

Advanced Graph Theory
Prof. Rajiv Misra
Department of Computer Science and Engineering
Indian Institute of Technology, Patna

Lecture – 15
Network Flow Problems

(Refer Slide Time: 00:15)

Preface

Recap of Previous Lecture:

In previous lecture, we have discussed the k -connected graphs, k -edge-connected graphs, Menger's theorem and Line graph.

Content of this Lecture:

In this lecture, we will discuss the Network Flow Problems *i.e.* Maximum Network Flow, f -augmenting path, Ford-Fulkerson labeling algorithm, Max-flow Min-cut Theorem and the Proof of Menger's Theorem using max-flow min-cut theorem.

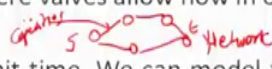
Advanced Graph Theory Network Flow Problems

Network Flows. Recap of previous lecture we have discussed k connected graphs, k edge connected graphs, Mengers theorem, and line graph, content of this lecture, we will discuss network flows that is maximum network flow, f augmenting path, Ford-Fulkerson labeling algorithm, Max-flow min cut theorem, and we will see the proof of Mengers theorem using max flow min cut theorem.

(Refer Slide Time: 00:50)

Network and Flows

- Consider a network of pipes where valves allow flow in only one direction.
- Each pipe has a capacity per unit time. We can model this with a vertex for each junction and a (directed) edge for each pipe, weighted by the capacity. We also assume that flow cannot accumulate at a junction.
- Given two locations s , t in the network, we may ask **“what is the maximum flow (per unit time) from s to t ?”**
- This questions arises in many contexts. The network may represent roads with traffic capacities, or links in a computer network with data transmission capacities, or currents in an electrical network. There are applications in industrial settings and to combinatorial min-max theorems.



Advanced Graph Theory Network Flow Problems

Network and flows consider a network of pipes where valves allow flow in only 1 direction, each pipe has a capacity per unit time, we can model this with the vertex for each junction and the directed edge for each pipe, weighted by the capacity we also assume that flow cannot accumulate at the junction.

Let us see this kind of model can be represented in a form of a network. So, given two locations s and t in the network we may ask what is the maximum flow per unit time from s to t , so this question arises in many context the network may represent the roads, with the traffic capacities, or links in a computer network with data transmission capacities, or the currents in electrical network, there are applications in industrial settings and to the combinatorial min max theorems. So, here we can add the capacities, which can be defined as per the different applications.

(Refer Slide Time: 02:31)

Network 4.3

- A **network** is :
 - A digraph with a nonnegative capacity $c(e)$ on each edge e and
 - A distinguished **source vertex s** and **sink vertex t** .
 - Vertices are also called **node s** .

Advanced Graph Theory Network Flow Problems

So, the network is a directed graph let us see that these edges which are connecting the nodes are having a direction, and also each edge is having a capacity which is denoted by c of e , and there is a distinguished source and sink vertex s and t they are denoted as s and t . The vertices are called here the nodes. So, modeling a particular network, in this particular format is nothing, but a network n . So, for any given graph we can convert it into a network, in this particular setting and then we can use it for the study of network flow problems.

(Refer Slide Time: 03:39)

Network Flow: Definitions and constraints 4.3

- A flow f assigns a value $f(e)$ to each edge e .
- Let:
 - $f^+(v)$: the total flow on edges leaving v and $f^-(v)$
 - $f^-(v)$: the total flow on edges entering v
- A flow is feasible if it satisfies
 - The **capacity constraints** $0 \leq f(e) \leq c(e)$ for each edge and
 - The **conservation constraints** $f^+(v) = f^-(v)$ for each node $v \notin \{s, t\}$.

Advanced Graph Theory Network Flow Problems

So, the flow f assigns a value to each edge so, in the previous figure if you see s and t . So, each edge is having a capacity and also we can assign a flow value a particular edge. So, every edge will have these 2 values assigned that is the flow on the edge e which is basically represented f of e . Let f^+ be the total flow on the edges leaving v , and f^- be the total flow on the edges entering v , a flow is feasible if it satisfies the capacity constraints so; that means, the value of the flow is upper bounded by the capacity of that particular edge which is defined so; obviously, here in this particular scenario the value of the flow is basically bounded by or at most c_e on particular edge.

So, every edge has this capacity defined, and the flow which will be there on that particular edge will be at most that particular value, and also the flow is basically non negative hence it is greater than or equal to 0 and so on. Now the flow also satisfies besides capacity constraints, another constraint which is called a conservation constraint. So, conservation constraints says that the flow for each node other than source and sink. So, the value of the flow out for a node v should be equal to the flow which is into the node, hence f^+ that is the flow out on a particular node is equal to f^- the flow which is in to the particular node other than source and sink.

So, again let us recall that there is a flow, which is defined on each edge and this particular flow is feasible if it satisfies the capacity constraints, and also satisfies the conservation constraints that we have defined.

(Refer Slide Time: 06:45)

Maximum Network Flow

- The **value** $\text{val}(f)$ of a flow f is the net flow $f^-(t) - f^+(t)$ into the sink.
- A **maximum flow** is a feasible flow of maximum value.

$\text{val}(f) = f^-(t) - f^+(t)$ — if $f^+(t) = 0$

Advanced Graph Theory
Network Flow Problems

So, maximum network flow the value of the flow that is val of f of a flow f is the net flow that is the flow out from, that particular sink to be subtracted from in to the sink. Now if the flow there is no out from the sink, then it is nothing, but the flow value which is the entering into the sink, if there is no flow out from the sink, then the value of the flow is equal to flow which is entering into the sink. Now, the max flow is the feasible flow of the maximum value that is called the maximum flow.

(Refer Slide Time: 07:39)

Example of Max Flow

- The **zero flow** assigns flow 0 to each edge ✓
- It is feasible. ✓

Advanced Graph Theory
Network Flow Problems

Let us take this particular example graph which has a flow value equal to 0 assigned to each edge and this is also a feasible flow. So, every edge the capacity is 2, and the flow is 0. So, it satisfies a feasible flow satisfies the capacity constraints meaning to say that here in; for example, in the edge $s u$ the capacity of this particular edge is 2 and the amount of flow is 0, hence this particular capacity constraint is satisfied on this particular.

Similarly in all of the edges this is satisfied, second condition for a flow to be feasible is that conservation constraint is satisfied should be satisfied for example, in this particular node u , for a particular node u , let us consider that the value of $f u$ plus equal to f minus and here, the flow in both the sides, is 0 hence conservation constraint is also satisfied, therefore the 0 flow that is the total flow which basically will go to the sink. That is the total flow is equal to 0 hence this is the feasible flow, and this is the example of the network with a flow of 0 value, example of a maximum flow.

(Refer Slide Time: 09:48)

Example of Max Flow

- In the network below we illustrate a non-zero feasible flow.
 - Capacities are shown in bold, flow values in parentheses.
 - Our flow f assigns $f(sx) = f(vt) = 0$, and $f(e) = 1$ for every other edge e . This is a feasible flow of value 1.

Advanced Graph Theory **Network Flow Problems**

So, here in this particular network we will illustrate that it has the nonzero feasible flow, the capacities are shown as the bold, and the flow is shown within the bracket it is the flow on the edge. So, our flow f assigns f of $s x$ here that is equal to the flow on $v t$, and both are 0 here in this case. Similarly $f 1$ and edge e is 1, this edge it is 1 this edge also it is 1, this edge it is 1, and this edge is 1 for every other edge of this network. Hence this is a feasible flow of value 1, example of a maximum flow here.

(Refer Slide Time: 11:15)

Example of Max Flow

- A path from the source to the sink with excess capacity would allow us to increase flow.
 - In this example, no path remains with excess capacity, but the flow f' with $f'(vx) = 0$ and $f'(e) = 1$ for $e \neq vx$ has value 2.

Advanced Graph Theory **Network Flow Problems**

The path from source to sink with excess capacity would allow us to increase the flow in this example no path remains with the access capacity, but a flow f' with $f'(v, x)$ is equal to 0, and $f'(x, v)$ is equal to 1. Now for e which is not equal to v, x has the value 2.

So, let us take an example here in this particular case, we can see on this particular edge the capacity is 2 and the flow is 0. So, there is a residual capacity to inject a flow on this particular edge, similarly this particular edge also has the residual capacity. Now it has a flow from v to x hence from x to v we can do a back flow, or we can also say it is a reverse flow possible and that capacity is 1. So, if you take a path from s to x and x to v and from v to t .

So, this particular s to x we have the capacity of 2 from x to v there is a capacity of 1, and from v to t there is a capacity of 2. So, we can inject the minimum of all these values that is 1 a flow of 1 unit can be injected from source to the sink, via this particular path. So, if we do then basically you see that here the value of the flow is changed from 0 to 1, and here since it is a back flow. So, 1 will become 0 here in this case.

Similarly, here the flow will become 1, and through this particular path the flow is injected, and the amount of flow in this particular network it will be now this particular flow is 1 and this flow is 1. So, total flow in this network is 1 plus 1 that becomes equal to 2 units of flow, and that is the maximum flow possible.

(Refer Slide Time: 13:55)

f-Augmenting Path 4.3.4

- When f is a feasible flow in a network N , an **f-augmenting path** is a source-to-sink path P in the underlying graph G such that for each $e \in E(P)$,
 - a) if P follows e in the forward direction, then $f(e) < c(e)$. ✓ Capacity constraint
 - b) if P follows e in the backward direction, then $f(e) > 0$.
- Let $\epsilon(e) = c(e) - f(e)$ when e is forward on P , and let $\epsilon(e) = f(e)$ when e is backward on P .
- The **tolerance** of P is $\min_{e \in E(P)} \epsilon(e)$. ✓

Advanced Graph Theory
Network Flow Problems

Now, we have seen that we have identified a path through which we can inject a flow into the network or the residual capacity. So, that kind of path is called augmenting path, let us see the definition when f is a feasible flow in the network n , and f augmenting path is a source to sink path in the underlying graph G such that for each edge on the path that, if P follows E in the forward direction, then it has to follow this particular capacity constraint, and if P follows E that is in the backward direction, then there must be a nonzero flow on that particular edge.

Now, let ϵ is nothing, but the residual capacity that is $c_e - f_e$ when e is a forward direction on P , and let ϵ is equal to f_e when e is in the backward direction on P . So, the tolerance of that particular flow or a path is nothing, but the minimum of for all the edges on that particular path that particular ϵ values, and that is called a tolerance. In the previous slide we have seen how to compute this particular tolerance.

So, here this is nothing, but we have computed the tolerance, we have computed the tolerance that is by these are the ϵ values, and this particular minimum of ϵ is nothing but the tolerance, and this amount of values is injected onto the path, and the path is and this particular value is basically introduce as an additional flow of 1 unit. So, f augmenting path is this particular process of identifying what is the maximum amount of flow which can be undertaken on a particular path, and that is called f augmenting path.

(Refer Slide Time: 16:32)

New Flow after Augmenting

- The edges of P incident to an internal vertex v of P occur in one of the four ways shown below.
- In each case, the change to the flow out of v is the same as the change to the flow into v , so

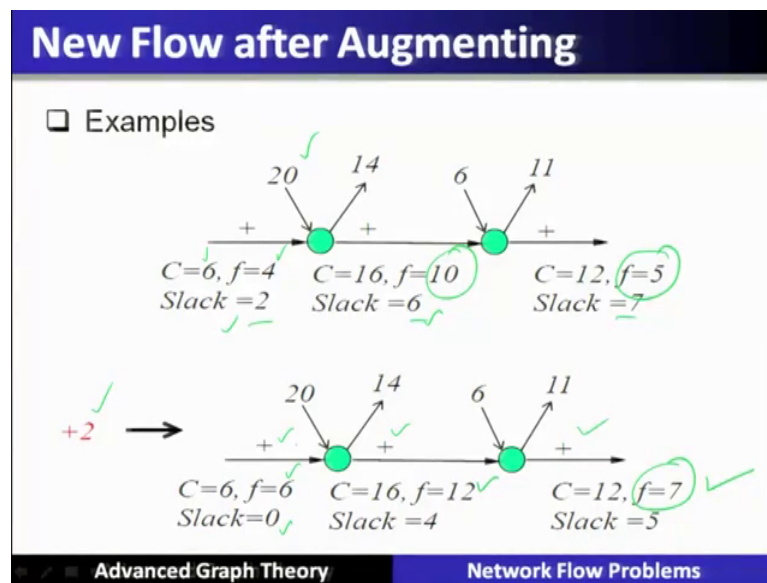
$$f^-(v) = f^+(v).$$

Flow in = Flow out

Advanced Graph Theory
Network Flow Problems

Now, when a new flow is injected through an augmenting path so, let us see what are the changes in the network will takes place, the edges are p incident to the internal vertex v of P occurs in 1 direction of the 4 different ways shown below, in each case the change to the flow out of v is the same as the change to the flow into v. Hence this is the flow out, this is the flow in, and that should be equal to the same for ensuring the property which we have designated as conservation constraint.

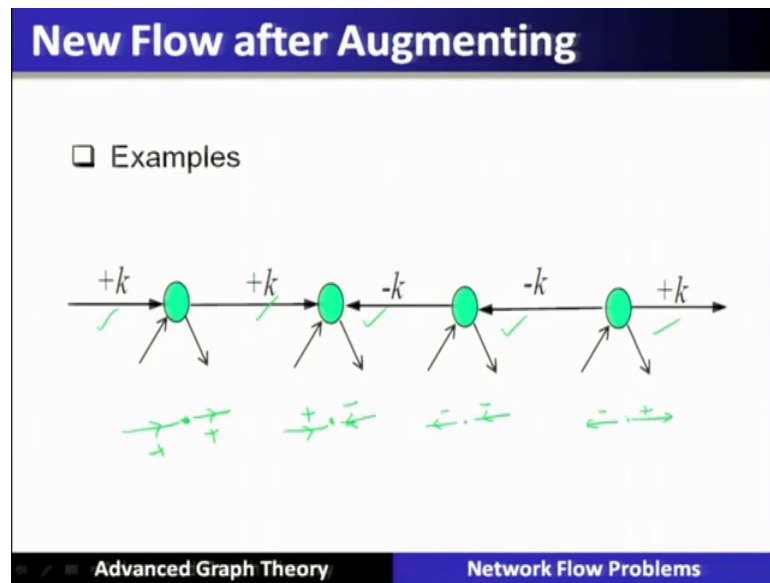
(Refer Slide Time: 17:29)



Now, let us see the examples of all these 4 different cases when a new flow of augmenting is basically carried out. So, here you consider the capacity is 6 and current flow is 4. So, there is a possibility of 2 units to be injected here. Similarly 20 unit is also flow is also there. So, basically here we can see that ten unit; that means, there is a slack of 6 units and so on.

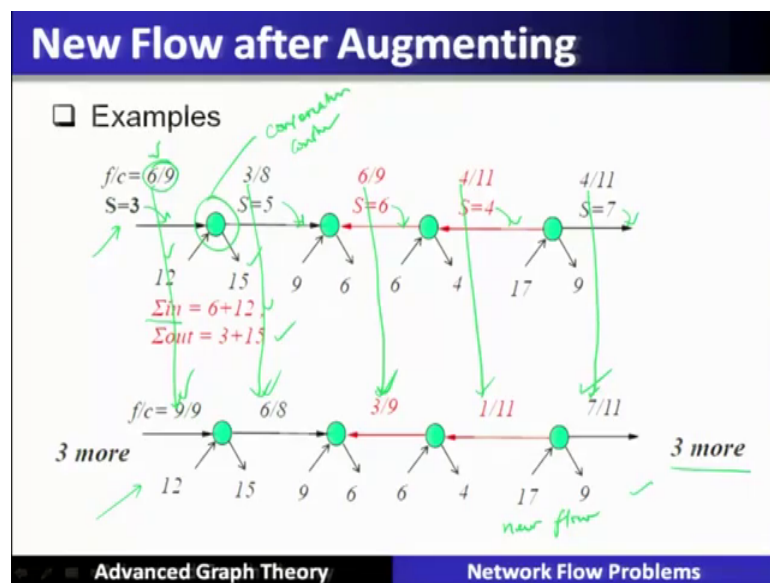
So, so if we take the minimum of all these slack units it becomes 2 units we can inject. So, the 2 units are injected. So, here the flow in this particular which will enter is equal to the to the capacity, and the slack will become 0 here in this case, the flow of 10 will become 12, and here the flow of 5 will become 7 here in this case, and these symbols are shown over here; that means, there is a possibility of forward direction flow on each vertex is possible.

(Refer Slide Time: 18:49)



Now, there is another example here we can see that on the forward direction there is a possibility of plus k , in this particular vertex also there is a possibility of plus k , then on this particular vertex there is also a possibility of minus k , and this is a minus k and plus k . So, the symbols which we have shown is a plus k and plus k , similarly another possibilities another case is plus k and minus k , and this is another condition where it is minus and minus, and this particular condition shows it is plus and it is minus. So, all 4 cases we can see here in this augmenting path to be illustrated.

(Refer Slide Time: 19:47)



Let us take another example here we can see here the total inflow is 6 plus 12 that is 18, and total outflow is 15 plus 3 that is 18 that is the conservation constraint is satisfied. So, let us see that if we can figure out what is that minimum value of that capacity constraint, the capacity is 9 the total flow here is undertaken is 6. So, there is a possibility of 3 units of residual capacity here, also there is a possibility of 5 units of capacity here, there is a 6 units of reverse flow possible, 4 units of reverse flow possible and 7 units of reverse flow possible. So, if you take the minimum. So, 3 more 3 units of more amount of flow which can be injected, hence the f augmenting path like this will allow 3 more units of the flow after doing that the flow values are changed accordingly that we are now seeing.

So, 4 will become 7, 4 will become 1, why because 3 unit will be subtracted it is a reverse edge 6 unit will become 3 why because 3 units will be subtracted, because this is the reverse edge or a back edge. Now here 3 will become 6 because it is a forward edge, 6 will become 9 why because it is a forward edge. So, after the augmenting the new flow will be introduced into the network. So, lemma P is an f augmenting path with the tolerance z .

(Refer Slide Time: 21:51)

Lemma. If P is an f -augmenting path with tolerance z , then changing flow by $+z$ on edges followed forward by P and by $-z$ on edges followed backward by P produces a feasible flow f' with $\text{val}(f') = \text{val}(f) + z$.

Proof:

- The definition of tolerance ensures that $0 \leq f'(e) \leq c(e)$ for every edge e , so the capacity constraints hold.
 - We need only check vertices of P , since flow elsewhere has not changed.
- For every vertex v , $f^+(v) = f^-(v)$ ✓
- Finally, the net flow into the sink t increases by z . ✓

Advanced Graph Theory Network Flow Problems

Then changing the flow by plus z on the edges forward by P , and minus z on the edges followed backward by P produces a feasible flow f prime with the value of value of flow f prime is equal to the value of f plus z that we have seen in the previous example. Let us

see the proof the definition of the tolerance ensures that, the capacity constraint is ensured for every edge. So, the capacity constraint holds.

So, we need only check the vertices of P since the flow elsewhere has not changed. Now for every vertex v conservation constraint is being followed. So, the amount of flow which is in equal to the amount of flow which will be exited from a particular vertex finally, the net flow into the sink t will be increased by this z value, that example we have seen earlier and that is also shown here in this particular picture.

(Refer Slide Time: 23:03)

Source/sink cut

- In a network, a **source/sink cut** $[S, T]$ consists of the edges from a source set S to a sink set T , where S and T partition the set of nodes, with $s \in S$ and $t \in T$. S+cut ✓
- The **capacity** of the cut $[S, T]$, written $cap(S, T)$, is the total of the capacities on the edges of $[S, T]$. S → T
sink node
- Keep in mind that in a digraph $[S, T]$ denotes the set of edges with tail in S and head in T . Thus the capacity of a cut $[S, T]$ is completely unaffected by edges from T to S . (*back edges*) S → T
CAP(S,T)

Advanced Graph Theory
Network Flow Problems

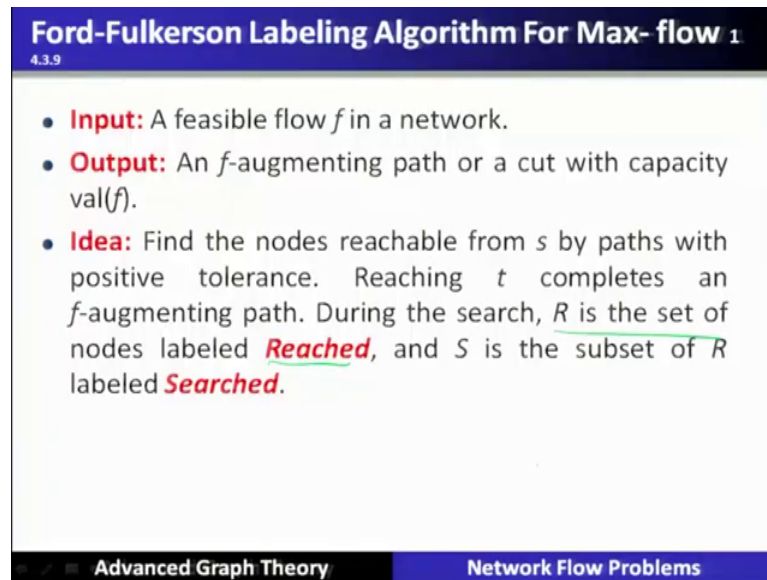
Source and sink cut so, in a network source a sink cut that is also called S T cut.

So, S T cut is nothing, but which is denoted as S comma T consists of the edges from the source set S and to a sink set P, where S and T partition the set of nodes with all the nodes small S belongs to S and P belong to T that means, source this is source node belong to 1 set that is S, and this is the sink node belong to T. So, all the other nodes are partitioned in S and T, and that is called basically S T partitions. So, the capacity of this particular S T cut is written as the capacity of S T is the total capacity on the edges of S and T; that means, the total capacities of the edges which crosses from S and T.

So, we can see over here S and T. So, all the edges which they crosses their capacities, if we sum them in the forward direction, then it is called the capacity of S T in the forward directions only. So, keep in mind that in a digraph S T denotes the set of edges with the

tail in S that I told you, and the head in T that is only the forward direction, edges thus the capacity of S T cut is completely unaffected by the edges from T to S ; that means, it is unaffected by the back edge capacities.

(Refer Slide Time: 25:22)



Ford-Fulkerson Labeling Algorithm For Max-flow 1
4.3.9

- **Input:** A feasible flow f in a network.
- **Output:** An f -augmenting path or a cut with capacity $\text{val}(f)$.
- **Idea:** Find the nodes reachable from s by paths with positive tolerance. Reaching t completes an f -augmenting path. During the search, R is the set of nodes labeled **Reached**, and S is the subset of R labeled **Searched**.

Advanced Graph Theory Network Flow Problems

Ford Fulkerson labeling algorithm for max flow, in this algorithm the input is a feasible flow f in a network, and the output is an f augmenting path or a cut with a capacity value of f . So, idea of this algorithm is to find the node reachable from S by the path with the positive tolerance, reaching T completes an f augmenting path. Now during the search R is the set of the nodes labeled as the reached, and s is the subset of R which are labeled as searched; Ford Fulkerson labeling algorithm.

(Refer Slide Time: 26:08)

Ford-Fulkerson Labeling Algorithm For Max-flow 2

Initialization: $R = \{s\}$, $S = \phi$.
Handwritten notes: Reached (green), Searched (green). $R_{Reached} = \{s, w, u, t\}$, $S_{Searched} = \{v\}$

Iteration: Choose $v \in R - S$.
Handwritten note: $v \rightarrow w$

- For each exiting edge vw with $f(vw) < c(vw)$ and $w \notin R$, add w to R .
Handwritten note: (Forward direction, Capacity)
- For each entering edge uv with $f(uv) > 0$ and $u \notin R$, add u to R .
Handwritten note: Backflow $u \rightarrow v$
- Label each vertex added to R as "reached", and record v as the vertex reaching it. After exploring all edges at v , add v to S .
- If the sink r has been reached (put in R), then trace the path reaching t to report an f -augmenting path and terminate. If $R = S$, then return the cut $[S, \bar{S}]$ and terminate. Otherwise, iterate.

Advanced Graph Theory Network Flow Problems

Initialization we will initialize the 2 sets are that is called reached, the another set S which is called searched set. So, initialized with s and this is initialized with ϕ .

Now, in the iteration let us choose a node v which is in R minus S , this is R and this is S . Now for each exiting edge vw with the capacity $f(vw)$ is less than $c(vw)$ and w is not in R , then we add w to R ; that means, we have reached to w in this labeling algorithm from v , because there is a capacity in the forward direction. Now for entering an edge uv with $f(uv) > 0$ with having the flow $f(uv)$ is non zero; that means, some flow is there, and u is not in reached then we have to add u in R .

So, this indicates the back flow along the f augmenting path. Now label each vertex added to R as the reached and record v as the vertex reaching it; that means, from which it has reached, after exploring all the edges at v add v to S . Now if sink R has been reached put in T put in R , and then trace the path reaching t to report the a f augmenting path and terminate. Now if r is equal to S then return the cut that is S and S prime, and terminate why because there is no f augmenting path.

So, we have to return the min cut. Otherwise we have to iterate, as long as f augmenting path is available the algorithm will iterate, and as soon as all the augment there is no augmenting path, then it will terminate and report the min cut of the network theorem, max flow min cut theorem.

(Refer Slide Time: 29:32)

**Theorem: Max-flow Min-cut Theorem-
Ford and Fulkerson [1956] 4.3.11**

- In every network, the maximum value of a feasible flow equals the minimum capacity of a source/sink cut.

Proof:

- In the max-flow problem, the zero flow ($f(e)=0$ for all e) is always a feasible flow and gives us a place to start. Given a feasible flow, we apply the labeling algorithm. It iteratively adds vertices to S (each vertex at most once) and terminates with $t \in R$ ("breakthrough") or with $S=R$.
- In the breakthrough case, we have an f -augmenting path and increase the flow value. We then repeat the labeling algorithm. When the capacities are rational, each augmentation increases the flow by a multiple of $1/a$, where a is the least common multiple of the denominators, so after finitely many augmentations the capacity of some cut is reached. The labeling algorithm then terminates with $S=R$.

Advanced Graph Theory Network Flow Problems

So, in every network the maximum value of a feasible flow equals the minimum capacity of the S T cut. So, the proof says that in max flow problem the 0 flow is always a feasible flow that we have seen, and gives us a place to start. Now given a feasible flow we apply the labeling algorithm, we have seen in the previous slide that is Ford Fulkerson algorithm. It iteratively add the vertices to S that is each vertex at most once and terminate with t when it reached, and that is why t is an element of R , and then we say it is a back through; that means, we have identified an augmenting path which is also called as a breakthrough here, or if it is not a augmenting path then S is equal to R , and we you have to report with a min cut.

So, in the breakthrough case; that means, when we have identified a augmenting path we have an f augmenting path and increase the flow value, we then repeat the labeling algorithm when the capacities are rational the each augmentation increases the flow by multiple of 1 by a where a is the least common multiple of the denominators. So, after finitely many argumentation the capacity of some cut is reached the labeling algorithm that terminates with S minus S is equal to R .

(Refer Slide Time: 31:18)

Theorem: Max-flow Min-cut-Ford and Fulkerson [1956] contd...

- When terminating this way, we claim that $[S, T]$ is a source/sink cut with capacity $\text{val}(f)$, where $T=S$ and f is the present flow, It is a cut because $s \in S$ and $t \notin R = S$.
- Since applying the labeling algorithm to the flow f introduces no node of T into R , no edge from S to T has excess capacity, and no edge from T to S has nonzero flow in f . Hence $f^+(S) = \text{cap}(S, T)$ and $f^-(S) = 0$. ✓
- Since the net flow out of any set containing the source but not the sink is $\text{val}(f)$, we have proved

$$\text{Val}(f) = f^+(S) - f^-(S) = f^+(S) = \text{cap}(S, T).$$

Flow out S →
→ S-cut

✓ ✓ ✓ ✓ ✓

Advanced Graph Theory Network Flow Problems

When terminating this way we claim that S, T is the source to sink cut that is S, T with the capacity of that cut is nothing, but a value f and f is the present flow it is the cut because source belongs to big S , and T does not belong to R is equal to S .

Since applying the labeling algorithm to the flow introduces no node of t into R no edge from S to T has access capacity, and no edge from T to S has non 0 flow in f hence f plus S ; that means, the flow out of flow out of S this is nothing but the capacity of S, T cut, and the flow inflow of S is equal to 0.

Since the net flow out of any set contains containing the source, but not the sink is the value of f , hence we have proved that the value of f is nothing but the flow out of source, and flow into the source is 0, hence the total flow out of source is nothing but the capacity of the S, T cut in the network, and that becomes the max flow value examples of a Ford Fulkerson algorithm.

(Refer Slide Time: 33:01)

Example (1)

Q. 1 Apply the Ford Fulkerson Algorithm to determine the value of maximum flow from the source x to the sink y.

Advanced Graph Theory Network Flow Problems

Let us apply the Ford Fulkerson algorithm to determine the value of the maximum flow from source to sink, sources as an sink is y. So, here let us see the conventions that this is the network, wherein the capacities which are placed on the edge are shown here, direct values and the flow value is shown within the bracket, which are there in the network. So, with this let us see the application of a Ford Fulkerson algorithm.

(Refer Slide Time: 33:50)

Solution:

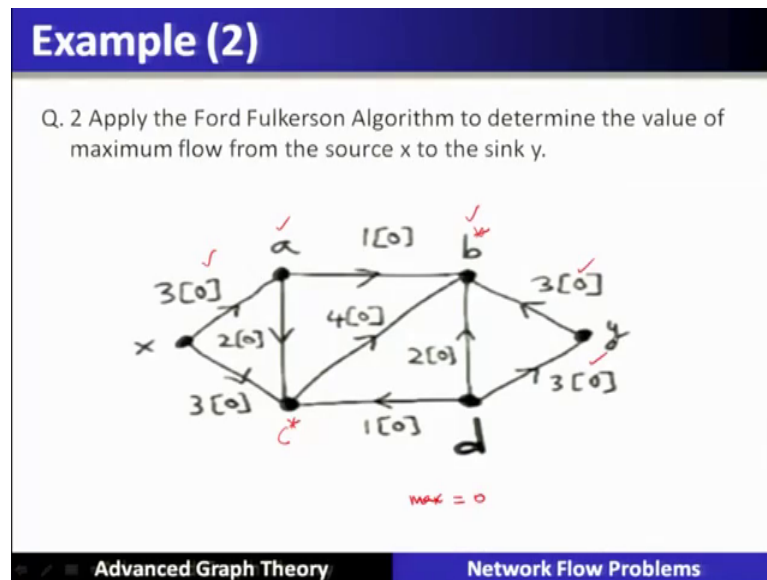
$x \rightarrow a \rightarrow d$
 $x \rightarrow d$

Q	Residual Capacity (forward)	Flow (reverse)	$\Sigma(Q)$
xaby	3,1,3	-	1
xdcy	3,1,3	-	1

Max flow = val(f)
 $= f^+(x) - f^-(x) = 1 + 1 - 0 = 2$

Advanced Graph Theory Network Flow Problems

(Refer Slide Time: 33:58)



Now here in this particular network, we see that from x we can reach to a or we can go to d , or it is already. So, from x we can either go to a , or we can go to d .

Now, $x \rightarrow a$ has the residual capacity because the capacity is 3 and the flow is 0. So, it can undertake 3 units of flow, similarly d also can take 3 units of flow. Now from a we can go to b , and we can also go to d . So, from b from a we can go to b why because there is a possibility of residual capacity of 1 unit. Now from b we can go to y . So, there is a augmenting path $x \rightarrow a \rightarrow b \rightarrow y$ by here there is a 3 unit of flow possible between $x \rightarrow a$ between $a \rightarrow b$, there is a 1 unit of flow possible and between $b \rightarrow y$ there is a 3 unit of flow.

So, the minimum value that is called epsilon is the minimum of all these that is called tolerance that is nothing but 1 unit. So, 1 unit of flow is possible let us inject that, now we will see again through d . So, 3 that means there is a possibility with $x \rightarrow d$ to take a flow from d to c also there is a possibility to take a flow, and from c to y also there is a possibility to take a flow. So, this will become $x \rightarrow d \rightarrow c \rightarrow y$. Now here $x \rightarrow d$ has the capacity 3 of residual, and $d \rightarrow c$ has capacity of 1, and $c \rightarrow y$ has capacity of 3. So, minimum value becomes 1.

So, let us introduce this particular flow in the network. Now let us go ahead again and see an any other augmenting path available or not. So, from x we can go to a , or we can go to d . So, if we go to a then this particular path is already saturated $a \rightarrow b$ is saturated. So, we cannot go ahead from a , from a we can go to d , and from d again we see there is a

saturated there is no saturated path. So, d also we cannot have any progress, similarly from x to d if we go then from d we cannot make any progress.

So, here we stop and we have to take the S T cut. So, the total amount of flow will become the S T cut that is nothing, but the total value which will basically out from the source that is or that is nothing, but the total amount of flow which will go into the sink all are equal. So, let us see that f plus s we can calculate that is 1 plus 1 that equal to 2. So, that is the total maximum flow in this particular network is 2, and that is all shown over here whatever we have already done. So, total flow in this particular example is 2.

Let us take another example let us find out the maximum flow. Now here in this particular example we can see that we can go there is a possibility, we can reach a we can reach b also because a b also, there is a capacity but from b we cannot go to y, because there is no flow which is being undertaken from y to b. So, here we stop so from b we cannot go ahead.

Now there is another way from a we can come down let us say that this is c and from c we can go to b, but from b we cannot go ahead. So, there is no possibility from these particular ends to undertake the flow hence, the max flow will be equal to the S T cut, why because there is no augmenting path, and that is nothing but the flow will be 0 which is there in this particular network.

(Refer Slide Time: 39:23)

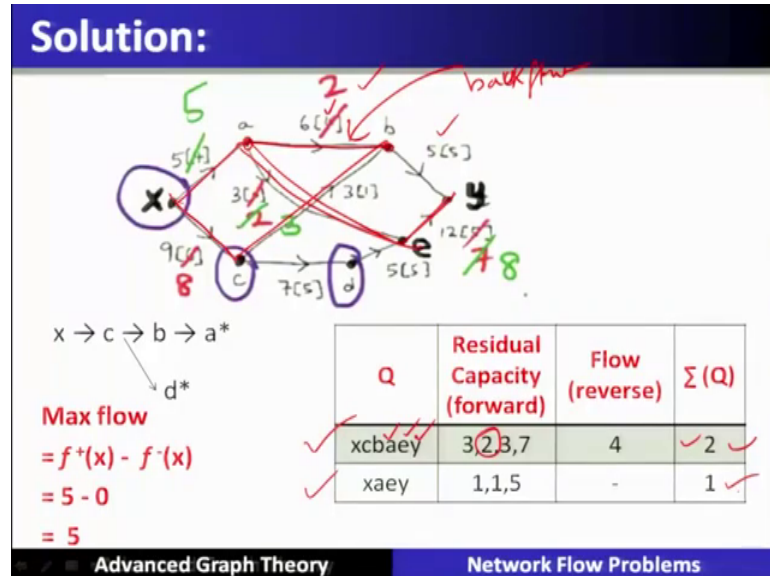
Example (3)

Q. 3 Apply the Ford Fulkerson Algorithm to determine the value of maximum flow from the source x to the sink y.

Advanced Graph Theory Network Flow Problems

Now, in the another example let us find out the max flow in this network using Ford Fulkerson.

(Refer Slide Time: 39:28)



Let us directly go and see this particular work worked out solution, for this particular problem. So, 1 such augmenting path which can introduce a flow of 2 which is injected, and that is x, then c, then b, and then from b we cannot go to y because this particular forward link is saturated, but we can go to a why because there is a flow from a to b. So, we can introduce a back flow, and we can come to a. So, from a we can go to e through the forward edge and from e we can go to y, and this particular way if we take the minimum of all this particular flow. So, the flow of 2 unit is being introduced.

Now, from x we can go to a, from a we can go to e, and from e we can go to y in this way we can also introduce 1 more flow. So, when there is no augmenting path we will find out the min cut, and that will be the maximum flow which will be working out in the network. Now we will see the Mengers theorem, and we will see the proof of Mengers theorem using max flow min cut theorem.

(Refer Slide Time: 41:18)

Remark: Menger from Max-flow Min-cut 4.3.13

- When x, y are vertices in a digraph D , we can view D as a network with source x and sink y and capacity 1 on every edge. Capacity 1 ensures that units of flow from x to y correspond to pairwise edge-disjoint x, y -paths in D . Thus a flow of value k yields a set of k such paths.
- Similarly, every source/sink partition S, T defines a set of edges whose deletion makes y unreachable from x : the set $[S, T]$. Since every capacity is 1, the size of this set is $\text{cap}(S, T)$.
- The paths and the edge cut we have obtained might not be optimal, but by the Max-flow Min-cut Theorem we have

$$\lambda'(x, y) \geq \max \text{val}(f) = \min \text{cap}(S, T) \geq \kappa'(x, y)$$
- Since always $\kappa'(x, y) \geq \lambda'(x, y)$, equality now holds.

Advanced Graph Theory Network Flow Problems

(Refer Slide Time: 41:20)

Menger's Theorem Proof

- **Goal:** Deduce Menger's Theorem (vertex version) from max-flow min-cut

Theorem (Menger's Theorem-vertex version) ✓

Let x, y be two distinct vertices in a connected graph $G = (V, E)$, such that $xy \notin E$. Then $\kappa(x, y) = \lambda(x, y)$

Proof:

We will show that

✓ (1) $\lambda(x, y) \leq \kappa(x, y)$ (use definitions) ✓

✓ (2) $\kappa(x, y) \geq \lambda(x, y)$ (use max-flow min-cut)

$\kappa(x, y)$ is the minimum size of an x, y cut

$\lambda(x, y)$ is the maximum number of pairwise vertex disjoint/internally disjoint paths

Advanced Graph Theory Network Flow Problems

So, this is the idea that lets go ahead and see the Mengers theorem of the vertex version, that says that if x and y be the 2 distinct vertices in the connected graph, such that between x and y there is no direct edge, then kappa of $x y$ is equal to lambda of $x y$. Let us see the proof of this particular theorem.

So, to prove this kappa $x y$ is equal to lambda $x y$ in the Menger theorem, what we will do is we will show this particular proof by two different steps. In the first step we will show that lambda $x y$ is less than or equal to kappa $x y$, and then we will show that kappa

$\lambda(x, y)$ is greater than or equal to $\kappa(x, y)$ using max flow min cut. So, $\kappa(x, y)$ you may be knowing it is a minimum size of x, y cut, and $\lambda(x, y)$ is basically the maximum number of vertex disjoint or internally disjoint x, y path. So, let us see the first one where $\lambda(x, y)$ is less than or equal to $\kappa(x, y)$.

(Refer Slide Time: 43:11)

(1) To show $\lambda(x, y) \leq \kappa(x, y)$: Proof

- Take a minimum (x, y) -cut U .
- Every xy -path must go through U .
- Number of (pairwise) internally disjoint paths is at most $|U|$

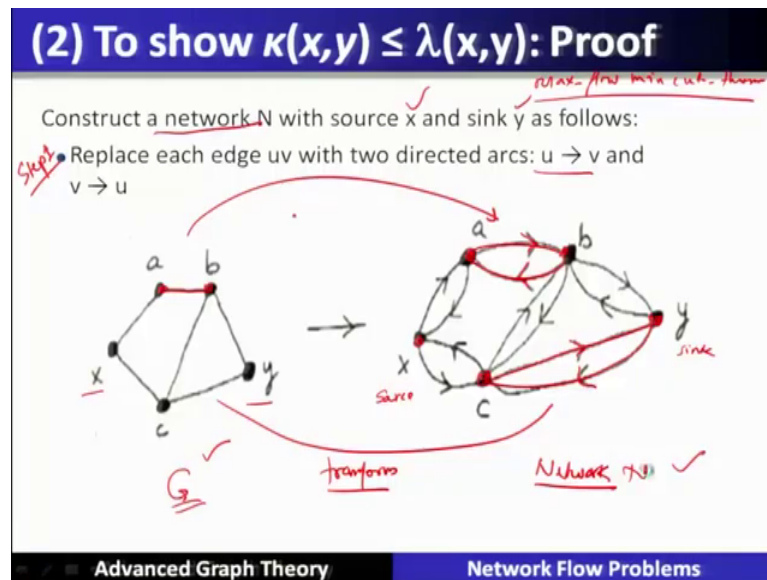
$\Rightarrow \lambda(x, y) \leq |U| = \kappa(x, y)$

Advanced Graph Theory Network Flow Problems

Now, if you take a x, y cut let us call it as U ; that means, in x and y the set of vertices let us call it as U , then if we remove it then x and y they become disconnected, when remove from the graph then it becomes disconnected hence, this U is nothing, but it is an x, y cut. Now if it is an x, y cut, then every x, y path must go through U , a vertex in U . The other path also will go to y through a distinct vertex in U why distinct, because these paths are internally disjoint paths that is between x and y . So, they are internally disjoint that means, no vertex can be common hence; that means, all the paths which will go from x to y they will be internally disjoint paths and all the vertices will be distinct.

So, how many such vertices will be present at least that means, 1 from each particular path, if we take so that means, the if we count how many different paths will be there between x to y that will be nothing but the size of this particular U , and this size of U is nothing but this is called x, y minimum x, y cut. Hence this path of the of the theorem is proved the other path we have to see that is $\kappa(x, y)$ is less than or equal to $\lambda(x, y)$. Now this we are going to prove using max flow, and min cut theorem.

(Refer Slide Time: 45:30)



Now, to prove this using max flow min cut theorem the underlying graph, we have to transform, it into a network to do this we will follow some steps. So, step number 1, so we will construct a network N let us call N out of this particular graph G . So, whenever there is a network as far as max flow min cut theorem there are 2 designated vertices, which are designated as source and sink. Let us say that x and y they are the source and sink respectively.

So, here x and y they become source and y become sink of the network, the second part says that for each edge $u v$ each edge of a graph, we will add 2 directed arcs $u v$ and $v w$ that is let us take an example that if there is a edge $a b$, then we have to add 2 arcs a to b and b to a with these particular directions.

Similarly, all other edges will be transformed in this particular manner, let us do it again for $c y$ between c and y there will be an edge from c to y , and from y to c so, we will transform this graph into this particular network, this is the step number 1. Let us see the second step.

(Refer Slide Time: 47:37)

Step 2

- Split each vertex $w \notin \{x, y\}$ into two vertices w^- and w^+ , together with a (new) directed edge from w^- to w^+ .
- Such an arc $w^- \rightarrow w^+$ is called an **internal arc** of the network.
- Other arcs $u \rightarrow v$ will be replaced by (Step 1):
 - $u^+ \rightarrow v^+$ if $u, v \notin \{x, y\}$;
 - $u \rightarrow v^-$ if $u = x, y$;
 - $u^+ \rightarrow v$ if $v = x, y$;

Diagram: A box containing u^+ and u^- on the left and v^+ and v^- on the right. Arrows indicate connections: $u^+ \rightarrow v^+$, $u \rightarrow v^-$, and $u^+ \rightarrow v$.

Advanced Graph Theory Network Flow Problems

In step number 2, what we will do we will take every vertex which is not the source and sink, and we will split this particular vertex w into two vertices w^+ and w^- , and we will place an edge between them from w^- to w^+ this edge is called internal arc of the network.

Now, other arcs will be replaced by u^+ that means, other arc means other than other than this internal arc all other arcs which are present in the graph, or in the network which we have obtained in a step number 1, will be replaced by from u^+ there will be an edge to v^- , if u and v they are not neither it is source nor sink. So, from u^+ to v^- we have to add an edge. Similarly if u is a source or a sink, then this edge will be from u to v^- . Similarly if v is a source of sink, then there will be an edge from u^+ to v . So, there are 3 different type of edges we will include, and we will split the node, and we will add internal arc now, after doing this we will obtain a network.

(Refer Slide Time: 49:44)

Imp • Every internal arc will have a capacity of 1, and other arcs will have a capacity of n (which is the number of vertices).

$G \longrightarrow N$

- In Graph G , there are 5 vertices all together. Therefore for every arc which is not an internal arc in network N , we have a capacity of 5 where as all internal arcs assigned the capacity 1.

Advanced Graph Theory Network Flow Problems

And then we will now place the capacities; that means, for every internal arc we will we will add a capacity of 1. So, internal arc if you recall it is from a minus to a plus b minus to b plus c minus to c plus there are 3 different internal arcs, the capacities we will include as 1, and all other edges will have the capacity of N . So, in this particular example here N is equal to 5. So, capacity of 5 will be included. So, this is the third step. So, after third step we will obtain this network out of this particular graph.

(Refer Slide Time: 40:33)

Observation:

- (1) Every vertex of the form w^- has **exactly** one arc going out from it, which is the internal arc $w^- \rightarrow w^+$
- (2) Every vertex of the form w^+ has **exactly** one arc coming into it, which is the internal arc $w^- \rightarrow w^+$

(one internal arc $w^- \rightarrow w^+$)

Approach: Show

Step 1 (1) $\lambda(x,y) \geq \max \text{ flow}$ ✓
Step 2 (2) $\kappa(x,y) \leq \min \text{ cut}$ ✓

$\kappa(x,y) \leq \lambda(x,y)$ ✓
by all edges N

Max-flow min cut \implies
 $\kappa(x,y) \leq \min \text{ cut} = \max \text{ flow} \leq \lambda(x,y)$

$\kappa(x,y) \leq \lambda(x,y)$

Advanced Graph Theory Network Flow Problems

So, we will see the observation that every vertex of the form w minus has exactly 1 arc going out from it that is this arc is present, but only 1 arc will be between w minus to w plus. Similarly, every vertex of the form w plus has exactly 1 arc coming into it, and which is an internal arc. So, if this is w minus, and this is w plus so, 1 arc is going out from w minus similarly 1 arc is coming into w plus which is an internal arc.

So, only 1 internal arc is present between w minus to w plus. Now what we will do is we will show to prove that $\kappa(x, y)$ is less than $\lambda(x, y)$ using the network, and which we have formed, we will use it to show that $\kappa(x, y)$ is less than or equal to $\lambda(x, y)$ to show $\kappa(x, y) < \lambda(x, y)$, we will again further take 2 steps, step number 1 we will show that $\lambda(x, y)$ is at least max flow, and then we will show in step number 2 that $\kappa(x, y)$ is at most min cut.

And then what we will do is you know that max flow is equal to min cut, now you know that if we can prove that $\kappa(x, y)$ is at most min cut, and min cut is equal to max flow by that particular theorem, and also we know that the max flow is less than or equal to $\lambda(x, y)$ therefore, we can show that $\kappa(x, y)$ is less than $\lambda(x, y)$.

So, hence it remains to show that $\lambda(x, y)$ is greater than or equal to max flow, and also it remains to show that $\kappa(x, y)$ is less than or equal to max min cut. So, these 2 things we are showing, we are going to show and once we will show that then basically the theorem will be proved or you will prove that $\kappa(x, y)$ is less than or equal to $\lambda(x, y)$, and Menger theorem will be proved accordingly.

(Refer Slide Time: 53:28)

To show that $\lambda(x,y) \geq \text{max flow}$

- Let f be a max flow, with $\text{val}(f) = m$.
- If there is a flow into u^- , then the value must be 1 (since there is only one arc directed from u^- - Observation (1))
- This one unit of flow must travel from x to y .
- m units flow transform into m internally disjoint xy -paths

$\Rightarrow \lambda(x,y) \geq m = \text{max flow}$

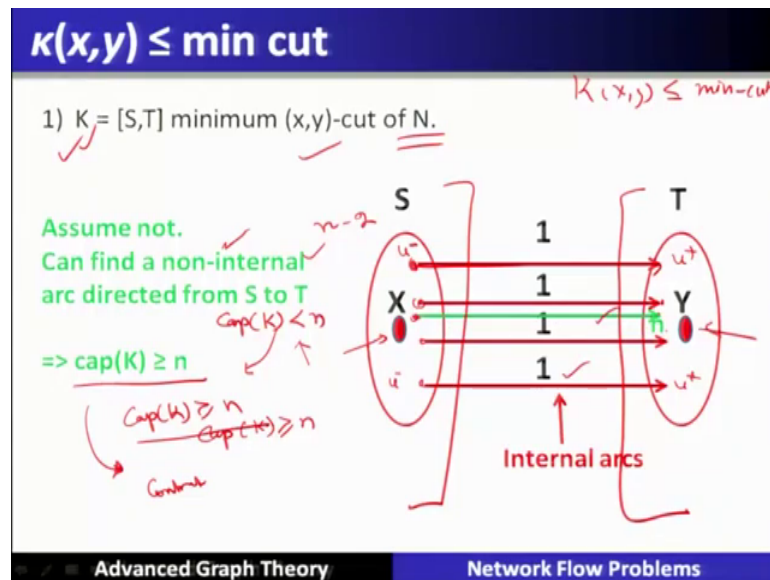
$\lambda(x,y) \geq \text{max flow}$

Advanced Graph Theory Network Flow Problems

Let us see the first thing let us see the first part of this particular proof that lambda x y is greater than or equal to max flow. Now, in the network and which we have formed let m be the maximum flow, and the value of that maximum flow is let us say some value that is m. Now if there is a flow into u minus, then that value must be 1, why because we have internal vertices, there will be only 1 internal vertices from u minus it will go to u plus and that capacity is 1, the previous observation. So, this if we trace back this particular flow of 1 unit, then we will find that this 1 unit must travel from x to y that is from source to the sink.

So, if m units are there so m unit flow will transfer me to m different internally disjoint x y paths. So, this is 1 such paths similarly other such paths must also be there, let us say u 2 minus to u 2 plus and so on. So, there are m internally disjoint path must be there from x to y hence, lambda means lambda x y is nothing, but internally disjoint paths. So, if m are there so the value of lambda must be at least m, and m is nothing but the maximum flow that we have assumed hence we have proved that lambda x y is at least the max flow.

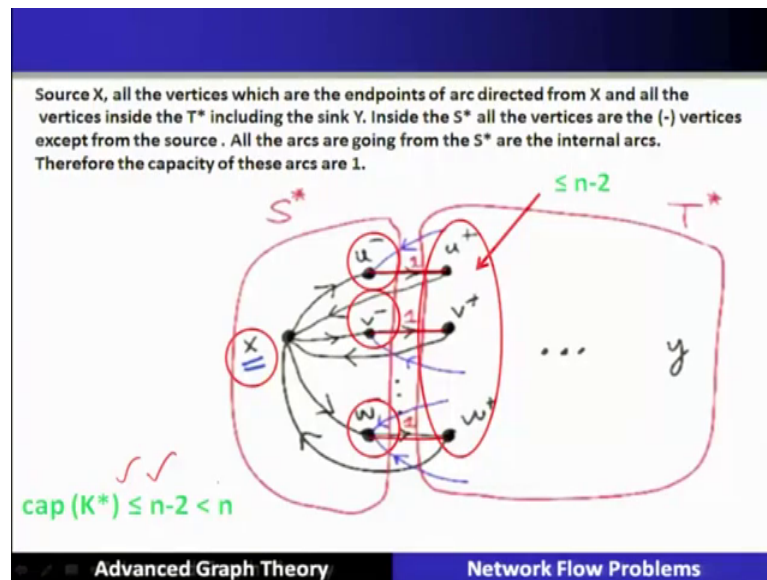
(Refer Slide Time: 55:35)



Now, we will see the remaining part of this particular proof which will see that we have to show that $\kappa(x,y)$ is at most the min cut of that network. Now let us take an S, T cut that is the minimum cut in the network N . So, here if you see an S, T cut what we will find out the arcs which will go from S to T they belong to the internal arc only that is u^- to u^+ , and each is having the capacity of 1 therefore, an S, T cut is basically of size the capacity of S, T cut is let us say k .

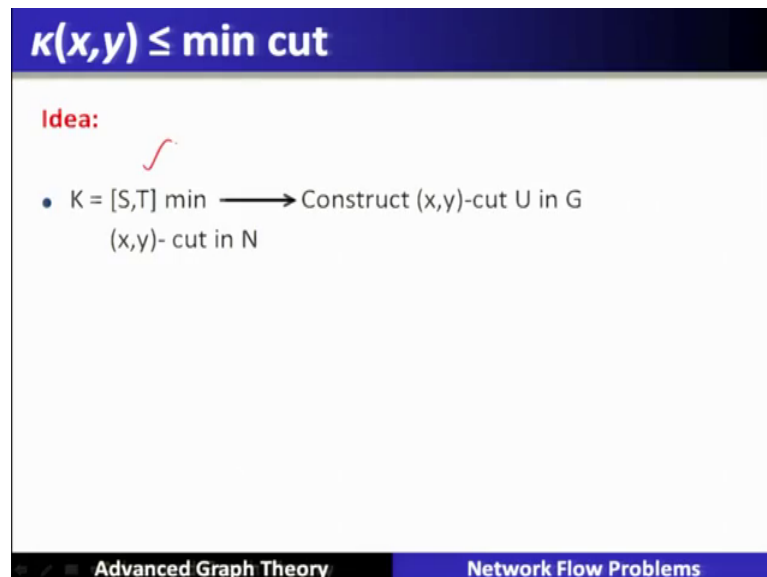
Now, let us see that if let us say that the capacity of S, T cut K , then in that case it should be less than is less than n why because x and y . So, n is the total number of nodes minus 2, then the capacity should be $n - 2$ at most hence strictly it is less than n . Now if some other edges other than the internal edge, basically is present with the capacity let us say n which we have included, then this capacity will not follow this particular bound, then the capacity of k will be greater than or equal to n , we will come to the contradiction hence it is not possible that other than internal edge or non internal edge will be a part of S, T cut.

(Refer Slide Time: 57:49)



So, this particular thing we can see through this particular example that if we find out the S T cut. So, only the internal edges are involved, and each internal edge is having a capacity of 1. So, that capacity of this S T cut is bounded by n minus 2 that is strictly less than N .

(Refer Slide Time: 58:13)



Let us take the S T cut K of minimum size.

(Refer Slide Time: 58:23)

- Let $U = \{ u \in V(G) : u^- \in S, u^+ \in T \}$
- Note that $|U| = \text{cap}(K) = \text{min cut}$.
- Aim: U is an (x, y) -cut in G .
 $\Rightarrow \kappa(x, y) \leq |U| = \text{cap}(K) = \text{min cut}$

$\kappa(x, y) \leq |U|$
 $= \text{cap}(K)$
 $= \text{min-cut}$

Advanced Graph Theory **Network Flow Problems**

In the network and we will see that this particular capacity of that minimum cut is nothing but U . So, what we will do is we will construct a U . So, U is nothing but they are the number of vertices and u minus is there in S and u plus is there in T . So, the cardinality of U is nothing but the capacity of K , and that is nothing but the minimum cut and we will show that U is an x, y cut in G , and that will prove that $\kappa(x, y)$ is at most the size of U and which is nothing, but the capacity of that K and that is nothing, but the minimum cut.

(Refer Slide Time: 59:17)

$P = x - u_1 - u_2 - u_3 - \dots - u_t - y$

Path P from source x to y . Start at source x then followed by u_1, u_2, u_3 before it reaches to vertex y

- $P = x u_1 \dots u_t y$ be an xy -path in G .
- P can be converted into a directed xy -path \vec{P} in the directed graph D underlying the network N as follows:
 $\vec{P} = x \rightarrow u_1^+ \rightarrow u_2^- \rightarrow u_2^+ \dots \rightarrow u_t^+ \rightarrow y$.

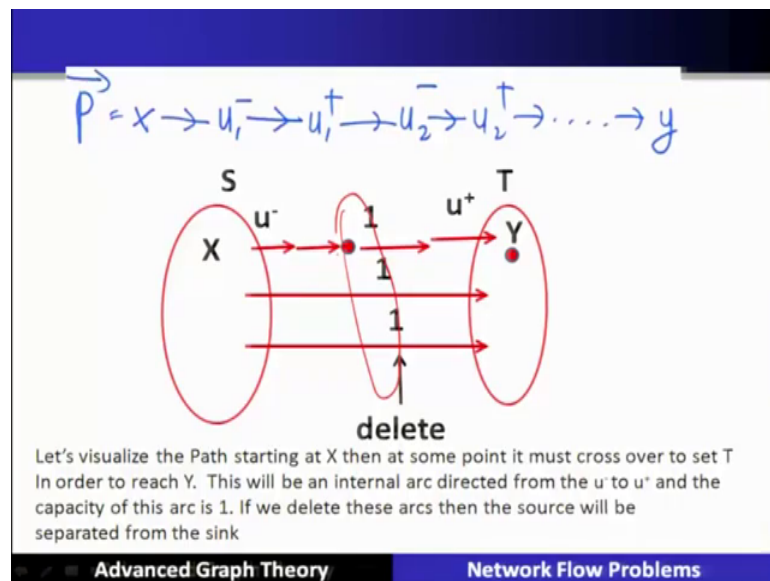
$\vec{P} = x \rightarrow u_1^+ \rightarrow u_2^- \rightarrow u_2^+ \rightarrow \dots \rightarrow u_t^+ \rightarrow y$

Directed Path in network N denoted by \vec{P}

Advanced Graph Theory **Network Flow Problems**

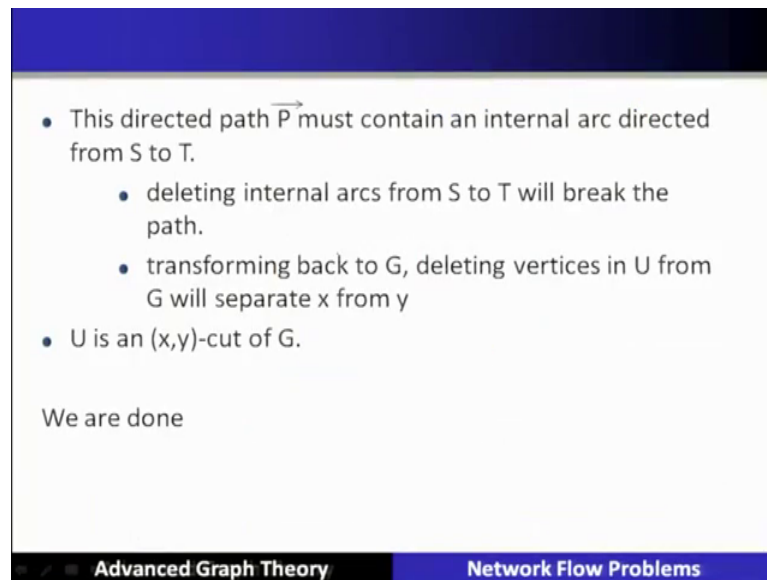
So, that is to be constructed then you will have different paths. So path we will start from source to sink followed by different internal vertices u_1, u_2 and so on up to u_t . Now in this particular path if you see the network, we will have a directed path corresponding. So, starting from the source it will be happening an edge from x to u^- , and from u^- to u^+ this is an internal edge, this arc we have already defined that it will if it is from source so; that means, u_2, u^- so basically source to u^- it will enter and so on. Similarly, here in the last arc will be from u^+ to y .

(Refer Slide Time: 60:22)



Now, if you delete these nodes then it will disconnect S and T hence it is an S T cut.

(Refer Slide Time: 60:33)



A slide with a blue header and footer. The header contains the text "Advanced Graph Theory" and "Network Flow Problems". The main content area is white and contains a bulleted list of three points. The first point states that a directed path \vec{P} must contain an internal arc directed from S to T. The second and third points are sub-points of the first, stating that deleting such arcs will break the path and that deleting vertices in U from G will separate x from y. The fourth point states that U is an (x,y)-cut of G. Below the list, the text "We are done" is written.

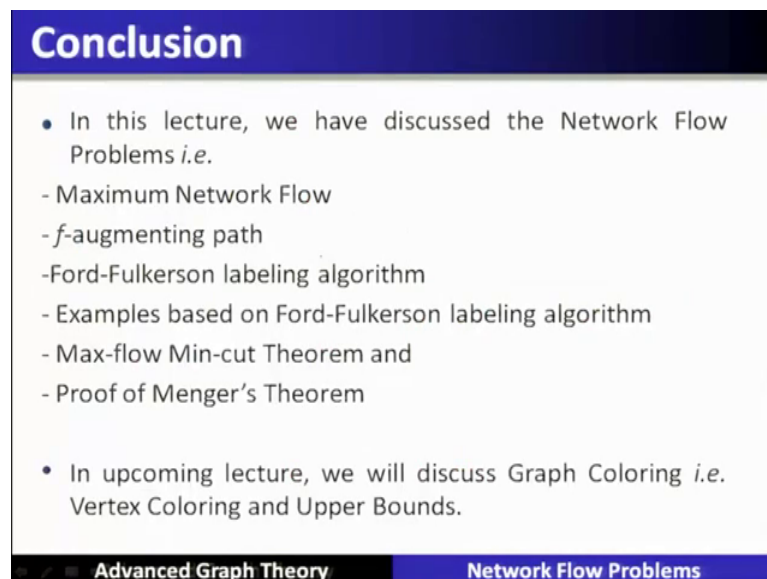
- This directed path \vec{P} must contain an internal arc directed from S to T.
 - deleting internal arcs from S to T will break the path.
 - transforming back to G, deleting vertices in U from G will separate x from y
- U is an (x,y)-cut of G.

We are done

Advanced Graph Theory Network Flow Problems

So, deleting the internal arcs from S to T will break the path transforming back to G deleting these vertices in U from G will separate x and y hence u is an x y cut of G therefore, we have shown that $\kappa_{x,y}$ is at most the minimum cut hence, we have proved this particular theorem conclusion.

(Refer Slide Time: 60:58)



A slide with a blue header and footer. The header contains the text "Conclusion". The main content area is white and contains a bulleted list of two points. The first point states that in this lecture, network flow problems were discussed, including maximum network flow, f-augmenting paths, Ford-Fulkerson labeling algorithm, examples based on Ford-Fulkerson labeling algorithm, Max-flow Min-cut Theorem, and proof of Menger's Theorem. The second point states that in the upcoming lecture, graph coloring (vertex coloring and upper bounds) will be discussed.

Conclusion

- In this lecture, we have discussed the Network Flow Problems *i.e.*
 - Maximum Network Flow
 - f -augmenting path
 - Ford-Fulkerson labeling algorithm
 - Examples based on Ford-Fulkerson labeling algorithm
 - Max-flow Min-cut Theorem and
 - Proof of Menger's Theorem
- In upcoming lecture, we will discuss Graph Coloring *i.e.* Vertex Coloring and Upper Bounds.

Advanced Graph Theory Network Flow Problems

In this lecture we have discussed the network flow problems maximum network flows, f augmenting path, Ford Fulkerson labeling algorithm, we have seen the examples based on Ford Fulkerson labeling algorithm, Max flow min cut theorem, proof of Mengers

theorem, using Max flow min cut theorem. Upcoming lectures, we will discuss Graph Coloring, Vertex Coloring and Upper Bounds.

Thank you.