Advanced Operations Research

Prof. G. Srinivasan

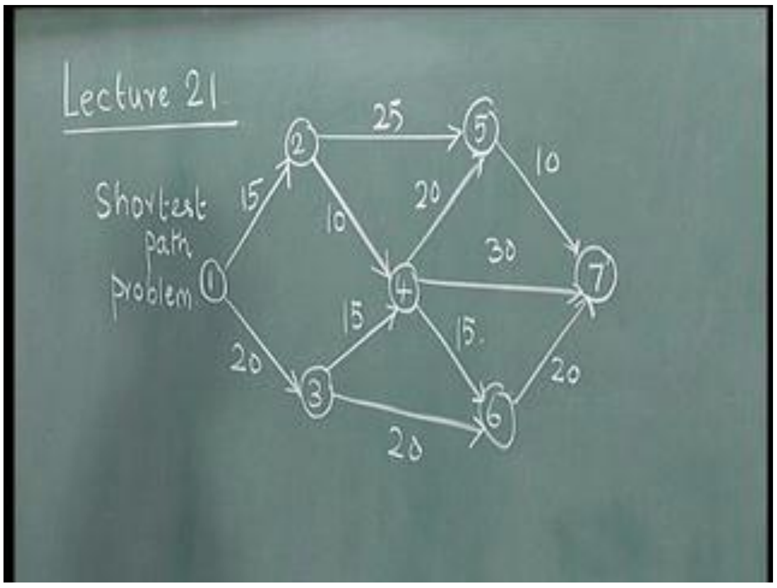
Department of Management Studies

Indian Institute of Technology, Madras

Lecture - 21
Successive Shortest Path Problem

In this lecture, we continue our discussion on the shortest path problem.

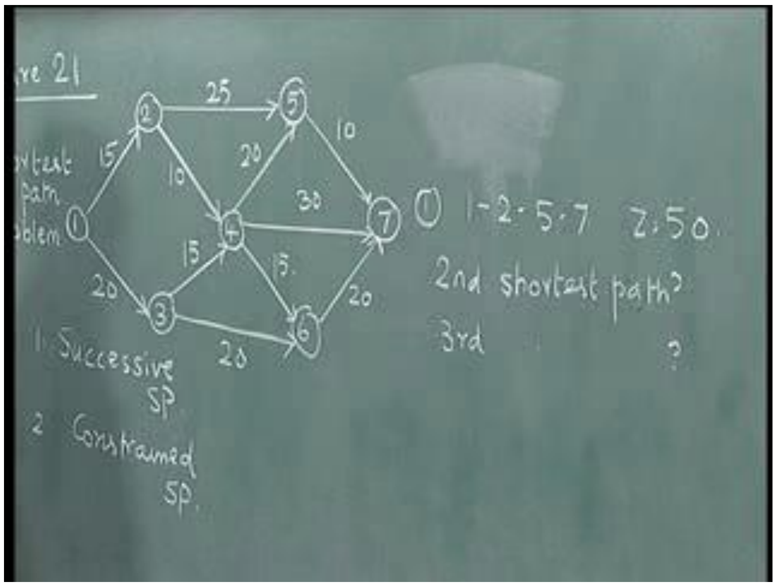
(Refer Slide Time: 00:29)



We have already described the shortest path problem, the Dijkstra's algorithm and the optimality of the Dijkstra's algorithm in earlier lectures. We have also looked at the cases where the C_{ij}s could be even less than 0. Ordinarily, in the Dijkstra's algorithm the C_{ij}s are greater than or equal to 0 as shown in this example. There could be cases where we could have negative costs also. We also saw how the Dijkstra's algorithm will have to be modified to handle negative cause and how the algorithm is capable of identifying a negative length cycle.

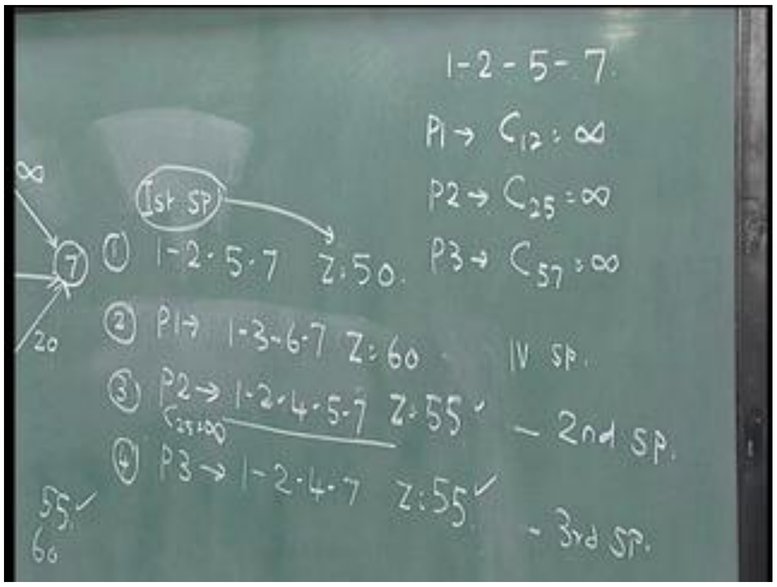
Today, we will look at two more aspects of the shortest path problem. The first is what is called successive shortest path and the second is constrained shortest path first. Let us look at the successive shortest path problem. Let us go back to the same network example which we have been using to understand the shortest path problem.

(Refer Slide Time: 01:50)



We already know for this example that the shortest path is 1 - 2 - 5 - 7 with Z is equal to 50. 1 to 2, 2 to 5, 5 to 7 is the shortest path with 15 plus 25 is equal to 40; 40 plus 10 is equal to 50 as the length of the shortest path. What do we do in the successive shortest path? This is obviously the shortest path or called as a first shortest path. In this example, we have only a unique shortest path. There is only one first shortest path. The question is what is the second shortest path? Is there a second shortest path, if so, what is the second shortest path? After we answer the question, what is the third shortest path and so on?

(Refer Slide Time: 02:57)



1-2-5-7.

P1 → $C_{12} : \infty$

P2 → $C_{25} : \infty$

P3 → $C_{57} : \infty$

1st SP

① 1-2-5-7 Z: 50.

② P1 → 1-3-6-7 Z: 60 IV SP.

③ P2 → 1-2-4-5-7 Z: 55 ✓ – 2nd SP.

$C_{25}: \infty$

④ P3 → 1-2-4-7 Z: 55 ✓ – 3rd SP.

55 ✓

60

Let us try to find out the second shortest path in this network. In order to find out the second shortest path in this network what do we do is, we take the first shortest path which is 1 - 2 - 5 - 7 and we observe that there are three edges in this shortest path, 1 to 2, 2 to 5 and 5 to 7. We solve three shortest path problems. We may be call problem P1 with C_{12} equal to infinity, that is, if we take out the edge 1 2 or if we assume that the edge weight becomes infinity from 15. Then, we are effectively, excluding 1 2 from the shortest path. We would solve problem P1 by excluding this from the shortest path or indirectly putting the value equal to infinity so that this does not lie in the solution.

The second problem will be P2 where we keep this as 15 and then we put C_{25} is equal to infinity. Then, we take the third case, where we look at P3 where we put C_{57} is equal to infinity. We have to solve three shortest path problems. If we look at the case where C_{12} is equal to infinity, which means we change this to infinity and apply one pass of the Dijkstra's algorithm. For the problem P1, we will get the solution as 1 - 3 - 6 - 7 with Z is equal to 60. That is, if we make this as infinity from 15, then the shortest path will now involve 1 to 3, 3 to 6, 6 to 7 with Z is equal to 60.

We now solve problem P2 by keeping this as 15 and by changing this to infinity. So that the solution to problem P2 or the shortest path to this problem, where this is kept as 15. This becomes infinity, is now given by 1 - 2 - 4 - 5 - 7 with Z is equal to 55. So, 1 to 2, 2 to 4, 4 to 5 and 5 to 7 with Z is equal to 55. You observe that in this problem P2, we have forced this value C_{25} to infinity, which is given here. Therefore, the edge or arc 2 5 does not lie in the solution.

We now solve the third problem, which is P3 that we keep here. For P3 what do we do is, we retain this to its original value of 25 and we change this from 10 to infinity; so that, we do not have this arc in the shortest path. We have excluded this from the shortest path, so we solve this new shortest path problem again, where these two values regain their original values and this is changed from the original value of 10 to infinity. Now, one more pass of the Dijkstra's algorithm would give us the solution 1 to 2, 2 to 4, 4 to 7 with Z is equal to 55. What we have done is this is the first shortest path 1 - 2 - 5 - 7 with Z is equal to 50. In order to get the second shortest path, we have created three more problems because there are three arcs in this shortest path and each of them we restricted to infinity. We take it out of the shortest path and apply one pass of the Dijkstra's algorithm. We have got P1, P2 and P3. Out of these three, the one that has the smallest distance is the second shortest path. In our example either this or this can be shown as the second shortest path. We may now say that this is the second shortest path. We could have also said this as the second shortest path. We have now found out the second shortest path on this network.

How do we find the third shortest path? In order to find the third shortest path, particularly for this example, since at this point we have two solutions with Z is equal to 55. This automatically qualifies to be the third shortest path, where the third shortest path and the second shortest path, at this instance, have the same value of 55. We may say that this is the third shortest path with 55. We may also say, for example, that we have two second shortest paths, both with 55. Therefore, we would be interested in third shortest path which has a value strictly greater than 55. Once again, in this instance, we have a solution with 60 which happens to be one out of the three. The way these numbers are given in this network, the minimum number being 10, this would automatically qualify to be the third shortest path.

If we wish to say that the third shortest path should have a value distinctly higher than that of the second shortest path, we qualify this to be the third shortest path with the same value of 55, then this automatically becomes the forth shortest path with value equal to 60. One should bear in mind that the three sub problems that we created only in this example have automatically qualified to become second, third and fourth shortest paths, respectively. On the other hand, the methodology would actually be slightly different from what I have described. Whatever I have said is clearly applicable for this particular instance or illustration of the problem.

In general, the algorithm would run like this. Take the first shortest path. Find out the number of edges in the first shortest path; in this example, there are three. Create three problems as shown here P1, P2 and P3 with the respective cost values set to infinity. Get the solutions for P1, P2 and P3. The smallest among them will automatically qualify to be the second shortest path; keep the other two in the solution; retain the solution which is 1 - 2 - 4 - 7 with 55 and retain 1 - 3 - 6 - 7 with 60.

In order to find the third shortest path, take the second shortest path. There are four arcs in the second shortest path.; set each one of them to infinity. Remember that this has already has C_{25}set to infinity because this comes from this path. Now, we will find out four more shortest paths where the first one will have C_{25}equal to infinity and C_{12} equal to infinity; second one will have C_{25} equal to infinity, C_{24} equal to infinity and so on.

There will be plus four solutions coming out of these four arcs that we have looked at. Effectively, there will be six solutions, some of which may even repeat. Out of these six solutions, find out that solution which has the least value? In this case it will turn out to be the 55. This will become the third shortest path and so on. Like this, we can get the second, third, fourth or any successive k^{th} shortest path for a given problem.

The only thing that we have to bear in mind is, as we move along, for example, when we did the first shortest path we had one pass of the Dijkstra's algorithm. When we did the second shortest path, we have as many passes as the number of edges that are here. Therefore, with every successive shortest path that we wish to compute, we end up computing more and more of shortest paths using the Dijkstra's algorithm.

The point that I am trying to make here is that the complexity of the algorithm will increase, if we wish to use this algorithm to find out successive shortest paths. Nevertheless, this provides us with a certain framework with which we can get or identify successive shortest paths in the network.

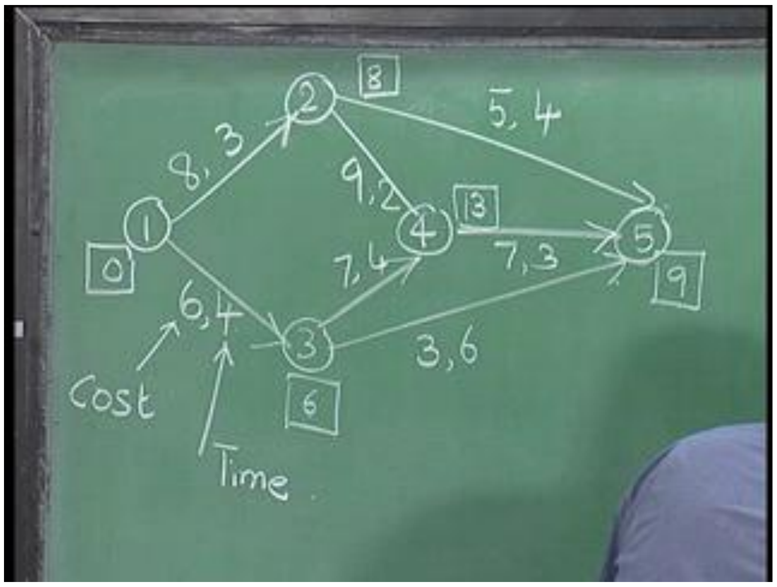
Now, we get to the other question of why should we study successive shortest paths in the network? The answer to that comes from something like this. So far in the shortest path problems that we have looked at, every arc has one associated weight which I described as either a cost of moving from i to j or the distance to move from i to j or the time taken to move from i to j. There can be situations and scenarios where we may define this with two parameters. It could mean the cost; it could mean the time. So, if each of these is now described on the basis of two parameters, which would be cost and time then the question is if we find out the least cost shortest path, can we find out the least time shortest path?

In some situations, we will have where we want the least cost shortest path subject to a condition that the total time taken should be less than or equal to something. In which case, the problem becomes a constrained shortest path problem where we do a cost minimization objective with an explicit constraint that the time taken should be less than or equal to something.

We will look at this problem. We will also understand that this problem is actually difficult to solve, but one of the reasons why successive shortest path is useful is that. If it turns out that the first shortest path violates the time constraint while the second shortest satisfies the time constraint, then we might say that the second shortest path is actually optimal to the constrained problem.

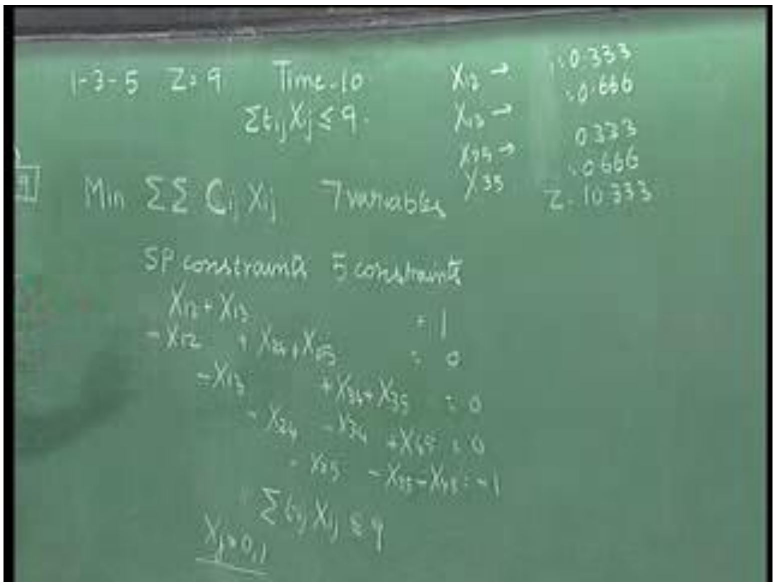
Successive shortest path is one of the ways of trying to solve a constrained shortest path problem, particularly, when we have more than one parameter that is describing the arc weight. The other reason why we do successive shortest paths is also to get a feel of perhaps the opportunity loss, if we do not choose the first shortest path and instead choose the second shortest path. We move from successive shortest path to studying the constrained shortest path and we take a different example to explain the constrained shortest path.

(Refer Slide Time: 15:30)



Let us look at this example and let us try to first find out the shortest path, assuming that each of these is a cost of moving from i to j. So, one pass of the Dijkstra’s algorithm would give us a 0 here, this would be 8, this would be 6, 6 plus 7 is equal to 13, 8 plus 9 is equal to 17, we will have 13; we have 8 plus 5 is equal to 13, 13 plus 7 is equal to 20, 6 plus 3 is equal to 9. One pass of the Dijkstra’s algorithm would give us the shortest path which is 1 to 3, 3 to 5 with Z equal to 9. Let us add one more parameter to this and let us say this as (8, 3), (6, 4), (9, 2), (7, 4), (3, 6), (7, 3) and (5, 4). Let us say that the second one that we have added actually represents time to move from i to j. Let us say this is the cost and let us say this is the time to move from i to j. Let us also have a restriction and first find out the time associated with the shortest path. The shortest path is 1 to 3, 3 to 5 and right now we have the time associated is equal to 10.

(Refer Slide Time: 17:40)



1-3-5 $Z = 9$ Time - 10

$\sum t_{ij} X_j \leq 9$

$X_{12} \rightarrow$ 0.333

$X_{13} \rightarrow$ 0.666

$X_{24} \rightarrow$ 0.333

X_{35} 0.666

$Z = 10.333$

Min $\sum\sum C_{ij} X_{ij}$ 7 variables

SP constraints 5 constraints

$$\begin{aligned}
&X_{12} + X_{13} = 1\\
&-X_{12} + X_{24} + X_{25} = 0\\
&-X_{13} + X_{34} + X_{35} = 0\\
&-X_{24} - X_{34} + X_{45} = 0\\
&-X_{25} - X_{35} - X_{45} = -1\\
&\sum t_{ij} X_{ij} \leq 9\\
&X_j = 0, 1
\end{aligned}$$

If we have a constraint that this time should be less than or equal to 9, then what happens to the problem? What happens is clearly the shortest path 1 to 3 and 3 to 5 is now infeasible. If we add an additional constraint that total time t taken which is t_{ij} X_{ij} should be less than or equal to 9. How do we solve this problem? This problem becomes a constrained shortest path problem. Let us first formulate the constrained shortest path problem and then try to see a way by which we solve this.

The constrained shortest path problem will look like this now. This will be to minimize sigma $C_{ij}X_{ij}$, subject to the shortest path constraints. Remember that we will have five constraints corresponding to these five nodes and seven variables. There are seven variables and five constraints and these five constraints will be: X_{12} plus X_{13} is equal to 1; minus X_{12} plus X_{24} plus X_{25} is equal to 0; minus X_{13} plus X_{34} plus X_{35} is equal to 0; minus X_{24} plus minus X_{34} plus X_{45} is equal to 0; minus X_{25} minus X_{35} minus X_{45} is equal to minus 1. We have this constraint sigma $t_{ij}X_{ij}$ less than or equal to 9, where X_i or X_j is equal to 0 1. This is the formulation of the constrained shortest path problem corresponding to this example.

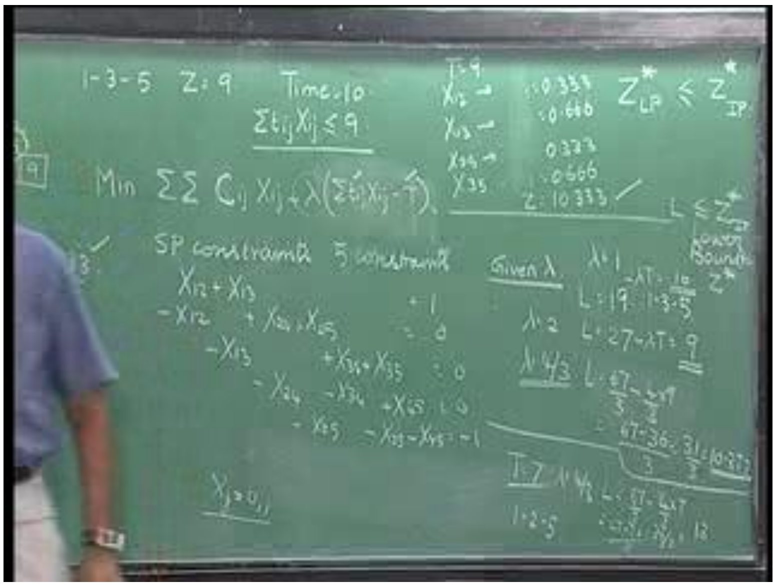
X_j is now a binary variable, so this problem becomes a 0 1 integer programming problem. Earlier, we have seen that the unconstrained shortest path problem, that is, if we leave out this constraint, it becomes a normal shortest path problem. We have already shown that the normal shortest path problem has a unimodular matrix and therefore, we can ignore this 0 1; solve as a linear programming problem and get solutions that have 0 value or 1 value for the variable. First of all, we need to verify whether the unimodularity property is retained with the addition of the constraint. In order to do that, let us first solve this as a linear programming problem and then see how the solution looks like.

Right now, I am not going to take you through the simplex iterations, but I am just going to give you the solution corresponding to this linear programming problem. That solution turns out to be X_1 is equal to 1 by 3 or 0.333, X_2 is equal to 0.666, X_5 is equal to 0.333, X_7 is equal to 0.666 and Z is equal to 10.333.

X_1 is nothing but X_{12} for our example and X_2 is nothing but X_{13} in our example, X_5 is X_{25}, X_7 is X_{35}. The solution to the linear programming problem is actually arc X_{12} is equal to 0.333, this is 0.666, this is 0.666 and this is 0.333. Total is equal to, you can see that it satisfies all these constraints plus is satisfies this constraint, it gives us Z is equal to 10.333. Clearly with the addition of this constraint this problem has lost its unimodularity characteristic. If it had not lost its unimodularity property, then it would have given us a solution which is 0 1 value which it did not do. The addition of this constraint takes away the unimodularity property. Therefore, it is now not possible to solve this using linear programming or linear programming based technique or a direct implementation of the Dijkstra's algorithm. We need to do something more.

What do we do in this case is something quite interesting.

(Refer Slide Time: 23:00)


1-3-5 Z: 9 Time: 10
$\sum t_{ij} X_{ij} \le 9$
Min $\sum\sum C_{ij} X_{ij} + \lambda(\sum t_{ij} X_{ij} - T)$
SP constraints 5 constraints
$Z^*_{LP} \le Z^*_{IP}$
$L \le Z^*_{IP}$
Lower Bound Z^*
Given λ
Z: 10.333
$X_{ij} = 0, 1$

We have already learnt earlier the method of Lagrangean multipliers where we could take a constraint into the objective function; then, do the optimization after we dualize this constraint or take this constraint into the objective function. We are going to try and apply that technique here. What we do is we are going to introduce a Lagrangean multiplier called lambda, which is a Lagrangean multiplier. We take this into the objective function. So this will become plus lambda times sigma $t_{ij}X_{ij}$ minus T.

We remove this and take this into the objective function by using a Lagrangean multiplier, where lambda is the Lagrangean multiplier. $t_{ij}X_{ij}$ is this constraint, t is equal to 9, in this example. We have said that the right - hand side value should be less than or equal to 9. We have got this into the objective functions. Now, if we look at the constraints, we go back to the normal shortest path constrains and therefore, this path is unimodular. The only difficulty is that we know this T, we know all these t_{ij}s, but we do not know this lambda. Lambda is another variable that we have introduced into the problem. It is like a dual variable associated with this constraint that we actually relaxed. This method is called Lagrangean relaxation and we describe it further.

What do we do is that, for given values of lambda, it is now possible to solve a new shortest path problem every time. For example, if we fix lambda equal to 1, then what happens to this? This becomes C_{ij} plus lambda t_{ij}; this becomes 8 plus 3 which is 11. I am just showing it in a circle, this becomes 5 plus 4 which is 9; this becomes 9 plus 2 is equal to 11 because this is C_{ij} plus lambda t_{ij}. C_{ij} is 8, lambda is 1, t_{ij}, is 3 so it becomes 8 plus 3 which is 11; 9 plus 2 is equal to 11, 6 plus 4 is equal to 10; 7 plus 4 is equal to 11; 3 plus 6 is equal to 9 and 7 plus 3 which is 10.

We now solve this or lambda equal to 1, we would get a shortest path, we would get a solution with Z or it is called L is equal to 19. Now, what happens? If we solve this for lambda equal to 1, you get this path is 11 plus 9 is equal to 20. This path is 10 plus 9 is equal to 19 so this path will come into the solution. 1 to 3, 3 to 5 with L is equal to 19, L is the value of the objective function. We need to do one more thing which is what we have got is 19, is the value of $C_{ij}X_{ij}$ plus lambda times $t_{ij}X_{ij}$, but then the full objective function is minus lambda T. We have to subtract this T from 19. So, this becomes 19 minus lambda T, which will b 10. Lambda is 1, capital T is 9; so, 19 minus 9 which is 10. When we put lambda equal to 1 and solve, the value L will become 19 minus lambda T, which is equal to 10. If you also see carefully the value here is 10.333, while the value here is 10.

Let us try and solve for some values of lambda. When we get lambda equal to 2, we got a solution with L is equal to 27 minus lambda T, which is 27 minus 18 is equal to 9. When lambda equal to 2, it means this value would become 8 plus 6 is equal to 14 and so on. When lambda equal to 2, we effectively got 27, but 27 minus lambda T is 27 minus 18 which is 9. What we can do is, for various lambdas, we can actually try and get some value for this L, which is the value minus lambda T. It is possible to show that for every value of lambda, this value that we get here is actually a lower bound to Z star, where the Z star is the objective function value of the constrained shortest path problem.

This is the LP relaxation of the constrained shortest path problem. So, this is Z_{LP} star is an LP relaxation of the constrained shortest path problem. We also know that this is less than or equal to Z star, where Z star is the Z_{IP} star where the constrained shortest path problem is an integer programming problem. This is also less than or equal to Z_{IP} star. Anyone of these L or effectively, L minus lambda T is less than or equal to Z_{IP} star.

Whatever we observe here by solving these shortest path problems for various values of lambda every one of them is actually a lower bound to Z_{IP} star. Automatically what we would like to do is to see, what is the highest value that we can get here, so that we can make the lower bound better and better? Though, we can do this for various values of lambda and show whichever is the highest, let us just try for lambda equal to 4 by 3. We would actually get L is equal to 10.333, which is 67 by 3 minus 4 by 3 into 9; this is 67 minus 36 by 3, which is 31 by 3, which is 10.333.

For lambda equal to 4 by 3, we would get a value which is 10.333. We tried with various lambdas but then it is possible to show that you cannot go beyond 10.333 in this example. If we look at this, when we put an additional constrained that $t_{ij}X_{ij}$ is less than or equal to 9 in this problem.

Right now, let me remove all these, these are specific values for lambda equal to 1. For this, if we put an additional restriction that $t_{ij}X_{ij}$ is less than or equal to 9, then we know that there is a solution with 1 - 2 - 5 with Z equal to 13 and time equal to 7. This is the optimal solution to the problem. We can now realise that every one of these values is a lower bound to this 13. The 10.333 that we got here is also a lower bound to this 13. For every lambda that we can try here we can actually get the lower bound to this 13. We wish to find out the lambda which would try to get as large a value here, so that this lower bound is tighter and close to this 13.

We also know that LP relaxation is a lower bound, so 10.333 is also a lower bound to this 13. Why are we using this now?

We are using this method for two reasons: one is we wish to know, is there a possibility that the lower bound obtained using these relaxations is actually better than 10.333? If so, then this method can give us a better lower bound than what LP relaxation can give. There are situations where Lagrangean relaxation can give a better lower bound than LP relaxation, but for the example that we have looked at which is a single constrained shortest path problem.

Whenever you have a single constrained problem and you relax it, it is always possible to show that the Lagrangean lower bound will only be as good as the LP lower bound. In this case it is not surprising that both the LP lower bound and the Lagrangean lower bound gave us 10.333. In fact, there is very interesting way to get the corresponding lambda. Actually, we can even show that if we solve the LP problem and we realise that $t_{ij}X_{ij}$ is exactly equal to 9, the dual variable corresponding to this constraint would have a value 4 by 3, which is exactly the value of lambda for which we get the best value of the lower bound.

This is also fairly clear because we said we took this constraint. We dualized this constraint and took it into the objective function. There is something to do with the dual value associated with this constraint. For a single constrained shortest path, one can only go as far as saying that the Lagrangean relaxation will only give a bound which is as good as the LP relaxation. Nevertheless, for other problems which are available in the literature, one can show that Lagrangean relaxation can give better bounds than LP relaxation. Therefore, this technique is used to get better bounds than LP relaxation.

The next question that we might ask is you know each one of these implies or involves solving a shortest path problem. The question is when I keep solving so many shortest path problems, how do I get the best value of lambda? One of which I said is this. We solved the LP, you know the dual and then you use it you will get this. If I do this and I know that by putting the dual I can only get up to 10.333, then the question that comes is why should we follow that approach? Is there a more structured approach to get the best value of lambda? There are approaches, one of which is called subgradient optimization which is used, extensively. We are not going to cover subgradient optimization in this lecture series. Right now, we would say that we would try and get the best value of lambda either by trial and error or by a structured procedure such as subgradient optimization. We also know that because it is a single constrained shortest path, the Lagrangean relaxation can only be as good as the LP relaxation.

As I mentioned, there are other problems where the Lagrangean relaxation can give a better lower bound than the LP relaxation. That leads us to the next question; can the Lagrangean relaxation give us the optimal solution? There are times Lagrangean relaxation can give the optimal solution though it is not guaranteed that for every instance, it is capable of giving the optimal solution.

For example, even when we put T is equal to 9, we realised that Lagrangean relaxation is only going up to 10.333 and it is not giving us 13. In this instance, this method is only able to give us a lower bound, but on the other hand, if we have put T is equal to 7 instead of T is equal to 19, which means we have changed this constraint to 7. If we did that, then, for lambda equal to 4 by 3, you realise that L is actually 67 by 3 minus 4 by 3 into 7, which is 67 minus 28 by 3, which is 39 by 3, which is equal to 13. When capital T is equal to 7 and if we have solved for lambda equal to 4 by 3, we would have got L is equal to 13. This is a separate case.

We also know from here that for the value of L is equal to 13, we have a feasible solution with Z is equal to 13. We would have got the same 1 to 2, 2 to 5 here as the shortest path, with Z is equal to 13. Here, you have a situation where the Lagrangean lower bound is 13, there is a feasible solution to the problem with 13 and therefore this 13 is optimal. There are situations where Lagrangean relaxation technique can give the optimal solution, but most of the situations it ends up giving a lower bound.

For many problems, it is possible to show that the lower bound obtained using the Lagrangean relaxation would be better than the lower bound obtained using the LP relaxation, though, for the particular example that we have taken, the LP relaxation and the Lagrangean relaxation give us the same bound. This brings us to the end of the discussion on the constrained shortest path problem.

What we have seen today is, we have first seen the successive shortest path problem and then we have seen the constrained shortest path problem. With this background, let us approach the successive shortest path. One of the ways of doing it is to go back to this; find out the first shortest path, then find out the second and third successive shortest path; the first of those which actually satisfies that constraint will be the optimal solution. It brings us the relationship between the successive shortest path and the constrained shortest path problems. I am not trying to suggest here that a single constrained problem should be solved using successive shortest path; it is not so, but then, if we do a successive shortest path on this problem and apply the additional constraint to each one of these paths, it is also possible to get to the optimal solution to the constrained shortest path.

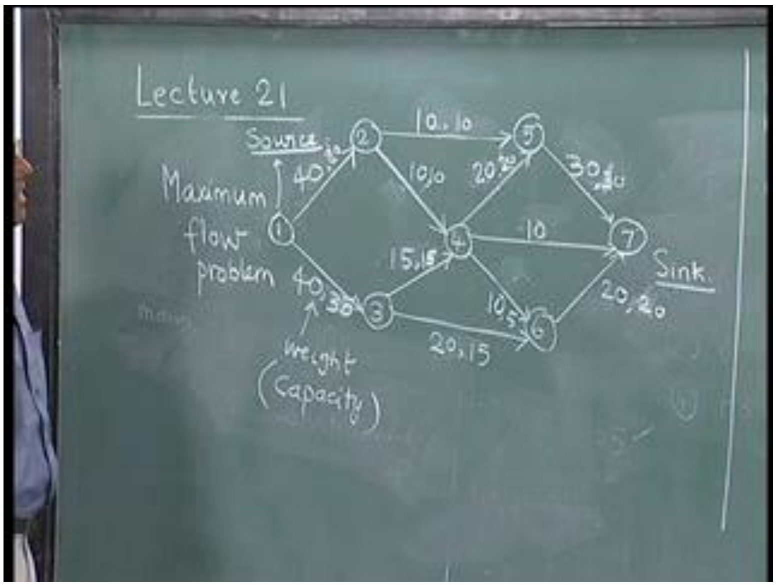
The thing that we understand is both successive shortest path and constrained shortest paths are computationally more intensive than finding the first shortest path. Successive shortest path involves computing many shortest paths. We do not know how many. That number can be very large, but we may end up computing many shortest paths. Therefore, this would be a computationally intensive algorithm. We had already seen that in first shortest path, we got one; to get the second shortest path we calculated three more; in principal, to get the third shortest path, we have to calculate four more and so on; so the number goes up. Similarly, with the constrained shortest path what we did was, we saw that the constrained shortest path problem immediately loses its unimodularity characteristic. Therefore, it has to be solved as an IP - Integer Programming, than as a linear programming problem. One of the methods we saw was to dualize the constraint such that the resultant constraint retains its unimodularity characteristics. Once you dualize it, you introduce another variable which is the Lagrangean multiplier; and then for different values of this multiplier lambda, we try to get the solution.

We also saw that in many instances, this can only give a lower bound and not the optimal solution and many times such a lower bound could be better than the lower bound obtained by LP relaxation. Though, in some very simple instances Lagrangean relaxation is capable of showing the optimum or identifying the optimum for the T is equal to 7 case.

This brings us to the end of the discussion on the shortest path problem. So far, what we have seen is the basic introduction to the shortest path problem; solving the shortest path using the Dijkstra's algorithm understanding the Dijkstra's algorithm is optimal by means of an LP formulation dual complimentary slackness. Then, modifying the Dijkstra's algorithm or looking at new algorithms particularly when the C_{ij}s were or could be negative, which means we looked at shortest path with arbitrary costs. Then, we also showed how a negative cycle can be identified if there is one. We moved into two other problems in shortest path, which are successive shortest path problems and constrained shortest path problems.

Now, we move to the next of the network models which is the maximum flow problem.

(Refer Slide Time: 41:40)



Let us describe the maximum flow problem where the maximum flow problem is like this: Given a network like this, it has seven nodes; it has eleven arcs; each network has a weight. This weight represents, what is called, the capacity of this network. We are going to assume that there is a source and there is a destination called the sink in the maximum flow problem terminology. This is called a sink and this is called the source. We are going to assume that a lot of material or fluid or liquid is available in the source and it has to be pushed to the sink. If we try to transport or send this from the source to the sink through this network, each of these edges or arcs represents a certain capacity.

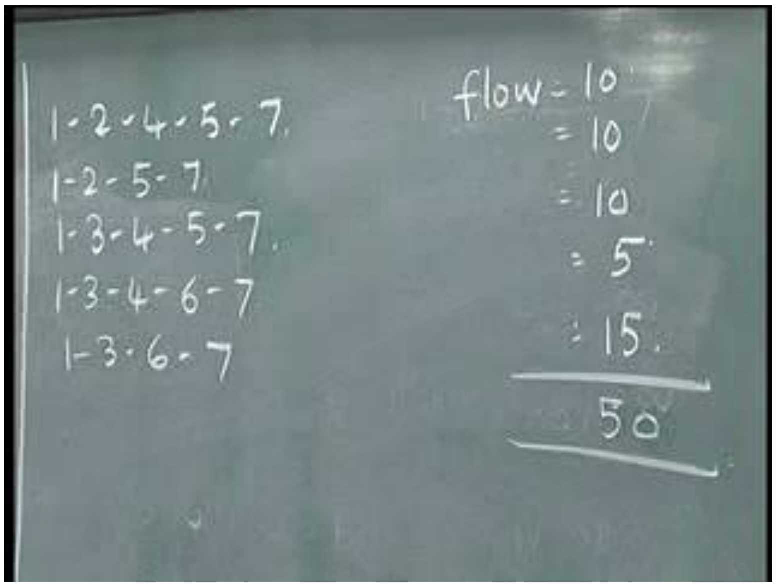
For example, if we want to send something in this 1 to 3, we realise that you cannot send more than 40 units from 1 to 3. It is like the quantity that it can handle, it is like a discharge.

Similarly, if you choose to use 1 to 2, then it cannot handle more than 40. So, given this network with weights, please remember that these weights are not costs. In the shortest path problem, they represented costs; in the maximum flow problem, the weight would represent the capacity. Given a network with the arcs, with the arc capacities, the problem is what is the maximum flow that I can have from the source to the sink? Remember that a large amount is available here or an infinite amount of whatever we are trying to push is available here.

What is the maximum flow in this network?

Let us first apply a very rudimentary algorithm to try and understand the maximum flow and the idea of the maximum flow. If we want to send something from 1 to 7, one of the easiest things to do is to identify a path and then try to send something in that path. There are several paths from 1 to 7. For example 1 to 2, 2 to 5, 5 to 7 is a path; 1 to 2, 2 to 4, 4 to 5, 5 to 7 is a path; 1 to 3, 3 to 4, 4 to 5, 5 to 7 is another path. We could identify a path and then try and send something through that path. There are several paths and we could pick the paths in any order we wish.. Let us try an arbitrary order of choice of paths and see how much we can send through this.

(Refer Slide Time: 45:35)


1-2-4-5-7 flow = 10
1-2-5-7 = 10
1-3-4-5-7 = 10
1-3-4-6-7 = 5
1-3-6-7 = 15
50

Let us first try and look at a path 1 - 2 - 4 - 5 - 7, 1 to 2, 2 to 4, 4 to 5 and 5 to 7. This has a capacity of 40, this has 10, this has 20 and this has 30. What we can send through that path is the minimum of the available capacities. There is a 40 capacity available here, there is 10, there is 20, there is 30, so, 1 to 2, 2 to 4, 4 to 5, 5 to 7. We will be able to send the minimum of them.

I would say flow equal to 10. I am going to put a 10 here saying that I have sent a flow of 10 in this. There is a flow of 10; from here it goes like this and it reaches 7. Let us look at another path that we have and let us look at 1 - 2 - 5 - 7. One other thing that we ensure is that whenever we take a path, we try to exhaust the path. For example, when we took 1 to 2, 2 to 4, 4 to 5, 5 to 7, the maximum that we could send was the minimum of the capacities which turned out to be 10 and therefore, we decided to send 10. We did not send 9 or 8 or anything like that. Whatever maximum flow that is possible we try to send that flow which is equal to 10. If we look at 1 to 2, 2 to 5, 5 to 7, 1 to 2, the actual capacity was 40. We already used 10, so available capacity is 30. Here, the actual capacity is 10, available the capacity is also 10.

Here the actual capacity is 30, the available capacity is 20. We look at 30, 10 and 20. The minimum is 10, so we put flow equal to 10. This gets updated to 20, this gets updated to 10 and this also gets updated to 20.

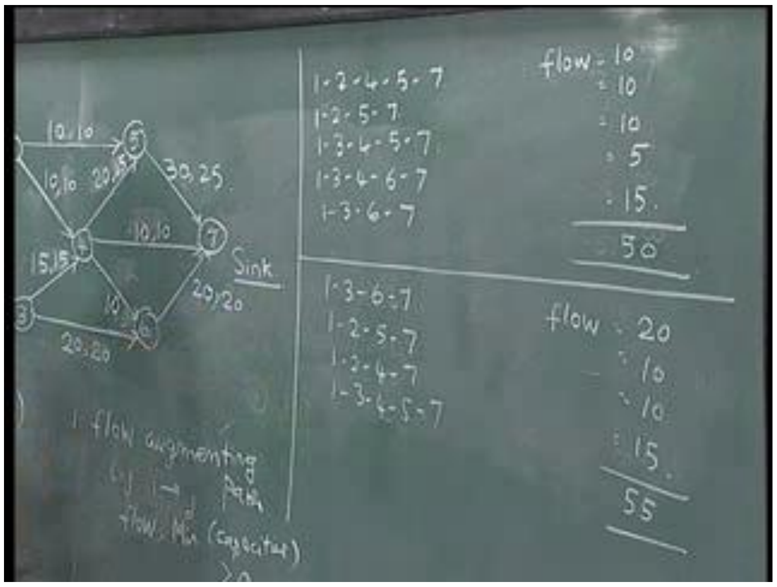
Let us look at the third, 1 to 2, 2 to 4 and 4 to 7. 1 to 2, the available capacity is 20; 2 to 4, there is no available capacity, this has been exhausted; 4 to 7, available capacity is 10. We will be not able to send anything using 1 to 2, 2 to 4, and 4 to 7. We then look at 1 - 2 - 4 - 6 - 7. Once again you realise that 1 to 2, available capacity is 20, here it is 0, here it is 10, and here it is 20. We will not be able to send much, so we leave out this.

Then we move to 1 to 3, 3 to 4, 4 to 5 and 5 to 7. 1 to 3, 40; 3 to 4, 15; 4 to 5, 10; 5 to 7, 10 because 20 minus 10 is equal to 10 is available, 30 minus 20 is equal to 10 available, so 40 15 10 and 10. We get 10 as the flow. We now update the flow. This becomes 10, this becomes 10, this becomes 20 and this becomes 30.Now, we look at 1 - 3 - 4 - 6 - 7. The available capacity is 30, it is 5, it is 10, and it is 20. The minimum is 5, so you have flow equal to 5. We update it so you get 15, you get 15, you get 5, and you get 5.

The last one that we will look at is 1 - 3 - 4 - 7, which is also available, but this is 25, this is 0, this is 10. So, we cannot send anything through 1 - 3 - 4 - 7.

We look at 1 - 3 - 6 - 7. This gives us a balance of 15, this is 20 and this is 15. So, we have another 15 that can be sent. This is 40 minus 15 is equal to 25 is possible, 20 is possible, 20 minus 5 is equal to 15 is possible. We have sent another 15, so this gets updated to 30, this gets updated to 15 and this gets updated to 20. Let us see if we have any more paths with which we can send so 1 - 2 - 5 - 7; this is exhausted so you cannot send. 1 - 2 - 4 - 5 - 7 is exhausted, 1 - 2 - 4 - 7 is exhausted, 1 - 2 - 4 - 6 - 7 is also exhausted, 1 - 3 - 4 - 5 - 7 is exhausted, 1 - 3 - 4 - 7 is exhausted, 1 - 3 - 4 - 6 - 7 is exhausted and 1 - 3 - 6 - 7 is also exhausted. We now realise that we have a total flow of 35 plus 15 is equal to 50 that we are able to send through this particular order or choice of the paths. Let us try doing this with a different configuration or order of these.

(Refer Slide Time: 52:00)



Let us find out a different order in which we send. Let us first look at 1 to 3, 3 to 6 and 6 to 7. This is 40, this is 20 and this is 20. So, we can send flow equal to 20. We update it with 20, 20 and 20. Let us look at 1 - 2 - 5 - 7. This is 40, this is 10 and this is 30. So, we get flow equal to 10; this becomes 10; this becomes 10 and this also becomes 10. Let us look at 1 - 2 - 4 - 7; this is 30, this is 10 and this is 10. So, this becomes 20, this becomes 10 and this becomes 10. If we look at 1 - 3 - 4 - 5 - 7, this is 20, this is 15, this is 20, another 20. The minimum is 15 here, so, we will be able to do 15, this becomes 35, this becomes 15, this becomes 15 and this becomes 25.

Let us see whether we have exhausted all the paths. 1 - 2 - 5 - 7 is exhausted; 1 2 4 5 7 is exhausted; 1 - 2 - 4 - 7 is exhausted; 1 - 2 - 4 - 6 - 7 is exhausted because this is complete; 1 - 4 - 5 - 7 is exhausted here; 1 - 3 - 4 - 7 is exhausted; 1 - 3 - 6 - 7 is exhausted; 1 - 3 - 4 - 6 - 7 is also exhausted. We have exhausted all of them but we realise that we could have flow equal to 55 units in this case.

What have we tried to learn is, one, we said that it is possible to increase the flow by identifying a path and if there is capacity in that path then we can increase the flow. What we also did consistently was that, out of all the capacities that are available, we picked the maximum capacity or maximum flow possible which is a minimum of the capacities that are available. Then we exhaustively looked at all these paths and tried to increase the flow and we tried to do that in two ways.

The first time we picked a certain order of these flows and then we got the flow of 50. Then we picked another order of these paths and then we got a solution equal to 55. Two things we actually did; one, in both these instances, we took a path and we increased the total flow or flow in that path, if it is possible to increase. Also, remember that there were times that we said, in certain paths we cannot increase the flow because that path has at least one link which is completely utilised or exhausted. Only when we could increase the flow in a path, we did it. So effectively, we defined something called a flow augmenting path.

If there is a path and if that path is made up of arcs, for example, if we consider 1 - 2 - 5 - 7 on the network, without these numbers we said 1 - 2 - 5 - 7 is a path, because we have 1 to 2, 2 to 5 and 5 to 7. It is made up of i, j, such that there is an arc from i to j and there is residual capacity that is available. If we leave out this 20, 10 and 25, we said the capacity that is available is 40, 10 and 30; we took the minimum. We defined a flow augmenting path as a path comprising of forward arcs where there is an arc from i to j; we could move from source to destination or source to sink.

We defined a flow augmenting path; then, we said we found out the maximum capacity that is available in each one of these. If all of them are positive, the flow can be augmented. The flow was equal to minimum of the capacities such that the capacities are strictly positive or greater than 0. In each of these methods we identified flow augmenting paths based on this condition. The second one that we observed was that different choice of paths gave us different solutions. Is there a structure by which we can get the correct order of these paths? That we will see in the next lecture.