Travelling Salesman and a Trip around India

No. of pages: 14

Introduction

After high school ends, I will have to go off to college. But before I do that, I need to meet my family scattered all across India in six different major cities. Knowing that the time I will have is limited, I need to find a way to travel to meet all of them in the least possible time and return back to my home city of Hyderabad. Since the most time friendly route is of most convenience, it might not be the most pocket friendly. On the flip side, lowest duration means lowest flight time and fuel use, so it might also be cost friendly. Thus, another very important factor : cost, needs to be incorporated in the decision.

Hence, the question that this exploration will answer will be "Which route requires the shortest amount of time to travel across six major Indian cities and is it also the most cost-effective?". The cities that will be considered are Hyderabad, Visakhapatnam (Vizag), Mumbai, Bengaluru, Delhi and Kolkata. To find the solution to my Real Life Problem , the Travelling Salesman Problem is going to be applied into this context and evaluated under the lenses of time and cost.

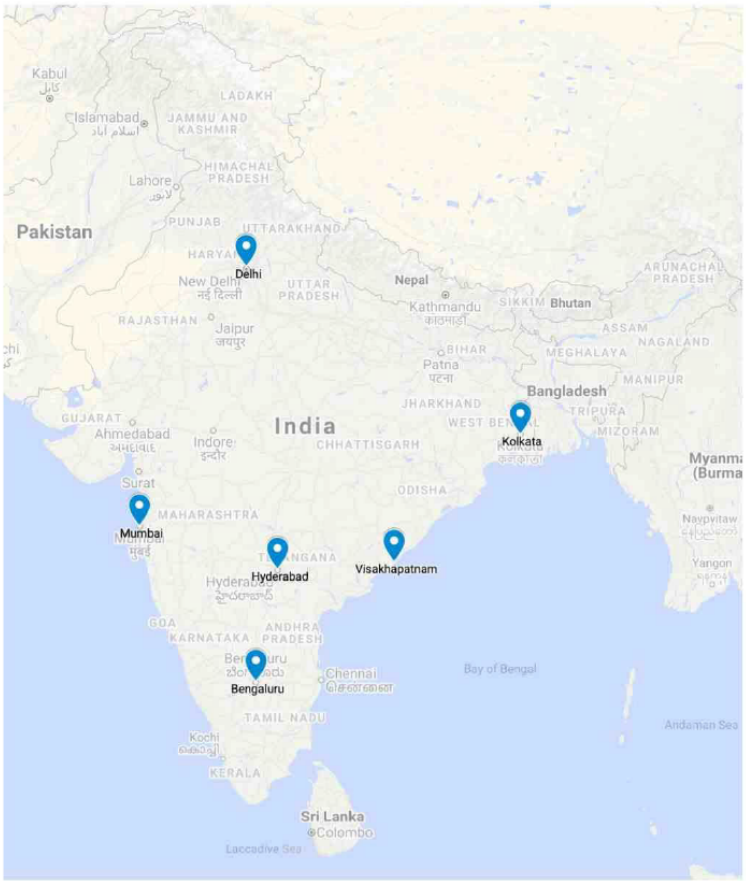
To provide some history, the origin of Graph Theory could be traced back to Euler's work in the 1735 as a solution to the Königsberg bridge problem. Travelling Salesman Problem is also a part of graph theory that helps to find a cyclic tour of minimum time or distance a salesman would need to travel a number of cities. This problem was popularised by RAND Corporation in the 1930's and led to a paper by G.B Dantzig who solved it with 49 cities. (Gross, et al.)

The classical TSP is a Hamiltonian graph of least weight identified on a complete weighted graph. Also, a hamiltonian graph is a cycle that visits every vertex in the graph exactly once except its starting and ending points. This exploration also attempts to use a similar classical TSP to find the route that would take the least time and cost for visiting the six cities.

Initial Graphing and Weights

The map of India with the selected cities can be viewed below to provide a visual representation of them on an actual scale. The six cities that are being considered have all been spotted with a blue marker icon

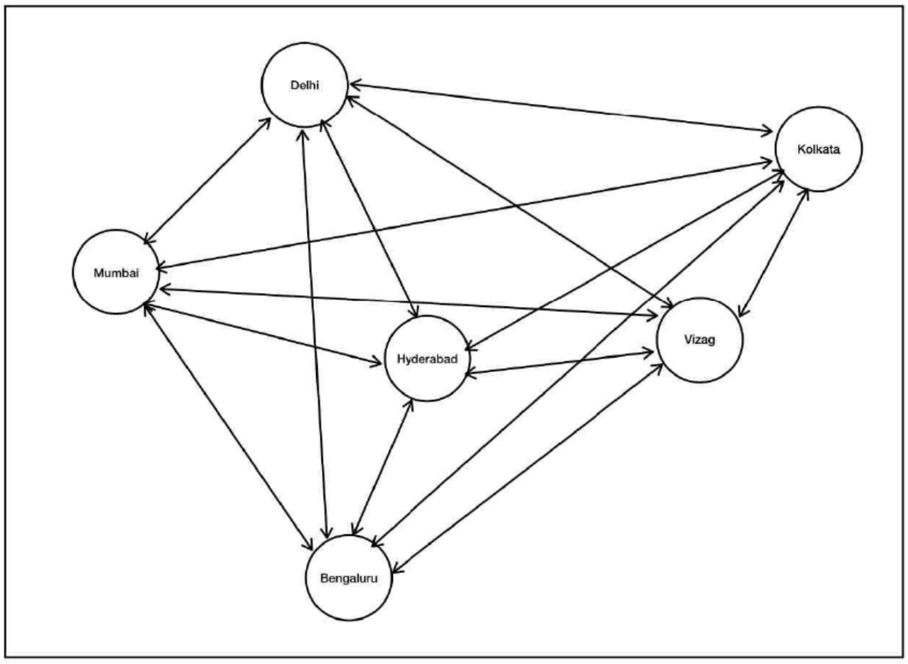
Figure 1 : Selected cities of the exploration



(Map made using Google My Maps)

The graph below is complete graph of the selected cities depicted as vertices and paths between them as edges. A complete graph is a graph in which all the vertices are connected to each other by an edge. To put it into context of the exploration, each of the six cities is connected directly to the other five by the means of an edge, so it shows a complete graph.

Graph 1 : Complete Graph of selected cities and paths



By visualising the above graph , the weighted adjacency tables based on duration and cost can be seen below. Table 1.0 depicts the time to travel between cities as the weight and Table 2.0 depicts the cost of travel between these cities as the weight. For generalisation, the time has been taken as the least time of the journey since different flights can be of different durations. As for the costs, the average price of journey has been considered for the dates in July (the actual time I plan on taking the trip). All the durations and prices are taken from Google Flights.

Table 1.0: Weights as Duration of Flight (in minutes)

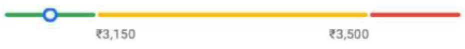
		Arrival Airport					
		Hyderabad	**Vizag**	**Kolkata**	**Bengaluru**	**Delhi**	**Mumbai**
Departure Airport	**Hyderabad**	-	70	125	55	120	75
	Vizag	70	-	95	90	145	110
	Kolkata	125	95	-	145	110	155
	Bengaluru	55	90	145	-	125	75
	Delhi	120	145	110	125	-	125
	Mumbai	75	110	155	75	125	-

Table 2.0: Weights as Cost of Flight (in INR)

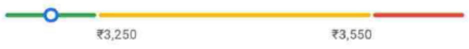
		Arrival Airport					
		Hyderabad	**Vizag**	**Kolkata**	**Bengaluru**	**Delhi**	**Mumbai**
Departure Airport	**Hyderabad**	-	3200	3600	2200	3700	2700
	Vizag	3200	-	4100	3400	4100	3800
	Kolkata	3600	4100	-	4300	3700	5100
	Bengaluru	2200	3400	4300	-	4400	3200
	Delhi	3700	4100	3700	4400	-	3300
	Mumbai	2700	3800	5100	3200	3300	-

* Table 2.0 uses costs as weightage, it is obvious that flight prices are dynamic and the price from city A to city B might not be the price for city B to city A. So, to provide generalisability the weights used are the average of the lowest average costs of the trip during the time. Refer example below.

Trip from Hyderabad to Vizag:



Trip from Vizag to Hyderabad:



Weight taken = (3150+3250)÷2 = ₹ 3200

Similarly, all the other weights were calculated and were rounded to the nearest 100.

Solution using TSP

The data being considered has 6 different vertices and all are connected to each other, so I figured it would be impractical to try and solve it by trial and error and hence I will be using the upper and lower bound method individually for calculating the path with lowest duration and least cost .

(a) Lowest Duration

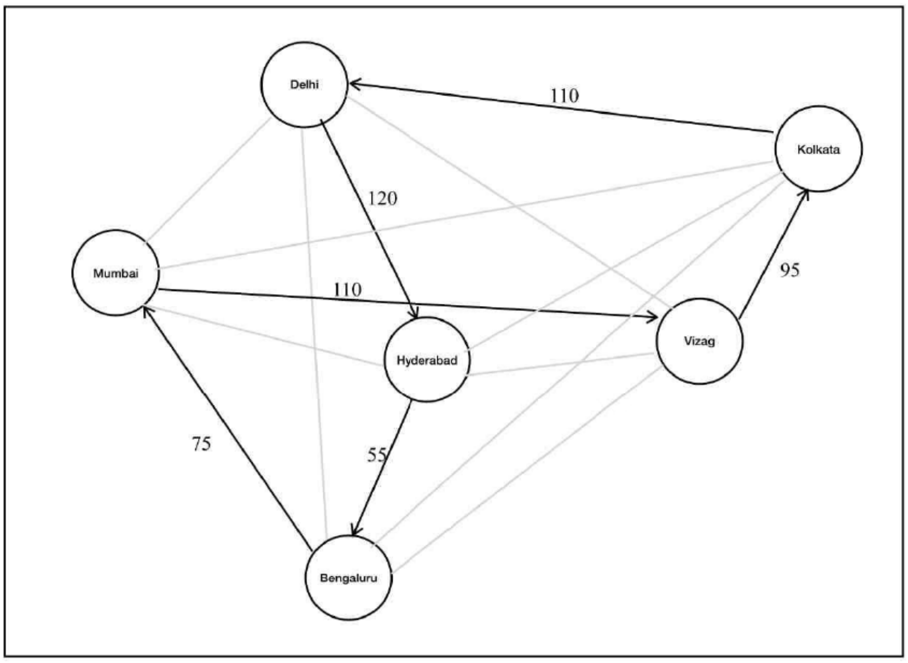
(i) Upper bound

The upper bound is found using the nearest neighbour algorithm. First, we know we have to start and end at Hyderabad, so we take the edge with the lowest duration from Hyderabad to another vertex, in this case Bengaluru, now we do the same with Bengaluru as the Departure Airport and since we cannot go to Hyderabad again, we choose the next shortest edge. This process is repeated till all the cities have been visited once and then create a cycle by traveling back to Hyderabad. It should be noted that no city is visited more than once. The weight of this route is the upper bound. The selected edges have been highlighted and numbered as seen in Table 1.1 below to provide a better understanding of the process. Graph 1.1 below, provides a visual representation of Table 1.1 by highlighting the chosen path.

Table 1.1: Upper Bound for lowest duration (in minutes)

		Arrival Airport					
		Hyderabad	**Vizag**	**Kolkata**	**Bengaluru**	**Delhi**	**Mumbai**
Departure Airport	**Hyderabad**	-	70	125	55 (1)	120	75
	Vizag	70	-	95 (4)	90	145	110
	Kolkata	125	95	-	145	110 (5)	155
	Bengaluru	55	90	145	-	125	75 (2)
	Delhi	120 (6)	145	110	125	-	125
	Mumbai	75	110 (3)	155	75	125	-

Graph 1. 1 : Path of Upper Bound for the lowest duration (in minutes)



The upper bound route suggests the following path:

Hyderabad → Bengaluru → Mumbai → Vizag → Kolkata → Delhi → Hyderabad

Therefore, upper bound = 565 min

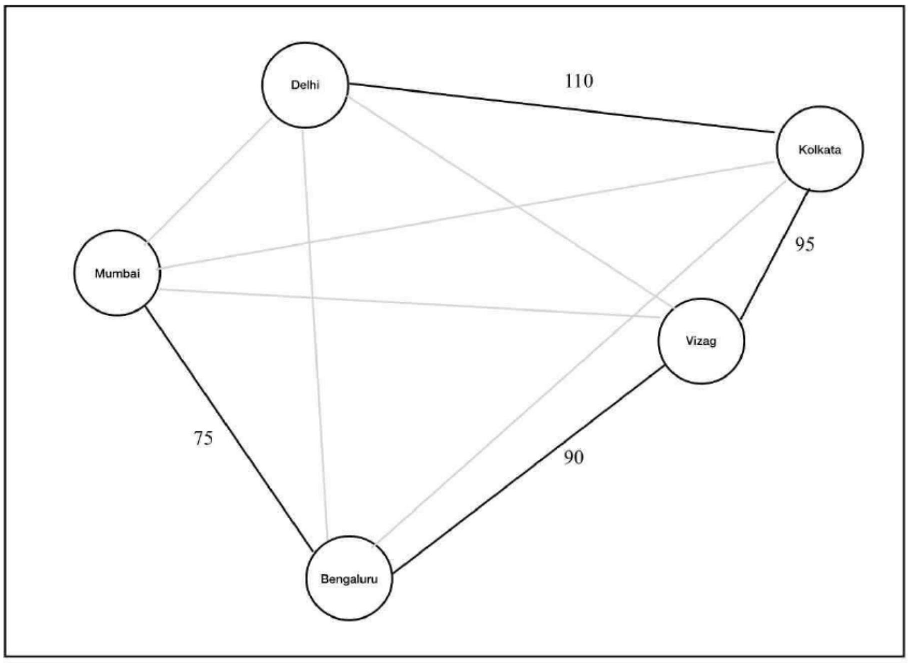
(ii) Lower bound

Lower bound will be found using the Kruskal's Algorithm. First we delete the starting and ending vertex along with connected edges, in this case Hyderabad and then make a minimum spanning tree. A minimum spanning tree is a collection of connected edges on a graph that includes every vertex and is of the least weight. It is important to note that a cycle is not formed, and to find the lower bound the weights of the two shortest deleted edges must be added to the weight of the tree. Since we do not want to make a cycle, the number of edges will be *one less than* the number of vertices. Here, there are 5 vertices, so the number of edges in the minimum spanning tree will be (5-1), which is 4. The minimum spanning tree is shown in Table 1.2 below and the path is highlighted in Graph 1.2.

Table 1.2: Edges in MST for lowest duration (in minutes)

Departure Airport	Arrival Airport: Vizag	Kolkata	Bengaluru	Delhi	Mumbai
Vizag	-	95	90	145	110
Kolkata	95	-	145	110	155
Bengaluru	90	145	-	125	75
Delhi	145	110	125	-	125
Mumbai	110	155	75	125	-

Graph 1. 2 : Kruskal's Algorithm for lower bound for lowest duration (in minutes)



So, lower bound using Kruskal's algorithm

= length of minimum spanning tree+lengths of two shortest deleted edges

= (75+90+95+110) + (55+70)

= 495 minutes.

(iii) Solution of Lowest Duration

Now that the upper and lower bound has been found, then the weight 'm' of the solution of TSP is going to be between those values.

So, $495 \leq m \leq 565$.

Answer to the TSP is the following route,

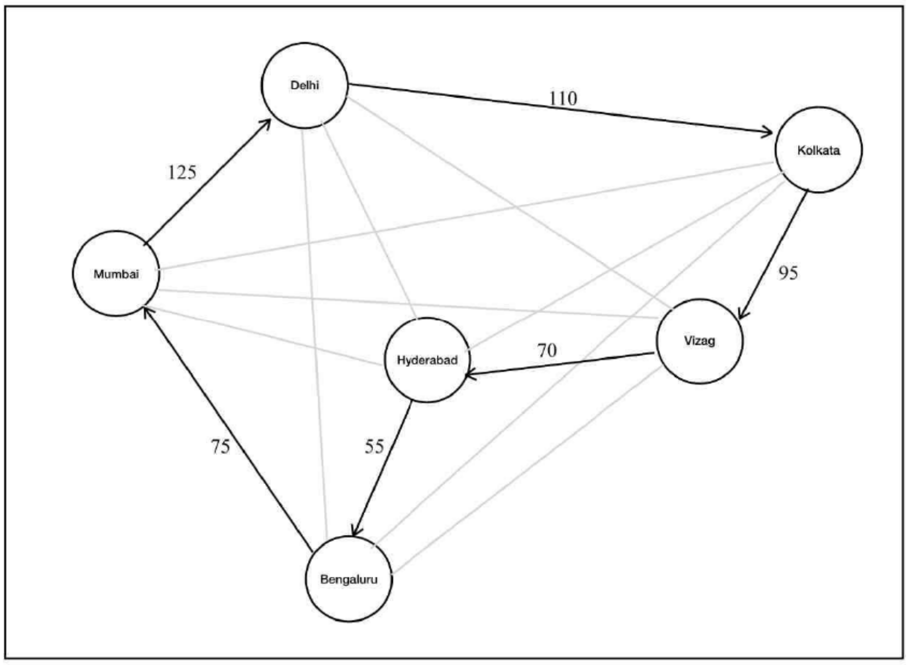
= Hyderabad → Bengaluru → Mumbai → Delhi → Kolkata → Vizag → Hyderabad

= 55+75+125+110+95+70

= Total weight = 530 mins

Therefore, m = 530 mins is the minimum duration of the complete flight time to and from Hyderabad to the other five cities. It can be viewed in the hamiltonian cycle below in Graph 1.3 .

Graph 1 . 3 : Solution to TSP for lowest duration (in minutes)



(b) Least Cost

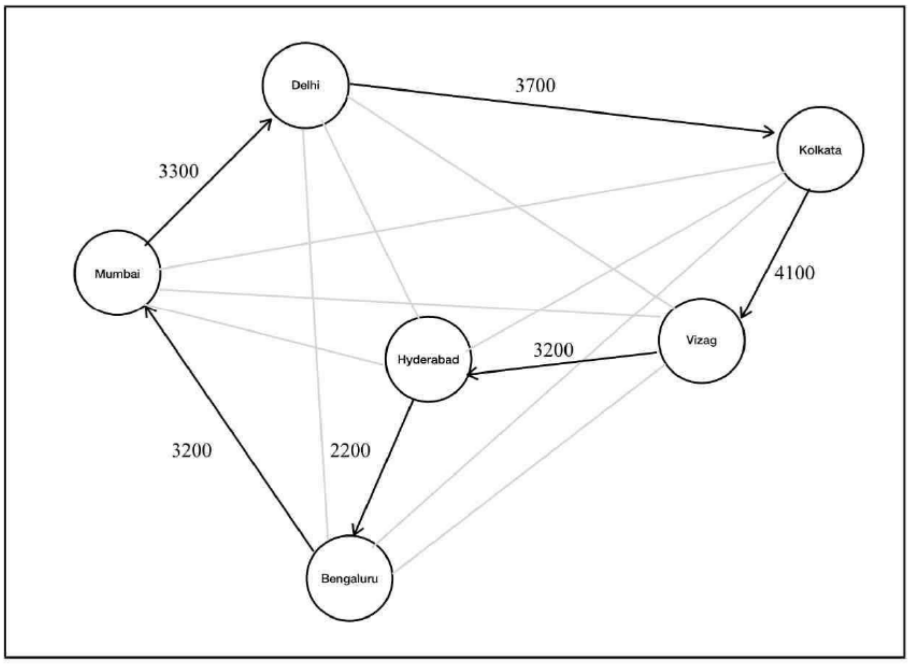
(i) Upper bound

The upper bound is found using the nearest neighbour algorithm again. The process used in (a) (i) will be repeated. The weight of this route is the upper bound. The selected edges have been numbered as seen in Table 2.1 and Graph 2.1 highlights the chosen path.

Table 2.1: Upper Bound for least cost (in INR)

		Arrival Airport					
		Hyderabad	**Vizag**	**Kolkata**	**Bengaluru**	**Delhi**	**Mumbai**
Departure Airport	**Hyderabad**	-	3200	3600	2200 (1)	3700	2700
	Vizag	3200 (6)	-	4100	3400	4100	3800
	Kolkata	3600	4100 (5)	-	4300	3700	5100
	Bengaluru	2200	3400	4300	-	4400	3200 (2)
	Delhi	3700	4100	3700 (4)	4400	-	3300
	Mumbai	2700	3800	5100	3200	3300 (3)	-

Graph 2. 1 : Path of Upper Bound for the least cost(in INR)



The upper bound route suggests the following path:

Hyderabad → Bengaluru → Mumbai → Delhi → Kolkata → Vizag → Hyderabad

Therefore, upper bound = ₹ 19700

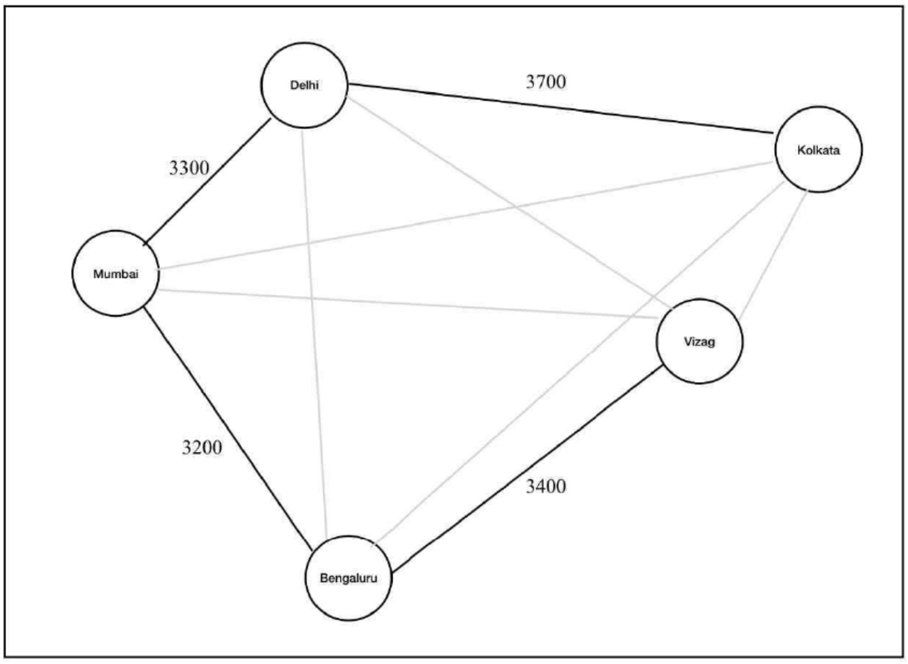
(ii)Lower bound

The lower bound will be found Kruskal's algorithm. Again, we want our path to start and end at Hyderabad, so we delete the vertex and all associated edges. Like the previous time, the minimum spanning tree will have 4 edges. The edges of minimum spanning tree are highlighted in Table 2.2 and the path has been shown in Graph 2.2.

Table 2.2: Edges in MST for least cost (in INR)

		Arrival Airport				
		Vizag	**Kolkata**	**Bengaluru**	**Delhi**	**Mumbai**
Departure Airport	**Vizag**	-	4100	3400	4100	3800
	Kolkata	4100	-	4300	3700	5100
	Bengaluru	3400	4300	-	4400	3200
	Delhi	4100	3700	4400	-	3300
	Mumbai	3800	5100	3200	3300	-

Graph 2. 2 : Kruskal's Algorithm for lower bound of least cost (in INR)



So, lower bound using Kruskal's algorithm

= length of minimum spanning tree+lengths of two shortest deleted edges

= (3200+3300+3400+3700) + (2200+2700)

= ₹ 18500

(iii) Solution of Least Cost

Now that the upper and lower bound has been found, then the weight 'm' of the solution of TSP is going to be between those values.

So, $18500 \leq m \leq 19700$.

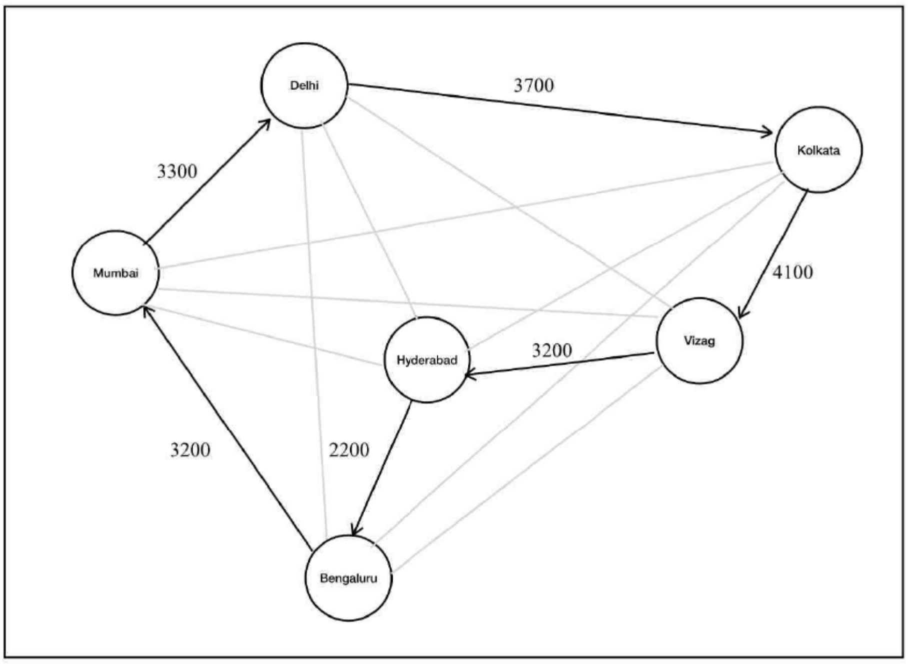
Answer to the TSP is the following route,

= Hyderabad → Bengaluru → Mumbai → Delhi → Kolkata → Vizag → Hyderabad

= Total weight = ₹ 19700

Therefore, $m =$ ₹ 19700 is the lowest cost to the cover the travel expenses to and from Hyderabad while visiting the other five cities. It can be viewed in the hamiltonian cycle below in Graph 2.3 .

Graph 2. 3 : Solution to TSP for least cost (in INR)



End Results and Analysis

The exploration aimed to find the least time taking route and analyse whether it is also the cheapest route to travel through to visit all the five cities after starting and ending the trip in Hyderabad. From solving it using the TSP, the trip with the lowest duration was to travel from Hyderabad to Bengaluru to Mumbai to Delhi to Kolkata to Vizag and finally back to Hyderabad and would take 495 minutes of flight time to complete. It was found that the trip that is the cheapest costs ₹ 19700 is also the same route that took the least time. This answers our question that the trip taking the least time is also the cheapest and the most optimum route to use.

Evaluation

The answer might seem very clear but in reality it is much more complicated. Though the result derived from this exploration might be true, other possibilities and limitations of this exploration need to be considered.

Firstly, the flight costs are taken as constant values though they are dynamic overall. The attempt to use an average value does make it more generalisable but it answers the question of the most cost-effective route 'in general' and not of this particular trip that I want to take in the beginning of July where the prices may be completely different.

Secondly, there are multiple flights for same routes and these different flights have different durations and costs. So, the flights selected for the least cost from City A to City B might not be the cheapest flight between the cities like it has been assumed in the exploration.

Thirdly, even if it is assumed that the flights that have the lowest duration are the cheapest, availability of seats on the flights also needs to be considered to reach a correct conclusion.

Lastly, the duration and the costs are taken from airport to airport, this neglects some important variables, distance and cost from city to airport. For example, the Hyderabad airport is nearly 90 minutes away from the city and costs ₹ 1400 by a taxi, while the Vizag airport is about 20 minutes from the city costing about ₹ 500 for a trip.

Conclusion

The exploration calculates the least time to visit all the six cities and return back to Hyderabad and checking whether this trip is also the least expensive. Due to the similarity to the Travelling Salesman Problem, the same method and understanding was used. The results suggest that the least time taking trip across the six cities to also be the least expensive trip. Though this is the desirable answer and deems the exploration as a success, the limitations identified under the evaluation section of this exploration are important to be considered.

Any study or exploration is not fulfilled until it has a scope for further research or allows for different perspectives/understandings to arise. Similarly, this exploration also identifies the limitations and agree that the answer in real life could be different since this is just a model.

However, the significance of the path to travel between these six cities should not be neglected either. Even if the costs might change and flight durations might be different, the values that are considered in this study are averages of real world values and close estimates of what the outcome would be if they were completely accurate.

So, even if the exploration does not give the answer to the question of the least cost and time taking trip for the particular instance being considered, it provides a close estimate and allows for a generalised answer to any trip involving these six cities. Also, with the right further study by mitigating the identified limitations, the derived result could be closer to real life answer of this situation.

Bibliography

Dantzig, G., et al. “Solution of a Large-Scale Traveling-Salesman Problem.” *Journal of the Operations Research Society of America*, vol. 2, no. 4, 1954, pp. 393–410. *JSTOR*, www.jstor.org/stable/166695. Accessed 2 Feb. 2021.

Gross, Jonathan L., et al. *Handbook of Graph Theory*. Chapman & Hall/CRC, 2017. Accessed 27 Jan. 2021.

Lenstra, J. K., and A. H. G. Rinnooy Kan. “Some Simple Applications of the Travelling Salesman Problem.” *Operational Research Quarterly (1970-1977)*, vol. 26, no. 4, 1975, pp. 717–733. *JSTOR*, www.jstor.org/stable/3008306. Accessed 30 Jan. 2021.

“Optimizing Complex Networks: Graph Theory.” *Mathematics: Applications and Interpretations*, by David Harris and Peter Gray, Oxford University Press, 2020, pp. 690–734.