Computational Intelligence for Optimization

MASTER'S DEGREE PROGRAM IN DATA SCIENCE AND ADVANCED ANALYTICS – MAJOR IN DATA SCIENCE

# Travelling Salesman Problem

Group Luidgi

Luis F. R. Agottani, m20200621

May, 2021

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

# TABLE OF CONTENTS

## 1. INTRODUCTION

The objective of this project is to implement genetic algorithms to solve the Traveling Salesman Problem (TSP) and analyze the different types of configurations to optimize the algorithm for the best fitness. The goal for this problem is minimization where the shortest possible route distance between all the cities from a list fits the best solution.

## 2. INITIATION

The test was carried out on three different lists of cities to understand the behavior of the algorithm in different sizes of value entries and to understand the viability of the model in question of computational consumption, where each list of cities has a different size, Berlin with 52 cities, Uruguay 734 cities and Italy 16862 cities. For the build solution, a random number is generated between 0 and the value of the length of the list of cities minus 1, for example, for Berlin it would be between 0 and 51, this procedure is repeated the same amount as the length of the list, for the example it would be 52 times, thus forming a list of random and not repeated destinations to start the algorithm.

## 3. EVALUATION SOLUTION

The Charles library was used to solve the problem, where some changes were made to record the results for all generations to facilitate the analysis of results for each configuration, was add self.scores and self.generations to the Charles function evolve. To calculate the fitness was used the Euclidean distance between cities coordinates (x,y) in the read_tsp_data.py and then the distance matrix was built.

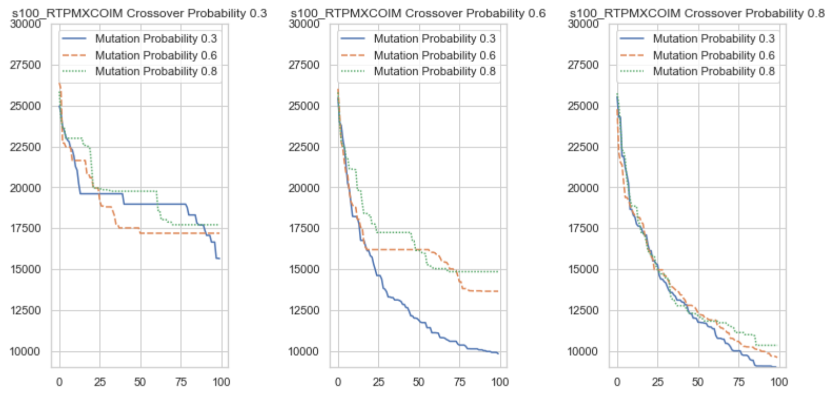
## 4. NEIGHBORHOOD FUNCTION

The neighborhood function was built to use with the initializations, Hill Climb and simulation, the function returns a population of n representations where n is the city list length and then two positions from each representation are changed to find the best fitness for this population, it is a good option to find the local optima before starting the main algorithm.

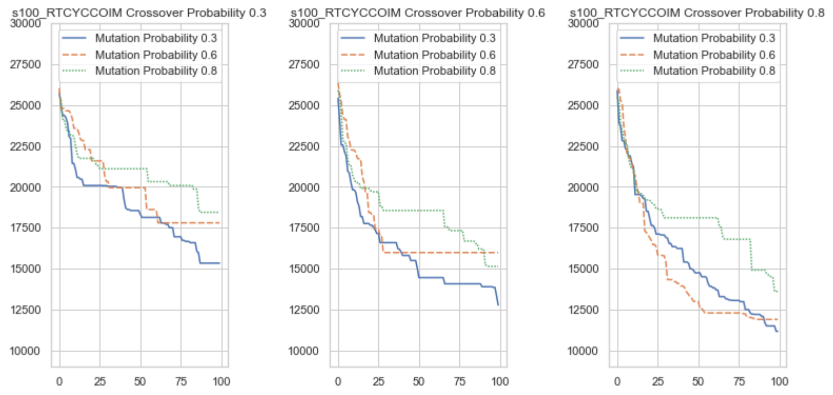
## 5. TEST FUNCITON

The test was made with the best fitness of 10 runs for each configuration of 100 generations each. All the configurations were tested with crossover and mutation probability of 0.3, 0.6 and 0.8. The graphic below shows the fitness for all generations in each configuration and each crossover and mutation probability. The tests were made with the berlin_52 for different configurations and for computational consumption tests the uy734 and it16862 were used to understand algorithms behavior for bigger datasets.

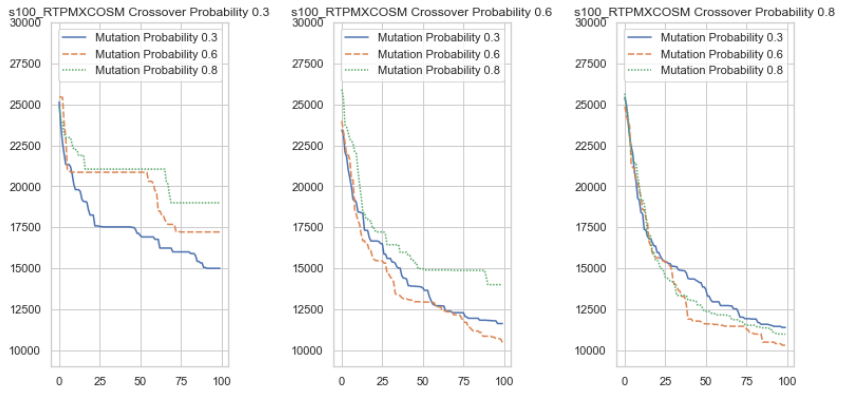
## 5.1. POPULATION SIZE OF 100, RANDOM INITIALIZATION, TOURNAMENT SELECTION, PMX_CO CROSSOVER, INVERSION MUTATION, ELITISM TRUE, BERLIN _52 DATASET



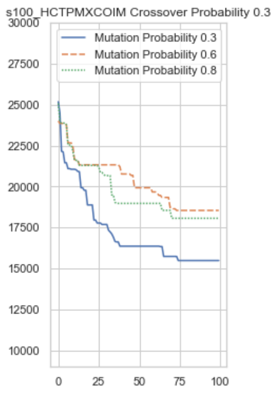
## 5.2. POPULATION SIZE OF 100, RANDOM INITIALIZATION, TOURNAMENT SELECTION, CYCLE CO CROSSOVER, INVERSION MUTATION, ELITISM TRUE, BERLIN _52 DATASET

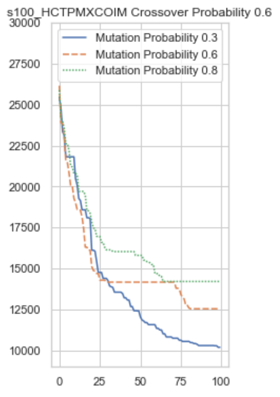


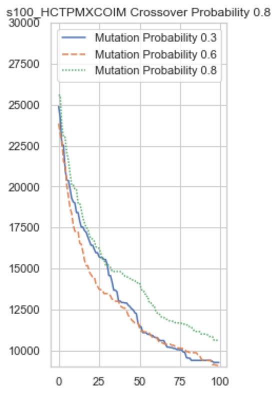
## 5.3. POPULATION SIZE OF 100, RANDOM INITIALIZATION, TOURNAMENT SELECTION, PMX CROSSOVER, SWAP MUTATION, ELITISM TRUE, BERLIN _52 DATASET



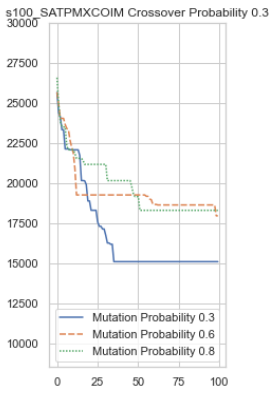
## 5.4. Population size of 100, hill climb initialization, tournament selection, PMX crossover, inversion mutation, elitism true, Berlin _52 dataset

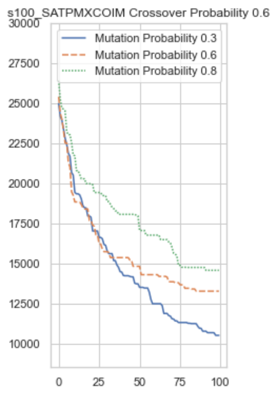


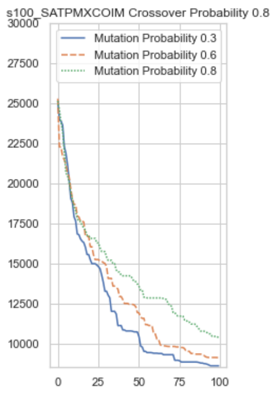




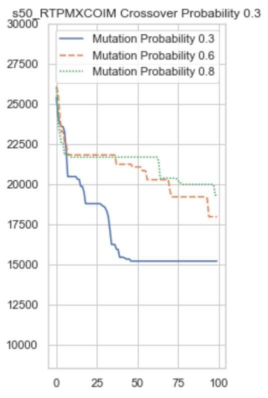
## 5.5. Population size of 100, simulated annealing initialization, tournament selection, PMX crossover, inversion mutation, elitism true, Berlin _52 dataset

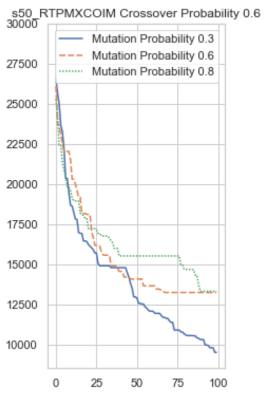


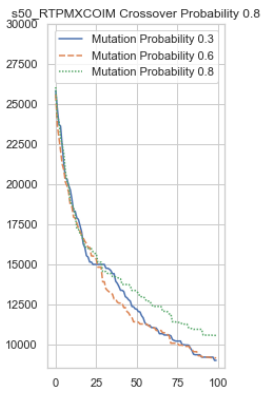




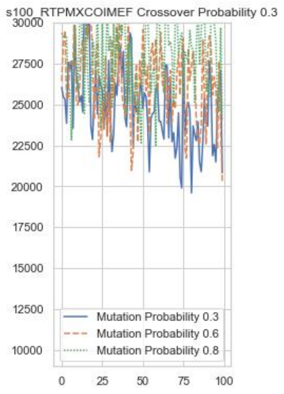
## 5.6. Population size of 50, random initialization, tournament selection, PMX crossover, inversion mutation, elitism true, Berlin _52 dataset

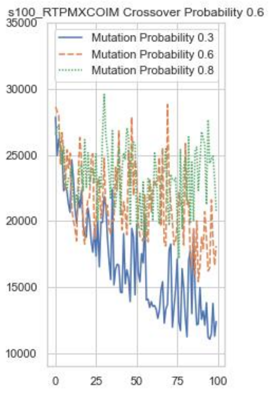


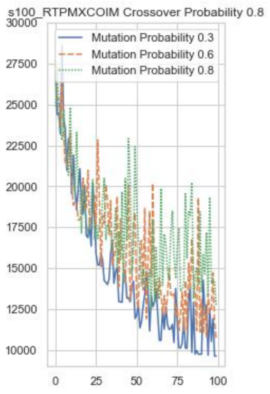




## 5.7. Population size of 50, random initialization, tournament selection, PMX crossover, inversion mutation, elitism FALSE, Berlin _52 dataset



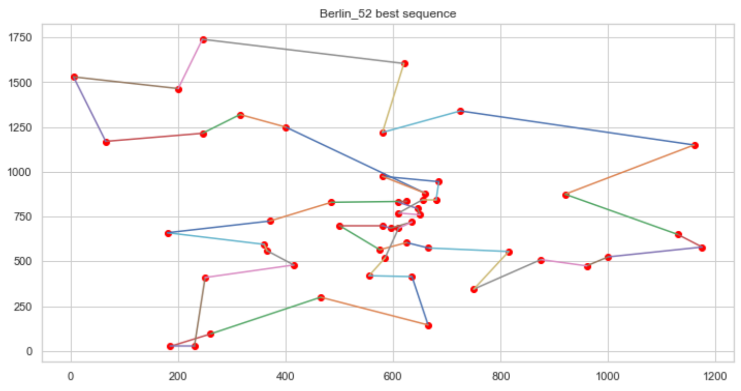




# 6. RESULTS

| Data Set | Population Size | Selection | Initialization | Crossover | CrossOver Probability | Mutation | Mutation Probability | Elitism | Fitness |
|---|---|---|---|---|---|---|---|---|---|
| Berlin_52 | 100 | Tournament | Random | PMX_CO | 0.8 | Inversion Mutation | 0.3 | True | 8719 |
| Berlin_52 | 100 | Tournament | Random | Cyclo_CO | 0.8 | Inversion Mutation | 0.3 | True | 11176 |
| Berlin_52 | 100 | Tournament | Random | PMX_CO | 0.8 | Swap Mutation | 0.6 | True | 10307 |
| Berlin_52 | 100 | Tournament | Hill Climb | PMX_CO | 0.8 | Inversion Mutation | 0.6 | True | 9087 |
| Berlin_52 | 100 | Tournament | Simulated Annealing | PMX_CO | 0.8 | Inversion Mutation | 0.3 | True | 8595 |
| Berlin_52 | 50 | Tournament | Random | PMX_CO | 0.8 | Inversion Mutation | 0.3 | True | 8982 |
| Berlin_52 | 100 | Tournament | Random | PMX_CO | 0.8 | Inversion Mutation | 0.3 | False | 9633 |

Best destination sequence: [27, 26, 25, 46, 13, 12, 51, 10, 11, 50, 32, 42, 9, 8, 7, 40, 18, 2, 44, 31, 48, 0 , 43, 33, 34, 38, 35, 21, 30, 17, 16, 20, 41, 1, 6, 29, 22, 19, 49, 28, 15, 45, 23, 47, 37, 39, 36, 4, 14, 3, 24 , 5]



### 6.1. SIZE

The results indicate a higher chance to get better results increasing the population, but consequently computational cost grows.

### 6.2. INITIALIZATION

The result for the hill climb initialization was not as expected, we had a fitness worse than the initial configuration for this dataset, however the simulated annealing initialization showed an improvement in the result, indicating an increase in the chance of obtaining a better result with a initialization that finds local optimal to start the main algorithm.

### 6.3. CROSS OVER

Cross over probability showed improvement for higher probabilities to do the cross over operation, which indicates that the chance to find a global optimal is higher while manipulating cross over probabilities to higher values. Comparing the operators, pmx and cycle cross over, the pmx_co resulted in better results than cycle_co.

### 6.4. MUTATION

The combination with higher cross over probability and lower mutation probability showed the best results for all configurations. Comparing the operators, inversion and swap mutation, the pmx_co resulted in better results than cycle_co.

### 6.5. ELITISM

The results demonstrated that elitism increase the chance to find better fitness where we can see in the graphics a big variation between the generations, indicating that we lose the best representation while generating news individuals. Elitism is a good choice for genetic algorithms for traveling salesman problem.

## 7. REFERENCES

Data Base link:

http://www.math.uwaterloo.ca/tsp/world/countries.html#DJ

http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html