Week 12: Reusable Software

Session 25: Creating Reusable Software for Multi-Product Supply Chain Planning

Problem Description

Nia is a data analyst at Trojan E-commerce, a medium sized online retailer with 17 fulfilment centers scattered across the US. She would like to apply optimization to minimize the weekly outbound shipping cost from fulfilment centers to customers. Trojan uses UPS 2-day delivery. Based on a regression analysis, she found that the cost of shipping each unit of item k from fulfillment center (FC) i to demand region j is $1.38w_k\delta_{ij}$, where w_k is the shipping weight of the item and δ_{ij} is the distance from FC i to region j.

While Trojan E-commerce sells hundreds of thousands of items, Nia conducted a clustering analysis to simplify the analysis and found 100 representative items, and she scaled up the demand for these so that they can serve as a proxy for all the items. Nia has also partitioned the US into 98 demand regions, and has estimated the weekly demand from each demand region for each of the representative items.

Trojan is committed to satisfying all customer demand for all of the items at all demand regions. The weekly demand for item k in region j is given as d_{jk}. However, Trojan can choose how much of this to provide from each FC. Using a closer FC would reduce the shipping cost, but capacity at each FC is limited. The company replenishes inventory every week. At any given FC, the amount of capacity required for processing each unit of item k is equal to s_k. The total capacity of FC i is given as q_i.

Write a function called `optimizeShipment` with two input parameters:

- `inputFile`: filename of the input file. The format is as in the `12-retail-toy-input.xlsx` and `12-retail-real-input.xlsx` files attached on Blackboard.
- `outputFile`: filename of the output file. The desired format is as in the `12-retail-toy-output.xlsx` and `12-retail-real-output.xlsx` files attached on Blackboard.

The function should be able to take in any input file of the same format, and create the corresponding output file.

Description of Data

- **12-retail-real-input.xlsx**: the data Nia prepared for her analysis. The excel workbook has five worksheets:
  - Fulfilment Centers: the set of FCs, as well as capacity q_i for each FC i.
  - Regions: the set of demand regions.
  - Distances: the distance δ_{ij} from each FC i to each region j. Each row represents a region and each column a FC.
  - Items: the set of items, as well as the shipping weight w_k and storage size s_k for each item k.
  - Demand: the demand d_{jk} at each region j for each item k. Each row represents an item and each column a region.
- **12-retail-toy-input.xlsx**: a toy dataset of the same format as the above, for development purposes.
- **12-retail-toy-output.xlsx**: the correct optimization output using the inputs from "12-retail-toy-input.xlsx."

- **12-retail-real-output.xlsx**: the correct optimization output using the inputs from "12-retail-real-input.xlsx."

In-Class Exercise: Understanding the Input and Output Data

Take a look at the above data files, and answer the following questions. All numerical answers can be rounded to two decimal places.

a) Based on "12-retail-toy-input.xlsx", what is the unit shipping cost $1.38 w_k \delta_{ij}$ for item $k = 0$, shipping from FC $i = A$ to region $j = 2$? $1.38 \times 3 \times 1.0475 = 4.23005$

b) Based on "12-retail-toy-input.xlsx", suppose that a certain FC ships a total of 5 units of item 0 and 3 units of item 1, what is the amout of capacity needed at the FC? $5 \times 1 + 3 \times 2 = 11$

c) In "12-retail-toy-input.xlsx", what is the demand of item 0 in region 2? 350

d) In the shipment plan in "12-retail-toy-output.xlsx", how much of item 0 is shipped from FC A to region 1? 100

e) In the shipment plan in "12-retail-toy-output.xlsx", how much of item 0 is shipped from FC A to region 2? 0

f) Based on "12-retail-toy-input.xlsx", what is the total cost of the following shipment plan:

FC	region	item	shipment	
A	0	0	150	x
A	1	1	100	x
B	2	0	200	x

$3 \times 1.5516 = 698.22$
$1 \times 1.4388 = 143.88$
$3 \times 0.43547 = 261.282$
1522.66716

In-Class Exercise: English Description and Concrete Formulation

On a piece of paper, jot down a succinct summary of the English description as well as fragments of the concrete formulation. You don't have to hand in anything. The purpose is to help you develop a clear idea of how to formulate the abstract formulation later. Think of this as doing scrap work on an exam.

w_k : weight of item k, δ_{ij} : dist between FC i & region j
d_{jk} : demand for item k in region j.
q_i : capacity of FC i
s_k : space for item k
x_{ijk} : quantity of item i to be sent from FC i to region j

Data:
I: fulfillment centers.
J: demand regions.
K: list of items.

Objective: minimize transportation cost.

$$1.38 \sum_{i \in I, j \in J, k \in K} (x_{ijk}) \times w_k \delta_{ij}$$

Constraints:

(capacity) $\left(\sum_{k \in K, i \in J} x_{ijk} \times s_k\right) \leq q_i$ for each FC i

Outbound delivery should be $\leq$ capacity of FC

(demand) $\sum_{i \in I} x_{ijk} \geq d_{jk}$ for each demand center j and item k

Shipment of each item into region j should be $\geq$ demand.

(non-negativity): $x_{ijk} \geq 0$ for each i, j, k

Exercise 12.5 Abstract Formulation for Multi-Product Supply Chain Planning

Complete the Jupyter notebook attached to the Blackboard link for this exercise and submit it there after completing it. The notebook asks you to write the abstract formulation for the above problem. To help you get started, the relevant data variables are already provided.

Abstract Formulation

Data:

- I: the set of FCs.
- J: the set of regions.
- K: the set of items.
- q_i: the capacity of FC i.
- δ_{ij}: the distance from FC i to region j.
- w_k: the shipping weight of item k.
- s_k: the storage size of item k.
- d_{jk}: the demand for item k in region j.

Decision Variables:

Objective and Constraints:

first add constraints to code & then add data frames & test.

Tips:

- Don't do outputflag false
- Use read, write to print model
- Code incrementally
- Code constraints first and then read data.
- Use structure of concrete for abstract.

```
[2]: # One useful Syntax for DataFrames: transposing
     import pandas as pd
     df=pd.DataFrame([[1,2,3],[4,5,6]],index=['A','B'],columns=['i','ii','iii'])
     df
```

```
   i  ii  iii
A  1   2    3
B  4   5    6
```

```
[3]: df.T
```

```
     A  B
i    1  4
ii   2  5
iii  3  6
```

Exercise 12.6: Reusable Software for Multi-Product Supply Chain Planning

Create a new Jupyter notebook and write in it your function `optimizeShipment`, following the instructions in the problem description. Make sure you put all of your final code in one cell with the comment `# Final Code` such that if the Kernel is restarted and only that cell is run, the code will work.

As seen in the test code below, it would be a good idea if the function prints something to let you know it has finished. You can do something similar to what is suggested in the sample outputs, but it is not mandatory.

Due to the size of the `12-retail-real-input.xlsx` file, the code may take a minute to run, and that is normal. Due to a numerical precision issue, in order to obtain the exact optimal solution for the larger test case, you need to include the following code before running `mod.optimize()`:

```
mod.setParam('MIPGap',1e-6)
```

```
[4]: # Final Code
```

```
[7]: # Test code
     optimizeShipment('12-retail-toy-input.xlsx','12-retail-toy-myOutput.xlsx')
```

```
Finished optimizing! Objective value: 3400.76919
```

```
[6]: # Test code (might take a minute to run)
     optimizeShipment('12-retail-real-input.xlsx','12-retail-real-myOutput.xlsx')
```

```
Finished optimizing! Objective value: 9841232.842925586
```