JM0100: Business Analytics
Assignment 1
2017 – 2018

Instructions

This assignment consists of 3 exercises regarding heuristics. The advice is to use Matlab or Python to solve the exercises, when appropriate. We don't have any preference.

Deadline for submission of solutions is **Monday, April 9, 23:59.**

Deliverables.
Submit all solutions electronically through Blackboard. Make sure that you submit everything in a single zip file that contains the solutions to the exercises, described below. The solution for each exercise must consist of the following:

- The pdf file Ex1.X-YYY.pdf that contains your description of the algorithm to the exercise of 1.X (X = 1, 2, …, 5). YYY is your student number. You can use up to ½ page (A4) for each task.
- The excel-file Ex1.X-YYY.xls that contains your answer to the exercise of 1.X (X = 1, 2, …, 5). YYY is your student number.
- The file Ex1.X-YYY.m (in case of using Matlab) or Ex1.X-YYY.py (in case of using Python), containing the corresponding programming code file.

Hence your zip-file consists of in total at most 15 files.
The structure of your excel-file with the result/outcome is as follows (see next page for the details of each exercise):

Route Nr.	City Nr.	City Name	Total Distance in Route (km)	Total Distance (km)

For example (sequence of stores and number of kilometers are arbitrary)

Route Nr.	City Nr.	City Name	Total Distance in Route (km)	Total distance (km)
1	0	EMTE HEADQUARTERS VEGHEL	0	0
1	36	EMTE GOIRLE	9	9
1	81	EMTE OOSTERHOUT NB	15	15
1	…	…	…	…
1	0	EMTE HEADQUARTERS VEGHEL	248	248
2	0	EMTE HEADQUARTERS VEGHEL	0	248
2	13	EMTE CULEMBORG	33	271
2	14	EMTE CUIJK	68	339
2	…	…	…	…
2	0	EMTE HEADQUARTERS VEGHEL	240	448
….	…	…	…	…

In other words: in the excel file you present the details per route and a final column where you present the total accumulated amount of kilometers. As a result, the last line of the excel-file includes the total amount of routes and the total amount of needed kilometers.

Some further remarks:

- Assignments are to be done individually. Plagiarism is not allowed (- all points).
  - Do not submit anything you have not written yourself.
  - Do not submit anything that is not your idea.
- Submit your files as a zip file named Ex1-YYY.zip (if not, -0.5 points).

Good luck!

Introduction Exercise 1

Compute the required amount of account managers and involved traveling distance for EMTE.

EMTE is a supermarket chain in the Netherlands with about 130 stores. EMTE likes to know how many account managers they need to visit all the stores in one day, and how many kilometers they need to travel in total together. Their main objective is to minimize the total traveling distance rather than the amount of account managers. Every account manager starts and ends at the headquarters. Due to the maximum working time of the account manager, every account manager can drive a maximum of 400 km per day. In other words: the task is to solve a VRP for this problem. The goal is to minimize the total distance. You can use the *haversine* formula in Matlab or Python to compute the distance between two locations.

In the attached excel-file (called **Data Excercise 1 - EMTE stores - BA 2018.xls**) you can find all the address details and coordinates (latitude, longitude) for every store and the headquarters in Veghel.

Exercise 1.1 (3 points)

Describe an ILP-formulation for the VRP-problem. Try to solve the problem on the entire case. For this you can use the MIP solver (function intlinprog in Matlab or install the Python package PuLP on your laptop, see https://pythonhosted.org/PuLP/index.html for more information) to solve the formulated ILP problem. When the solver cannot find the optimal solution within time limit (default value: 7200 seconds), it will return the best-found solution, with performance gap.

The data is given in the file **Data Excercise 1 - EMTE stores - BA 2018.xls**, in which the starting location (the depot = the headquarters) has location 0.

Exercise 1.2 (1 point)

Write a construction heuristic to create a starting solution for the VRP. To do so, you can use the Nearest Neighbor Heuristic, Insertion Heuristic, Savings Heuristic or any other heuristic, which will be explained during the lectures.

Exercise 1.3 (2 points)

Write a local search improvement heuristic to improve the solution you have created in Exercise 1.2. If you do not have a solution in exercise 1.2, just start with an arbitrary sequence of the stores.

The following local search heuristic needs to be implemented:

- City-swap
- 2-edge exchange
- Tabu-search

For the tabu-search you can define your neighborhood based on the city-swap and the 2-edge exchange moves. You are free to choose your own tabu- and stopping criterium, as well as the length of your tabu-list. Notice that you have to write your own tabu-search algorithm. You are **not** allowed to call a library-function.

Exercise 1.4 (1 point)

Improve your heuristic algorithm (result from exercise 1.3) by including at least 2 library-functions. You can choose for the following library functions:

- Simulated Annealing (see https://nl.mathworks.com/help/gads/examples.html#simulated-annealing when using Matlab or https://github.com/perrygeo/simanneal when using Python)
- Genetic Algorithm (see https://nl.mathworks.com/help/gads/examples/custom-data-type-optimization-using-the-genetic-algorithm.html, or DEAP: Distributed Evolutionary Algorithms in Python)
- Ant Colony Optimization (see http://nl.mathworks.com/matlabcentral/fileexchange/52859-ant-colony-optimization--aco- for more information when using MATLAB, or https://pypi.python.org/pypi/ACO-Pants when using Python)

Exercise 1.5 (3 points)

We extend the problem in the following way: suppose that the average driving speed of the account manager is 80 km / hour, that every visit takes 1 hour, and that every account manager works at most 10 hours a day. Every store is open between 09 and 17h. Find a solution for this problem, where the first objective is to minimize the amount of account managers, and the second objective is to minimize the amount of kilometers. You can choose whether you create an ILP-formulation or a heuristic algorithm for this problem.

Our grading rules

The grading rules are the following for each exercise:

- When your algorithms are correct, and your excel-sheet has the correct answer (or a very good answer in case of a heuristic) for a certain exercise, you receive 50% of the points.
- When your program (in Matlab or Python) has the right structure, you also receive (the other) 50% of the points for that exercise. The right structure means:
    - The program is easily readable and includes sufficient functions and procedures for the various steps.
    - The program contains logic names for the variables used and contains sufficient comments to explain the steps.