Introduction

In this project we are given a problem that is similar to a Traveling Salesman Problem with some constraints. We have 30 islands that we have to travel and since the road between them differs as Asphalt, Concrete and Gravel, the cost of traveling between 2 islands is different from the others. Also, coordinates of the islands can make time between 2 islands longer or shorter.

In addition to that we have storm constraints so that when a storm happens between 2 islands, the truck cannot use that road. To solve this problem we mainly used a TSP algorithm. You can see the details in the Methodology section.

Methodology

To solve a model we first create a Decision Variable which is:

X_{ij} = 1 if we use the road between island i and j, 0 if we do not use the road between island i and j.

Then **Objective Function** becomes:

$$min \sum_{i=1}^{30} \sum_{i \neq j,\, j=1}^{30} X_{ij} * T_{ij}$$

where T_{ij} is the truck's time spent (with the consideration of road type) between island i and j.

This Objective function is subject to some constraints:

Constraint 1:

Truck must enter each city only once

$$\sum_{i \neq j,\, i=1}^{30} X_{ij} = 1 \quad \text{for } j = 1,2,3\ldots..30$$

Constraint 2:
Truck must leave each city only once

$$\sum_{i \neq j, j=1}^{30} X_{ij} = 1 \quad \text{for } i = 1,2,3.....30$$

Constraint 3:
Make sure that there is only a single complete tour and eliminate subtours.

$$Y_i - (30+1) * X_{ij} >= Y_j - 30 \qquad 2 <= i \neq j <= 30$$
$$0 <= Y_i <= 29 \qquad 2 <= i <= 30$$

where Y is the set of islands.

Constraint 4:
Make sure that truck is not pathing through the stormy road.
$X_{ij} * S_{ij} = X_{ij}$ for i = 1,2,3…..30, j = 1,2,3…..30 $i \neq j$
where S_{ij} is binary and equals to 1 if **there is not** a storm between the road i and j.

After determining the Objective function and Constraints we did some preprocessing of given data and parameters.

Since the truck has max speed in roads: Asphalt (100), Concrete(65), Gravel(35), we must firstly create the time data that contains the duration of travels between islands. To do that a method takes the coordinates of 2 islands, calculates distance and divides the distance by speed (depending on road type). After that we have a collection of time durations of all roads (which model wants to minimize).

Moreover, to create Constraint 4 we created a Storm array that contains binary values. To decide if there is a storm or not between 2 islands, we check if the distance between the center of the storm to the road is greater or not than the radius of the storm. If a storm is detected between 2 islands we change the corresponding index of the Storm array to 0 (if there is a storm and road is not usable) or 1(if there is not a storm and road is usable).

Finally, we run our model and get the path which will be discussed in the conclusion section.

Constraint 5:

A road between island i and j are either usable or not. If it is usable, X_{ij} will be equal to 1. If it's not usable, X_{ij} will be equal to 0.

$X_{ij} \in \{0,1\}$ for i = 1,2,3…..30, j = 1,2,3…..30

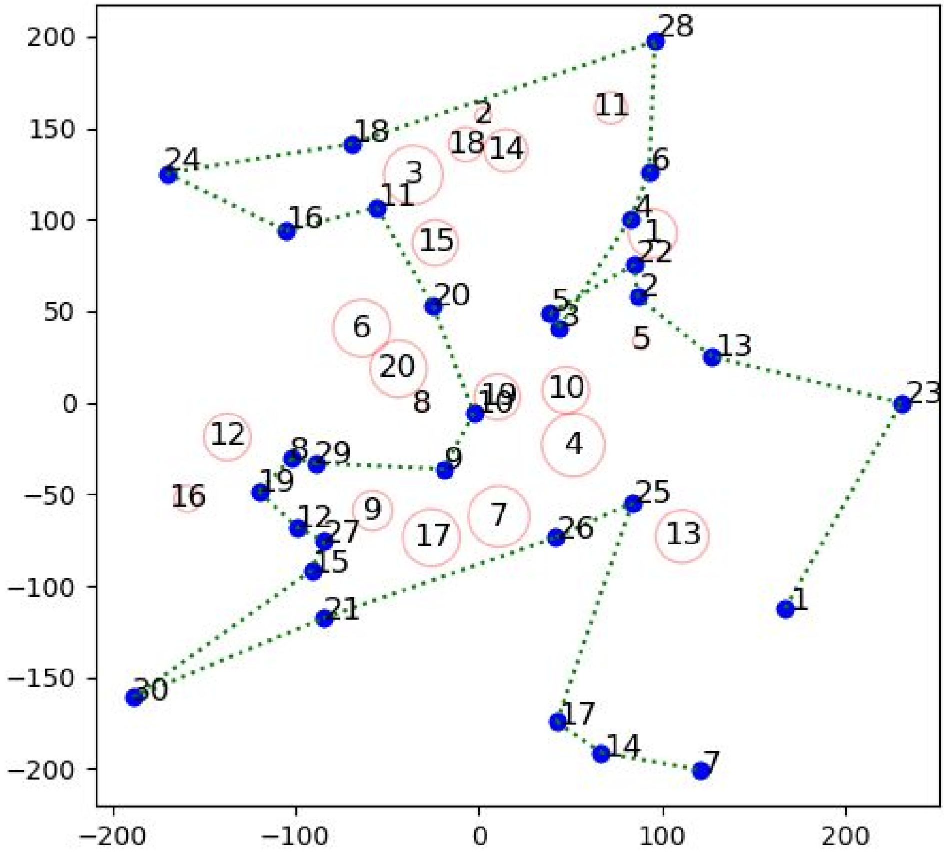
Conclusion

After running our solver (with calculation time of 59.34 CPU seconds), we obtained the **optimal solution** of **19.899226952973244** for the minimized travelling time of the cargo truck. Optimal route (starting from the first node) we have obtained is the following:

1, 23, 13, 2, 22, 5, 3, 4, 6, 28, 18, 24, 16, 11, 20, 10, 9, 29, 8, 19, 12, 27, 15, 30, 21, 26, 25, 17, 14, 7, 1

We have also visualized the problem in the cartesian coordinate system.

Blue dots represent the nodes visited by the cargo truck and are numbered according to their corresponding order in "data.xlsx". Red circles represent the storms (with their given center coordinates and radii) which are to be avoided by the cargo truck. Finally, the green line starting from Node 1 and ending on Node 7 represents the optimum route which can be taken by the cargo truck.



Python scripts that we have used to run our solver and plot our data are provided in the following appendix section.

It is necessary to note that we have encountered a memory issue when running the Windows version of Python MIP Solver and it prevented us from running our script in Windows operating systems. This issue is unrelated to our solution and it is caused by the way the MIP Solver runs in Windows. However, the script can be run in Linux and MacOS and it will successfully exit with corresponding objective value and optimum route.