Documentation of Optimization Tool

May 11, 2020

This is the documentation of our optimization tool. You can gain the following information after reading through the documentation.

1. Introduction of the optimization tool
2. Instruction for running from command line
3. Instruction for running in a Jupyter notebook

1 Introduction of the optimization tool:

Our optimization tool `optimize.py` contains a function called `optimize` with two input parameters:

- `inputeFile`: the path to the input file. (An example of an input file is "Sample_Input.xlsx".)
- `outputFile`: the path to the output file to be created by the function. (An example of an output file is "Sample_Output.xlsx")

1.1 Input data format

The input data should consist of ten sheets. These sheets include information about the classrooms within JKP, the timeslots available, the department's hourly needs, and each department's timeslot preference. Each of these input Excel sheets will be discussed below:

`classroom - Pr`: Describes the available graduate classrooms within JKP.

classroom	size	Big
102	52	0
104	56	0
110	77	1
112	77	1
202	54	0
204	54	0
210	78	1
212	78	1

- The first column is called `classroom`, which contains the identifying classroom number.
- The second column is called `size`, which contains the capacity of the classroom.
- The third column is called `Big`, which countains a binary identifier that signals whether the classroom is considered "Big" (1), meaning the capacity is greater than or equal to 77, or "Small" (0), meaning the classroom capacity is less than 77.

`hours - ht`: Describes the time-slots across the different weekdays.

- The column index is weekday. In the column index, weekdays are displayed numerically – Monday through Friday corresponds to the numbers 1 to 5.
- The row index is timeslot. It consists of eight timeslots from 8:00 a.m. to 21:30 p.m.
- In each cell, it provides the amount of teaching hours within each of the eight timeslots for each weekday.

	1	2	3	4	5
800-930	1.5	1.5	1.5	1.5	1.5
930-1100	1.5	1.5	1.5	1.5	1.5
1100-1230	1.5	1.5	1.5	1.5	1.5
1230-1400	1.5	1.5	1.5	1.5	1.5
1400-1530	1.5	1.5	1.5	1.5	1.5
1530-1700	1.5	1.5	1.5	1.5	1.5
1700-1830	1.5	1.5	1.5	1.5	1.5
1830-2130	3	3	3	3	3

`md`: Describes the 7 graduate departments and the total hours of teaching that each will provide in that semester.

- The first column is called `department`, which contains the abbreviation of the seven graduate departments.
- The second column is called `hours to teach in a semester`, which contains the total teaching hours of each department.

department	hours to teach in a semester
BUCO	31.5
DSO	42
FBE	36
ACCT	49.5
MOR	21
BAEP	31.5
MKT	31.5

`MKT`, `BUCO`, `...`, `BAEP`: These seven sheets represent the seven graduate department's individual preference scores for specific timeslots across the five weekdays.

- These sheets are all formatted the same, as shown in the following graph, with each cell corresponding to the proportion(between 0 to 1) of teachers within each department that want that time-slot for that particular weekday.

weekday	1	2	3	4	5
timeslot					
800-930	0.5	0.5	0.5	0.5	0.5
930-1100	0.5	0.5	0.5	0.5	0.5
1100-1230	0.5	0.5	0.5	0.5	0.5
1230-1400	0.5	0.5	0.5	0.5	0.5
1400-1530	0.5	0.5	0.5	0.5	0.5
1530-1700	0.5	0.5	0.5	0.5	0.5
1700-1830	0.5	0.5	0.5	0.5	0.5
1830-2130	0.5	0.5	0.5	0.5	0.5

1.2 Output data format

The output data should consist of one sheet, formated as the following figure.

Weekday	Timeslot	102	104	110	112	202	204	210	212
Mon & Wed	800-930	DSO	ACCT	MKT	ACCT	ACCT	BAEP	FBE	MKT
	930-1100	DSO	DSO	ACCT	BUCO	ACCT	DSO	MOR	MKT
	1100-1230	DSO	MKT	MKT	MOR	DSO	DSO	FBE	MKT
	1230-1400	DSO	DSO	DSO	BAEP	BAEP	BAEP	DSO	MKT
	1400-1530	FBE	DSO	FBE	BUCO	BAEP	ACCT	MOR	FBE
	1530-1700	BAEP	MKT	MKT	ACCT	BAEP	FBE	MKT	MKT
	1700-1830	BAEP	ACCT	MOR	MKT	BAEP	MOR	MKT	BAEP
Mon	1830-2130	MKT	ACCT	BUCO	BUCO	MOR	MOR	BUCO	BUCO
Wed	1830-2130	DSO	MOR	MOR	BUCO	MOR	BUCO	MOR	MKT
Tue & Thur	800-930	BAEP	BAEP	DSO	MKT	BAEP	FBE	MKT	MKT
	930-1100	BUCO	DSO	FBE	FBE	ACCT	FBE	FBE	MOR
	1100-1230	BAEP	ACCT	BUCO	FBE	ACCT	DSO	DSO	MOR
	1230-1400	DSO	ACCT	MKT	BUCO	MOR	FBE	MKT	FBE
	1400-1530	BUCO	BAEP	FBE	BUCO	BAEP	DSO	MOR	MOR
	1530-1700	MOR	ACCT	BUCO	FBE	DSO	FBE	FBE	ACCT
	1700-1830	BAEP	BAEP	MOR	MOR	BAEP	FBE	MOR	BAEP
Tue	1830-2130	ACCT	BUCO	BUCO	MOR	MOR	ACCT	ACCT	ACCT
Thur	1830-2130	MOR	ACCT	BUCO	DSO	DSO	BUCO	BUCO	MOR
Fri	800-930	FBE	DSO	MKT	BUCO	BUCO	BUCO	MKT	MOR
	930-1100	ACCT	ACCT	MOR	MKT	BUCO	BAEP	MKT	FBE
	1100-1230	BUCO	DSO	MOR	BUCO	ACCT	ACCT	BUCO	MOR
	1230-1400	MOR	FBE	ACCT	ACCT	BUCO	BUCO	FBE	DSO
	1400-1530	BUCO	BAEP	ACCT	DSO	BUCO	BAEP	MOR	MKT
	1530-1700	ACCT	BUCO	FBE	DSO	MKT	BUCO	BAEP	MKT
	1700-1830	ACCT	MOR	MKT	BUCO	ACCT	BUCO	FBE	MOR
	1830-2130	ACCT	BUCO	FBE	BUCO	ACCT	BUCO	BUCO	BAEP

The column denotes the classrooms and double-index denotes the weekdays and the time slots. Noted that on Wednesday and Thursday we only show the 18:30-21:30 time slot since all other time slots are the same as Monday's and Tuesday's and thus redundant. As you can see for each timeslot on each day, a particular department is assigned.

2 Instructions for running from command line

You can use the following command to run our optimization tool from command (in Anaconda prompt in Windows and in a Terminal in Max) .

```
python optimize.py inputFile outputFile
```

where "inputFile" should be replaced by the path to the input file, and "outputFile" should be replaced by the path to the output file. (For example, inputFile might be "Sample_Input.xlsx" and outputFile might be "Sample_Output.xlsx".)

3 Instructions for running in a Jupyter notebook

You can follow the three steps below to successfully run our optimization function in a Jupyter notebook.

Step 1:

You should first import the necessary modules and functions using the following three commands by copy pasting them into a code cell and running them:

```
import numpy as np
```

```
import pandas as pd
```

```
from gurobipy import Model, GRB
```

Step 2:

To import the optimization function, you can copy the function part from `optimize.py`, to a code cell and run it.

The function part starts from

```
def optimize(inputFile,outputFile):
```

and ends with

```
writer.save().
```

Step 3:

Finally, you can run the following command in a new code cell:

```
optimize('Sample_Input.xlsx', 'Sample_Output.xlsx')
```

You should make sure the inputFile is in the same directory where your Jupyter notebook is. The code will produce the desired output of an optimized schedule and export it to an Excel file located in the same directory with the output name just as you defined.