ISyE 6669 Project - Team 7 - Part C - Fall 2020

Connor Owen, Matthew Mendez & Peter Williams

date: 2020-11-19

Part C (30 pts)

Question 8 (15 pts)

Often businesses have to take into account considerations beyond just the cost. In our case, due to trade regulations, our warehouses and customer orders have been assigned to one of four different regions. These assignments are given in WarehouseRegions.csv and OrderRegions.csv. Orders from one region should be fulfilled by warehouses from the same region until the supplies are depleted. After that, they can be fulfilled from any region. Write the model for the updated problem. Explain any additional parameters, variables, and constraints you had to introduce.

To start we have products, warehouses and orders with the addition of notation for regions. Thisis required as we now need to factor in the region assignments due to trade regulations:

$$k = 1, ..., K \text{ Products,} \tag{1}$$
$$j = 1, ..., N \text{ Warehouses, and} \tag{2}$$
$$i = 1, ..., H \text{ Orders} \tag{3}$$
$$r = 1, ..., R \text{ Regions} \tag{4}$$

We also define,

$$d_{ik} \text{ , the demand for product } k \text{ in order } i, \tag{5}$$
$$s_{jk} \text{ , stock of product } k \text{ in warehouse } j, \tag{6}$$
$$c_{ij} \text{ , cost of sending 1 lbs. of product from warehouse } j \text{ to customer } i, \tag{7}$$
$$w_k, (k = 1, ..., K) \text{ denotes the weight of product k in lbs.} \tag{8}$$
$$f_{ij} \text{ , the fixed cost of sending a product from warehouse } j \text{ to customer } i. \tag{9}$$

The inclusion of the regions also means we need to account for stock of product in each region as well:

$$s_{kr} \text{ stock of product } k \text{ in region } r \tag{10}$$

We also create sets that describe which region a warehouse belongs to as well as the order region,

$$N_r \text{ the set of warehouses in region } r \tag{11}$$
$$N_{r*} \text{ the set of warehouses not in region } r \tag{12}$$
$$H_r \text{ the set of orders in region } r \tag{13}$$

The formulation of the decision variables becomes,

$$x_{ijk}, \text{ number of units shipped of product } k \text{ from warehouse } j \text{ to customer } i, \text{ and} \tag{14}$$
$$\delta^*_{ik} = \text{ the number of units of product } k \text{ not fulfilled in order } i \tag{15}$$
$$t_{rk} \in \{0,1\} \text{ is an indication of if there is stock-out of product } k \text{ from region } r, 1, \text{ if stock-out, 0 o.w} \tag{16}$$
$$z_{rk} = \text{ regional flow of product } k \text{ in region } r \tag{17}$$
$$z^{'}_{rk} = \text{ nonregional flow of product } k \text{ in region } r \tag{18}$$

Note that the variable t allows us to factor when a region has stock out. The z and $z^{'}$ variables help us track when a stock is staying within a region vs when it is making its way to destinations beyond the region.

Our goal is to minimize the cost with this additional complexity resulting in the following objective function:

$$\min z = \sum_{i=1}^{H}\sum_{j=1}^{N}\sum_{k=1}^{K} x_{ijk}c_{ij}w_k + \sum_{i=1}^{H}\sum_{j=1}^{N} f_{ij}y_{ij} + M\sum_{i=1}^{H}\sum_{k=1}^{K}\delta^*_{ik}. \tag{19}$$

subject to,

$$\sum_{j=1}^{N} x_{ijk} + \delta^*_{ik} = d_{ik}, \forall i \in H, k \in K, \tag{20}$$
$$\sum_{i=1}^{H} x_{ijk} \le s_{jk}, \forall j \in N, k \in K \tag{21}$$
$$z_{rk} = \sum_{j=1}^{N_r}\sum_{i=1}^{H_r} x_{ijk} \forall r \forall k \tag{22}$$
$$z^{'}_{rk} = \sum_{j=1}^{N_{r*}}\sum_{i=1}^{H_r} x_{ijk} \forall r \tag{23}$$
$$My_{ij} \ge \sum_{k} x_{ijk} \forall j \in N, \forall i \in H \tag{24}$$
$$1 - t_{rk} \le s_{kr} - z_{rk} \forall k, \forall r \tag{25}$$
$$s_{kr}t_{rk} \le z_{rk} \forall k, \forall r \tag{26}$$
$$Mt_{rk} \ge z^{'}_{rk} \forall k, \forall r \tag{27}$$

and,

$$M > 0, \text{ arbitrary large positive number (i.e. 'Big M').} \tag{28}$$

$$x_{ijk}, d_{ik}, s_{jk} \in \mathbb{N} \tag{29}$$

$$x_{ijk} \geq 0 \forall i \in H, j \in N, k \in K \tag{30}$$

$$\delta^*_{ik} \geq 0 \ \forall i \in H, k \in K \tag{31}$$

Several constraints that are new to this model require additional explanation. Constraint (22) accounts for the regional flow, defined as product sent from a warehouse in region r to an order in region r, while the non regional flow is product sent to an order in region r from a warehouse outside of region r is accounted for in constraint (23).

Constraint (25) ensures that when there is a regional stock out for region r and product k, that t_{rk} is assigned the value 1, while constraints (26) ensures that t_{rk} is 0 when there is not a regional stock out for region r and product k. Finally, constraint (27) ensures that non-regional flow for region r and product k is 0 until there is a stockout for that product in that region.

Question 9 (10 pts)

Implement your new model in Xpress or Gurobi/Python. In your submission, this script should be named ModelC.mos or ModelC.py. As in Question 2, make sure you explain any new lines of code you have added.

The addition of the regions results in multiple new lines of code in ModelC.py:

- We must now read in the necessary data from WarehouseRegions.csv and Order Regions.csv into dataframes named *warehouse_ regions_ df* and *order_ regions_ df*. The Warehouse regions *.csv* includes the data for each warehouse j and their respective region r. The order regions matches and order i to a region r.

- With the addition of the new dataframes we then have to turn them into dictionaries. The first of which is the warehouse regions using the command *warehouse_ regions = {warehouse_ regions_ df.at[i, 'Warehouse ID']: warehouse_ regions_ df.at[i, 'Region'] for i in warehouse_ regions_ df.index}*.

- An important part of adding the regions is tracking the amount of stock of a product ion a specific region. We can accomplish this by using a command to track the amount of product k in region r. The command is as follow: *region_ stock = {r: {k: sum([warehouse_ stock[w][k] for w in warehouses if warehouse_ regions[w]==r]) for k in products} for r in regions}*.

- Within the variables we must now factor in the regional demand and consider the maximum demands. This will ne used in the non regional flow constraint. To set this regional demand we used the command *regional_ demand = {r: {k: sum(orders_ data[o][k] for o in orders if k in orders_ data[o]) for k in products} for r in regions}*.

- There are a number of variables needed to fully build out the necessary infrastructure for the inclusion of the regions. Firstly we introduce the region indices with r for regions and k for products with *region_ indices = [(r, k) for r in regions for k in products]* . We then create a Binary variable for whether or not a region has stock out to keep track of when there are transregional orders; *region_ stock_ out = m.addVars(region_ indices, vtype=GRB.BINARY)*. To keep track of the intra regional flow we must include a regional flow variable using *regional_flow = m.addVars(region_ indices, lb=0, vtype=GRB.CONTINUOUS)*. And in conjunction with that we use the line of code, *nonregional_flow = m.addVars(region_ indices, lb=0, vtype=GRB.CONTINUOUS)*, to track the non regional flow.

- Now for the lines of code added to account for the new constraints. In defining the regional flow we use the sum of a product sent to destinations within a region meaning the destination is in the same region as the warehouse. This is done by identifying when the "r" region is the same for both warehouse and the order and is implemented with the following command: *m.addConstr(regional_flow[(r,k)] == sum(flow[(w, o, k)] for w in warehouses if warehouse_regions[w]==r for o in orders if k in orders_data[o] and order_regions[o]==r))* A similar command was used to define the nonregional flow as when the *r* region was not identical between the warehouse and the order using the difference of *if warehouse_regions[w] != r* in the code.

- With the stock out variable being binary we needed constraints to activate and deactivate. This was done by making sure that when the region stock was not great enough to satisfy the regional flow, the stock out would be triggered. In our code this is reflected as *m.addConstr(region_stock[r][k] - regional_flow[(r,k)] >= 1 - region_stock_out[(r,k)])* and *m.addConstr(region_stock[r][k]*region_stock_out[(r, k)] <= regional_flow[(r,k)])*. In addition to this, a constraint to guarantee that nonregional flow is only allowed when the stock out is active is as follows: *m.addConstr(regional_demand[r][k]*region_stock_out[(r,k)] >= nonregional_flow[(r,k)])*.
- The final additions to the previous code include solution print commands for the region summary table and the regional statistics such as regional flow, stock out, and non-regional flow. Additionally warehouse region and order region are added into the outputs with adding *Warehouse_Region* and *Order_Region*.

Question 10 (5 pts)

Solve your model. What is the objective value of your solution? What does it mean in words? How does it compare to the solution of Question 8? What is the optimal solution? Which orders are satisfied from which warehouse? What quantities of different items have been sent?

With the inclusion of the complexity of regional areas our model provides the following solution. In this topline solution we provide the realized value from our objective function, as well as the total order costs, which are the sum of total fixed costs + unit costs. Our realized objective function value can also be understood as the product of *Unfulfilled Demand Quantity* × *M* + *Total Order Cost*:

Table 1: Model C: Topline Results (Rounded)

Model Result	Value
Objective Function Value	173,416.66
Total Order Cost	13,416.66
Total Fixed Cost	4,324.00
Total Unit Costs	9,092.66
Penalty	M = 10,000
Unfulfilled Demand Quantity	16

With the results from our model we can quickly break-down the stock, stock-out, and regional and non-regional flows summarized below:

Table 2: Model C: Regional Summary

Region	Product	Regional Stock	Region Stock-Out	Regional Flow	Non-Regional Flow
1	1	0	1	0	13
1	2	0	1	0	9
1	3	0	1	0	16
1	4	0	1	0	8
1	5	0	1	0	9
2	1	13	0	9	0
2	2	10	1	10	15
2	3	15	0	10	0
2	4	15	0	14	0
2	5	14	1	14	4
3	1	6	1	6	14
3	2	5	1	5	11
3	3	10	1	10	6
3	4	10	0	6	0
3	5	7	1	7	13
4	1	35	0	12	0
4	2	44	0	7	0
4	3	37	0	20	0
4	4	37	0	20	0
4	5	39	0	13	0

Similar to our solution in part B, we can also retrieve the unfulfilled demand by product and order as shown below. The total unfulfilled demand was 16:

Table 3: Model C: Unfulfilled Demand Summary

Order	Product	Unfulfilled Demand
9	5	2
12	5	2
17	5	1
26	1	3
28	3	5
30	1	2
43	5	1

We can also get insights on the leftover supply in each warehouse for each product as shown below:

Table 4: Model C: Leftover Stock Results

Warehouse	Product	Leftover Supply
1	4	1
2	4	2
3	4	4
6	2	1
7	4	7
8	2	1

Finally we provide the complete solution to the problem, which is shown in the appendix following. This solution contains for each warehouse and region, the order, its region along with the quantity and associated unit costs to fulfill.

Appendix

The complete solution to problem C is provided below in three parts in order to fit in this document in a readable way:

Table 5: Full Solution Problem C: Minimizing Order Fulfillment Costs (1/3)

Warehouse	Warehouse Region	Order	Order Region	Product	Quantity	Unit Costs
1	2	9	1	1	4	54.02
1	2	9	1	3	1	27.01
1	2	25	2	3	3	224.13
1	2	25	2	5	2	112.07
1	2	26	2	4	5	190.92
1	2	26	2	5	3	343.65
1	2	30	2	1	1	16.04
1	2	30	2	2	2	64.15
1	2	31	3	3	1	45.1
1	2	39	2	2	3	121.16
1	2	39	2	5	4	121.16
2	3	10	3	2	1	27.27
2	3	10	3	3	8	218.16
2	3	10	3	4	1	6.82
2	3	21	3	1	2	65.47
2	3	21	3	4	1	16.37
2	3	23	3	3	1	24.33
2	3	23	3	5	4	72.99
2	3	31	3	1	1	62.88
2	3	31	3	3	1	125.76
2	3	31	3	5	3	282.97
2	3	36	3	2	2	274.88
2	3	36	3	4	4	137.44
2	3	40	3	1	3	114.96
2	3	40	3	2	2	153.27
2	3	42	1	4	2	20.94
3	4	1	3	2	5	240.36
3	4	3	2	2	3	64.68
3	4	12	1	1	3	183.05
3	4	17	4	1	1	37.5
3	4	17	4	4	3	56.25
3	4	17	4	5	5	281.27
3	4	18	3	1	3	68.98
3	4	18	3	5	1	34.49
3	4	22	4	3	1	77.94
3	4	32	4	1	2	10.74
3	4	32	4	3	3	32.23
3	4	37	3	3	2	29.98
3	4	37	3	5	4	44.97

Table 6: Full Solution Problem C: Minimizing Order Fulfillment Costs (2/3)

Warehouse	Warehouse Region	Order	Order Region	Product	Quantity	Unit Costs
4	2	2	2	3	4	72.13
4	2	2	2	4	1	4.51
4	2	3	2	3	2	144.1
4	2	3	2	5	2	108.07
4	2	7	2	2	5	133.24
4	2	19	2	4	3	91.21
4	2	27	2	1	4	3.96
4	2	27	2	5	3	4.45
4	2	29	2	4	4	91.51
4	2	35	2	1	3	100.06
4	2	41	1	3	3	52.97
4	2	45	2	1	1	32.5
4	2	45	2	3	1	65
4	2	45	2	4	1	16.25
5	4	5	1	3	1	54.33
5	4	5	1	5	2	81.49
5	4	24	1	1	2	7.81
5	4	31	3	1	2	113.23
5	4	31	3	3	1	113.23
5	4	36	3	1	5	39.38
5	4	36	3	2	2	31.51
5	4	38	4	2	4	24.74
5	4	38	4	4	5	7.73
5	4	43	4	4	2	36.77
5	4	43	4	5	3	165.46
5	4	44	1	2	4	115.52
5	4	44	1	3	4	115.52
5	4	44	1	5	2	43.32
6	4	4	4	1	2	51.02
6	4	4	4	3	2	102.03
6	4	14	4	1	3	88.96
6	4	14	4	4	4	59.31
6	4	15	3	3	1	5.43
6	4	17	4	3	4	62.29
6	4	17	4	5	1	11.68
6	4	25	2	2	2	26.45
6	4	30	2	2	2	120.6
6	4	34	3	2	3	1.28
6	4	34	3	3	1	0.43
6	4	34	3	5	4	1.28
6	4	43	4	2	2	102.61
6	4	43	4	3	2	102.61
6	4	43	4	4	2	25.65
7	4	8	3	2	1	20.93
7	4	8	3	5	1	15.69
7	4	11	2	2	4	266.79
7	4	12	1	5	1	41.32
7	4	15	3	5	3	30.78
7	4	20	4	1	4	99.51
7	4	20	4	3	3	149.27
7	4	24	1	2	5	265.56
7	4	41	1	1	1	17.88
7	4	41	1	3	4	143.06
7	4	41	1	5	4	107.29

Table 7: Full Solution Problem C: Minimizing Order Fulfillment Costs (3/3)

Warehouse	Warehouse Region	Order	Order Region	Product	Quantity	Unit Costs
8	4	6	2	5	3	2.39
8	4	13	4	5	4	96.15
8	4	16	4	2	1	89.07
8	4	16	4	4	4	89.07
8	4	22	4	3	5	368.89
8	4	23	3	1	4	1.27
8	4	29	2	5	1	3.11
8	4	33	1	3	3	101.26
8	4	33	1	4	6	50.63
8	4	41	1	1	3	173.14
8	4	45	2	2	4	241.6