Module 6: Descriptive and Predictive Modeling

Exercise 3: Using Genetic Algorithms to solve a TSP

Preliminary note: this exercise is to be completed in groups of 2 students.

Given a list of cities and the distances between each pair of cities, the travelling salesman problem (TSP) is a combinatorial optimization problem that aims to discover the shortest possible route that visits each city and returns to the origin city. This problem is recurrently addressed with meta-heuristic in the related literature and permits to model many practical scenarios in last-mile logistics and transportation.

Groups are asked to design a genetic algorithm capable of solving any TSP problem efficiently. To this end, we will utilize the set of public TSP problem instances available in the TSPLib repository:

http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/

(please start with small-sized instances, e.g. berlin52)

Students must develop a parser for the unified format in which the above instances are provided, as well as a fitness evaluation function that computes the total distance for any given candidate route. Such a route must be encoded as a permutation of the integer set [1, …, N], where N is the number of cities.

Next, they must investigate public implementations of genetic algorithms published by other parties and develop one of their own. Some useful links follow:

https://towardsdatascience.com/evolution-of-a-salesman-a-complete-genetic-algorithm-tutorial-for-python-6fe5d2b3ca35

https://www.geeksforgeeks.org/traveling-salesman-problem-using-genetic-algorithm/

https://towardsdatascience.com/python-genetic-algorithms-and-the-traveling-salesman-problem-f65542fae5d

The exercise will be evaluated in terms of:

1) The explanation of the evolutionary operators in use (crossover, mutation), and the novelty with respect to other implementations found during the literature search
2) The quality of the produced solutions (e.g. difference with respect to the total distance of the optimal route, which is known: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html)
3) The visual inspection of the routes output by the algorithm
4) (Advanced) The insertion of local search operators
5) (Advanced) Scalability and programming efficiency
6) (Advanced) Comparison to third-party solvers

Reports can be a DOC document, a PDF document (along with the Python scripts that generate the reported figures and results) or a Jupyter Notebook (with saved checkpoint). Other formats (e.g. link to Google Colab) must be agreed with the professor.

When uploading the report, please indicate name, surname and ID (DNI number) of all members of the team.

Delivery deadline: March 7th, 2021