Contributing Team Members: Cainan Parrish, Vijaya Mummidi, Keval Mehta, Yu-Ling Hsiao

FINAL PROJECT

About the Company

About Care is a Non-profit Organization based out of Chandler that caters to the health / physical needs of people who are in need. Few of the services they provide include transporting the elderly / physically disabled to clinics, helping them understand the prescriptions etc.

The volunteers who take part in the NGO's works are usually certified nurses. On a regular day, the volunteers approach the client's location for their needs and provide them with the services required. The assignment of the available volunteers to clients is usually done in a random manner.

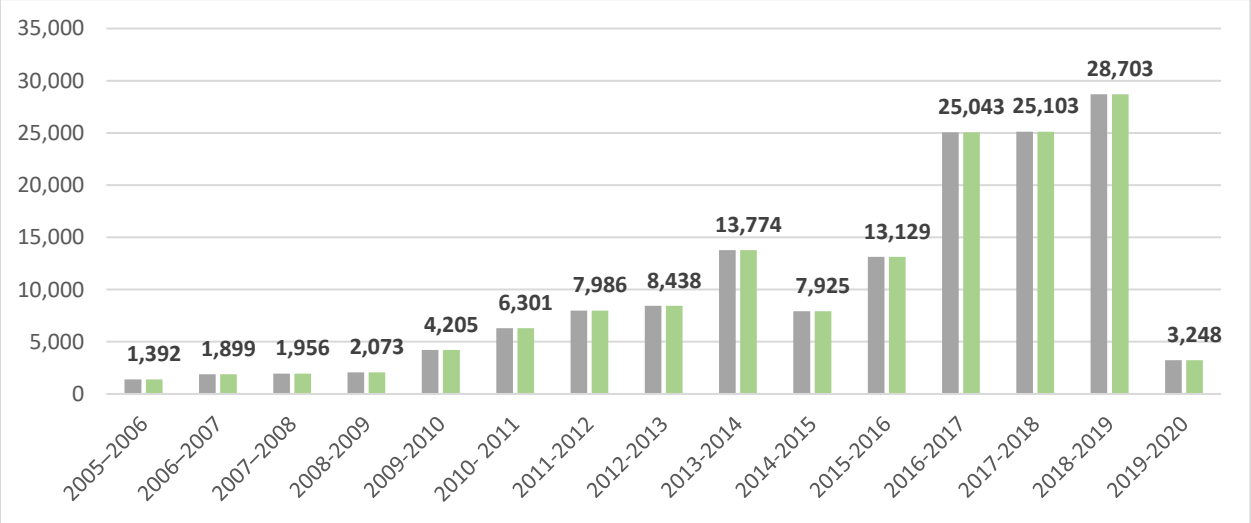
In order to contribute to their good work and apply our knowledge on modelling to the real-world problem, we thought it was a great opportunity for the team to analyze their problems and come up with an optimization solution.

Data Collection

We received a large set of data from the NGO which included details about the clients who were served and the volunteers who served them. We also had financial reporting data and a list of grants that were applied for, accepted or denied. We narrowed down the information provided to only that information which was needed in order to answer this problem. This meant that we were not going to look at the commercial financial information or the grant information.

Upon analyzing the relevant data provided to us in detail, we identified that the services provided by the NGO had increased exponentially over the years. This meant, they had to utilize the volunteers available effectively in order to meet all their demand, unlike their regular approach of random allocation.

A plot on the services of the NGO across years



Problem Statement

Based on our analysis of the positive impact our solution is going to have on the NGOs productivity, we decided to optimize the number of visits made by a specific volunteer to a specific client.

We had a specific list of business requirements for the same which were considered as constraints in our model

1) Requirements of the client (in terms of number of visits made by volunteer) must be met.
2) A volunteer's allocation shouldn't exceed 4 visits per day
3) Every volunteer should make at least one visit

Here are the examples of the datasets.
(Click here for the complete client data file)

Client Number	Age	City	Sex	Zip	Cross Streets	Demand
1	69	Chandler	F	85286	Germann & McQueen	1
2	62	Chandler	F	85225	Cooper & Chandler	4
3	86	Chandler	F	85249	McQueen & Riggs	1
4	80	Gilbert	F	85297	Germann & Power	2
5	81	Chandler	M	85226	Chandler Blvd & Rural	1

(Click here for the complete volunteer data file)

Volunteer ID	City	Cross Streets	Zip
1	Chandler	Pecos & McQueen	85225
2	Chandler	Pecos & McQueen	85225
3	Tempe	202 & Rural	85281
4	Tempe	Warner & Priest	85284
5	Chandler	Gilbert-Riggs	85249

Challenges

While working on the solution, we came up with few challenges:

1) Data provided by About Care did not include the volunteer availability.
2) The dataset was too large for the excel solver to process it.
3) We did not have readily available data that showed the distance between each volunteer and client's location.

To overcome the challenges, we had to create volunteer availability data based on our assumptions (the Demand column in client data file). The NGO was informed of this and was suggested to start maintaining a record of the volunteer's availability so that they could use the model efficiently.

We used Google API in combination with python to retrieve the distances between two locations. This way we came up with a distance matrix between each volunteer and client which was an input parameter to our model.

To handle the problem of large datasets, we used Gurobi and other libraries in Python to achieve our objective.

Mathematical Model

Parameters:

i= Client

j= Volunteer

V_i: Visits needed by client i in a month

D_{ij}: Distance between client i and volunteer j

Decisions:

X_{ij}: Number of visits made by volunteer j to client i

Objective:

Minimum Total Distance: $\sum_i \sum_j X_{ij} \times D_{ij}$

Constraints:

$\sum_j X_{ij} = V_i, \forall i$ (Visits needed by clients should be meet)

$\sum_i X_{ij} \leq 4$ (Volunteers should have at most 4 visits)

$\sum_i X_{ij} \geq 1$ (Volunteers should have at least 1 visit)

$X_{ij} \in Integer, X_{ij} \geq 0$ (Number of visits should integer and positive)

Python Code

First, since we only used zip code to locate the volunteers and clients, we extracted out the unique zip code pairs and used Google Matrix API to get the distance (miles) between regions. (Click here for the complete distance matrix file)

	85286	**85225**	**85249**	**85297**	**85226**	**85224**	**85233**	**85298**	**85234**
85225	3.9	1	6.2	9.2	9.8	4.5	3.6	12.2	7.8
85281	19.1	16.2	21.4	24.4	14.4	12.3	13.7	27.5	16.3
85284	11.6	8.6	13.9	16.9	4.3	4.7	8.3	20	16.2
85249	4.2	6.2	1	10.1	11.8	10.4	9.5	7	16.6
85224	8.2	4.5	10.5	13.5	7.3	1	5.2	16.5	10
85142	15.6	18	12.3	10.2	24	22.6	21	7.5	18.5
85248	5.8	7.9	3.4	11.7	9.8	8	11.1	8.9	18.7
85234	13.3	8.4	16.7	8.7	19.6	10.3	5.4	11	1
85226	9.7	9.8	12	15	1	7.4	12.2	18	20.2

Then we populate the zip code distance data to the volunteer and client matrix. (Click here for the complete volunteer and client distance matrix file)

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	3.9	1	6.2	9.2	9.8	1	4.5	3.6	3.6
2	3.9	1	6.2	9.2	9.8	1	4.5	3.6	3.6
3	19.1	16.2	21.4	24.4	14.4	16.2	12.3	13.7	13.7
4	11.6	8.6	13.9	16.9	4.3	8.6	4.7	8.3	8.3
5	4.2	6.2	1	10.1	11.8	6.2	10.4	9.5	9.5
6	8.2	4.5	10.5	13.5	7.3	4.5	1	5.2	5.2
7	15.6	18	12.3	10.2	24	18	22.6	21	21
8	11.6	8.6	13.9	16.9	4.3	8.6	4.7	8.3	8.3
9	5.8	7.9	3.4	11.7	9.8	7.9	8	11.1	11.1

After we obtained the necessary data, we used Gurobi to build model and solve our problem. Following is the example of our code. (Click here for the Python code file)

Create model

```
model = gp.Model()
```

Create decision variables.

```
decision = model.addVars(matrix.shape[0], matrix.shape[1],
name='decision', vtype='I')
```

Create distance dictionary for every volunteer and client pair

```
distance = {}
for i in matrix.index:
    for j in matrix.columns:
        distance[(i - 1, int(j) - 1)] = matrix.iloc[i - 1, int(j) - 1]
```

Set objective - minimum total distance

```
objective = decision.prod(distance)
model.setObjective(objective, GRB.MINIMIZE)
```

Add constraints

```
# number of visits should be positive
for i in decision.values():
    model.addConstr(i >= 0)
# number of sevice for client should meet their demand
for c in matrix.columns:
    model.addConstr(decision.sum('*', int(c) - 1) ==
demand_list['Demand'][int(c)])
for r in matrix.index.values:
# each volunteer should service less than 4 times
    model.addConstr(decision.sum(r - 1, '*') <= 4)
# each volunteer should service at least 1 time
    model.addConstr(decision.sum(r - 1, '*') >= 1)
```

Optimize and solve the problem.

```
model.optimize()
```

After successfully processing the above python code, we got **1685.3 miles** as our optimized objective and arrived with the result of decision variables. (Click here for the complete result file)

Below is a small snippet of the result (This is modified to show both the scenarios):

	Client Number						
Volunteer ID		**1**	**2**	**3**	**4**	**5**	**6**
	1	0	0	3	0	0	0
	2	2	0	0	0	0	0
	3	0	1	0	0	2	0
	4	0	0	0	0	0	4
	5	0	0	0	1	0	0
	6	0	0	0	1	0	0

In the above matrix, 0 represents that the corresponding volunteer is not assigned to the corresponding client, whereas the cells in green show the number of visits between each volunteer and client.

Conclusion

We were successfully able to achieve our target of finding the optimal solution for number of visits between the clients and volunteer.

We believe this should significantly improve the NGO's productivity in catering needs of clients.

This project helped us to utilize the knowledge we gained in the class to apply to real-time problem effectively.