JM0100: Business Analytics
Assignment 2 (2019 – 2020)

Instructions

This assignment consists of 3 exercises focusing on heuristics. The objective is to use Python to solve these exercises. Deadline for submission of solutions is **Friday, July 10, 6pm, by e-mailing** your deliverables to **E.J.Roos@tilburguniversity.edu and g.kant@tilburguniversity.edu.** Notice that the final grade counts, so if you submit resit deliverables, then the grade for the resit overrules your current grade for Assignment 2.

Deliverables

Make sure that you submit *everything* in one single zip file that contains the solutions to the exercises, described below. The solution for each exercise must consist of the following:

- The excel-file Ex2.X-YYY.xls that contains your answer to the exercise of 2.X (X = 1, 2, 3). YYY is your student number.
- The file Ex2.X-YYY.py (or ipynb, a Jupyter Notebook), containing the programming code file.

Hence your zip-file consists of in total at most 6 files. You can include the code of all 3 exercises in one Jupyter Notebook, we have no preference. The structure of your excel-file with the result/outcome is as follows (see next page for the details of each exercise). This layout is very important, since we will check your answer by automatic processing!

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	…	…	…	…
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 number 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. Wrong layout leads to subtraction of points. Attached is an example of the layout of an output-file (with only 2 routes) to help you.

Some further remarks:

- The assignment must be executed individually. Plagiarism is not allowed (- all points). We will do some proper checking on plagiarism and might lead to exclusion from this course for this year.
- Do not submit anything you have not written yourself, or which is not your idea!
- Submit your files as a zip file named Ex2-YYY.zip (if not, subtraction of points).

Introduction Resit Assignment 2

Compute the required number of kilometers for account manager John of EMTE Supermarkets.

EMTE was a supermarket chain in the Netherlands with 133 stores, owned by Sligro Food Group. In 2018, Sligro has sold EMTE, with 79 stores moving to Jumbo Supermarkets, 51 stores moving to Coop and 3 supermarkets to Other chains ([see this link]() for the press release). Sligro has given the task to its account manager John to visit all 133 stores, to discuss the transition. Some aspects related to this:

- John starts and ends every day at the headquarters (HQ) in Veghel.
- Average driving speed for John is 90 km/h. Visiting time is 30 minutes when this EMTE-store moves to Jumbo Supermarkets and 20 minutes for the others (= stores who moves to Coop or Other chain).
- John works at most 11 hours a day.
- Every store can be visited from 09:00h until 17:00h (then the visit should be finished).

EMTE likes to know how many kilometers (and how many days) John needs in total to visit all the stores.

So, the problem is to solve a VRP for this problem. The goal is to minimize the total distance. To compute the distance between two locations in Python, you have to use the *haversine* formula. For computing the corresponding travel time, you have to use the average speed of 90 km/h. After calling the *haversine* function for a pair of locations, you must round the distance to kilometers and the driving time to minutes.

The data can be found in **Data Excercise 2 - EMTE stores - BA 2020.xls** (including details and coordinates (latitude, longitude) for every store and the headquarters in Veghel). This excel-sheet is equal to the one in Assignment 2 (see attachment under Assignment 2).

Exercise 2.1 (10 points)

Write a construction heuristic to create a starting solution for the VRP based on the Nearest Neighbor Method. This means the following:

1. Start a new route *R* with the store, say *x*, which is unplanned and nearest to the HQ in Veghel.
2. Select the nearest store from *x*, say *y*, which can be added after *x* in a feasible way to route *R*. Set *x* = *y* and repeat step 2, until it is not possible anymore to add a store y in a feasible way.
3. If there are still unplanned stores, go back to step 1, otherwise stop.

A feasible route means that its duration is at most 11 hours, and the total time between the arrival time at the first store and the departure time at the last store in the route is at most 8 hours (to respect the opening times between 09:00 – 17:00h).

Exercise 2.2 (10 points)

Write the following Tabu-search improvement heuristic to improve the solution you have created in Exercise 2.1.

1. The neighborhood structure is defined by a **Swap**, which means selecting randomly two arbitrary stores (which can belong to the same route, or to different routes), and swapping them. In other words, if you select store s_1 and s_2, then you try to put store s_2 at the position of store s_1 in the current plan and vice versa. Of course, the new solution must be remain feasible.
2. For the tabu-list you create a list of length 50, where you store the last 50 swaps you have executed.
3. If swapping s_1 and s_2 gives an improvement, you apply the swap, otherwise you apply the swap only if swapping s_1 and s_2 is not tabu (is not in the tabu-list of the last 50 swaps).
4. Stopping criterium: 100 steps without any improvement, or when exceeding a total time limit of 1000 seconds. Maintain and return the best solution found.

Exercise 2.3 (10 points)

Improve your constructed solution (the result from Exercise 2.1) by applying a Simulated Annealing approach, quite similar to the method in Exercise 2.2:

1. The neighborhood structure is defined by a **Swap**, which means selecting randomly two arbitrary stores (which can belong to the same route, or to different routes), and swapping them. In other words, if you select store s_1 and s_2, then you try to put store s_2 at the position of store s_1 in the current plan and vice versa. Of course, the new solution must be remain feasible.
2. For the Simulated Annealing, the starting temperature T is 1000, the cooling down parameter is 0.999.
3. Let *Delta* be the change in kilometers after swapping s_1 and s_2. If *Delta* is negative (there is an improvement), you apply the swap, otherwise you apply the swap with a probability of $e^{-Delta/T}$.
4. Stopping criterium: if T < 0.001, or when exceeding a total time limit of 1000 seconds. Maintain and return the best solution found.

Our grading rules

The grading rules are the following:

- We should be able to run your Python-code, so do not refer to local input/output files in your code.
- When your Python code runs but does not lead to strong results (in your excel-sheet) points will be subtracted (5 points if the answer is missing).
- When your Python program does not have the right structure, points will be subtracted. The right structure means:
  - The program is easily readable and includes enough functions and procedures for the various steps.
  - The program contains logic names for the variables used (e.g., use a constant `MaxTimeLimit = 1000`) and contains enough comments to explain the steps.
  - The program includes comments within the procedures and code (so, not only outside the program, for which you use the comment structure in Jupyter, but also inside the program).