Linear Programming and its Extensions

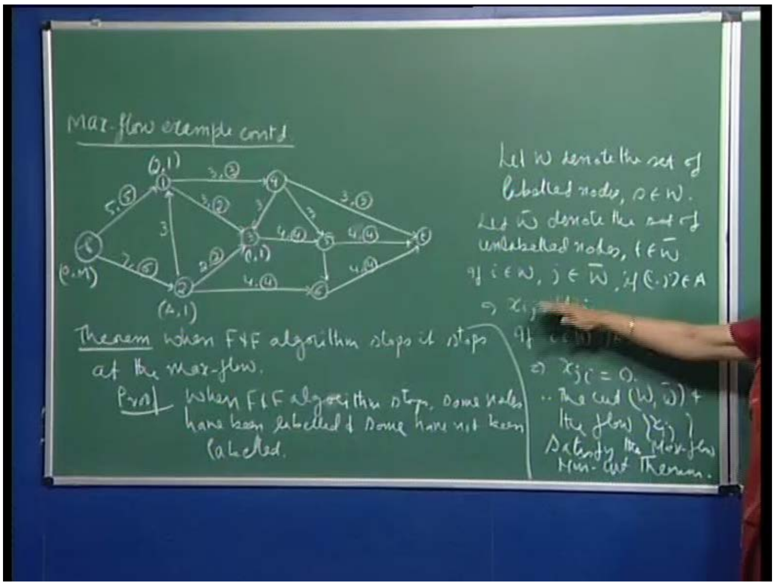
Prof. Prabha Sharma
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Module No. # 01
Lecture No. # 35

Max- Flow-Critical Capacity of an ARC, Starting Solution for Min-cost Flow problem

I will continue with the ford and Fulkerson labeling algorithm, that we were that I was that I showed to you last time and we had obtain and max-flow, sorry we had obtained a flow of 11 units in a network.

(Refer Slide Time: 00:30)



Max-flow example contd

Theorem When F&F algorithm stops it stops at the max-flow.

Proof When F&F algorithm stops, some nodes have been labelled & some have not been labelled.

Let W denote the set of labelled nodes, $s \in W$.
Let $\bar{W}$ denote the set of unlabelled nodes, $t \in \bar{W}$
If $i \in W$, $j \in \bar{W}$, if $(i,j) \in A$
$\Rightarrow x_{ij}$ [illegible]
If [illegible]
$\Rightarrow x_{ji} = 0$.
∴ The cut $(W, \bar{W})$ & the flow $\{x_{ij}\}$ satisfy the Max-flow Min-cut Theorem.

So, last time I had shown you the augmenting paths, how we obtain through the labeling procedure we obtain augmenting path. And then once we label t, we know by how much to argument the flow and also the labels, the second label tells us how to trace back the augmenting path and so we had mange to obtain this flow. So, now, we continue some more and so I will start with the label here. So, this as we said is 0 and m, for this m and now you look for for example s 1, is a saturated arc, you cannot increase the flow any more on it, so, therefore, I cannot label 1 from here.

But 2 still has a scope for increasing the flow on it, so I will label this and the difference is 1 unit, so I will label this as s and 1, right. So, that tells us that we can come to 2 from s and we can argument the flow by 1 unit here. Then we look for the arcs here, so here for example, this is not used arc at all, the capacities is 3, I can label 1 and so this will be 2 and the corresponding second label would be minimum of 1 and 3, which is 1.

So, that means, I can reach up to 1 from s via a path, which has a capacity 1 unit on it. Then you can label 3, for example from here, because you have a capacity of 1 unit left here, so that label here would be 1 and 1. From 2 I could not have labeled 3 or 6, the idea I mean of course, again as I told you there is no hard and fast rule, how you operate the list that that you use for scanning the nodes and then finding out augmenting path.

So, here, from 2 I could not have labeled 3 and 6, I could only label 1 and so then from 1 can label, I cannot label 4, I can label 3. Now, 4 3 is a backward arc, there is no flow on 4 3, so I cannot label 4 from 3. If there was some flow positive flow from 4 to 3 I could have gone to 4 from 3, right, by reducing to the flow then 5 is saturated, so that is it. So, my labeling algorithms stops here, and I am not able because I am not able to label the node t any more, right, so. So, therefore, yes and I will now prove the Ford and Fulkerson's theorem, that is validity of the algorithm, which says that when Ford and Fulkerson's algorithm stops, its stops at the max-flow.

So, we need to prove the theorems, so maybe I will write out here, theorem. The theorem says that when Ford and Fulkerson's algorithm stops, it stops at the maximum flow. So, let us prove this simple proof, so here when ford and Fulkerson's algorithm stops, stops some nodes have been labeled, right, labeled labeled and some have not been labeled. So, let us call, let w denote the set of labeled nodes, right and s belongs to w always because I can always label s, right and let w bar denote the set of unlabeled nodes and t belongs to w bar, right, have stopped because we cannot labeled t any more, right, ok.

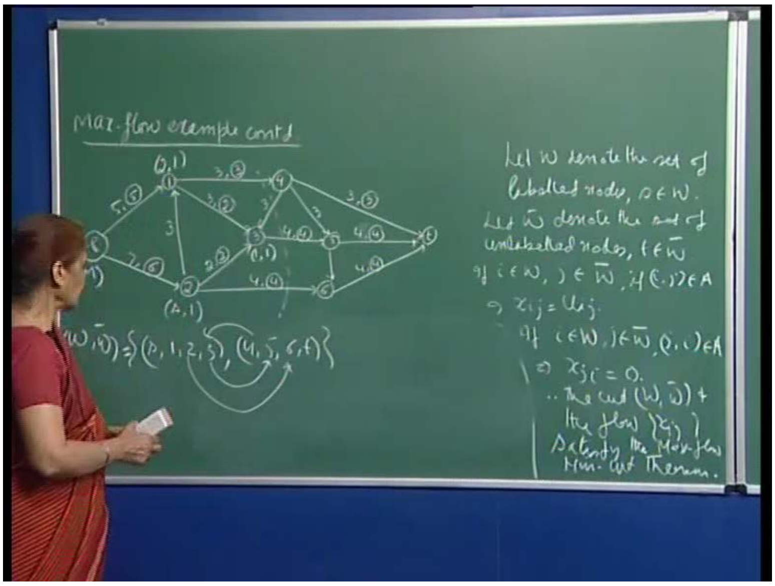
So, now, let us see what is happening, so if I belong to w and j belongs to w bar, that means I was labeled and j was not labeled, right, so this implies and if and if i j belongs to a, that means, the arc is from i to j, this implies that x i j, the current flow in the network is equal to u i j, right. If you have a remembered our labeling procedure was that if you have a forward arc, if you have a arc like this and that is what we are doing here also i and j, so if the flow here x i j is u i j and i is labeled, I cannot labeled j that is what was happening here, also I could not label 1 from s, because the arc is saturated.

So, if i is in w and j is in w bar and i j is an arc of a, then x i j must be equal to u i j, right. Similarly for the backward arc, yeah, so the idea here is this and if if I belongs to w and j belongs to w bar and j i is the arc and j i belongs to a, right, so i was labeled and j, so I am saying that here this is your w and this is your w bar, i is here and j is here, so you have an arc like, this I should add these 2, right. So, this is an arc like this, now i is labeled and if I want to label j, there must be some positive slope on the arc x j i on the arc j i, right.

So, but I am not able to label j, j belongs to the unlabeled set, so this implies, so this would implies this implies that x j i is 0 and now, if you recall the max min max-flow min-cut theorem when this, so this, therefore, the cut the cut w w bar satisfies the conditions of that theorem and therefore, the current flow in the network is maximum and the corresponding cut is the minimum cut, so very simple proof, right.

So, what we are saying is that now, so this in therefore the cut w w bar and the flow x i j satisfy the max-flow min-cut theorem. And this is the complementary slackness conditions right proof to show to you and so many ways of looking at the same theorem, but essentially it said that if you have a cut and you have a flow such that these conditions are satisfied. That means, for i and w, j and w bar, i j forward r the arc must be saturated and if i is in w, j is in w bar and j i is the arc, then x j i must be 0, if these conditions are satisfied by a feasible flow in a network and the corresponding cut w w bar, then the cut must be the minimum cut and the flow is maximum, right. And we can now quickly verify the theorem here for this for the flow that we have, because we are not able to proceed any further.

(Refer Slide Time: 09:35)



Max-flow example contd

$(W, \bar{W}) = \{(s, 1, 2, 3), (4, 5, 6, t)\}$

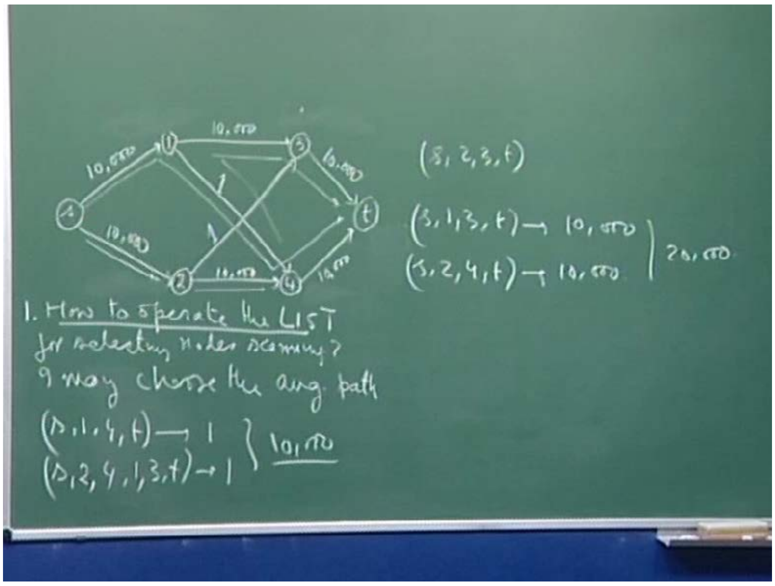
Let W denote the set of labelled nodes, $s \in W$.
Let $\bar{W}$ denote the set of unlabelled nodes, $t \in \bar{W}$
If $i \in W$, $j \in \bar{W}$, $\forall (i, j) \in A$
$\Rightarrow x_{ij} = u_{ij}$.
If $i \in W$, $j \in \bar{W}$, $(j, i) \in A$
$\Rightarrow x_{ji} = 0$.
$\therefore$ The cut $(W, \bar{W})$ + the flow $\{x_{ij}\}$ satisfy the Max-flow Min-cut Theorem.

So, what is your cut here? So, the cut w here would be what? We are able to label s 1, 2 and 3, right and there was the remaining nodes 4, 5, 6 and t these belong to, so should not say w, this is w, w bar, ok. So, this is w w bar and now you can see that what are the forward arcs that you have, say for example, from 1, 2, 3, so just check for the arcs, so your cut maybe, I can I can just show you that the cut is like this, right.

4, 5, 6, t are in w bar and 1, 2, s, 1, 2, 3 are there, so look at all the forward arcs, this 3 to 5 is a forward arc, right and this is saturated, then you have 2 to 6, 2 to the 6 is a forward arc, this is also saturated and the arc 4, 3 arc 4, 3 has no flow on it, so the corresponding x 4 3 is 0 in a current flow. So, the last arc of this cut is 1, 4, maybe I can show it this way 1, 4, you see the cut is this 1, 2, 3 as 1, 2, 3. So, this is the cut, so 1 4 and therefore, now you see that this cut will satisfy the conditions for the max-flow min-cut, because 1 4 is also saturated and you can check for the other arcs also that they satisfy. For example, 4, 3 there is no flow and all other arcs the forward arcs are saturated, therefore this is a min-cut and the corresponding flow is the max-flow.

So, a very simple basic algorithm and I showed you that the that the the the ground work for this algorithm was done by the duality theory for linear programming, because I told you that if you look at the restricted dual problem and it is equivalent to obtaining augmenting path, if it is exist in the problem, in in the network for a given flow. So, a very basic algorithm, but certainly it has a many drawbacks and this was done let say thirty years ago, almost. So, lot of improvement have taken place, but let me point out to you some of the draw backs and let see see the first thing is the algorithm because yeah..

(Refer Slide Time: 12:37)



1. How to operate the LIST for selecting nodes scanning?

I may choose the aug. path

(s,1,4,t) → 1
(s,2,4,1,3,t) → 1
} 10,000

(s, 2, 3, t)

(s,1,3,t) → 10,000
(s,2,4,t) → 10,000
| 20,000

So, let me make the point clear, I will make the point, first point is that the how to how to operate the list for selecting nodes for scanning. Remember, how? This is a question mark, because nothing is said, you can do it in any way you like and so, say for example, when you scan a node, that means, you a node has been labeled, so that is the part of the list. Then you take it out from the list and then you scan, that means, you try to see what are the nodes that can labeled of this node and then you put the labels and you add them to the list. So, now, you may you may operate a list like this, like for example, you take an node 2 and then your try to label 3 or 4, so you will add, they may already be some nodes here, so you will put 3 and 4 here.

Or you may decide put 3 and 4 on top or maybe somewhere in between and then when you want to pick up a node from the list to this thing, then you may either take this one or you may take this one or you may choose a node at random. So, there is no hard and fast rule given in the ford and Fulkerson's algorithm to how to operate the list, how to pick a node from the list, right.

So, therefore, I want to show you what can happen, say for example, a pathological case I am showing you, so what will happen here look at this network. So, here the capacities for these 6 arc is for each of them is 10000 and for arc 1 4 and 2 3 the capacities are only 1 unit, right. Now, if it is possible I may choose my augmenting path, so I may choose the augmenting path, I may choose an augmenting path s, 1, 4 and t, nobody stops because from s I will be able to label this node and this node, right.

So, this would be 10000, I mean this will be s and 10000, then this will also be s and 10000 and then I may decide to pick up one, so both these nodes get added to the list then I pick up node 1 and from here I can label this as 1 and 10000 and I will label 4 as 4 as 1 and 1, right. And then I now I may decide to choose a 4 from, I may decide to pickup node 4 from the list, because 3 and 4 will both get added to the list, so once I pick 4 then I will be able to label t and the label to t, would be 4 and 1, yes, so and the moment I label t, I stop, because I have to label t, so I found an augmenting path. And so, this will be an augmenting path and the flow will be well for 1 unit, right.

Now, in the next and so, then again you erase the labels, because that is what the algorithm says, at once you are form an augmenting path, you erase all the labels and start all over again. So, when I do it all over again, so I guide now my flow is this direction this and this, right and 1 unit of flow. Now, in the next iteration, I start labeling and let us say I label 2, I can again label 1 and 2, then I pick up 2 from the list and 2, I will labeled 3 and 4, right, so I will go this way and this way.

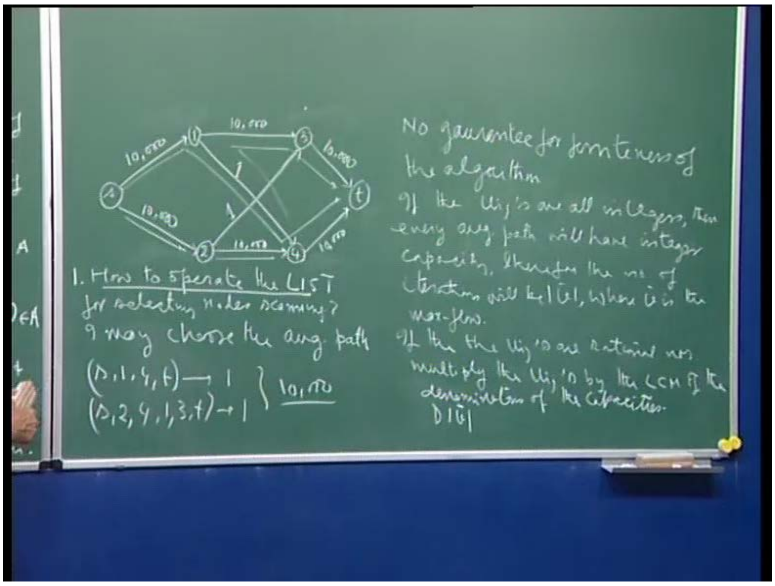
Then from 4 I may decide to label 1, because there is a 1 unit positive flow on the arc 1 4, I can label 1 from 4, so I will go this way, then go this way and then this way, so that means the next augmenting path that I will choose would be s, 2, 4, 1, 3 and t and this also has capacity 1. And I may nobody stop me from repeating this, choosing this augmenting path, because once I have done this then I can again choose s, 1, 4, t as an augmenting path, right. Remember, because these are the these are enough capacities here, I because I have now removed the flow in this direction, so, therefore I can again label 1 to 4 rights, I will do it this way and then go to t.

So, this I will then end up repeating 10000 times, but if I were to simply choose the augmenting paths s, 1, 3, t, then this has a capacity of 10000 and this will be s, 2, 4, t, which will also have a capacity of 10000. So, in 2, if I just my choice of 2 augmenting paths and I would have got the max-flow, which is 20000.

So, therefore, lot depends on how you operate the list, which has not been spelled out in ford and Fulkerson's algorithm, so this is what the later algorithm they try to give lot of improvement and now we have very efficient algorithm. So, yes, here probably I think I would just like you to sit down yourself and work out as to how you can get 20000 flows by doing, you know alternatively using this as an augmenting path and this as an augmenting path and then reversing the flow and you should be able to obtain it, ok, so this is one. So, how to operate the list is (( )) is there in the algorithm and that can play havoc with the algorithm because you cannot ok…

Now, second thing is even though I showed you that the basis for the algorithm is from the primal dual algorithm, we are not working with basic feasible solutions, because at every iteration I am augmenting the flow and so, therefore, we are no longer actually using the primal dual algorithm. Because the way we go on augmenting the flow, we lose the basic feasible solution of the in the character of basic feasible solution, right, so, therefore, finiteness is in danger, so let me or one cannot get guaranteed.

(Refer Slide Time: 19:25)



1. How to operate the LIST for selecting nodes scanning?
I may choose the aug. path
$(s,1,4,t) \to 1$
$(s,2,4,1,3,t) \to 1$
10,000

No gaurantee for finiteness of the algorithm
If the u_{ij}'s are all integers, then every aug. path will have integer capacity, therefore the no. of iterations will be $|\bar{v}|$, where $\bar{v}$ is the max-flow.
If the the u_{ij}'s are rational nos multiply the u_{ij}'s by the LCM of the denominators of the capacities.
$D|\bar{v}|$

So, no guarantee for finiteness of the algorithm, now of course if all in of if if the u i j are all integers, then every augmenting path will have integer capacity, right, all integer capacity, therefore the number of iterations, because you will at least argument the flow by 1 unit, at every iteration if you have u j are all integers, so, therefore the number of iterations will be where v bar is the max-flow is the max-flow, ok.

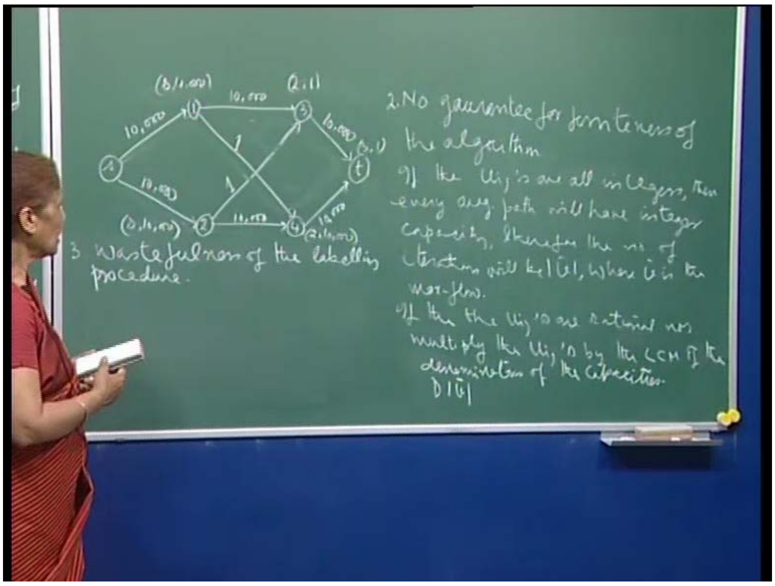
So, it is I hope it is clear, because at every augmenting path if the flow must increase by 1 unit and so you go on adding and since this is the max-flow you cannot have more than so many iterations and the amount of work that you do at each iteration is, because you scan all the arcs, so the idea is that you do not do more than order m work or order n square. So, that not really so important, but the algorithm is fine here, right. Now, if if the capacities, if the if some u i j are, if the u i j are rational numbers, see then I can sort of, I can multiply, I can change if the u i j are all rational numbers, multiply the u i j by the LCM D of the capacities, right, of the capacities.

LCM by the LCM of the denominator by the LCM of the denominators of the denominators of the capacities, right. So, you to and then if you multiply by the denominator see and then all the capacities, the new capacities will become integers and so again the flow will be, the number of iterations will be finite, because you have integer capacities, that means it will be something like d into v. If p of v bar is again the flow, v bar is the flow and since you have multiplied the capacities by d, so, therefore, this will be the total flow in the current network.

So, again the number will be finite and I can then to get original flow I can scale it by the number d and get the right, so this is fine, but if the capacities are not rational numbers, they are real numbers, irrational numbers, which again is is is the only you know normally in physical life you are not going to have capacities as irrational numbers and therefore, that is only you know for the sake of mathematical completeness, right. In case some of the capacities are irrational numbers, then your algorithm and Ford and Fulkerson actually worked out an example and this showed at if the capacities are irrationals, then the algorithm will not terminate. Because your argumentation maybe remembered, right and irrational numbers can be written as an infinite decimal, so, therefore the increments maybe such that label never add up to the total numbers, so you the the algorithm will not terminate, right.

But that is again you know just for the sake of completeness, that you want to discuss the case of irrational capacities, but in any case, even this is not a very good bound on the number of iterations, because you can improve on the (()). The third thing is the wastefulness of the algorithm, right, so yeah this was number 2, this was our number 2 and now let me talk of the, you saw yourself here, maybe I will just, yeah.

(Refer Slide Time: 24:34)



2. No gaurentee for finiteness of the algorithm

If the u_{ij}'s are all integers, then every aug. path will have integer capacity, therefore the no. of iterations will be $|\bar{v}|$, where $\bar{v}$ is the max-flow.

If then the u_{ij}'s are rational nos multiply the u_{ij}'s by the LCM of the denominators of the capacities.

$D|\bar{v}|$

3 Wastefulness of the labelling procedure.

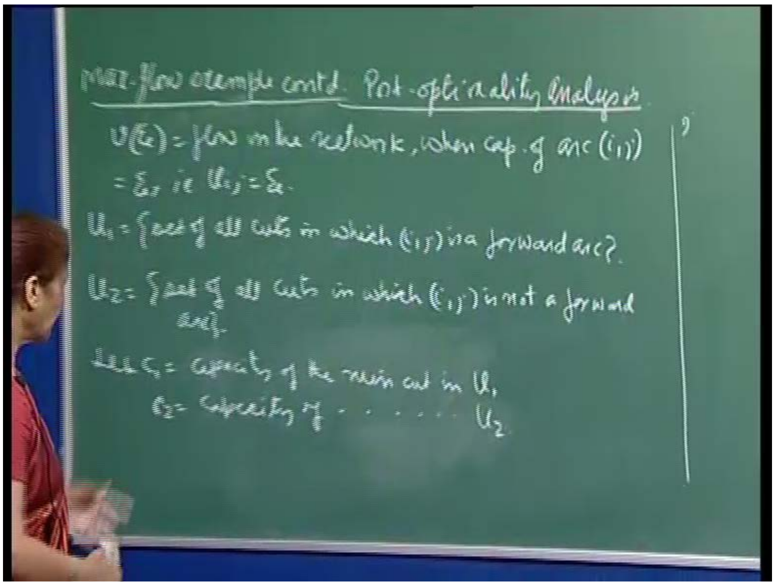
So, third point is wastefulness of the labeling procedure, labeling procedure. See yourself saw that, I maybe, I can, I can just repeat this here again, what we are saying is that, see the algorithm says that once you are found an augmenting path, you can just begin again a fresh and drop all the labels. But, see for example, here as we saw, we could label this as s and 10000 and here also I could label s and 10000, then depending on, suppose I choose this one, then the label would be 2 and 1, right and here the label would be 2 and 10000. And send again if I choose 3 for example, then the label would be, for t would be 3 and 1, right.

So, once I have labeled t, I will just say that I can augment the flow along the path s, 2, 3, and t, because I have come to t from 3, 3 I came to 2, 3 i came from 2, so 2 here and 2 i came from s, so this is my augmenting path, i augment the flow by 1 unit. And so the algorithm says erase all the labels, but you see here if I do not erase, then I can also find some other augmenting path. For example, this i can find here, right, s, 1, 3, t or s, 1, 4, t were two more augmenting paths that are available, but I will not find them right now.

Because the algorithm tells me to go back, erase the labels and start again. So, this is where people like Edmonds and Dynic, the references would be available in the text books that I have given you. So, these people try to improve on it, so they try to improve the how to operate the list and then you know not you you try to find as many log augmenting path is possible in one go, right, just do not find one augmenting path at a time and that requires some extra knowledge of how you search network and these are how you search data list and they are things like depth search, depth first and breadth first and so on. So, breadth first search is will give you more than one augmenting path at a time and then therefore you can really increase the flow along all these paths simultaneously and then work out.

So, therefore, your effort of labeling will be used to the to a greater extent, because the labeling that you have done, you will able to find more than one augmenting path, so this is the idea behind it. So, therefore, this all that I want to comment about the labeling algorithm, right and as I said that we are talking of linear programming and and its extensions, so, therefore the scope of the course does not really allow us to talk about the various improvements, then the algorithm here. So, let us do some post of optimality analysis here also as usual, because the data can keep changing and so you should be able to update your solutions according to the changes. So, here, let me let me start by some definitions.

(Refer Slide Time: 28:17)



Max-flow example contd. Post-optimality Analysis

$v(\xi)$ = flow in the network, when cap. of arc (i,j) $= \xi$, i.e. $u_{ij} = \xi$.

$U_1 = \{$set of all cuts in which (i,j) is a forward arc$\}$.

$U_2 = \{$set of all cuts in which (i,j) is not a forward arc$\}$.

Let C_1 = capacity of the min cut in U_1

C_2 = capacity of . . . . . . U_2.

So, what we are saying is that suppose let let me denote this v psi or not v psi, I want to use v psi as the equal to flow in the network when capacity of arc i j is equal to psi, that is I am saying that, I am considering the case when u i j is equal to psi. So, this is my notation, that v psi is the flow in the network when capacity of arc i j is psi, ok.

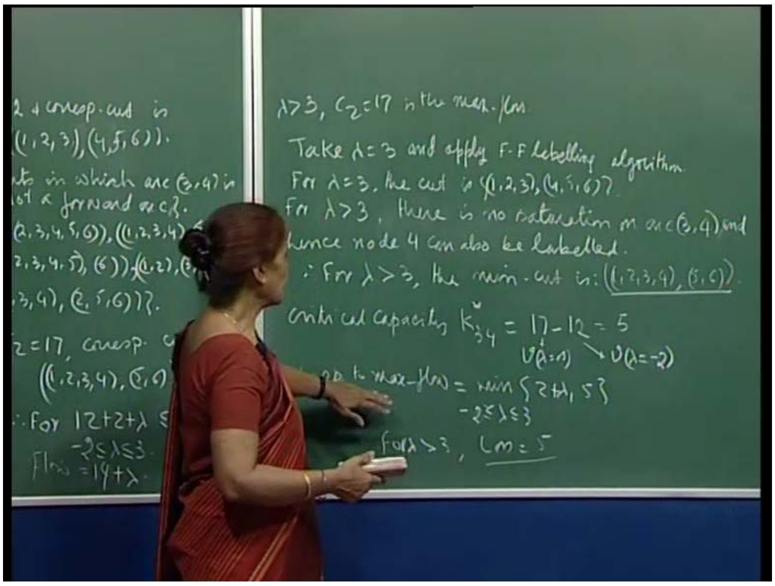
Now, lets us see here, let me define u 1 as the set of all cuts, in set of all cuts in which i j is a forward arc, yeah, so that means every cut here will have i in the first set and j in the second set, right. So, this is the collection of all cuts in the form, in which i j is a forward arc, right and lets define u 2 as the set of all cuts in which i j is not a forward arc; that means, if you have cut here and so either in this case either i or j both are here, right or i j are here.

So, they are not, it is not occurring or or it can be i j is not a forward arc, so it means that i is here and j is here, then this is an arc. So, this covers three cases and the forward case was that only this is the thing, that is i is here and j is here, so then it is a forward arc, if it is not a forward arc then we are looking at this thing, right.

So, now, you see that here yeah and now let me define let c 1 will the capacity, capacity of the minimum cut, minimum cut in u 1, yes and c 2 is the capacity of the minimum cut in u 2, not that I am saying that you really have to find it and just denoting because this is a finite set of cuts and so among all these cuts we compute the capacity and select the one for which is a minimum capacity and the say that the capacity is equal to c 1, similarly this is c 2 here, right.

Now, see for. So, it is like this, if yeah and this set of all cuts capacity of minimum cut in u 1 when psi is 0, so I am just putting the capacity of the arc i j as 0 and then finding out the cut with the minimum capacity.

(Refer Slide Time: 32:13)


λ>3, C₂=17 is the max-flow
Take λ=3 and apply F-F labelling algorithm
For λ=3, the cut is {(1,2,3),(4,5,6)}.
For λ>3, there is no saturation in arc(3,4) and hence node 4 can also be labelled.
∴ For λ>3, the min-cut is ((1,2,3,4),(5,6)).
Critical capacity k₃₄ = 17-12 = 5
U(λ=1) ↘ U(λ=-2)
max-flow = min{2+λ, 5}
-2≤λ≤3
For λ>3, Lm=5

So, now when, yeah so when u i j is equal to psi flow or not flow, capacity of all the cuts, in u 1 will go up by psi units, yeah, it is not look at all complicated, because as we said that since i is here and j is here, so this is a forward arc, c 1 was computed as the capacity of a minimum cut when psi was 0. Now, if I increase the capacity of this arc then the flow will go up by psi units, right, because remember this is all the cut the capacity will go by psi units.

So, the minimum capacity will also go up by psi units right and so, now, we said that the flow the max-flow in the network and therefore max-flow is equal to minimum of c 1 plus psi and c 2. Remember your max-flow min-cut theorem, so if c 2 is a smallest number then smaller than this and this will be the max-flow and so here this is is in. So, now, if c 1 is greater than c 2, this implies that c 1 plus psi, because psi we are taking to be non-negative, the either size 0 or positive, this implies that c 1 plus psi is greater than c 2, therefore c 2 is the max-flow, right. And if otherwise what do we get psi, if it is less than c 2 minus c 1, you see if psi is less than c 2 minus c 1 and psi plus c 1 will be less than c 2, ok.

And so yes, so therefore psi less than this will imply that c 1 plus psi is less than c 2 less than or equal to, so, therefore the cut in u 1 will be will remain min-cut for psi greater than or equal to c 2 minus c 1. So, this gives you a good idea, therefore, that means one would have to solve to get this number, one would have to solve two max-flow problems, one with psi equal to 0. And with the other one, so when you obtain when you want to obtain the values c 2 here, so let me just define, yeah, so for to obtain c 2 what we will do is we will put the capacity of arc i j as infinity, very large number.

So that it never occurs as a forward arc in a flow, in a min-cut, right and so that means you can compute the max-flow or psi equal to 0 and for psi equal to infinity to get the numbers c 1 and c 2 and that tells you psi is this, right. So, here that means and so we say that the capacity or the critical capacity, critical capacity of arc i j is equal to c 2 minus c 1, which is v infinity minus v 0 that is another notation, right.

That means, it was v psi, I started with the notation v psi, so v infinity means that when you have made that capacity of arc i j very large and this is when you have made the capacity of arc, so now 0, right. So, the critical capacity and then the question is, so that means what would be the and the loss in the max-flow, so the this you can also call as k star i j, we call it the critical capacity (( )).

So, loss in the max-flow is equal to minimum of u i j and k i j star, which have a (( )), right, if psi is less than this, then of course this will be, that it means if u i j is less than k i j star then this is the critical capacity though loss in the max-flow would be this. If u i j is greater than k i j star, then the loss in the max-flow would be k i j star, so this is what the idea. And so it has a very interesting application, that for example when in the time of war, if the enemy wants to destroy some supply line to the enemies this thing, they would want to know which will which will give the maximum damage.

That means, they want to stop the supplies reaching the camps of the enemy, camp of the enemies, so, therefore they would want to know which one they should strike, so that they get maximum, they do maximum damage. And they may be other or for example, the you may want to improve a certain route, so then you want to know that which is a very most the question is of finding the most vital arc in a network. And the most vital arc will tell you, so for example for i arc and i j, I know that this is the bound by which maximum flow would be lost in the network.

So, then I can do this exercise for each arc and then try to find out which is the arc with which we get destroyed and there will be maximum loss to the max-flow in the network and many more interesting exercises you will do with this. So, I will try to show you some more applications of the max-flow problem in the next lecture.

Let us consider this example and I will try to explain all these concepts that we have discussed, so here you see currently there is a 12 units of flow in the network, make sure that this flow conservation at each node. So, 9 units of going on arc 1 2 and 3 units on 1 3 and this 3 units are diverted on arc 3 5, right and then this 9 units, 6 of them come on arc 2 5, 3 go from 2 to 4. So, this 6 plus 3 9 finally go from 5 to 6 and this 3 units go from 4 to 6. So, the 12 current flow, total flow in the network is from 1 to 6 is 12 units, then you see that their capacity of arc 3 4 is a function of lambda and so your max-flow will also be a function of lambda, right.

Now, let us try to look at all the cuts in which 3 4, arc 3 4 figures as a forward arc and just to explain to you the concept of cut and you become familiar with it, I have just written down all the cuts, all possible cuts, this is a small example. So, you see in all these cuts 3 and 4 are in on different sets, so therefore they these cuts separate nodes 3 and 4. So, here and then you write down the capacities of each of these cuts and it turns out that the cut 1 2 3, 4 5 6 is the minimum cut among all these cuts. So, therefore, c 1 equal to 12 is the flow in the network when you treat the capacity of arc 3 4 as 0, right.

And as the arc the capacity of arc 3 4 will go up, you know it is from 2 2 plus lambda, so from 0 to positive, it becomes more and more positive, then the flow, the total flow from 1 to 6 also will go up till as we said. So, now, if you consider the cuts in which 3 4 does not figure as a forward arc in that case these are the possible cuts 1 2 3 4 5, again we write down the capacities and this will turn out to be the min-cut and the capacity of this cut is 17, ok.

So, therefore, now you want to know how far the capacity of arc 3 4 can be increased, so that the cut 1 2 3, 4 5 6 remains the min-cut and for that you will want to say that 12 plus 2 plus lambda, because the capacity of arc 3 4 is 2 plus lambda, so this total number should be less than or equal to 17. So, as long as your lambda and so this implies lambda is less than or equal to 3 and here lambda can be as low as minus 2, so as as long as lambda is between minus 2 and 3, the flow, if the current the cut 1 2 3, 4 5 6 will remain the min-cut, right.

And the moment lambda is equal to 3, then your capacity of arc u 3 4 will become 5, right and the flow in the network can be update, so let us just see how you can update the flow in the network. So, that is why I am trying to show you here, so we will do the labeling algorithm, right, we will start with 0 m here and if you want to label, right, so you will label node 2. So, for example, this is I am showing you the addition of the flow, but here currently 20 is the capacity, 9 is the flow, so therefore I can label node 2 as 1, 11, right, the difference, that means 11 more units can come from 1 to 2, right.

And similarly for node 3 flow current flows is 3, so the first node, first number indicates node from which you are coming and then 2 is the additional flow that you can have 1 arc 1 3 right. And then see from 3, from 2 you cannot label these two, right, so again as you saw that in the ford and Fulkerson algorithm, there is no hard and fast tool how you operate the list. So, for example, from here I can try to label four so from 3, therefore the first label is 3, 2, so because this number is 2. So, I can at most flow to 2 more units from 3 4 and therefore the label is 3, 2, right and once you have this, then if I decide to scan this node, then you will have to more additional units of flow going from 4 to 6 and so the total flow will go up to 14 unit's, right.

So, 4 2 this is the final this thing, so initially your flow in the network was 12 units, say 2 more units, right and so this becomes saturated now, again if you want to label from you start labeling from node 1, then what is happening, this is 3 plus 2 as we said, this gets, so this yeah this I cannot label 3 from 1, but I can label 1 2 from 1 and again the flow then label is 1, 11, right, because still the flow here is 9 units, right. Then from 2 you can label 3, right, because the capacity of this arc is 4 units, so therefore the label would be 2, 4 and these 2 these 2 nodes I cannot label from here, but from 3 I can again label node 4, because I have used only 2 units of flow here, so 3 more, I should have said 2 plus 2 plus 3.

So, because right now I had 2 units of flow going from 3 to 4, so now 3 more units can go, so therefore the label for 4 is 3, 2, right, 2 more units are coming. What is happening, why am I having yeah 3 plus 2, this should be 2 units, right, now I have 3 units coming here from here, right and so 3 units more are going on 3 to 4, so the label is 3, 3 from this thing, right. And so you have additional unit of flow 3 units from 4 to 6, right, because from 3 I cannot labeled 5, I can only label 4, so 3 more units. So, 5 more units in the network and therefore total flow is now of 17 units, the label for 6 4, 3 because from 4 you could come 2 and 6 and you could add the flow up to 3 units, right.

So, this is the updating of the flow when your capacity of the arc 3 4, that means the lambda is 3 then the capacity of arc 3 4 is 5 units, right and this is what we found out from here also and the capacity. And therefore, now, when for lambda greater than 3, if you want to label (( )) see how far your labeling algorithm will go, so then again this arc has extra capacity, because this is only 12. So, here the label would be 1 8, right, because 12 units of flow, then from here the label would be 2 1, because you have 1 unit of capacity left here.

And since lambda is greater than 3, so you will have some extra capacity here whatever it is, so you it will be lambda minus lambda minus 3, right, lambdas greater than 3 because I am considering the case lambda greater than 3. So, here the label would be 3 and minimum of minimum of 1 and lambda minus 3, I am treating the capacities are integers, so, therefore it will be 1 at most, yeah, because if lambda is 5 or 6 even then this will be only 1, the minimum, right.

And then you could you cannot label anything from here, right, there is no flow on arc 5 4, so I cannot label 5 from 4 and this 1 is saturated, so that is it. So, therefore, your labeling algorithm will stop here, right and you would have been able to label the nodes 1 2 3 4, therefore by the labeling algorithm, your cut now is 1 2 3 4, 5 6 and this must be the minimum cut, because I am not able to increase the flow anymore. And the critical capacity of arc 3 4, which we talked about a few minutes ago denote by k star 3 4, which is the difference of the flow max-flow when lambda is infinity minus the max-flow when lambda is minus 2, right.

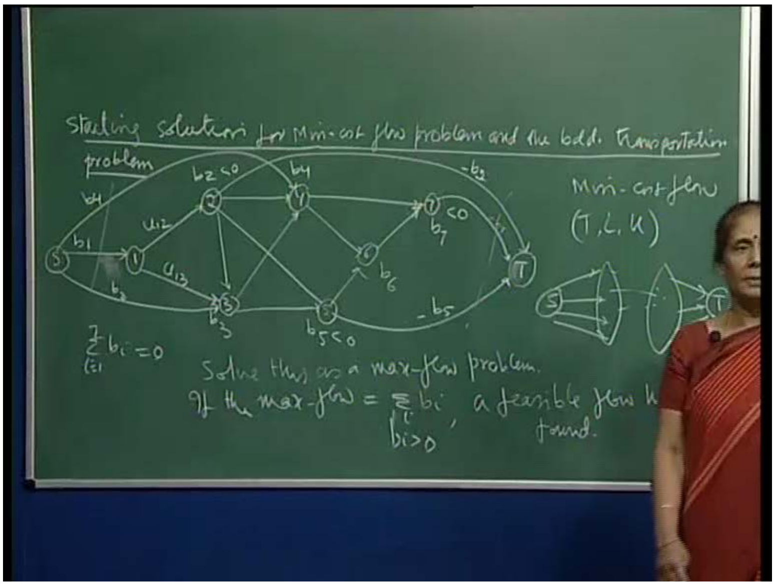
So, this is 17 minus 12 which is 5 and therefore the loss to the max-flow would be the minimum of these 2 numbers when lambda is between minus 2 and 3, but when lambda is greater than or equal to 3, the loss in the network will be 5 units. So, it would not help if you increase the capacity of arc 3 4 any further, because the max-flow will not increase, right. So, this is gives an idea how much invest on a particular route and so on and as I told you some any other applications of this concept of critical capacity of an arc.

So, now, you would want to find out the most vital arc that means destroying an arc would, the most vital arc would be that arc destroying which you will give maximum loss to the max-flow. And therefore, as I was mentioning earlier, in war times, you want to if you have a destroy more than 1 supply chain then you would want to know which is the chain which will cause the most damage.

That means you cannot the supplies cannot be, the maximum amount of supplies cannot be reached to the enemy camp that is the idea. So, the concept of most vital arc will come through here, right and as I had mentioned earlier also that you would end up solving, if you want to find the most vital arc you will end up solving two max-flow problems for each arc one with capacity 0 and the other with capacity infinity. And then out of those max-flows the two end max-flows that you get or the number of arcs you will have to do it.

If m is the number of arcs, 2 m max-flows you will be solving and then from there you will compute the most vital arc. So, remember, when we were talking of the bounded transportation problem and the min-cost flow problem, I had told you that how to obtain a starting feasible solution, I will discuss once I had introduce the concept of max-flow and this is what I want to come back to here. So, here the idea, see this is just I have drawn without showing you the cost, because they are not needed here, we will not be using them, its only the capacity is that you would be requiring so and so on.

(Refer Slide Time: 49:10)



Starting solution for Min-cost flow problem and the bdd. Transportation problem

Solve this as a max-flow problem.
If the max-flow = $\sum_{\substack{i \\ b_i > 0}} b_i$, a feasible flow is found.

You can just fill up the other numbers, so the idea here is that this is a min-cost flow problem and the b i add up to 0. So, the negative ones that means, the demand things I have denoted by less than 0, the others are all positive. So, the idea here is that you add a super source and then connect the nodes which have positive supply available at those nodes and so here then I will make the capacity of this arc as b 1. Similarly this is positive, so I will add another arc here and the capacity of this arc will be b 3 and then you have b 4.

So, I will have to add an arc like this and the capacity of this arc will be b 4. So, I will have to add an arc like this and the capacity of this arc will be b 4, fine. And then for the demand nodes I will add a super source t, super sink t and then connect b 2, so I will have to draw it this way, right and the capacity here will be minus b 2, remember because this is negative. So, I need to this capacity to be positive, then you have b 5, so this will be like this and the capacity here will be b 5 and finally, this is minus b 7. So, you see that this is now a max flow problem, because I am ignoring the I am ignoring the coos here, the capacity is given to you in this arcs.

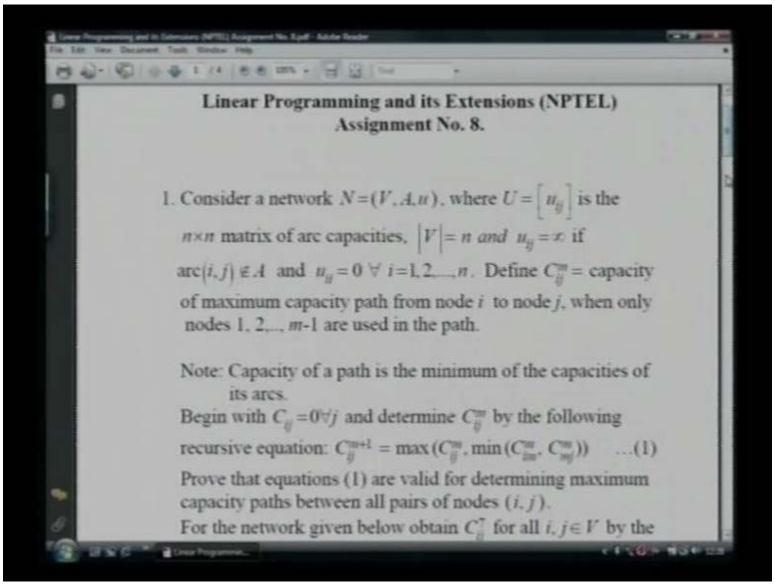
So, the the new arcs that I have added, I have defined the capacities, so now you want to find, solve this as a as a max-flow problem and you can see that immediately if the max-flow comes out to be, see for example, here the cut, because remember you you want all the supplies to be used up then only and once you use up all the supplies, because sigma b i is 0 all the demands will also be met and so that will be a feasible flow in the network, right. So, all this as a max-flow problem if the max-flow is equal to summation b i, i summation over is such that b i is greater than 0, a feasible flow has been found, feasible flow has been found. And you can see that the cut will either be this or a cut will be this and both these cuts have the same capacity which is equal to either sum of the b i, b i positive or sum of b i, b i less than 0.

So, once you have a feasible flow, that is ok, because you do not need yeah in the min-cost flow problem we were working, remember for the min-cost flow problem you require a solution in this form, where this was the tree, spanning tree, then arcs at their lower bounds and arcs at their upper bounds. So, it is not necessary that when you solved the max-flow problem you will get it in this form, but then you know how to reduce any feasible flow to this form by you know if you have flow along cycles you can drop one of the arcs and then try to. So, we know enough of the simplex network, simplex algorithm to be able to reduce a feasible flow to this form, so you can do it. And now there is no difference between the min-cost flow problem and the bounded transportation problem except that the underlying network for the bounded transportation problem is (( )), right.

You have this set of, so you have the supplied points, these are the demand points, so then it is simple, you will be adding arcs like this and of course these arcs are already there, then you will be adding arcs like this with the required capacities, whatever is demanded at these nodes, right and so again you can solve a max-flow problem. And here also you will not get a basic feasible solution, because you will have to work with the basis of the size m plus n minus 1 arc. So, then you again you know how to using theta loops, you can reduce it to a basic feasible solution where some of the arcs will have to be at their upper bounds but non-basic structure, right.

So, this was the idea that using of course I gave you a (( )) methods for finding out optimal solution, starting basic feasible solutions and so on for min-cost and for transportation problem bounded, transportation problem, but this is a neat way because you can just program the problem and the max-flow algorithm. Which now we have very efficient max-flow algorithm will help you to find a starting feasible flow and then you can reduce it to a basic feasible flow, the way you want it for working with the algorithm further on. That is what I have told you, I will come back to when I have introduced the concept of max-flow, so I will discuss assignment 8 today, which is based on your shortness path and max-flow problems.

(Refer Slide Time: 54:46)



Linear Programming and its Extensions (NPTEL)
Assignment No. 8.

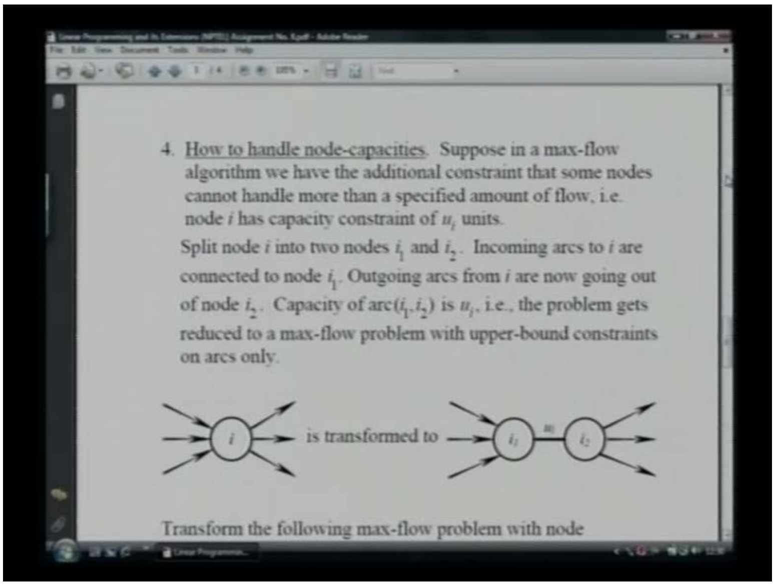
1. Consider a network $N=(V,A,u)$, where $U=\left[u_{ij}\right]$ is the $n\times n$ matrix of arc capacities, $|V|=n$ and $u_{ij}=\infty$ if arc$(i,j)\notin A$ and $u_{ii}=0\ \forall\ i=1,2,\ldots,n$. Define C_{ij}^m = capacity of maximum capacity path from node i to node j, when only nodes 1, 2,..., m-1 are used in the path.

Note: Capacity of a path is the minimum of the capacities of its arcs.
Begin with $C_{ij}=0\forall j$ and determine C_{ij}^m by the following recursive equation: $C_{ij}^{m+1}=\max(C_{ij}^m,\min(C_{im}^m,C_{mj}^m))$ ...(1)
Prove that equations (1) are valid for determining maximum capacity paths between all pairs of nodes (i,j).
For the network given below obtain C_{ij}^7 for all $i,j\in V$ by the

So, in the first problem, I have asked you to find a maximum capacity path, so capacity of maximum capacity path from node i to node j, I am defining c i j m, so this is trying to modify the floyd-warshall algorithm, which was finding out the shortest path between every pair of nodes. Now, I want you to find out between every pair of nodes a path which has maximum capacity and remember the capacity of a path is the minimum of the capacity is of its arcs.

So, that means, it is a double edge problem, in the sense that you want to find a path whose capacity is maximum and the capacity of a path is the minimum of its arcs which make up the path. So, this is what and therefore the recursive equation is a simple modification of the floyd-warshall equation, so here you will be choosing maximum of c i j m, where of course c i j m as we said is the maximum capacity of a path from node i to node j when only nodes 1 2 m minus 1 are used.

(Refer Slide Time: 59:03)



4. How to handle node-capacities. Suppose in a max-flow algorithm we have the additional constraint that some nodes cannot handle more than a specified amount of flow, i.e. node i has capacity constraint of u_i units.

Split node i into two nodes i_1 and i_2. Incoming arcs to i are connected to node i_1. Outgoing arcs from i are now going out of node i_2. Capacity of arc(i_1, i_2) is u_i, i.e., the problem gets reduced to a max-flow problem with upper-bound constraints on arcs only.

Transform the following max-flow problem with node

Exactly the same way as we defined for the floyd-warshall algorithm, but here we are talking about the capacity, so up to m minus 1 nodes have been used. So, now, when you want to include the node m, you will compare the two values c i j m, which is not including node m and then you see when you include the mode m, you have c i m m and c m j m. So, the minimum of the 2, because whichever arc you used you will have to say that the capacity will only increase by the minimum of the arcs that you are adding to the to the path and so and then you will choose the max, because if you have choice, for for example, this is i m and m j, so this will be c i j m plus 1 will be max of c i j m and minimum of c i m m comma c m j m.

So, this we will use repeatedly and then finally, for a for a node, for a for a network with n nodes you will have to compute c i j n plus 1. So, here I have asked you to now apply this recursive method to to find out the maximum capacity path between every pair of nodes for the given network, so this is your problem 1. Lets go to problem 2, see problem 2 I am asking you I have given you this matrix of distances between nodes, right, so that means the entry i j, entry of this matrix will give you the length of the arc from i to j, node i to node j and I am asking you to find the longest path, I want to you to the solve longest path problem here from node 1 to all other nodes, ok.

And I have discussed this with you how to apply the shortest path algorithm here, that means, you will replace the c i j by the minus c i j and then solve for the shortest path and then that those paths will correspond to the maximum path, so this is your problem 2, right. Now, problem 3, I have given you network and first number gives you the capacity and the circle numbers give you the flow on the arc, so here conservation of flow is maintained, I have to check it and you should also find out that the conservation equations are satisfied, then I this to, so that you get familiar with the cuts and their capacities and so on. I have asked you to specify 4 s t cuts, 1 cut with arc 6 5 as a forward arc, so find out an arc cut which has 6 5 as a forward arc.

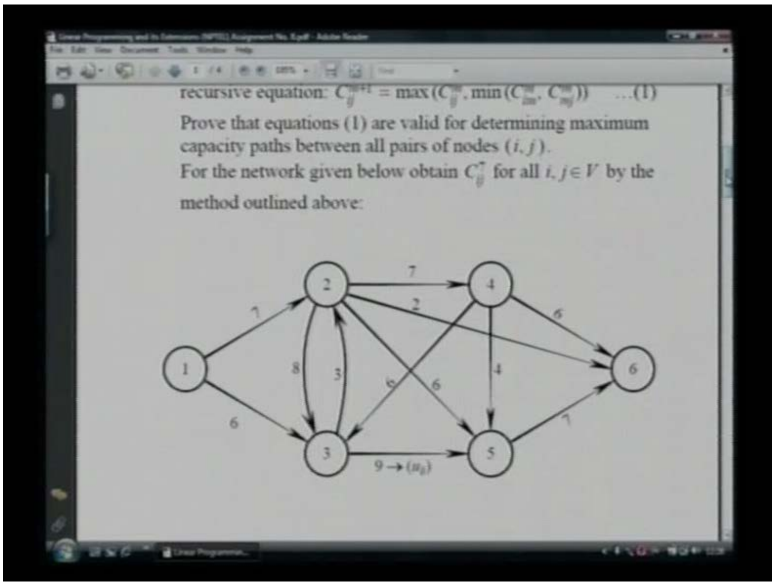
Then compute net flow across these cuts, remember I define these system for you and verify that it is equal to the flow in the network. So, the current flow in the network is 20 units, so you have to, whatever 2 cuts you find out you will have to find out the flow, net flow across these cuts and then show that for both the cuts the value is 20. Then I and the second part I want you to find out two augmenting paths and augment the flow along along them, so this is problem 3. Now, problem 4, I have because we did not discuss this in the lecture, so I thought though the through the assignment sheet, I will familiarize you with the situation when you have node capacities also.

See you have arc capacities that means you have the limitation on the amount of flow that will arc can handle, but suppose you also have limitation on the amount of flow that a node can handle, because your warehousing thing may not be the capacity may be limited and so on, the many situations which you can formulate with the node capacities also.

So, now, here, for example, if node i has a capacity of u i units, the amount of flow it can handle, then I am saying that you transform this, you split the node into two nodes i 1 and i 2. So, the incoming arcs to node i1 will now be incoming to i 1, then i join over the error should have been there i 1 i 2, there is an error, this arc has a capacity u i units. And then the outgoing arcs from node i will be now outgoing from i 2, so you see that because the arc i 1 i 2 has a capacity u i, therefore neither u i, neither the node i 1 not the node i 2 can handle of flow of more than u i units.

So, I take care of the situation that node, I cannot handle more than u i units by doing this transformation and once you do this, you have been able to transform the node capacity problem to the usual max-flow problem, in which the only the arc capacities are required to be satisfied, ok, so this is your problem 4.

(Refer Slide Time: 57:06)



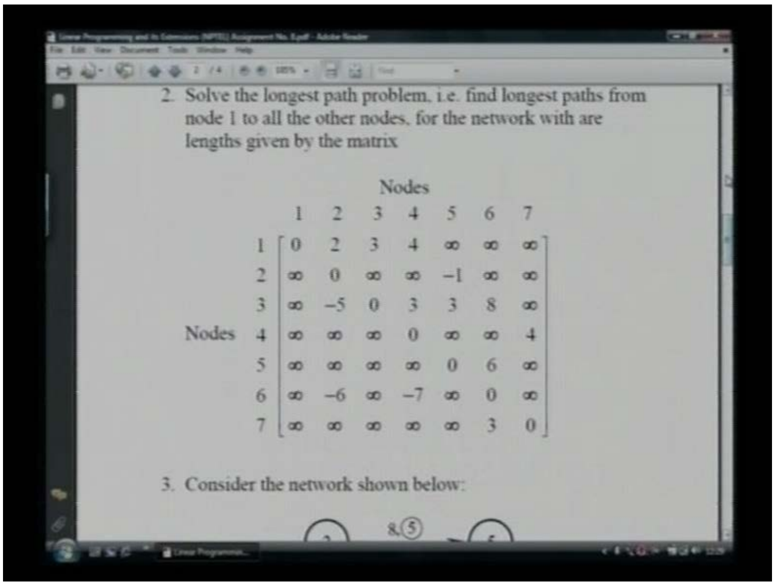
recursive equation: $C_{ij}^{m+1} = \max(C_{ij}^{m}, \min(C_{im}^{m}, C_{mj}^{m}))$ ...(1)

Prove that equations (1) are valid for determining maximum capacity paths between all pairs of nodes (i, j).

For the network given below obtain C_{ij}^{*} for all $i, j \in V$ by the method outlined above:

And then I have asked you to solve this problem, I have give you anyone network flow, so covert this to regular max- flow problem. And I will mention that the nodes 6 from s 6 and t do not have any node capacities that mean they have infinite node capacities. So, that is your problem 4.

(Refer Slide Time: 57:14)



2. Solve the longest path problem, i.e. find longest paths from node 1 to all the other nodes, for the network with are lengths given by the matrix

$$\begin{array}{cc} & \text{Nodes} \\ & \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \text{Nodes} \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} & \begin{bmatrix} 0 & 2 & 3 & 4 & \infty & \infty & \infty \\ \infty & 0 & \infty & \infty & -1 & \infty & \infty \\ \infty & -5 & 0 & 3 & 3 & 8 & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty & 4 \\ \infty & \infty & \infty & \infty & 0 & 6 & \infty \\ \infty & -6 & \infty & -7 & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & \infty & 3 & 0 \end{bmatrix} \end{array}$$

3. Consider the network shown below:

Now, problem five, I will have to yeah…So, problem five, when you have presence of undirected arcs there will it is possible that some arcs you have not specified the direction in which they can be used. So, the idea here is that you replace the arc i j which is undirected and has capacity of u i j units, you replace it by two directed arcs i j and j i and make the capacities of both the arcs is u i j equal to u j i, which is the which is given to you, then you solve the max-flow problem.

And if x i j minus x j i is greater than 0, that is in the optimal flow, x i j is your vector of flow, then if x i j is greater than x j i, then that means you will be using the flow, the vector, the arc in the direction i to j and so, therefore, we will replace the edge i j with the arc i j. And if x i j is less than x j i then we will be using the arc in the backward direction or that means that the edge i j will be replace by the arc j i and with capacity u i j. So, solving a max-flow problem will tell you, because this again in a traffic situation, you see the people who want to avoid rushes and traffic jams they may want to know which which will be profitable.

So, they want to see that the max -flow would be possible while using the street or the road in a particular direction and this will help them to determine the direction in which they should allow. So, they will make it one way and they will allow the traffic in a direction in which you have the max-flow possible in the network, so this is a situation we are trying to model here and yes hence, so, yeah.

So, now this is your diagram I have drawn for you, arc arcs 3 5 and 2 4 are undirected and so you have to replace them by 2 directed arcs, solve the max-flow problem and then find out which is the best direction to give to these two arcs. So, now, here I have referred to the these three books, which I have been referring to you will earlier, but they are lot of problems in that these three books on the topics of shortest paths and max-flow and lot of applications. But of course, all these three books are good for your for trying out your problems on whatever material we are discussing here, so they should take care of your topics shortest path and max-flow.