Project 2

Heuristic Optimization

Professor:

João P. Carvalho

Students (G2) :

Aurora Nora | 93573

João Lopes | 93584

Introduction

In this report we will explain how we developed a solution to find the shortest route that starts in the Central, runs through all the given EcoPoints and ends in Central. The distances between EcoPoints are stored in a matrix in "Project2_DistancesMatrix.xlsx".

For that we used two different approaches, as requested. In the first approach we used Genetic Algorithms and in the second approach we used Ant Colony Optimization. There are two *.py files, each with a different approach.

The input (EcoPoints that require to be emptied) is given inside a *.csv file. There is an example where all EcoPoints (1 to 99) need to be emptied, its name is "p2_input.csv". You need to write the name of input file as an argument, when running the python file.

Genetic Algorithm

For the implementation of the Genetic Algorithm (GA) approach, we started by doing the exercises given in the lab classes. We also followed a YouTube tutorial [1] in order to better understand how the libraries worked and how to adapt the tutorial code to our use case.

In our case, the GA is defined as a "FitnessMin", as we are trying to find the path with lower distance. We called "EcoPoint" to the EcoPoint number registered in the toolbox. An "Individual" is a list of "EcoPoints", containing the list given as input in a random order. A "Population" is the number of "Individuals" per generation.

After defining all this, we chose as a crossover technique the Partially Mapped Crossover.

As "evaluate" function, we created a function called "get_path_distance" that receives an "Individual" and returns the sum of the distance between Central and the first element of the input list, the distance between each element of the list and the distance between the last element and Central. The best result would be the lowest possible.

After some testing, by trial and error, we ended up with the following configuration for our GA:

- Number of generations: 400
- Size of a population: 1000
- Crossover probability: 70%
- Mutation probability: 20%

We noticed that if we continued to increase the number of generations and/or the size of the population, the improvements were despicable, yet the runtime would increase substantially.

The results are the following:

- If all EcoPoints are full (from EcoPoint 1 to 99)
    - Total distance: 47.2 km
    - Runtime: 197.55s (3mins 18s)

[1] https://www.youtube.com/watch?v=kTSZ-QPBVD4

    - Route (Starting and ending in Central):
      ['C', 'E48', 'E66', 'E14', 'E90', 'E69', 'E52', 'E28', 'E18', 'E82', 'E64', 'E44', 'E62', 'E12', 'E17', 'E34', 'E89', 'E21', 'E80', 'E50', 'E36', 'E42', 'E37', 'E95', 'E93', 'E83', 'E94', 'E86', 'E5', 'E45', 'E72', 'E59', 'E67', 'E84', 'E41', 'E76', 'E85', 'E16', 'E91', 'E31', 'E70', 'E38', 'E35', 'E24', 'E8', 'E22', 'E51', 'E88', 'E32', 'E39', 'E4', 'E96', 'E23', 'E33', 'E27', 'E79', 'E77', 'E58', 'E54', 'E57', 'E60', 'E81', 'E55', 'E74', 'E99', 'E97', 'E56', 'E78', 'E13', 'E71', 'E2', 'E3', 'E6', 'E46', 'E7', 'E92', 'E75', 'E40', 'E73', 'E9', 'E53', 'E87', 'E30', 'E1', 'E43', 'E19', 'E26', 'E61', 'E15', 'E63', 'E25', 'E98', 'E68', 'E10', 'E29', 'E47', 'E65', 'E49', 'E11', 'E20', 'C']

- If half of the EcoPoints are full (ex.: from EcoPoint 1 to EcoPoint 50)
    - Total distance: 23.7 km
    - Runtime: 129.51s (2mins 10s)
    - Route (Starting and ending in Central):
    - ['C', 'E20', 'E3', 'E47', 'E46', 'E7', 'E49', 'E19', 'E29', 'E17', 'E38', 'E35', 'E31', 'E50', 'E36', 'E24', 'E8', 'E23', 'E28', 'E27', 'E18', 'E1', 'E9', 'E12', 'E30', 'E45', 'E11', 'E10', 'E48', 'E6', 'E14', 'E44', 'E21', 'E39', 'E32', 'E22', 'E4', 'E13', 'E2', 'E16', 'E42', 'E37', 'E5', 'E40', 'E33', 'E43', 'E34', 'E26', 'E15', 'E25', 'E41', 'C']

- If there are 20 full EcoPoints (average) (ex.: from EcoPoint 1 to 20)
    - Total distance: 10.3 km
    - Runtime: 42.47s (43s)
    - Route (Starting and ending in Central):
      ['C', 'E20', 'E14', 'E17', 'E1', 'E12', 'E9', 'E5', 'E8', 'E4', 'E13', 'E16', 'E2', 'E18', 'E7', 'E11', 'E19', 'E15', 'E10', 'E3', 'E6', 'C']

We had some better results when using a Population of 2000 and 1000 Generations (under 40km), however it would take 20mins or even more to calculate and the difference was about 10km, so we ended up reducing these values until reaching the values referred above. The best result was the following:

- If all EcoPoints are full, Population=2000 and Number of generations=1000
  (from EcoPoint 1 to 99)
    - Total distance: 36.5 km
    - Runtime: 1193.91s (19mins 53s)
    - Route (Starting and ending in Central):
    - ['C', 'E67', 'E98', 'E48', 'E1', 'E60', 'E58', 'E54', 'E12', 'E30', 'E82', 'E55', 'E80', 'E35', 'E5', 'E94', 'E95', 'E86', 'E91', 'E51', 'E88', 'E22', 'E32', 'E39', 'E71', 'E42', 'E93', 'E83', 'E40', 'E87', 'E9', 'E62', 'E44', 'E33', 'E64', 'E18', 'E69', 'E85', 'E81', 'E28', 'E23', 'E96', 'E52', 'E74', 'E13', 'E24', 'E8', 'E31', 'E70', 'E50', 'E36', 'E16', 'E21', 'E34', 'E7', 'E26', 'E61', 'E3', 'E68', 'E65', 'E10', 'E29', 'E41', 'E6', 'E14', 'E90', 'E72', 'E76', 'E79', 'E77', 'E27', 'E53', 'E75', 'E38', 'E37', 'E78', 'E56', 'E97', 'E99', 'E4', 'E2', 'E92', 'E73', 'E57', 'E45', 'E17', 'E89', 'E43', 'E49', 'E11', 'E46', 'E15', 'E47', 'E19', 'E25', 'E66', 'E63', 'E59', 'E20', 'E84', 'C']

Ant Colony Optimization

For the implementation of the Ant colony Optimization Algorithm, we used the python library "ACO-pants". We heard about this library during theoretical classes and, later on we did some research about it. Its documentation is very helpful however there's not much information about it on the internet.

We treated the input list the same way we did in the previous approach however we couldn't find a way to insert the Central as final destination (it was throwing errors), so we proceeded without it.

While creating the "World" of our problem, we used "get_distance" as a function who "calculates" the distance between two points. In this case it just gets the value from the distances' Matrix.

"ACO-pants" has built-in functions that calculate the shortest path but, in order to specify the beginning of our path (Central) we had to create our own "Ant" objects, recurring to "ACO-pants" library. We tried several values as colony size (number of ants) and we ended up choosing 10 ants, as we couldn't get improvements having more than that.

The results were the following:

- If all EcoPoints are full (input has length 99)
    - Total distance: 37.9 km (39 km, with distance to Central manually added)
    - Runtime: 5s
    - Route (Starting and ending in Central):
    - ['C', 'E20', 'E41', 'E66', 'E63', 'E14', 'E6', 'E59', 'E67', 'E10', 'E26', 'E19', 'E48', 'E49', 'E61', 'E3', 'E98', 'E90', 'E84', 'E43', 'E17', 'E73', 'E54', 'E30', 'E1', 'E12', 'E45', 'E60', 'E87', 'E9', 'E58', 'E57', 'E42', 'E81', 'E85', 'E82', 'E64', 'E18', 'E27', 'E69', 'E62', 'E53', 'E75', 'E40', 'E92', 'E21', 'E80', 'E39', 'E51', 'E88', 'E99', 'E56', 'E50', 'E71', 'E78', 'E22', 'E31', 'E16', 'E91', 'E13', 'E97', 'E4', 'E74', 'E70', 'E28', 'E23', 'E96', 'E36', 'E2', 'E8', 'E35', 'E38', 'E37', 'E5', 'E95', 'E93', 'E86', 'E94', 'E83', 'E34', 'E72', 'E89', 'E76', 'E77', 'E33', 'E55', 'E44', 'E52', 'E24', 'E32', 'E79', 'E7', 'E46', 'E47', 'E11', 'E15', 'E68', 'E29', 'E65', 'E25']

- If half of the EcoPoints are full (ex.: from EcoPoint 1 to EcoPoint 50)
    - Total distance: 21.7 km (23.2 km, with distance to Central manually added)
    - Runtime: 1s
    - Route (Starting and ending in Central):
    - ['C', 'E20', 'E41', 'E14', 'E6', 'E29', 'E26', 'E11', 'E47', 'E48', 'E15', 'E3', 'E10', 'E46', 'E7', 'E43', 'E17', 'E12', 'E30', 'E45', 'E1', 'E34', 'E18', 'E35', 'E5', 'E42', 'E37', 'E38', 'E40', 'E8', 'E50', 'E24', 'E13', 'E36', 'E39', 'E22', 'E4', 'E32', 'E31', 'E16', 'E44', 'E21', 'E23', 'E28', 'E2', 'E33', 'E27', 'E9', 'E49', 'E25', 'E19']

- If there are 20 full EcoPoints (average) (ex.: from EcoPoint 1 to 20)
    - Total distance: 11.3 km (12.8 km, with distance to Central manually added)
    - Runtime: < 1s
    - Route (Starting and ending in Central):
    - ['C', 'E20', 'E14', 'E6', 'E3', 'E10', 'E11', 'E15', 'E7', 'E17', 'E1', 'E12', 'E9', 'E5', 'E4', 'E13', 'E16', 'E8', 'E2', 'E18', 'E19']

GA vs ACO

Considering both approaches, the Municipality restrictions (maximum of 20 minutes to decide the route) and that the Municipality uses the same computer that we're using.

As results were almost similar between both approaches, with exception on the route of 100 EcoPoints, we would choose the Ant Colony Optimization approach as it is almost instantaneous to obtain a route, comparing to the Genetic Algorithm approach.