Supply Chain - Linear Optimization & Abstract Formulation

Author: Gihyeon Kwon

Questions and data derived from *DSO 570 – The Analytics Edge: Data, Models, and Effective Decisions*

- This project consists of 3 different supply chain planning problems

In [1]:
```
import pandas as pd
import os

os.chdir("12-data-files")
```

1) Warehouse Planning: Concrete Formulation & Optimization

The following table provides the shipping cost for one-pound packages, from 7 of a company's fulfillment centers (FC) to 4 regions.

Region	1	2	3	4	5	6	7
A. Kings County, NY	20.25	7.70	24.59	23.26	7.69	7.70	7.69
B. Los Angeles County, CA	18.43	23.30	7.69	7.69	24.16	22.12	24.91
C. King County, WA	21.28	24.18	7.70	17.67	23.91	22.98	24.57
D. Harris County, TX	7.69	7.70	18.73	7.71	18.79	7.70	19.47

A shipping cost of $10 or less indicates that the package will be transported via ground shipping; otherwise, it will be transported via air shipping.

For a certain item that weights a pound, the company would like to stock it in as few FCs as possible, while guaranteeing that it can fulfill demand in all 4 regions via ground shipping. Moreover,

- the item must be stocked in at least one of FCs 5 or 7;
- the item cannot be stocked in FC 4 unless it is also stocked in FC 1;
- if the item is stocked in FC 2, then it cannot also be stocked in FC 3.

Goal: Formulate a linear optimization problem to find the minimum number of FCs needed.

English Description

Decision: Whether or not use utilize each FC

Objective: Minimize number of FC's needed

Constraints:

- Each FC fulfills demand for all 4 regions in ground shipping
- At least one of FCs 5 or 7
- Cannot be in 4 unless in 1
- If in 2, cannot be in 3

Concrete formulation

Decision Variables: Let $X_1, \cdots, X_7$ denote whether to use each FC. (Binary)

Objective and constraints:

$$\begin{aligned} \text{Minimize} \quad & X_1 + X_2 + \cdots + X_7 \\ \text{s.t.} \quad & X_2 + X_5 + X_6 + X_7 \geq 1 \\ & X_3 + X_4 \geq 1 \\ & X_3 \geq 1 \\ & X_1 + X_2 + X_4 + X_6 \geq 1 \\ & X_5 + X_7 \geq 1 \\ & X_4 \leq X_1 \\ & X_2 + X_3 \leq 1 \end{aligned}$$

Code implementation

In [2]:
```
import pandas as pd
from gurobipy import Model, GRB
mod = Model()

variables = range(1,8)
X = mod.addVars(variables, vtype=GRB.BINARY, name='X')
mod.setObjective(sum(X[i] for i in variables), sense=GRB.MINIMIZE)

constraints = [[2,5,6,7], [3,4], [3], [1,2,4,6], [5,7]]
for c in constraints:
    mod.addConstr(sum(X[i]for i in c) >= 1)

mod.addConstr(X[4] <= X[1])

constraint_2_3 = [2,3]
mod.addConstr(sum(X[c] for c in constraint_2_3) <= 1)

mod.write('10-2.lp')
%cat 10-2.lp
mod.setParam('OutputFlag', False)
mod.optimize()

print()
print(f'Minimum # of FCs needed: {int(mod.objval)}')
print('Stock item in the following:')
for i in variables:
    if X[i].x == 1:
        print(f'\tFC{i}')
```

```
Set parameter Username
Academic license - for non-commercial use only - expires 2025-10-15
\ LP format - for model browsing. Use MPS format to capture full model detail.
Minimize
  X[1] + X[2] + X[3] + X[4] + X[5] + X[6] + X[7]
Subject To
 R0: X[2] + X[5] + X[6] + X[7] >= 1
 R1: X[3] + X[4] >= 1
 R2: X[3] >= 1
 R3: X[1] + X[2] + X[4] + X[6] >= 1
 R4: X[5] + X[7] >= 1
 R5: - X[1] + X[4] <= 0
 R6: X[2] + X[3] <= 1
Bounds
Binaries
 X[1] X[2] X[3] X[4] X[5] X[6] X[7]
End

Minimum # of FCs needed: 3
Stock item in the following:
        FC1
        FC3
        FC7
```

Summary of Warehouse Planning Problem

- **Objective:** Minimize the number of fulfillment centers (FCs) while ensuring all regions are served via ground shipping.
- **Constraints considered:** Geographic restrictions, minimum FCs required, and mutual exclusivity conditions.
- **Optimization Insight:** The model identifies the fewest FCs needed to meet demand while minimizing shipping costs.

2) Supply Chain Planning: Abstract Formulation

Company produces and sells a certain product. There is a certain set of production plants, each with a given capacity, which is the maximum number of units that can be produced there in a given month. There is a certain set of demand regions, each with a given estimated demand, which is the maximum number of units you can sell in that region per month. It is possible to not fulfill all the demand, as that corresponds to having the item being stocked out for some customers. The price that you charge when sell the item may be different in each region, but the prices are determined by another department, so is outside of your control.

As an illustration, suppose there are 3 plants, with the following production costs and capacities:

Plant	1	2	3
Cost per unit	60	60	55
Capacity	35	40	35

Suppose there are 4 regions, with the following selling prices and demand estimates:

Region	A	B	C	D
Price	80	85	70	75
Demand	30	50	10	20

The following table provides the transportation cost of shipping each unit from each plant to each region.

Region \ Plant	1	2	3
A	20	8	25
B	18	23	8
C	21	24	8
D	8	8	19

Goal: To create an abstract formulation of a linear optimization problem to determine a plan for producing and shipping items, so as to maximize total profit, which is the revenue from the selling the product minus the production and transportation costs. Formulation must be flexible enough to handle arbitrary sets of production plants and regions, and none of the numbers in the above description can be hardcoded.

(In this problem, fractional units are allowed, since we are working with rates.)

Abstract Formulation

Data:

- I: Representing each plant
- c_i: Cost per unit for each plant i
- k_i: Capacity for each plant i
- R: Representing each region
- p_r: price of unit in each region r
- d_r: Demands of each region r
- t_ir: Transportation cost for each unit form i to r

Decision Variables:

- X_i: How many units to produce in each plant i (continous)
- Y_{ir}: How many units to transport to each region r from each plant i

Objective:

$$\text{Maxmize: } \sum_{i \in I, r \in R} p_r Y_{ir} - \sum_{i \in I} c_i X_i - \sum_{i \in I, r \in R} t_{ir} Y_{ir}$$

Constraints:

$$\begin{array}{rll} \text{s.t.} & & \\ \text{(Capacity)} & X_i \le k_i & \text{for each } i \in I \\ & \sum_{r \in R} Y_{ir} \le X_i & \text{for each } i \in I \\ \text{(Demand)} & \sum_{i \in I} Y_{ir} \le d_r & \text{for each } r \in R \\ \text{(Non-Negativity)} & X_i \ge 0 & \text{for each } i \in I \\ & Y_{ir} \ge 0 & \text{for each } i \in I \text{ and } r \in R \end{array}$$

Summary of Supply Chain Planning: Abstract Formulation

- **Objective:** Develop an abstract supply chain optimization model to **maximize total profit**, considering production costs, transportation costs, and regional demand.
- **Constraints considered:** Plant capacity limits, production-to-demand allocation, and transportation costs.
- **Optimization Insight:** The model **determines the optimal production and shipping strategy** by balancing **cost minimization and profit maximization**, ensuring an **efficient supply chain planning framework** adaptable to various production scenarios.

3) Multi-Product Supply Chain Planning: Abstract Formulation & Reusable Optimization Software

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 $1.38 w_k \delta_{ij}$, 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. (For simplicity, assume that fractional amounts of items is allowed, so the amount shipped does not have to be an integer.) 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-sampleOutput.xlsx` and `12-retail-real-sampleOutput.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-sampleOutput.xlsx**: the correct optimization output using the inputs from "12-retail-toy-input.xlsx."
- **12-retail-real-sampleOutput.xlsx**: the correct optimization output using the inputs from "12-retail-real-input.xlsx."

3a) Abstract Formulation for Multi-Product Supply Chain Planning

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: X_{ijk} : How much to ship from FC i to region j of item k

Objective and Constraints:

$$\begin{array}{rl} \text{Minimize:} & \sum_{i \in I, j \in J, k \in K} 1.38 X_{ijk} w_k \delta_{ij} \\ \text{s.t.} & \\ \text{(Capacity)} & \sum_{j \in J, k \in K} X_{ijk} s_k \le q_i \text{ for each FC } i \in I \\ \text{(Demand)} & \sum_{i \in I} X_{ijk} \ge d_{jk} \text{ for each item } k \in K \text{ and region } j \in J \\ \text{(Non-Negativity)} & X_{ijk} \ge 0 \text{ for all } i \in I, j \in J, k \in K \end{array}$$

3b) Reusable Software for Multi-Product Supply Chain Planning

Goal: Create the 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. **At the end of your function, you should print a line to report that the function is finished, and print the optimal objective value, as in the sample outputs below.**

In [3]:
```
#Data
input_file='12-retail-toy-input.xlsx'
Regions = pd.read_excel(input_file, sheet_name='Regions', index_col=0)
FC = pd.read_excel(input_file, sheet_name='Fulfilment Centers', index_col=0)
Items = pd.read_excel(input_file, sheet_name='Items', index_col=0)
Demand =  pd.read_excel(input_file, sheet_name='Demand', index_col=0)
Distances = pd.read_excel(input_file, sheet_name='Distances', index_col=0)

I = FC.index
J = Regions.index
K = Items.index
q = FC['capacity']
delta = Distances
w = Items['shipping_weight']
s =Items['storage_size']
d = Demand
```

In [4]:
```
def optimizeShipment(input_file, output_file):
    import pandas as pd
    from gurobipy import Model, GRB

    Regions = pd.read_excel(input_file, sheet_name='Regions', index_col=0)
    FC = pd.read_excel(input_file, sheet_name='Fulfilment Centers', index_col=0)
    Items = pd.read_excel(input_file, sheet_name='Items', index_col=0)
    Demand =  pd.read_excel(input_file, sheet_name='Demand', index_col=0)
    Distances = pd.read_excel(input_file, sheet_name='Distances', index_col=0)

    I = FC.index
    J = Regions.index
    K = Items.index
    q = FC['capacity']
    delta = Distances
    w = Items['shipping_weight']
    s =Items['storage_size']
    d = Demand


    mod = Model()
    X = mod.addVars(I, J, K)

    mod.setObjective(sum(1.38 * X[i,j,k] * w[k] * delta[i][j] for i in I for j in J for k in K))
    for i in I:
        mod.addConstr(sum(X[i,j,k] * s[k] for j in J for k in K) <= q[i])

    for k in K:
        for j in J:
            mod.addConstr(sum(X[i,j,k] for i in I) >= d[j][k])

    mod.setParam('OutputFlag', False)
    mod.optimize()

    print(f"Finished optimizing! Objective value: {mod.objval}")


    writer = pd.ExcelWriter(output_file)

    summary = pd.DataFrame([mod.objVal], columns = ['Minimum Total Cost'])
    summary.to_excel(writer, sheet_name='Summary', index=False)

    data= []
    for i, j , k in X:
        if X[i,j,k].x > 0:
            data.append([i, j, k, X[i,j,k].x])

    result = pd.DataFrame(data, columns = ['FC_name', 'region_id', 'item_ID', 'Shipment'])
    result.to_excel(writer, sheet_name = 'Solution', index=False)
    writer.close()
```

In [5]:
```
# Test code 1
output_file='12-retail-toy-myOutput.xlsx'
if os.path.exists(output_file):
    os.remove(output_file)
optimizeShipment('12-retail-toy-input.xlsx',output_file)
display(pd.read_excel(output_file,sheet_name='Summary'))
display(pd.read_excel(output_file,sheet_name='Solution'))
```

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

	Minimum Total Cost
0	3400.76919

	FC_name	region_id	item_ID	Shipment
0	A	0	1	125
1	A	1	0	100
2	A	1	1	200
3	A	2	1	100
4	B	0	0	500
5	B	0	1	75
6	B	2	0	350

In [6]:
```
# Test code 2
output_file='12-retail-real-myOutput.xlsx'
if os.path.exists(output_file):
    os.remove(output_file)
optimizeShipment('12-retail-real-input.xlsx',output_file)
display(pd.read_excel(output_file,sheet_name='Summary'))
display(pd.read_excel(output_file,sheet_name='Solution'))
```

```
Finished optimizing! Objective value: 9841229.288170155
```

	Minimum Total Cost
0	9.841229e+06

```
      FC_name  region_id  item_ID  Shipment
0        SAT1         10        0    2524.0
1        SAT1         10        1    2485.0
2        SAT1         10        3    2300.0
3        SAT1         10        4    3480.0
4        SAT1         10        5    6208.0
...       ...        ...      ...       ...
7870     MSP1         95       95      71.0
7871     MSP1         95       96      43.0
7872     MSP1         95       97      97.0
7873     MSP1         95       98      28.0
7874     MSP1         95       99      38.0

7875 rows × 4 columns
```

Summary of Multi-Product Supply Chain Planning: Abstract Formulation & Reusable Optimization Software

- **Objective:** Develop a scalable supply chain optimization model to **minimize weekly outbound shipping costs** while ensuring all customer demand is met.
- **Constraints considered:** Fulfillment center (FC) capacity, demand fulfillment, and transportation cost variations based on shipping weight and distance.
- **Optimization Insight:** The model **allocates shipments strategically across fulfillment centers**, reducing costs while maintaining efficiency. A **reusable software implementation** enables seamless application across different datasets and business scenarios.