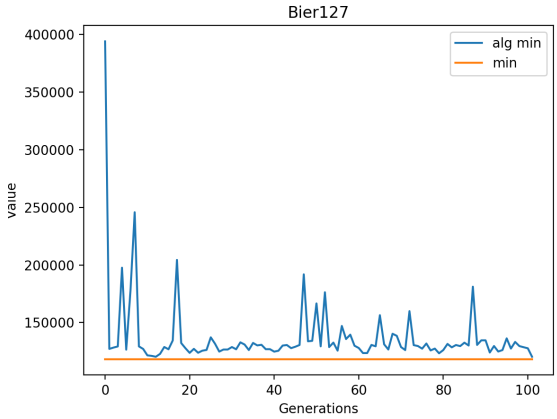


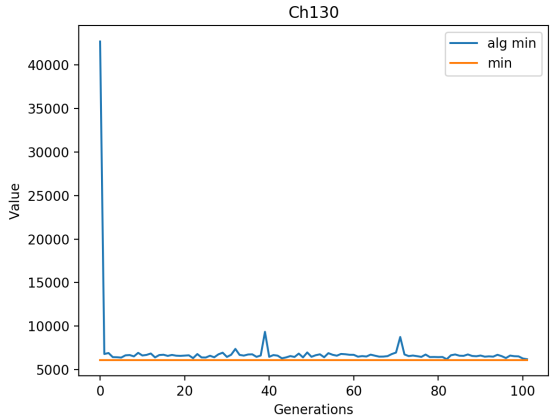

Pr76
220000
200000
180000
160000
140000
120000
vaiue
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Restarts
hcb min
min

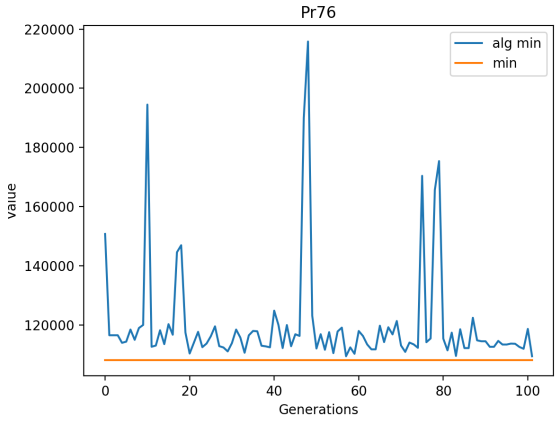
6 Evolution of Generations

I executed the GA, on a population of 100 individuals, over 100 generations. The pm = 0.02 and pc = 0.7. I added a new function that executes 2-otp-search on random spots for 5 times on every member of the population, and a full 2-otp-search on the best member of that generation. This is the evolution of the minimum over the generation.









7 Conclusions

I think that, the Genetic algorithm did better than the Hill Climbing even though the only comparasion is between an execution. Looking at how a execution takes place and the results given back by the GA we can say that it gives us a better solution overall as the Hill Climbing tends to get stuck in a local optimum. In many situations GAs locates the neighborhood of the global optimum extremely efficiently but have problems converging onto the optimum itself, that's why I used an 2-opt search over the minimum, which turned out to do the trick.
Seeing the result of the GA over 30 run times gives us a good view of the consistency of the result.
The times when over an execution the result is way higher than it should be is because of the parameters chosen. A high pc for crossover and even a pm =

0.02 makes the algorithm go a little bit offtrack sometimes.

References

http://www.cleveralgorithms.com/nature-inspired/advanced/visualizingalgorithms.html
https://twitter.com/vroomproject/status/649314769518882817
http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html
http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/
http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html