Anticipatory balancing of a RoboTaxi systems – Read Me File

1. First step is to split the service area into separate zones and create the demand files.
   1. First, we use the read\_json2.py file :

The file take as an input the file 'Manhattan.json' which contains all the bus stations in Manhattan and creates a list of tuples – all the stations that are 3 minutes or less from one to another.

* 1. The 'station.mod' solves an optimization model and select all the stations we will use for the model.
  2. The filter\_input\_new.py filter only the trips that started and ended inside the Manhattan polygon.
  3. The file demands.py takes as an input the dataset name(output of the file from step 1.c) and list of the stations location (output of the optimization file from step 1.b) and create 3 pickle files – time factor table, demands, and Stations and two csv files – dist mat and time mat.

The demands and time factor are 3D matrix in size of (num\_of\_stationXnum\_of\_stationsX24).

The station file creates a list of objects of type station, where each object has a few attributes – location, id, locations ( a list of real locations from the dataset that are in the zone of the station, top10min – a list of all the stations that are less than 10 minutes' drive from the station.

The dist mat is a matrix that holds the distance in meters from each station to each station, and the time mat is the same but keeps the time between stations in seconds.

1. The optimization files are - rebalance\_TR.mod, rebalance\_NE.mod, and rebalance\_pavone.mod. those files solves the different optimization models. The TR file is for the anticipatory model, the NE file is for the anticipatory neighborhood model, and pavone file is for the myopic model.

* All the files' outputs from 1.d and 2 has to be in the same folder as the simulation in order for the simulation to work.

1. **In order to run the simulation :** 
   1. First, we need to open the server\_car.bat (by double-clicking it) in order to open a server for the OSRM.
   2. You need oplrun on your computer and IBM Cplex in order to run the simulation.
   3. If you don’t have python on your path and the packages numpy and pandas you can use 'Anaconda' – first open the program, then move on the left menu bar to 'Environments', press right on 'base' and then 'open terminal'.

In the new window move to the files that contain all the simulation files by the command 'cd [folder location]'. Next, open the file 'set\_path'.

* 1. Now, when all the environment is ready we can run the simulation.

**Simulation input :**

1. Data set name (in our simulation is - train.sorted)
2. Number of cars
3. Patince of the customer (in seconds) – what is the maximum time we can pick a customer up before we reject him.
4. How many days we want to run the simulation (30 in our test)
5. Beta – the weight parameter in the objective function
6. Planning horizon (relevant only to TR and NE models, but need a value for the other models as well).
7. Period length in seconds (in our test it was 900 – 15 minutes)
8. Cplex running time limit (we set it to 30 seconds)
   * + **In order to simulate the JIT model we set this parameter to 0.**
9. Demand factor (we set to 8)
10. Model type:

TR – Anticipatory model

Ne – Anticipatory neighborhood model

P- myopic model

Example :

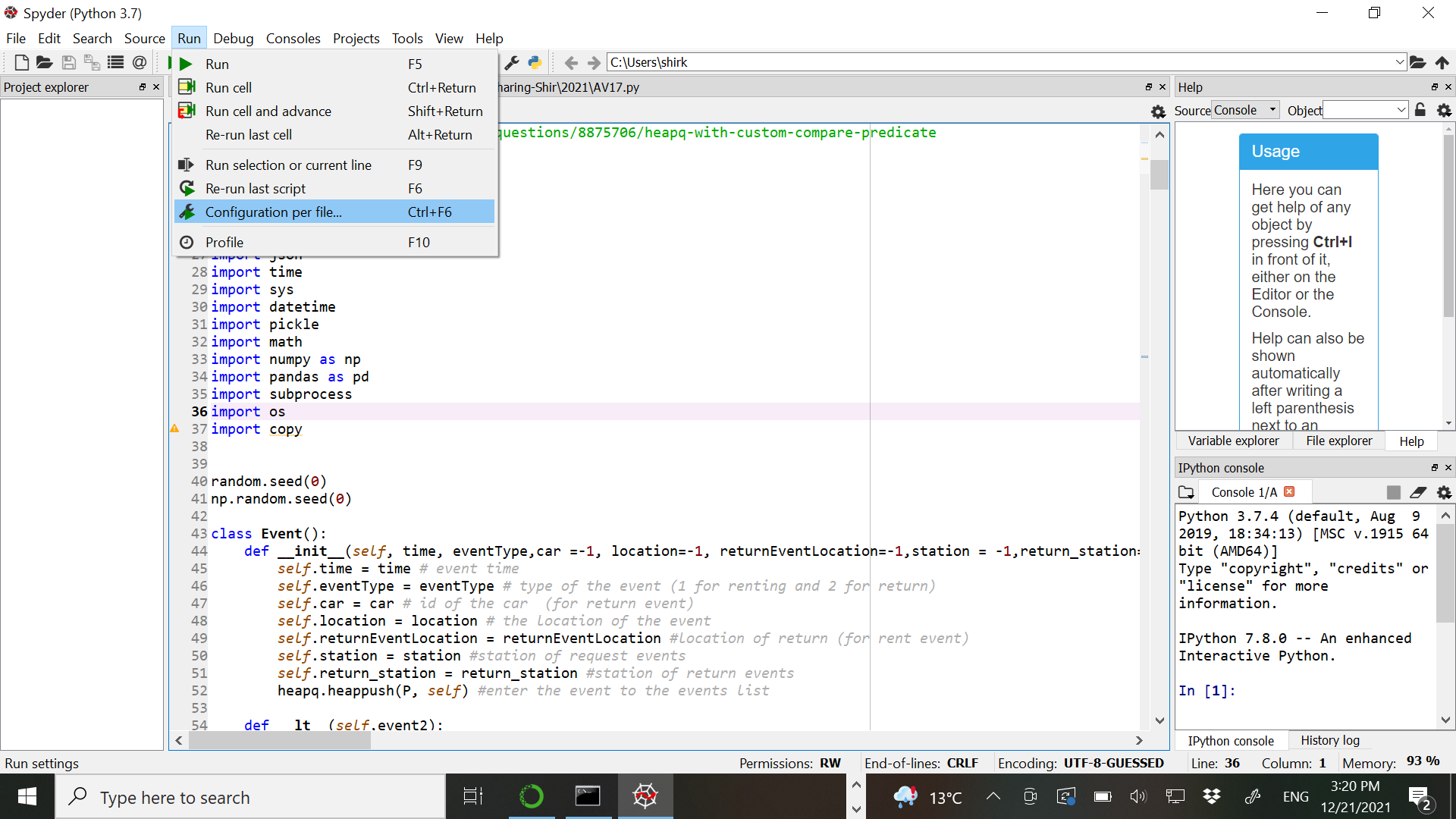
train.sorted 1080 720 30 4200 6 900 30 8 NE

These parameters runs the model Anticipatory neighborhood model with 1,080 cars, 12 minutes(720 seconds) of passenger patience, 30 days of simulation days, beta = 4,200, planning horizon = 6 periods, each period is 15 minutes (900 seconds), cplex running time limit of 30 seconds, dimand factor of 8.

There are two ways to run the simulation:

1. One single run from a python program (such as spyder/ pycharm etc.)
2. We first open the simulation file in the profram (AV17.py)
3. We set the command line parameters :

In the 'Run' bar we select 'Configuration per file'



Then, we check the box 'command line parameters' and then enter the parameters. תמונה שמכילה טקסט, צילום מסך, מחשב

התיאור נוצר באופן אוטומטי

1. Second way is to run by batch file (we can run as many simulations as we want)

The batch file contain in each row the line:

python AV17.py train.sorted 1080 720 30 4200 6 900 30 8 NE

first we write the word 'python', then the simulation file name, and then all the parameters. The bat file looks:

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

In order to run the second way we run the bat file from the terminal we opened via Anaconda.

The simulation file writes the results into csv file called ' ResultsByDays-AV17.csv' in resolution of one day.