Course Name - Operations and Revenue Analytics

Professor Name - Prof. Rajat Agrawal

Department Name - Department of Management Studies

Institute Name - IIT, Roorkee

Week - 07

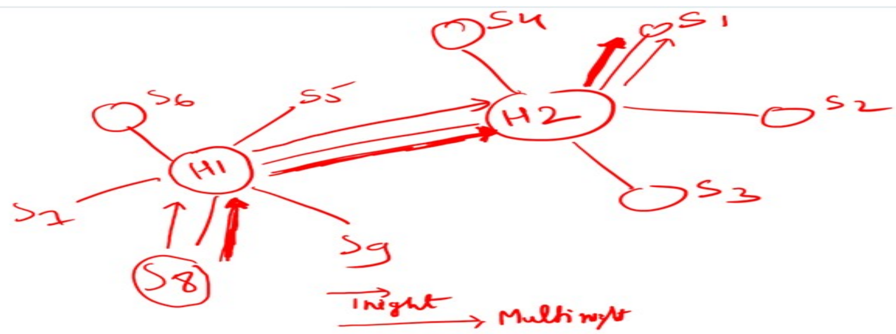
Lecture - 31

Welcome friends. In our earlier sessions, we were discussing particular cases of our revenue optimization, where we have situations like customers coming to my hotel just for a one-night stay, customers renting cars for one day, or customers traveling from Mumbai to Delhi, and that is all. But we all know that in global operations management, there is a very important concept we all know: the Hub and Spoke model. A large number of airlines, for optimizing their resources, have started using this Hub and Spoke model. There are some very popular hubs in the world, particularly in Middle Eastern countries.

There are hubs in European countries because all passengers traveling from, let us say, Asian countries or South Asian countries are being repackaged in those hubs, and then there are many spoke stations related to these hub stations. If I talk about India, in India also, there are some very important hubs we all know. For example, one hub can be our New Delhi airport, and from this New Delhi airport, there are a large number of spokes in North India. So, all of Punjab, Haryana, Uttarakhand, UP and Bihar are the spokes attached to the hub of New Delhi airport. So now, we actually have to see that we want to optimize the revenue of this entire network. When we were looking only at a particular resource, the optimization of revenue was much easier.

We saw how we could apply booking control, how we could apply some kind of simple heuristics to get the allocations, and then the optimization of our revenue happened. But here, we are going to discuss revenue maximization in a situation where you have hubs: Hub 1, Hub 2. And then there are multiple spokes: Spoke 1, Spoke 2, Spoke 3, Spoke 4. And to Hub 1 also, there are multiple spokes attached: Spoke 5, Spoke 6, 7, 8, 9. So, you have to see the optimization of revenue in this entire network. Someone may start from this spoke, travel to this hub, then from this hub to this second hub, and from this second hub to this particular spoke.

So, there are in fact three legs involved in this entire journey of this customer. And we need to see that how we are going to allocate request of this particular customer for this entire journey. Because, there may be on the same time many passengers who will be travelling only from Hub 1 to Hub 2, they are let me have a slightly thicker line for



representing the different types of cases. And then there may be few customers who are only travelling between this particular leg or this particular leg. So, you have to see that whether allocating seats to only a leg customer or to the entire network customer is more profiteering to you.

And therefore, in this particular session, we are going to discuss the allocation of our capacity for this network management. This network management is basically allocation of your capacity for network kind of customers who are going to take. Similarly, in the hotels also when you are having a customer only coming for one night it is easier to decide, but there are many examples where customers are staying multi night. And in fact, in this multi-night situation, maybe a weekend or a peak weekday may also fall where you may have a different rates of your hotel rooms. So, how to accommodate all those requests which are running into a network kind of situation.

So, we will be talking about a very simple way of allocating the capacity in the network which is known as greedy algorithm. We will discuss that this greedy algorithm though it is very easy but, it is not very successful in simple situations because, in all practical situations you will be having some limited kind of capacity as we already discussed. So, in the limited capacity cases greedy algorithms may not work. Linear programming we will be doing in this particular session. I hope all of us are very much familiar with LP modelling.

So, we will see how we can do LP modeling for such network situations, and when you have very definite demand labels—proper demand labels—like, 'This is the forecasted demand for leg 1, leg 2, leg 3, then linear programming is also possible. But we will discuss that it also has some shortcomings, and a lot of computational effort is also needed for linear programming to actually execute. And in all real-life situations, it is almost impossible to forecast your demand for a particular leg very accurately. There will always be some kind of uncertainty. So, how to incorporate that uncertainty in linear programming will also be a challenge.

Some researchers and mathematicians have suggested using mean demand for linear programming rather than an accurate estimation of the forecast. But that also has its own limitations. Therefore, another possible way of allocating capacity in network cases is virtual nesting, which is like our simple case when you have only one particular category of resources, as we have already discussed with the help of EMSR methods. So, a kind of variation of EMSR is possible to handle cases of network optimization, where we will try to see if we can divide the network resources into some kind of product classes—and that is virtual nesting. And when we discuss the concept of virtual nesting, one very important concept that will be required is indexing.

So, we will also introduce indexing—what it means and how indexing will be used in virtual nesting. So, I hope we have already understood the context of network management—in which particular type of situations network management will be applicable—and as we just discussed, it is very much possible to use when you have multi-leg travel. That is, you are traveling and have some kind of changeovers at different airports, and you have a single ticket from the start to the end of your journey. So, in that case, in which particular category the tickets will be booked or not booked. You must have seen on your own that if you are traveling from city A to city C via city B.

Sometime, sometime the ticket window will allow you to book a single ticket for A to B and B to C. A single ticket is possible sometime and most of the ticket booking engines they allow you single booking also. But, sometime in some rare situations, it is also possible that you may not be able to get a single ticket. And in that case manually you have to book a ticket for A to B manually this is auto and sometime you have to book manually ticket for A to B and another ticket for B to C also manually. So, whether and all these things forcing you to go for manually not automatic this is also the design of the optimization algorithms and so that is a very popular case in the network management. Similarly, as we discussed that when you are going to stay in a hotel for multiple nights and during that multiple nights may be a weekend is coming or a peak weekday is coming that will also affect the availability of the resources.

Because, there are vacation hotels in the vacation hotel the demand will go very high during the weekends and that time it will become a case of constrained resource. And in the business hotels, demand may go to a particular peak level during the mid of the weekdays and in that case again it will become a case of constrained demand, when the demand is shooting up beyond the capacity of the hotel. So, if those days are coming in your multiple night stay again it will become a case of network optimization, whether to allow you a single booking for all those nights or you will be charged different rates for different days of booking. So, hotel should see this problem as a network optimization problem and the same applies for the rental cars for multiple days. So, if there are vacation days or you are expecting that demand is going to peak in a particular day.

So, rental car agency may try to get more revenue for those days when they are expecting higher demands. So, all these things are the different types of examples of our network management situations. Now, as in the network management situation, we have a very simple example here, where we are travelling from Mumbai to Delhi that is leg number 1 and Delhi to Roorkee that is leg number 2. Now, here, you have three possibilities, three types of customers will come to you. The customers who are looking to travel from Mumbai to Delhi only.

Customers, so they will use only leg 1. The other type of customers who are traveling between Delhi and Roorkee, so they are using only leg 2. And then, there may be another type of customer who wants to travel from Mumbai to Roorkee via Delhi, so, both legs. It is very obvious, it is very obvious that customers who want to take a complete ticket from Mumbai to Roorkee, they are expecting some kind of economies. So, therefore, you see generally the fare charges if you are traveling between Mumbai and Delhi, this is 200.

And let me also tell you that for all these three legs, all these three routes, there may be different fare classes also. Just to explain this example, we are considering only one fare class in each leg. But, there may be a full fare class, there may be a discounted fare class in each of these legs also. And here, you see that from Mumbai to Delhi, let us say if there is only one fare class, the price is 200 dollars, Delhi to Roorkee the price is 160 dollars but, you are getting a benefit if you are booking a single ticket from Mumbai to Delhi to Roorkee and that is coming at 300. Otherwise, if you buy two separate tickets, the total price will come to 360.

No	Route	Fare
1	Mumbai => Delhi	$200
2	Delhi => Roorkee	$160
3	Mumbai => Delhi => Roorkee	$300

So, as a customer, if I am traveling from Mumbai to Roorkee, I will like to book a single ticket. While, the airline or the railway company will like to take benefit of two individual tickets because, in any case, if I am traveling from Mumbai to Roorkee, I am paying 300. So, the revenue which the company is getting is 300. But, if the company forces me that okay, I am not giving you a 300 rupees ticket, you buy one ticket of 200 and one ticket of 160. So, the revenue expected is 360 and if I buy a single ticket, the revenue is 300.

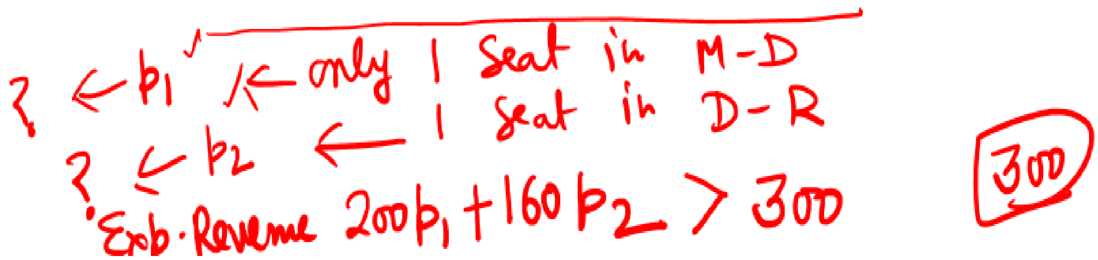
Ultimately the capacity is same there is one flight between Mumbai to Delhi in the same flight whether you are holding a full ticket from Mumbai to Roorkee or you are holding a ticket from Mumbai to Delhi you are sitting in the same flight. Similarly, there is a same flight between Delhi to Roorkee. So, whether you are holding a ticket only between Delhi to Roorkee or a ticket which is started from Mumbai and coming to Roorkee, you both are sitting in the same flight. And in that way, you can consider that these are three different fare classes. These are three different fare classes.

One fare class is 300, another fare class is 200 and another fare class is 160. These are three different fare classes we are having. Now, how the greedy algorithm works in this particular case? Now, with high demand from Mumbai to Delhi and only one seat left for both legs, should the airline accept a booking request for route 3 at 300 dollars? That is the question.

The name greedy algorithm comes because of this only. Now, what it says that there is a customer who is coming to book you a combined ticket from Mumbai to Delhi to Roorkee and at that time the booking control office checks that only one seat is left, only one seat in Mumbai-Delhi sector and only one seat in Delhi-Roorkee sector. Now, this booking control is in a confusion that should I give this seat to this single customer who is looking for Mumbai-Delhi-Roorkee. Now, this single customer is going to give you revenue of 300. So, you can ensure that this 300 revenue comes to me and you book that seat and all the capacity is exhausted, booking is closed or you can say that you can reject the request of this customer.

Because, you are anticipating that there will come another request just for Mumbai Delhi sector and there may come another request for Delhi Roorkee sector. And if you get a possibility to book two passengers for each of these sectors, you will get revenue of 360. So, should I wait for 360 revenue or should I book 300 that is the question and here comes and here comes this greedy algorithm because we are greedy. We will like to maximize our profit and in this case, let us say p_1 is the probability that there may come a customer between Mumbai to Delhi. p_2 is the probability that there may come a customer between Delhi to Roorkee.

So, therefore, at this point when this 300 dollar customer is waiting for my decision what I will calculate that is expected revenue at this stage will be 200 p_1 + 160 p_2, 200 p_1 +160 p_2 this is the expected revenue.



And this expected revenue should be more than 300. If expected revenue **is more than 300 then I will reject the request** of this customer who is looking for a complete sector ticket from Mumbai to Roorkee. Otherwise, if this 200 p_1 + 160 p_2 is **less than 300 then I will accept this request of customer**. So, this is the greedy algorithm.

So, here the entire approach is based on our estimations of these probabilities. What is these probabilities? These estimations are very very important and if these estimations are not up to the mark, it will be difficult to take benefit of this greedy algorithm. So, that is one thing. Now, greedy algorithm is giving you the optimal choice when you have very specific probabilities available with you.

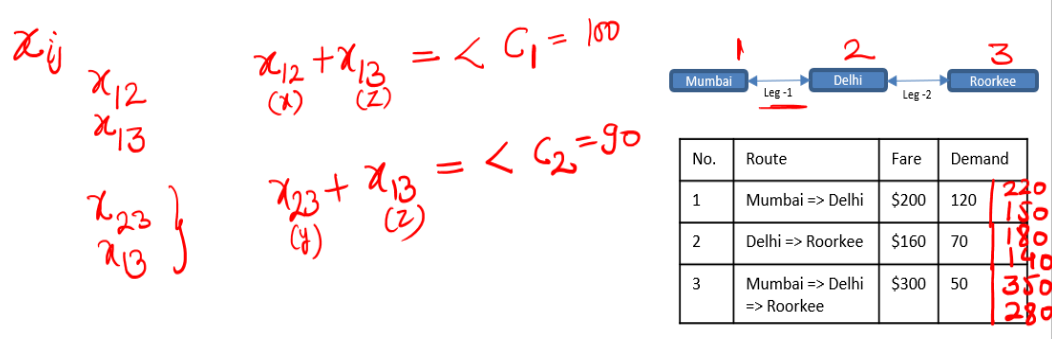
When these p_1, p_2, etc., are known to you or you can compute them regularly, and just for this example, I am considering only one seat is left. But you have to continuously run this greedy algorithm for all these seats whenever a decision is made to accept the request. Your available capacities are reduced, and therefore, your new probabilities will come into the picture: what is the possible demand of this sector, and what is the possible demand of that sector? So, continuously, you keep updating your expected revenue calculations, and that is decided. So now, as we discussed, this greedy algorithm is not a very robust method for determining capacities or deciding the acceptance and rejection of requests. Because if you consider a case where the demand for Route 2 (Delhi-Roorkee) is low and the airline does not get any bookings for that leg, then by rejecting the request, it will make a loss of a hundred dollars by flying with a vacant seat on that leg.

So, whenever even a single sector—even a single route in this network—gives you any kind of issue, then this particular greedy algorithm will be very, very challenging. So, if one particular leg is in a constrained situation, the greedy algorithm is not going to work in that case. Now, to improve the greedy algorithm, where we require probabilities and capacity constraints, these are very important reasons for the failure of greedy algorithms. A slightly improved version is linear programming. Linear programming is considered much better for solving network management problems.

And we hope that all of you are well-versed with LP formulations, where all the equations are linear in approach. And in this case, again taking the same example of the Mumbai-Delhi-Roorkee problem, we have fares and the demand available to them. Now, we consider that the plane flying between Mumbai and Delhi has a capacity of, let us say, 100—let me write it as C1. The capacity of the first plane is 100, and the capacity of the second plane is 90. So now, there are two or three possibilities: what are the chances that customers may start? There are, let us say, x_{ij}. So, customers may start from 1 and go to 2, or customers may start from 1 and go to 3.

This is 1, this is 2 and this is 3. So, customers who are starting from 1 to 2 plus 1 to 3, this should be equal to or less than the total capacity available because, all these customers who are travelling from 1 to 2 or 1 to 3 will be using the flight which is operating between Mumbai to Delhi leg 1. So, this capacity should be less than C1. Similarly, there are customers who will travel from 2 to 3 and customers who are coming from station number 1 and will ultimately going to 3, they all will also use flight 2. So, x_{13} will come here also.

So, $x_{23} + x_{13}$ should be equal to and less than C2 and therefore, based on this and C1 is equals to as I just said 100, C2 is 90. x_{12} is known as x in our this class, x_{23} is known as y, x_{13} is known as Z. So, we have simply summarized all these things like x + z should be less than equals to 100, y + z should be less than equals to 90.



No.	Route	Fare	Demand
1	Mumbai => Delhi	$200	120
2	Delhi => Roorkee	$160	70
3	Mumbai => Delhi => Roorkee	$300	50

And the revenue which we can expect from this particular situation that is, how many customers are going to travel from this sector, from this sector and from this entire sector? So, 200x+160y+300z that is the total revenue which we want to maximize subject to these two constraints plus three more constraints. That demand x, y, z that what we have forecasted this is forecasted demand.

So, x is less than equals to 120, y is less than equals to 70 and z is less than equals to 50. And one more important thing that no allocation x, y, z can be in the negative. So, x, y, z all need to be non-negative. So, with this my LP is formulated. This is LP formulation.

Considering the previous case with demands available for the three routs to be 120, 70 and 50, and the allocations to x, y and z, the objective is to

$$Maximize\ Z = 200x + 160y + 300z$$

And the constraints

$$x + z \leq 100$$

$$y + z \leq 90$$

Similarly, $x, y\ and\ z \leq 120, 70\ and\ 50\ respectively$

Solving the above model using Excel Solver we get the allocations to be 80, 70 and 20 respectively.

No.	Route	Fare	Demand
1	Mumbai => Delhi	$200	120
2	Delhi => Roorkee	$160	70
3	Mumbai => Delhi => Roorkee	$300	50

In fact, as I said that we have taken a very simple case to explain you the formulation of LP. Otherwise, in all the classes, in all the routes, there may be multiple classes. For example, there if I consider a real life case, it can be let us say there is a fare 150, another is 80, there is a fare 120, another is 70. And similarly, there can be a fare which since it is 200, so it will be appropriate to take consider is 250. So, fair maybe let us say 220 and 150, fare maybe 180, 140 and fare maybe 350 and 280.

These may be the different types of fares which are possible. And, you will see you will see that our problem will become even more complex here we took only three variables. But, if we have so many sub classes of the fare in each route we will be having six decision variables. Because, how many seats will go to a full fare or discount fare so generally every airline every, hotel company, every car rental company may have multiple fares in each of these classes. And just to demonstrate you, this problem is formulated and you can use whatever software package you are familiar with for solving such cases, you will find that solution appears to be 80, 70, 20 for this particular case.

Where you have allocations of 80 units for x, 70 units for y and 20 units for z. So, in this way you are using your capacity in the optimal way that 80 seats you are giving for full fare for leg 1. 70 you are giving for leg 2 and there will be 20 customers who will be using the entire network from Mumbai to Roorkee in this particular capacity allocations. So, I hope we are able to understand how the linear programming situation is better than greedy algorithm and linear programming in this particular case is looking very simple but, actually whenever you are applying linear programming, you also know that our capacities keep changing in the dynamic environment. Every time you are booking a particular passenger either in leg 1, leg 2 or for the overall you have to continuously update your forecasted demand that now what is the new forecast and you can run this LP model again and again with that new forecast value. And similarly whenever cancellations are happening you have to add that cancellation into the capacities of all the legs and therefore, generally, we do linear programming in our other courses in a very static manner but, here, we have to do linear programming in a more regular manner so that, you are able to get the best out of this.

And since you are not maintaining only one particular case, you are maintaining a large number of networks at the same time on a particular day. Therefore, it appears that this may become a more complex method if you are supposed to maintain 1000 air networks, and in all those 1000 networks, multiple classes are also present. So, the problem of linear programming will also become a complex problem to solve. So, with this, we come to the end of this particular session, where we discussed greedy algorithms and linear programming approaches for solving network optimization problems. In our next session, we will be discussing virtual nesting and indexing concepts for network optimization.

Thank you very much.