Foundation of Decision Modelling
Project - Assigning Regions to Sales Representatives at Pfizer

Gaël De Léséleuc Alexandre Duval Sai Deepesh Pokala

December 16, 2019

In this report, we give an overview of the problem that we attempted to solve and how we approached it.

1 The Problem

Pfizer Turkey is a major player in the Turkish pharmaceutical market. The VP of Logistics at Pfizer thinks that it is about time to renew the territories of Sales Representatives (SRs) country-wide. It is important to keep the total travel distance short and it is not a good idea to make drastic changes in the existing territories. Also, we need to note that maintaining a somewhat balanced workload among the SRs is important.
To solve this assignment problem, we must take a look at the bigger picture of what is at stake here. Therefore, we must first look at the data available, the stakeholders involved, the constraints and challenges that we have.

2 The Data

The country is broken down into several territories (22) and each SR is assigned a group of territories called **bricks**. Each brick is associated with a workload value V_i that is also available to us. Furthermore, each SR has exactly one brick that is his office. He will be located here and will have to travel to the rest of his bricks for work purposes. Therefore, the distance between his office i and brick j D_{ij} and more generally between any two bricks i and j, denoted B_{ij} are key aspects to consider. These values are also provided.

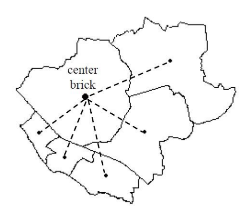

Figure 1: Territory of an SR

Figure 1 above is an example of territory for one SR. The centre brick gives the location of his office, and the other bricks the regions he must take care of. The challenge here is to find better ways to assign these bricks to the SRs.

3 The Stakeholders

There are two main stakeholders in this problem, they are:

1. **Pfizer Organization** : This reassignment problem is important to Pfizer. This is because of the dynamic structure of the market. With time, the index values of bricks that quantify the workload, keep changing. Therefore, territories should be periodically reassigned in order to maintain a workload balance. Another aspect is disruption, i.e. when an SR is assigned a new territory. This means the SR has to start building relations with Medical Doctors (MDs) from the ground up, which is not very efficient.

2. **Sales Representatives** : The reassignment is also going to directly impact the SRs as the decision taken by the organization will increase/decrease their workloads. In the case of disruption, they will

no longer be able to utilize their ties with the MDs from their old bricks and will have to start building relations from the ground up with the MDs of their new bricks.

4 Mathematical Formulation

4.1 Constants

- **numSR** and **numBrick** : Respectively the number of Sales Representatives and the number of Bricks : 4 and 22 in our case
- **P** : The previous Assignment Matrix (*numBrick* x *numSR*), so that P_{ij} **= 1 if brick i is assigned to SR j, else 0.**
- **D** : The Distance Matrix (*numBrick* x *numSR*) so that D_{ij} **= distance of brick i to SR j's office.**
- **B**: The Brick Distance Matrix (*numBrick* x *numBrick*), so that B_{ij} **= distance of brick i to brick j.**
- **V** : The value vector so that V_i = the workload value involved with brick i.

4.2 Variables

- **X** : The new Assignment Matrix (*numBrick* x *numSR*). X_{ij} **= 1 if brick i is assigned to SR j, else 0.**

4.3 Parameters

- **min_load** : The lower bound of the workload tolerance interval.
- **max_load** : The upper bound of the workload tolerance interval.
- **max_disruption** : The upper bound for the disruption. We change this value in an iterative manner to eliminate solutions.

4.4 Constraints

- **Assignment**: each brick must be assigned to one and only one SR:

$$\forall i, j \;\; X_{ij} \in \{0, 1\} \tag{1}$$

$$\forall i \;\; \sum_j X_{ij} = 1 \tag{2}$$

Because we have three objectives : maximize workload equity, minimize distance, and minimize disruption, two of them will be treated as a constraint : workload and disruption.

- **Workload**: each SR must have a workload comprise between max_load and max_load.

$$\forall i, j \;\; \text{min\_load} \leq \sum_i V_i \cdot X_{ij} \leq \text{max\_load} \tag{3}$$

- **Disruption**: we specify a upper bound for the disruption.

$$\text{disruption } = \sum_{i,j} \frac{1}{2} V_i \mid X_{ij} - P_{ij} \mid \leq \text{max disruption} \tag{4}$$

Because some solver do not allow the use of absolute function, we can replace $\mid X_{ij} - P_{ij} \mid$ by $X_{ij} \cdot (1 - P_{ij}) + P_{ij} \cdot (1 - X_{ij})$. This work because X_{ij} and P_{ij} are binary variables.

4.5 Objective

So now, we only have one objective left : the distance. But between two solutions that have the same distance value, we prefer the one that have a smaller disruption. In order to obtain directly this solution, we minimize the distance + epsilon · disruption.

$$\textbf{Minimize} \sum_{i,j} D_{ij} \cdot X_{ij} + \epsilon \cdot \sum_{i,j} \frac{1}{2} V_i \mid X_{ij} - P_{ij} \mid \tag{5}$$

5 Questions

5.1 Do the obtained solutions have properties that are worth bringing to the attention of the decision makers ?

	SR	Center-brick	Bricks	Disruption	Workload	Distance
0	1	4	[4, 5, 6, 7, 8, 15]	0.0	0.9507	19.30
1	2	14	[10, 11, 12, 13, 14]	0.0	1.3377	33.32
2	3	16	[9, 16, 17, 18]	0.0	0.7048	10.05
3	4	22	[1, 2, 3, 19, 20, 21, 22]	0.0	1.0068	124.74

Figure 2: Initial assignment

The initial solutions obviously don't cause any disruption, since no re-assignment is made. The disruption value is thus equal to 0. More importantly, the workload constraint is not satisfied. Indeed, workload is not always contained in the interval $[0.8, 1.2]$. For instance, SR 2 has a workload of 1.3377.

5.2 Discuss the properties of the current solution in comparison to the solutions you have obtained

We implement a model that finds every efficient solution to the problem defined above. Using the cvxpy solver in python, we obtain iteratively 20 efficient solutions. Each new solution presents a lower disruption value than previous ones, until there is no alternative solution. As expected, the more constraint the disruption, the bigger the distance.

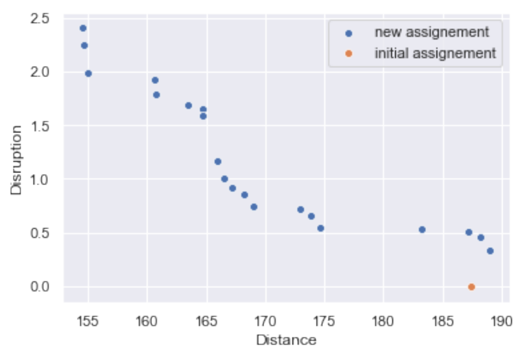

Figure 3: Efficient solutions

5.3 Varying workload is currently an objective taken into account in the form of constraints ? Try at least one interval and discuss the result

In this question, we try different intervals to constraint the workload, namely $[0.825, 1.175]$, $[0.85, 1.15]$, $[0.875, 1.125]$, $[0.9, 1.1]$. As the interval chosen gets smaller, the number of efficient solutions decreases (from 20 in the original model to 10 in the last interval). Furthermore, as we add more constraints, the solutions found are less interesting, meaning that they show a a higher distance for an equivalent disruption level. This makes complete sense as restricting the problem even more reduce the range of possibilities and is thus likely to yield less optimal solutions.

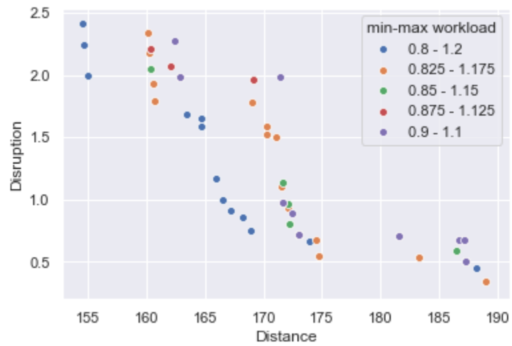

Figure 4: Efficient solutions when varying workload bounds

5.4 How to model the case for partially assigning bricks ?

The process is extremely similar to what was done in question 2. The only difference is that we relax the integer constraint on the assignment matrix. More precisely, we remove the binary constraint on X and let each X_{ij} vary between 0 and 1 so that X_{ij} **is now the % of brick i assigned to SR j**. The rest of the problem is defined identically. This relaxation leads to obtaining many more efficient solutions (60), which is logical as it offers more possibilities. Note that most of these new efficient solutions involve a large disruption.

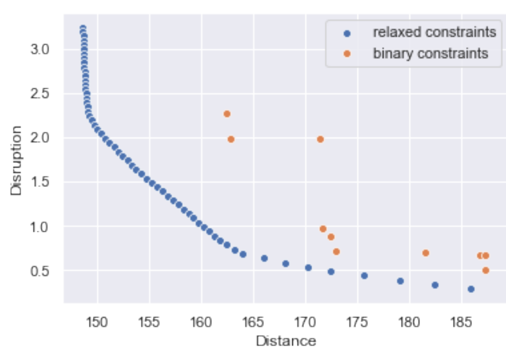

Figure 5: Efficient solutions when partially assigining bricks to SRs

5.5 If the demand increases uniformly in all bricks, it may be necessary to hire a new sale representative. Where to locate his office?

The first idea we had was to add two new variables to the initial problem.

- **X'** the assignment vector for the new SR, such that $X'_i = 1$ if brick i is assigned to the new SR, else 0
- **L** the new localisation vector such that $L_i = 1$ if the new SR is located at brick i

These variables enable us to compute the distance traveled by the new SR via the formula:

$$\text{distance } = \sum_{k,i} L_i \cdot B_{ik} \cdot X_k \tag{6}$$

We had to linearize every product $L_i \cdot X_k$ by adding new variables but ultimately, we found out that cvxpy optimization solver was two slow to solve the problem.

As a result, we adopted a simpler approach, which consists in considering all possible locations for the new SR office independently. As we assume that two SR can share the same office and therefore be located in the same brick, we have 22 possible cases to consider. We ran the solver as we did in previous questions, adapting correspondingly the formulation of the problem. The constraints of the model are not modified except for the fact that j now goes from 0 to $numSR + 1 = 5$ (for the assignment matrix and the workload). However, the new seller is not considered in the disruption constraint equation as it did not exist before. In the end, we are left to minimise the (same) objective function in order to find all efficient solutions, for each case. Once this is done, we concatenate the results and retrieve only the efficient solutions, as it can be observed on the graph below.

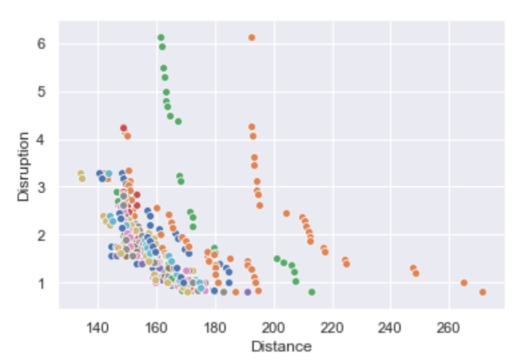

Figure 6: All efficient solutions for each possible office location of the new SR

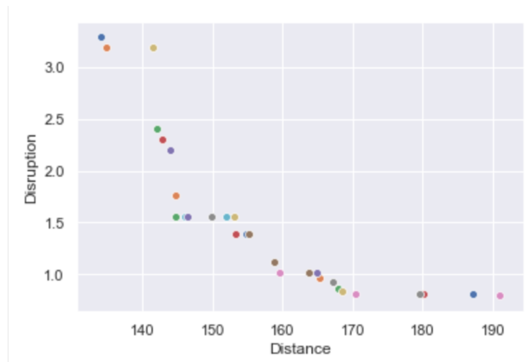

Figure 7: Remaining efficient solutions when results are combined

5.6 Generalise the model so as to allow a modification of the center bricks

This question is a generalisation of the above concepts. We now want to find the optimal center bricks for each of the four original SR. We thus need to define a new variable :

- **L** : localisation matrix so that $L_{ij} = 1$ if office of SR j is located on brick i. For this variable, we have to add a new constraint so that an SR office is located at one and only one brick.

.

$$\forall i, j \;\; L_{ij} \in 0, 1 \tag{7}$$

$$\forall j \;\; \sum_i L_{ij} = 1 \tag{8}$$

We now also have to compute the distance without using D matrix but instead by using brick to brick B matrix.

$$\text{total distance} = \sum_{i,j,k} B_{ik} \cdot L_{kj} \cdot X_{ij} \tag{9}$$

As a consequence, we have to linearize each $L_{kj} \cdot X_{ij}$ product. To do so, we introduce for each product a new variable K_{ijk} with constraints that insure than $K_{ijk} = L_{kj} \cdot X_{ij}$

$$\forall i, j, k \;\; K_{ijk} \in \{0, 1\} \tag{10}$$

$$\forall i, j, k \;\; K_{ijk} \geq L_{kj} + X_{ij} - 1 \tag{11}$$

$$\forall i, j, k \;\; 2 \cdot K_{ijk} \leq L_{kj} + X_{ij} \tag{12}$$

So we end up with : 22x4 (matrix X) + 22x4 (matrix L) + 22x22x4 (tensor K) = 2112 binary variables. Because, we saw in question 5 that cvxpy optimizer is limited when dealing with an important of variable, we used another solver : CP-SAT (from OR-TOOLS framework) which is a constraint solver that still allow to do some optimization on the top (but is not always able to find the optimal solution).

In practice, we went a step further than what was asked in the doc by adding a constraint restraining the number of possible SR's office moves. We started by making it possible to move at most 1 center brick, then printed the set of efficient solutions and progressively relaxed this constraint, until making it possible to change all four center bricks. The results are observable on the plot below.

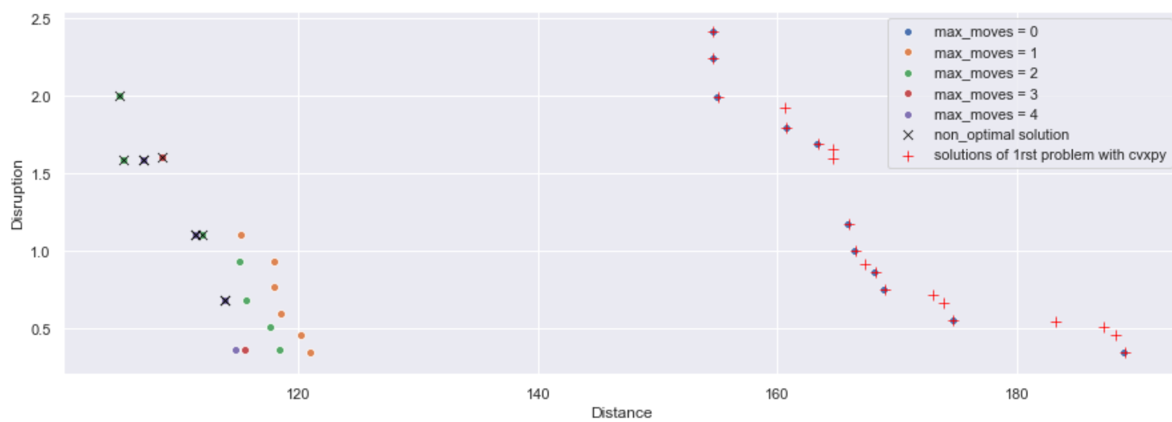

Figure 8: Efficient solutions when restricting the number of center bricks we change

5.7 How to incorporate in the model the SR preferences for their area ?

To incorporate this, we propose to ask each SR to rank all possible office locations according its preferences. We therefore obtain a matrix R of dimension 22x4, where each cell R_{ij} represents the rank (number from 1 to 22) attributed to the office location i by SR j. For each SR j, we calculate the dot product $R_{ij} \cdot W_{ij}$, which outputs the rank associated with the attributed office location. By summing the score over all SR, $\sum_j R_{ij} \cdot W_{ij} \cdot X_{ij}$, we obtain a preference score that is relevant to minimise if we want to please them as much as possible.

To emphasize even more preferences within our model, we could square rankings. Repeating the same process would give a higher weight to the preference value in the minimisation problem.
A last improvement we have thought of consists in weighting the preference score of each SR by its workload. This induces that we will try to please a bit more the SR having a higher workload than others, which seems to be a fair compensation.
One limitation of this approach resides in the difficulty to produce a ranking for the SR.