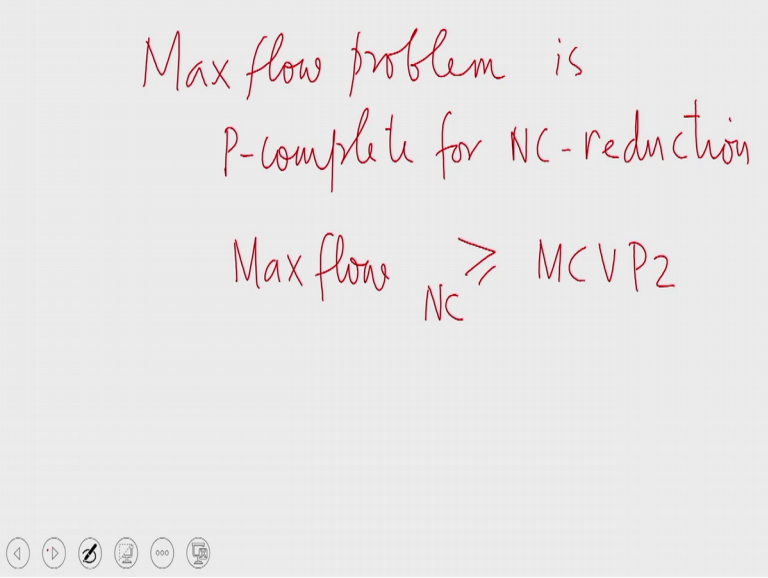
Parallel Algorithms
Prof. Sajith Gopalan
Department of Computer Science & Engineering
Indian Institute of Technology, Guwahati

Lecture – 37
Max Flow is P-complete for NC-reductions

Welcome to the 37th lecture of the MOOC on Parallel Algorithms. Today we shall show that the Max Flow problem is P complete for NC reductions. You must be familiar with the max flow problem. In a max flow problem we are given a flow network. A flow network is a graph with a set of vertices and directed edges and with two designated nodes s and t and then every edge in the graph has got a capacity. And we have to define a flow so that the net flow out of the source which is equal to the net flow into the sink is maximized.

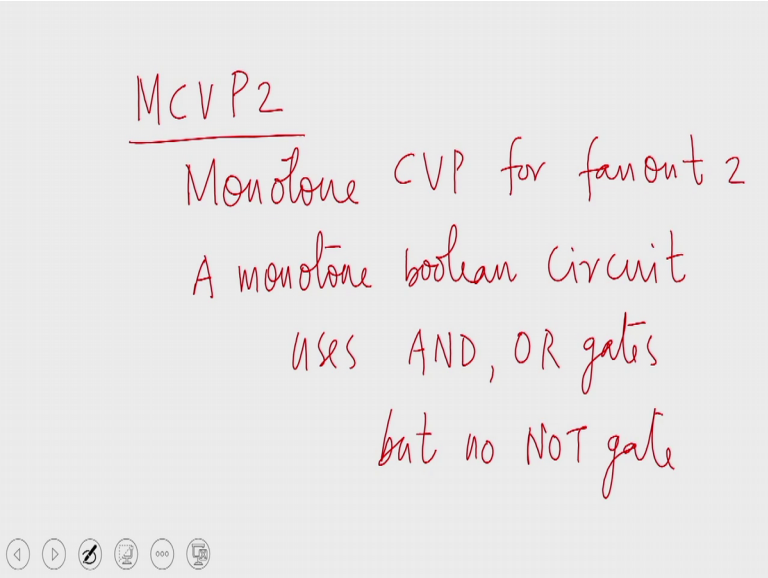
So, the decision version of the problem is what we consider today, here we have to answer questions of the sort given a network and a flow value check if this is the maximum flow the network can have.

(Refer Slide Time: 01:15)



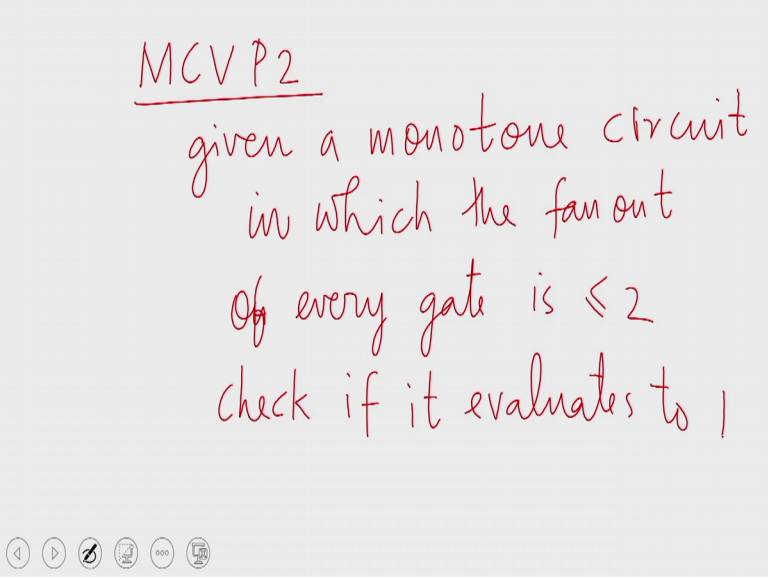
So, we shall show that the max flow problem P complete for NC reductions by showing that the max flow problem, is at least as hard from the perspective of NC reductions to MCV P 2.

(Refer Slide Time: 01:38)



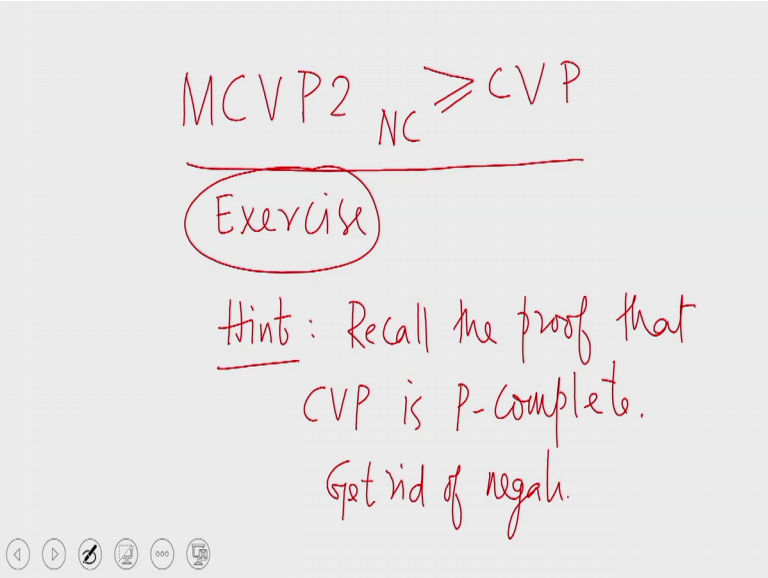
MCV P 2 is a short form for monotone C V P or fan out two a monotone Boolean circuit uses AND OR gates, but no NOT gate. So, what we have to check us this in a monotone CVP problem.

(Refer Slide Time: 02:01)



An MCV P 2 problem we have to check this given a monotone circuit the circuit that uses only AND OR gates in which the fan out of every node, the fan out of every gate is less than or equal to 2. So, given such a circuit check if it evaluates to 1, this is the monotone CVP 2 problem.

(Refer Slide Time: 02:33)

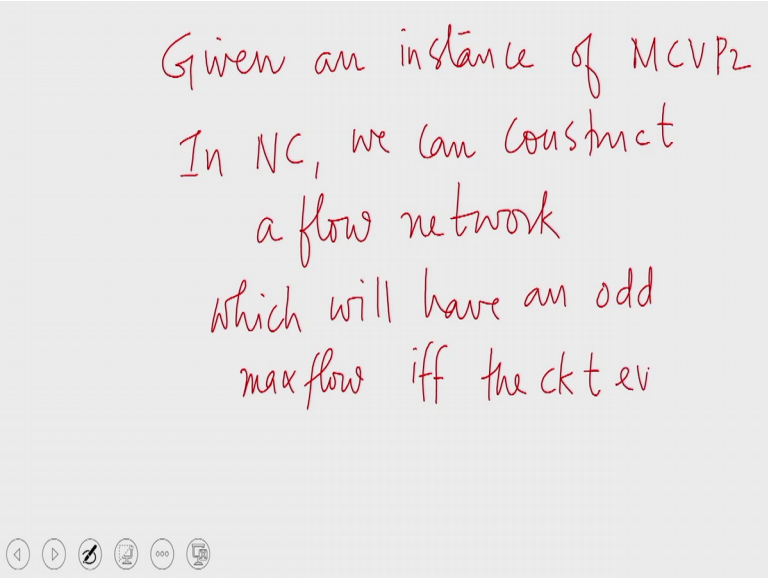


It can be shown that even though monotone CVP 2 is restricted variant of CVP this is at least as hard as the CVP problem from the perspective of NC reductions. I will leave the reduction as an exercise to you, I will give you a hint nevertheless. Recall the proof, that CVP is P complete for NC reductions get rid of negation from the circuit constructor there.

Recall the proof that CVP is a P complete for NC reductions in that proof we assumed we considered a language l an arbitrary language l belonging to p. Since l belongs to p there is a deterministic turing machine that is polynomial time bounded for deciding l, we took the transition table of this Turing machine and used it to construct a circuit. In this circuit we had used NOT gates in one place see how you can get rid of the use of the NOT gate here, that is you have to replace this NOT gate with AND and OR gates. And then you have to ensure that every gate that is used has a fan out of two. Once you do that you would have reduced the CVP problem to MCV P 2 problem.

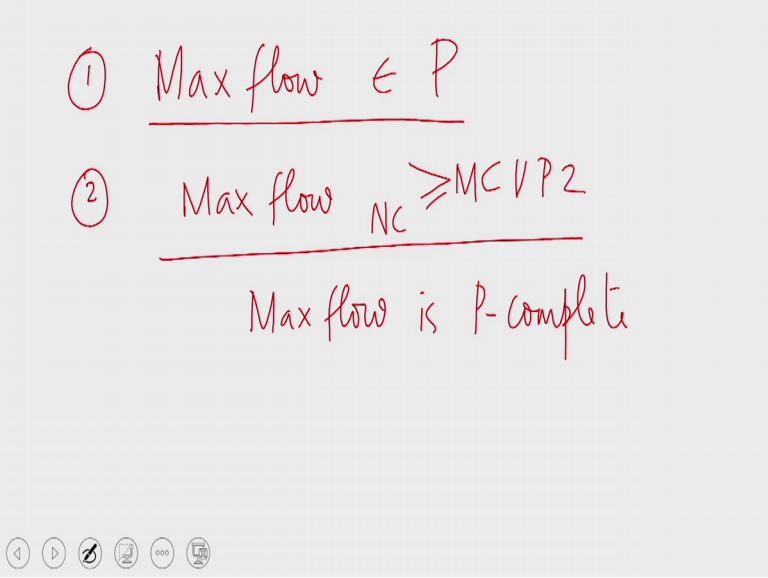
Therefore, we can show that MCV P 2 is P complete for NC reductions it is this proof that we are going to depend on now.

(Refer Slide Time: 03:59)



So, we assume that we are given, an instance of MCV P 2 in NC we can construct a flow network, which will have an odd flow an odd maximum flow if and only if the circuit evaluates to one.

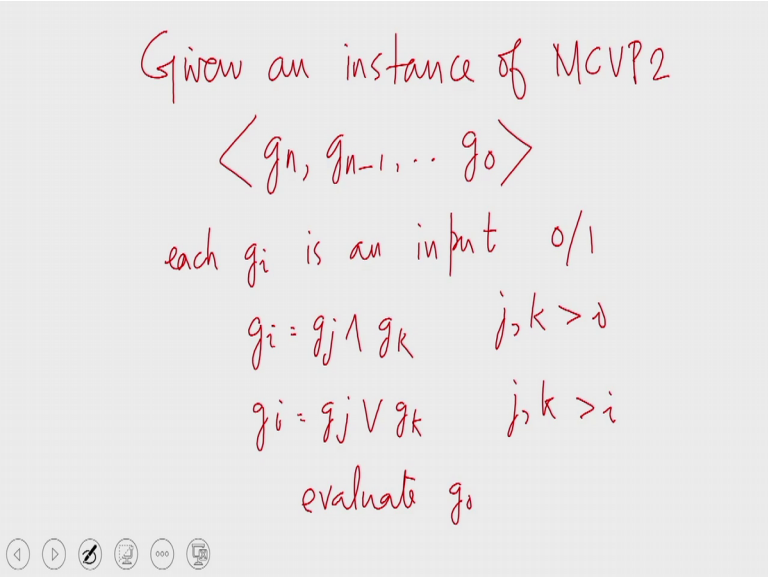
(Refer Slide Time: 04:35)



We of course, know that max flow belongs to P you would have studied a polynomial time algorithm for finding the max flow of a flow network in the first algorithms course. So, that establishes that max flow belongings belongs to P. So, this is the first part of the proof, the second part of the proof will show that, MCV P 2 NC reduces to max flow,

when you combine these two parts we would have established that max flow is P complete.

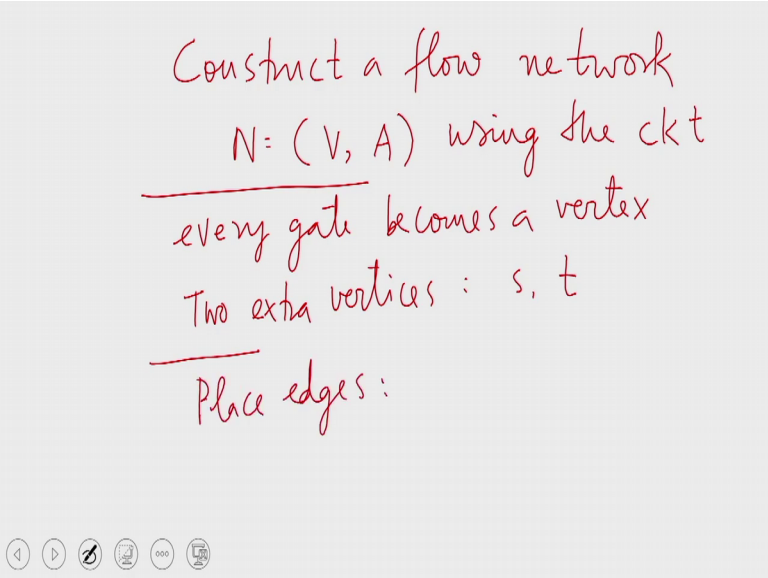
(Refer Slide Time: 05:08)



So, now let us prove the second part. So, we are given an instance of MCV P 2 let that this instance be represented like this we are numbering the gates from the other end. So, the last gate will be now numbered g 0 in this each g i is an input 0 or 1 or it is the AND of g j and g k where j and k are greater than i. Since the numbering is now from the other end the input side is numbered higher in every AND operation the inputs will have larger numbers than the output or it is g j or g k where again j and k are greater than i.

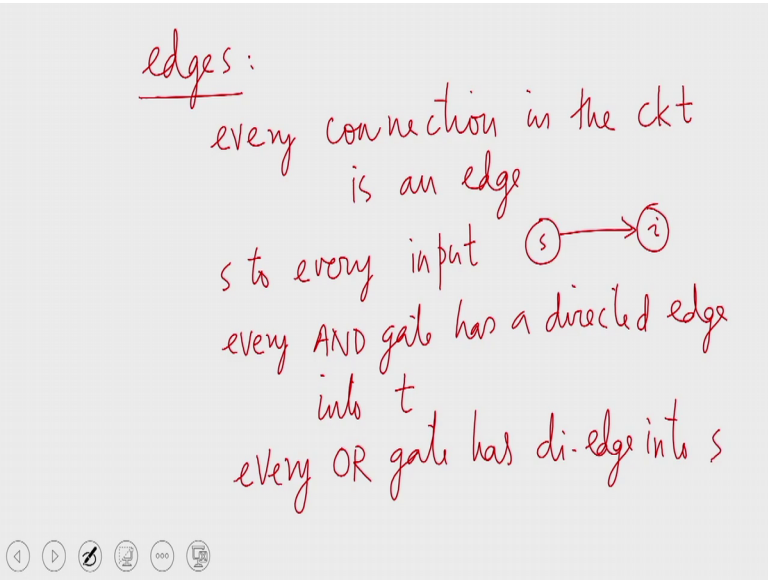
So, we are given such a circuit and the requirement is to evaluate g 0 that is we have to check whether g 0 evaluates to one or not. So, that is the MCV P 2 problem.

(Refer Slide Time: 06:22)



Now, given this instance of MCV P 2 what we are do going to do is to construct a graph a flow network using the inputs of the MCV P 2 circuit. So, in the flow network every gate becomes a vertex and there are two extra vertices named s and t, these will function as the source and the destination of the flow network and then we place edges. Now what are the edges like? Every connection in the circuit is an edge.

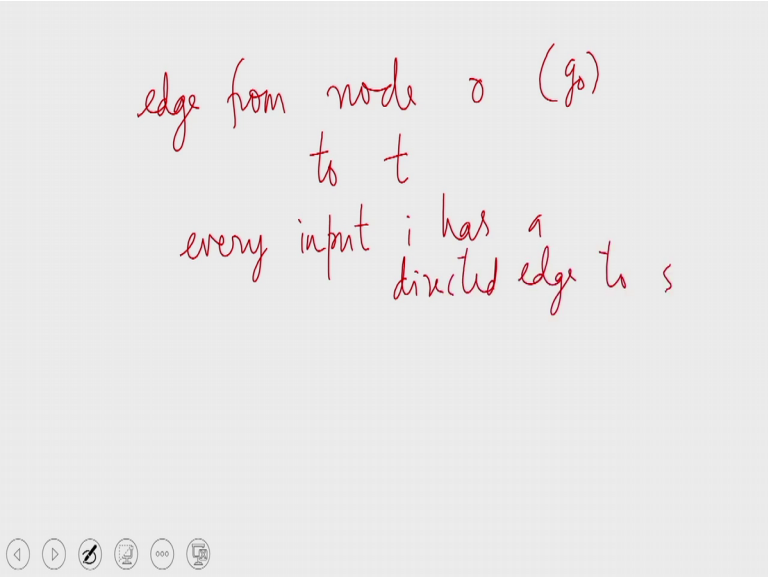
(Refer Slide Time: 06:52)



There is a connection from s to every input gate or an input value, every s will s will be connected to every input. So, this is an edge which is directed from s to i when g is an

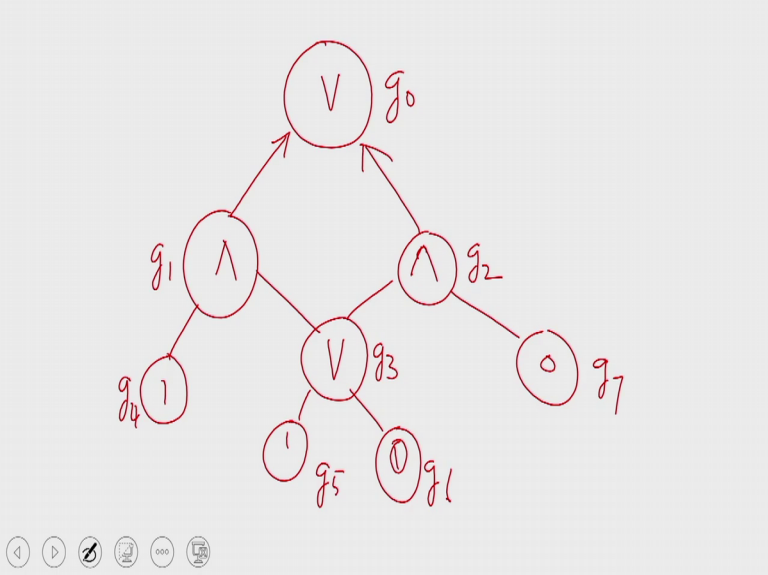
input and then there is an extra vertex t every AND gate has a directed edge in to t and every OR gate has a directed edge in to s.

(Refer Slide Time: 07:39)



And finally, we have an edge from node 0 which corresponds to g 0. So, these are the directed edges of course, there are some more edges every input g i has a directed edge. So, this will form the set of edges of the graph.

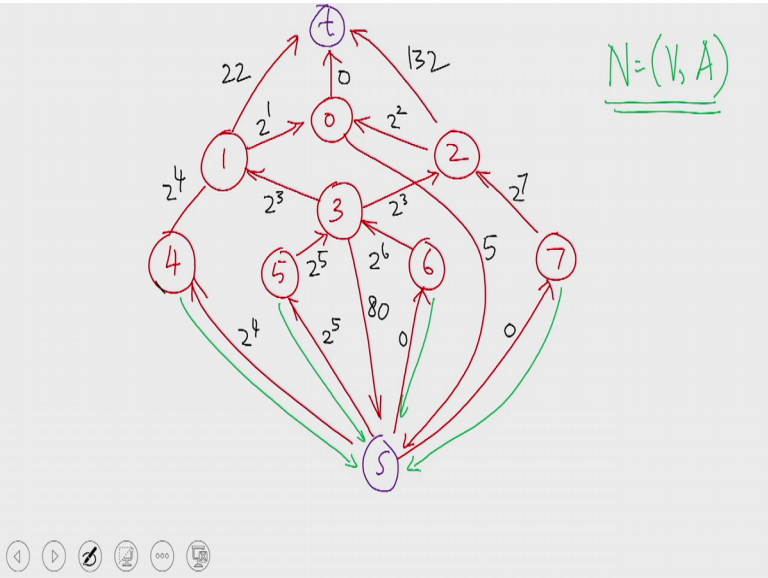
(Refer Slide Time: 08:04)



So, let us take an example let us say we are given the circuit g 0 is an OR gate this has 2 inputs; one input is the output of g 1; g 1 is an AND gate and the other input is the output

of g 2 which is also an AND gate, g 4 is an input there is an OR gate g 3 the output of which will drive an input of g 1 and g 2 each. So, here g 4 g 5 g 6 and g 7 are inputs they have values 110 and 0 respectively. So, let this be 0.

(Refer Slide Time: 08:45)

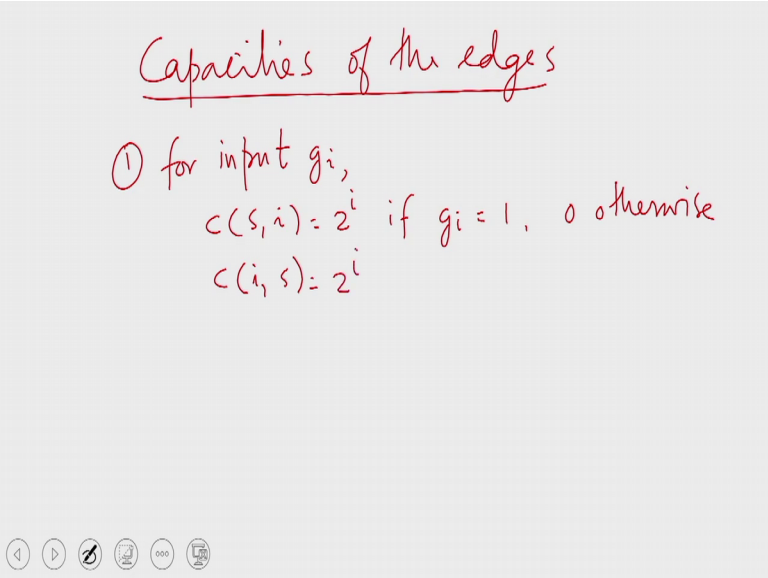


So, this is the circuit that we are given let us say. Then what we do is this for gates 1 and 2 we take vertices like this, this is vertex 0 we have directed edges from 1 to 0 and 2 to 0 these are of course, the circuit edges there is a directed edge from 1 to 0 and 2 to 0.

Similarly, we replicate the circuit, this is g 4 these are the outputs of g 3 the inputs to g 3 are from 5 and 6 respectively. And the inputs to 2 are from 3 and 7 and then we have a new vertex s and a new vertex t and we have connections of the sort from 0 we have a connection to t, from 1 we have a connection to t that is because 1 is an AND gate. Similarly from 2 also we have a connection to t that is because 2 is an AND gate s is connected to 4, 5 and 6 and 7, these are the input values. And we have edges from 3 to s 3 is an OR gate so we have an edge from 3 to s and we also have an edge from 0 to s.

So, this is the flow network that we have, but of course, for the flow network to complete we also should specify a capacity for the edges. Of course, some edges have not been drawn. Here we are going to have edges from 4,5,6 and 7 back to x those edges are not drawn here let me draw them. Now, these are edges going back from the input nodes to s. So, this will complete the flow network n equal to V A; A is the set of all the edges here.

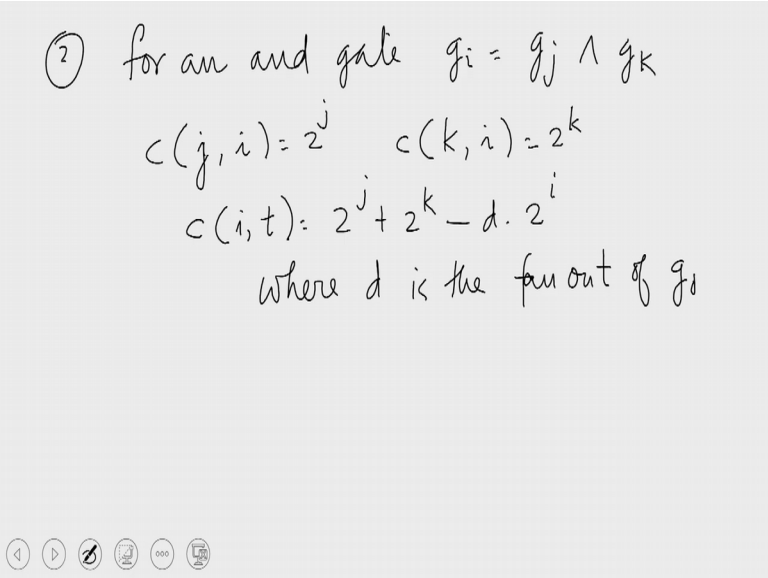
(Refer Slide Time: 10:29)



Now, let us specify the capacities of the various edges. For an input g i capacity of S i is defined as 2 power i. If g i equal to 1 0 otherwise and the capacity of C i S is defined as 2 power i coming back to this figure. Therefore, the capacity of the edge from S to 4 will be 2 power 4 that is because this node is numbered 4 and the capacity of the edge from S to 5 will be 2 power 5.

The capacity of the edge from S to 6 will be 0, that is because 6 is an input 0 and the capacity of the edge from S to 7 is also going to be 0. And the backward capacities are all going to be 2 power 4 2 power 5 2 power 6 and 2 power 7 respectively. So, that is what we have said C S i is 2 power i if g i equal to 0 g i equal to 1 0 otherwise and C i S is 2 power i irrespective of the value of the input.

(Refer Slide Time: 11:39)

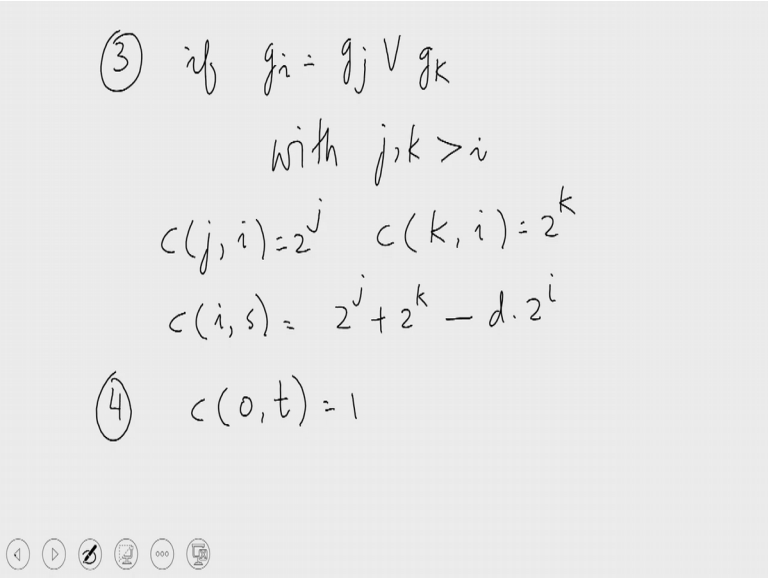


And then for an AND gate g i that is g j into g k we define C j i as 2 pi j C k i as 2 power k and C i t as 2 power j plus 2 power k minus d into 2 power i where, d is the fan out of g i. So, this is the second case for every AND gate we define the capacities of the involved edges in this manner. So, going back to our figure the AND gates here are g 1 and g 2. So, capacities for g 1 and g 2 will be defined in this manner g 1 has a capacity of 2 power 1 and a capacity of 2 power 2 on these edges, that is the outgoing edge of g 2 will have a capacity of 2 power 2. And C i t would be 1 and 2 are the AND gates here.

Therefore, for the incoming edges the capacities would be 2 power 4 here, 2 power 3 here for this gate the j and k values are 4 and 3 respectively. So, the capacities on those edges would be 2 power 4 and 2 power 3 here for gate 2; the capacities would be 2 power 3 and 2 power 7.

So, these are the incoming edges into the AND gate and for the edge from i to t the capacity is going to be 2 power j plus 2 power k minus d into 2 power i. Here for gate 1 d is 1 its driving only the input to gate 0 therefore, the capacity here is going to be 16 plus 8 24 minus 1 into 2 which is 22 whereas, for gate 2 the capacities the incoming capacity is 8 plus 128 which is 136 and there is only 1 input being driven by the output of gate 2 which is that of gate 0 therefore, d equal to 1 for gate 2. So, d into 2 power 2 is 2, so, 136 minus 4 is 132. So, the capacities of the edges involved with AND gates are marked thus.

(Refer Slide Time: 14:12)



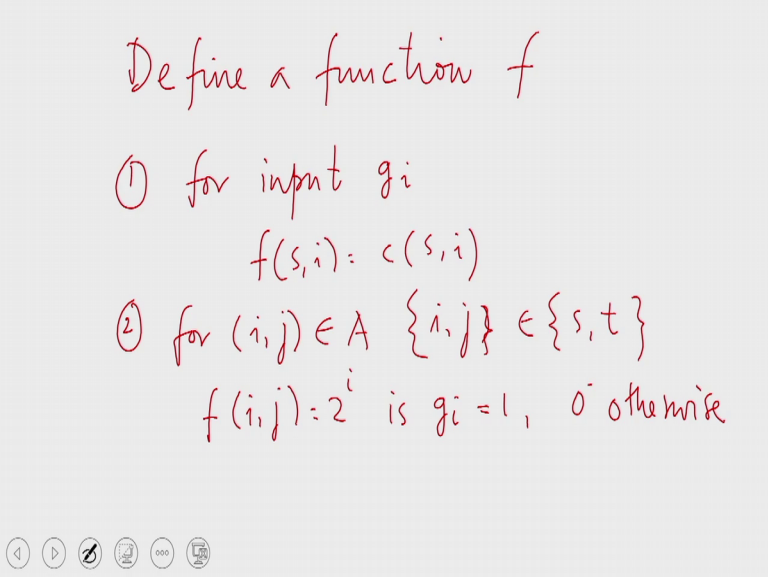
Then for the OR gates if g i is an OR gate with j k greater than i then the capacities are defined like this C j i is 2 power j and C k j k i is 2 power k these are exactly as in the case of the AND gates. But now we have an edge back into the sink the capacity of this edge is going to be 2 power j plus 2 power k minus d into 2 power i and finally, we define the capacity of z 0 t to be 1.

So, marking these capacities in the figure you have to consider the OR gates. So, which are the OR gates here 3 is an OR gate and 0 is an OR gate. So, when you consider 3 it is a incoming edges have capacities of 2 power 5 and 2 power 6 respectively. And the outgoing edges as we as have been marked have capacities 2 power 8 and 2 power 8 respectively. Therefore, the edge which is going back from 3 to s has a capacity of 80; 2 power 5 plus 2 power 6 minus 2 into 2 power 3 this 80 that is the capacity of the edge from 3 to s.

Now, coming to vertex 0, these edges have capacities of 2 power 1 and 2 power 2 the edges from 1 to 0 and 2 to 0 have capacities of 2 power 1 and 2 power 2 respectively. And according to the 4th clause the edge from 0 to t has a capacity of 0 and according to clause 3 the edge from 0 to s has a capacity of 5. Now, the capacity of every edge is marked except the green edges that are from the inputs to the source for these edges the capacity would be 2 power the ordinal value of the input.

So, for the edge from 4 to s the capacity would be 2 power 4 and the edge from 7 to s will have a capacity of 128. So, this is how we assign the capacities. So, this is a flow network.

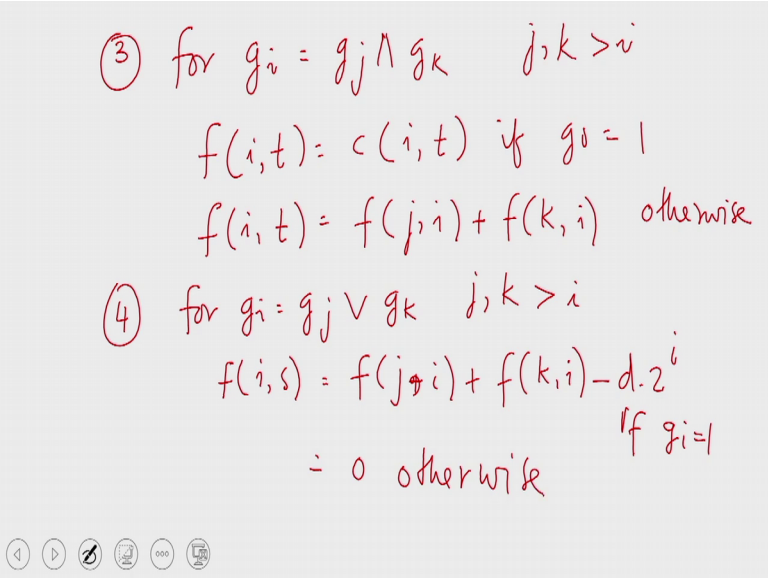
(Refer Slide Time: 16:31)



Now, once the capacities are assigned we define a function flow f, after defining the function f we shall claim that f is a flow and that it and shall show that it is in particular a max flow. So, we defined flows in this manner for an input g i, f of s i is defined as c of s i.

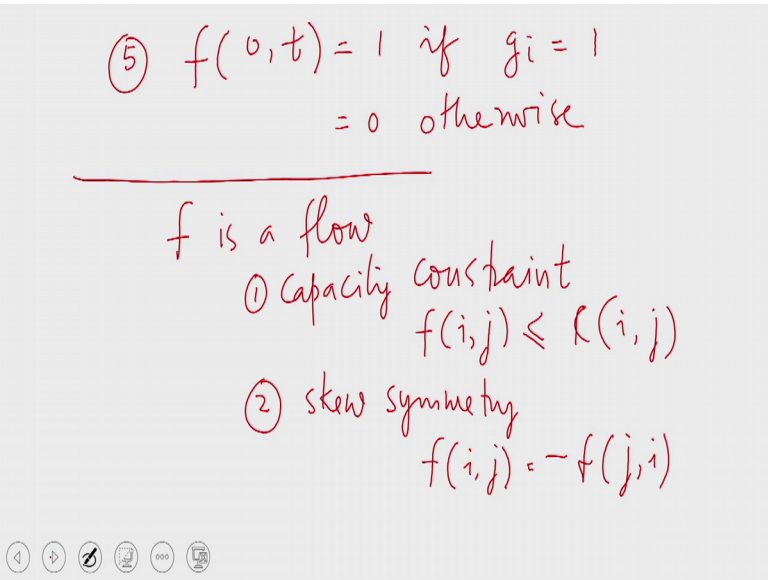
So, all the edges from s to i are saturated for any i j belonging to the network where neither i nor j is s or t; we define f of i j as 2 power i if g i evaluates to 1, 0 otherwise. So, in step two we are considering the edges that are in fact, edges of the network every network edge every network connection became an edge in the graph. And these are the edges that do not involve s or t, for these edges we define the flow in this manner for an edge that is from i to j. So, this is from the output of gate i to an input of gate j for this edge we define the flow as 2 power i if g i evaluates to one otherwise the flow is defined as 0 f is 0.

(Refer Slide Time: 17:52)



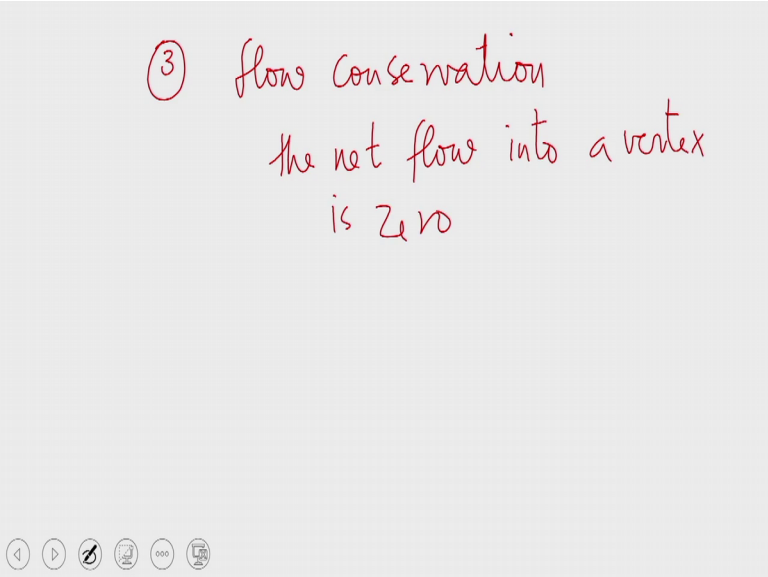
Now, let us come to the edges that involve s or t, for g i equals g j and g k an AND gate where j and k are greater than i. We define f i t as c i t if g i equal to 1, f i t as f of j i plus f of k i otherwise. So, here we consider all the edges that are from the AND gates to the sink t. Similarly let us consider the Or gates we define f of i as the flow back into s as f of j plus i sorry f of j i plus f of k i, minus d into 2 power i. If g i equal to 1 equal to 0 otherwise where, d as before is the fan out of the node.

(Refer Slide Time: 18:53)



And finally, we define f of 0, t as 1 is g i equal to 1 as 0 otherwise. So, this is the definition of a function so we have defined a function f in this manner. Now we claim that f is a flow for f to be a flow it should satisfy 3 conditions; one is the capacity constraint. The capacity constraint says that the flow from i to j for every edge i j must be less than the capacity of the edge less than or equal to the capacity of the edge. Then the requirement is that of skew symmetry which says that the flow from i to j is the negative of the flow from j to i.

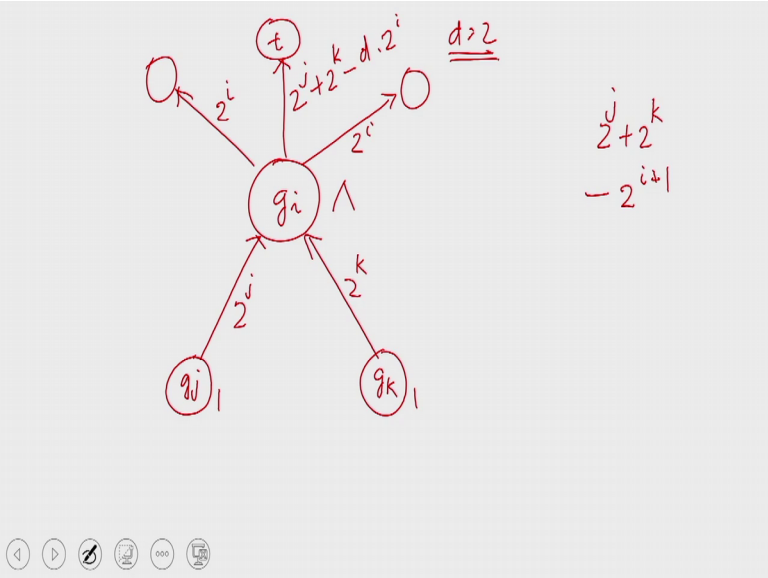
(Refer Slide Time: 19:41)



And finally, the third condition is that of flow conservation that is the net flow into a vertex. So, when these conditions are satisfied we say that f is a flow; now you can easily verify that these conditions are satisfied. Compare the capacity function that we defined earlier and the flow function that we have now we find that on every edge the flow that we have defined is less than the less than or equal to the capacity of the edge. Some of the edges are saturated for example, the edges from the source vertex s to the input signals these are all saturated, but some edges are not, but everywhere the flow that we have defined is less than or equal to the capacity.

So, the capacity constraint is satisfied. And then you can verify that the flow conservation is satisfied everywhere and the way we have defined the flows q symmetry is also satisfied. So, talking about the flow conservation we can look at a few examples.

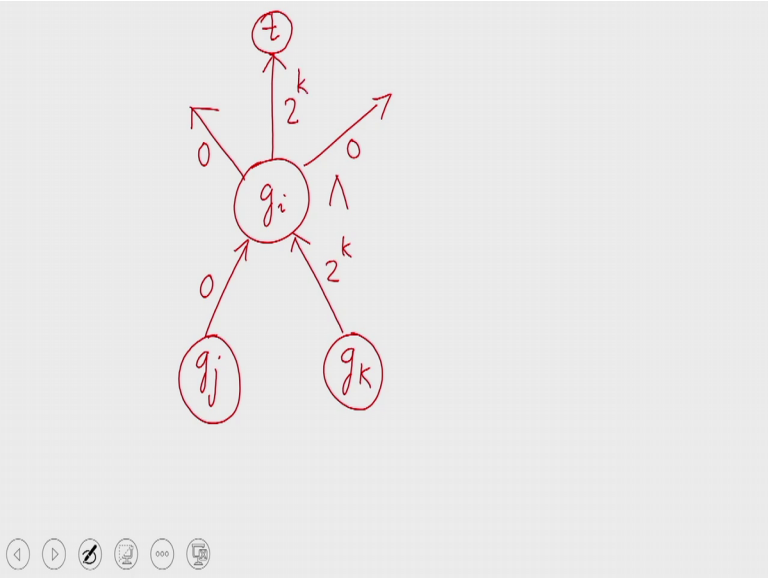
(Refer Slide Time: 20:41)



Consider a gate g i let us say this is an AND gate let us say it has inputs from g j and g k where j and k are greater than i. Let us say both g j and g k evaluate to 1 if both of them evaluate to 1, then they would be putting outflows of 2 power j and 2 power k on these edges.

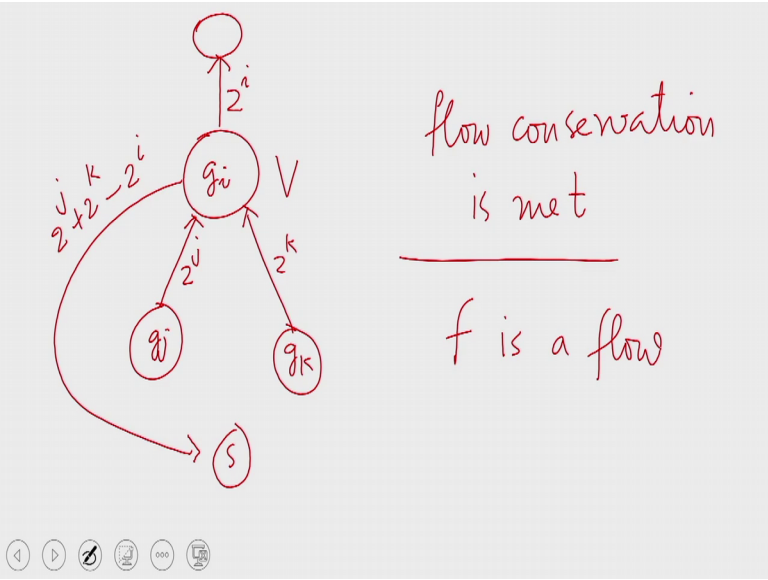
So, the total incoming flow into g, i is 2 power j plus 2 power k; and let us say g i drives 2 inputs, since g i evaluates to 1 here on these edges it would be putting out a flow of 2 power i each. So, the net inflow is 2 power j plus 2 power k and the net outflow is 2 power i plus 1 and the edge that is from g i to t is going to get 2 power j plus 2 power k minus d into 2 power i. So, we find that all these edges are saturated when g i is an OR gate with 2 input signals both are 1 and the flow conservation is met d is 2 here, because this gate drives 2 inputs, so, every edges saturated.

(Refer Slide Time: 21:55)



Instead, if g i were an AND gate with 1 0 input and 1 input that is let us say g k evaluates to 1, but g j evaluates to 0 therefore, the flow put out by g j on this line is going to be 0. Therefore the total inflow is 2 power k, since the gate evaluates to 0 it would put out 0 on both the outgoing edges, but the edge to t will have 2 power k thereby maintaining the flow conservation. So, as you can see here f of i t is f of j i plus f of k i the net inflow into the gate when the gate evaluates to 0.

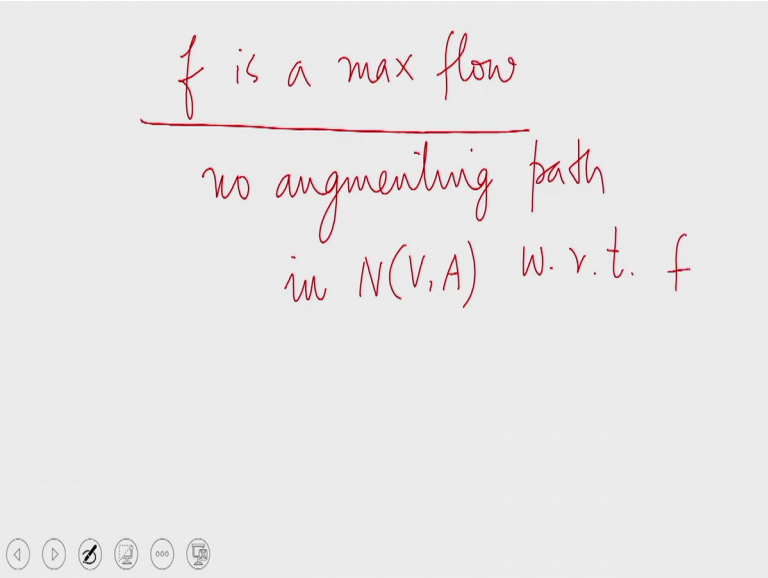
(Refer Slide Time: 22:39)



Another example consider NOR gate let us say g i is an OR gate with both inputs 1 and let us say it drives 1 input on that input it puts out the flow of 2 power i then on the edge which is going back to s it would put out 2 power j plus 2 power k minus 2 power i.

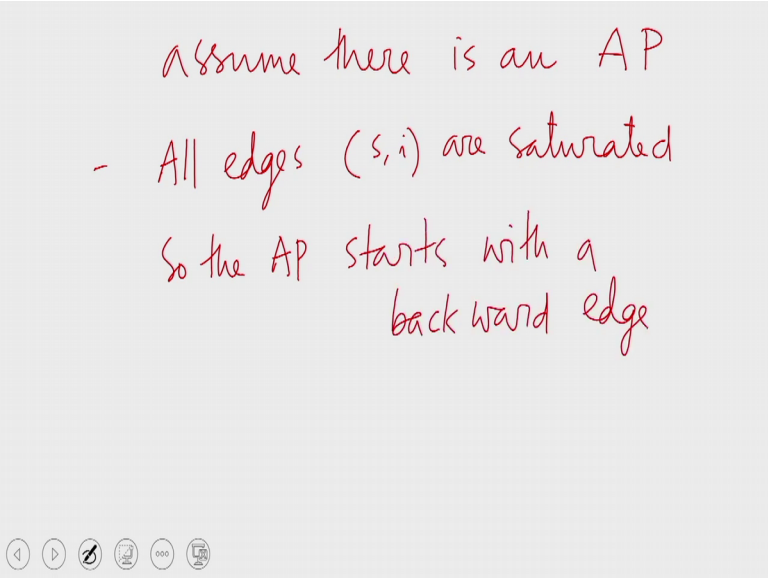
The net inflow minus the outflow, so again flow conservation is satisfied. So, likewise you can work out all the cases and show that flow conservation is therefore, f is a flow f is a flow function.

(Refer Slide Time: 23:16)



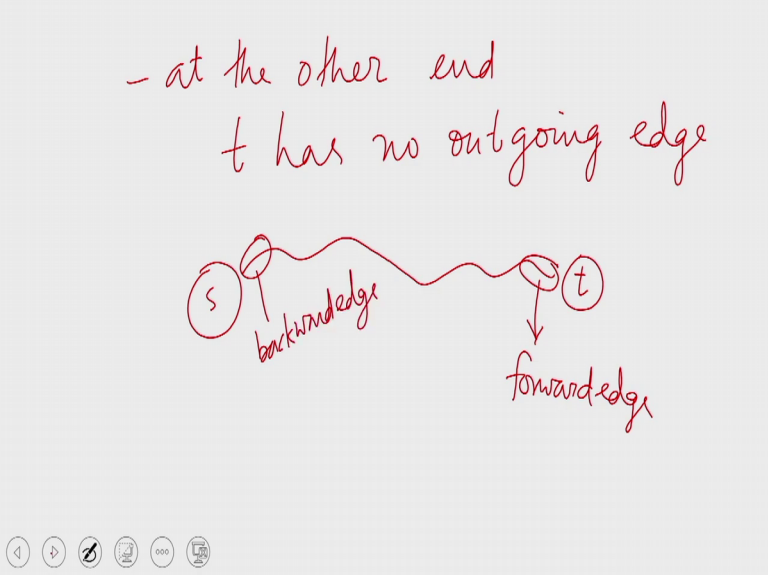
Next we claim that f is a max flow, we know that a flow function f is a max flow in a flow network, if and only if the flow network does not have an augmenting path with respect to this flow. So, there should be no augmenting path in N equal to V A with respect to the flow function f, if f is a max flow this is how it is going to be.

(Refer Slide Time: 23:48)



So, let us assume the contrary assume that there is a augmenting path AP for Augmenting Path. So, we assume that there is an augmenting path.

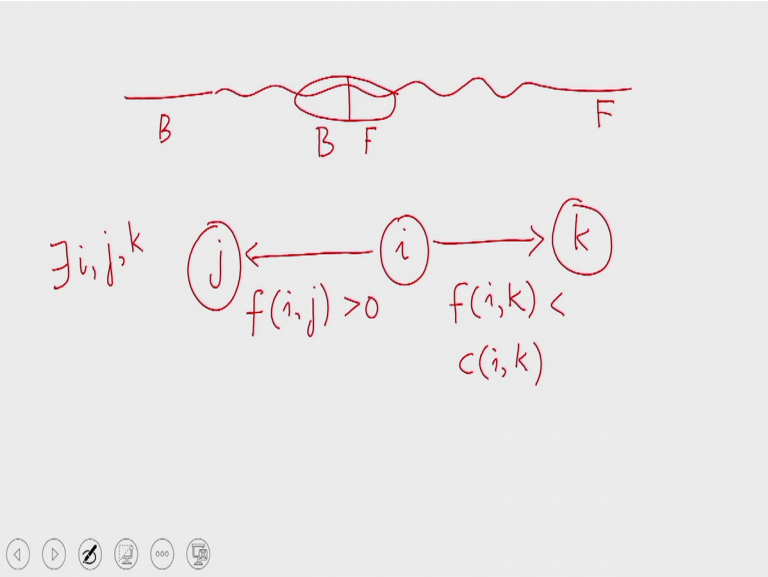
(Refer Slide Time: 24:29)



Now, when you look at the graph we find that, all edges of the form s, i are saturated; if an input is of value 0 then s i has a capacity of 0 and the flow on it is 0 as well. If the input has a value of 1 then the capacity is 2 power i and the flow on it is also 2 power i so all these edges are saturated.

Therefore, the augmenting path starts with the backward edge. Now when you look at the other end, t has no outgoing edge. Now an augmenting path is a path of the underlying undirected graph of the flow network and this path is from s to t and since t has no outgoing edge, we assure that the last edge which is used by this path is a forward edge what we have just now establishes that the first edge is a backward edge.

(Refer Slide Time: 25:00)



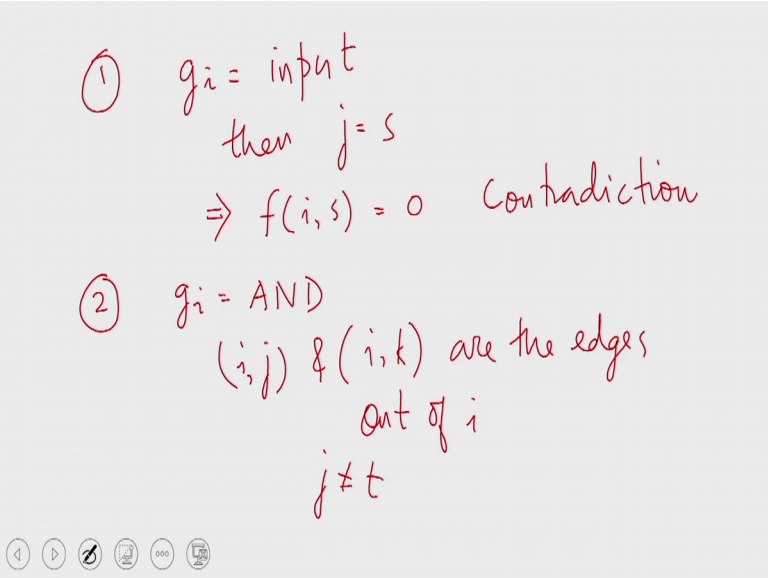
Therefore in this path we are beginning with the backward edge and then ending with the forward edge. So, there must be a pair of consecutive edges somewhere, where the first one is backward and the second one is forward. Since we are beginning with the backward edge and ending with the forward edge at some point we should be switching from backward edges to a forward edge. So, let us pick out one such case, let us say the vertices involved are j, i and k. So, we have backward edge from j to i in the augmenting path we have an edge from j to i which is a backward edge which means the network in fact, has an edge from i to j and a forward edge from i to k.

Now, this is an augmenting path which means, f of i j is greater than 0, the actual flow is from i to j, so we would be pushing more flow along this path by reducing the flow from i to j. On the other hand since i k is a forward edge and it is a part of the augmenting path, we have that its flow f i, k is less than c i, k this edges unsaturated. So, it is possible to push more flow along this path. In total when you push more flow from j to k what you do is this, you reduce the flow from i to j there is an existing flow from i to j which

we reduce and then we push more flow from i to k more real flow is pushed from i to k, that is how we managed to increase the traffic from j to k.

So, what we know is that, there exist vertices i, j, k where on in the augmenting path they appear in the sequence j i k so that this condition is satisfied.

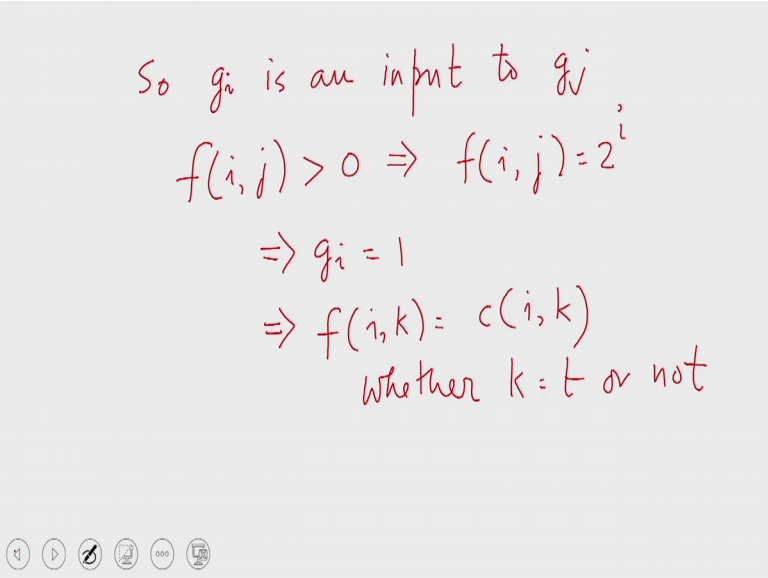
(Refer Slide Time: 26:47)



So, now let us see what this entails this breaks down into 3 cases if g i happens to be an input then j equal to s. The only incoming edge into an input vertex is from s, but then everywhere edge from s to i for an input i is saturated; in other words or we know that the edge from i to s has a flow of 0 this is the contradiction, because what we want is that f of i j is greater than 0.

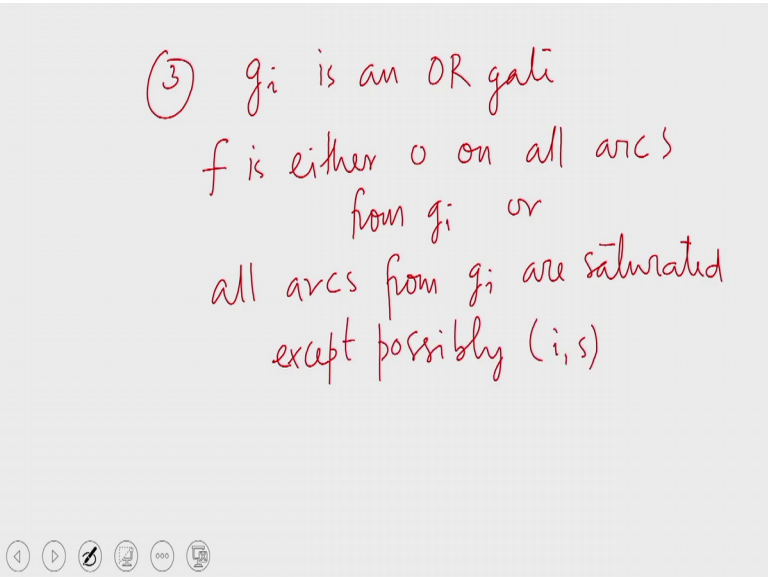
So, if g i is not an input then g i must be an AND gate. So i, j and i, k are the edges going out of this AND gate and we assume that j is not equal to t because in the augmenting path which is directed from s to t, j is on the side of s so j is not equal to t.

(Refer Slide Time: 27:38)



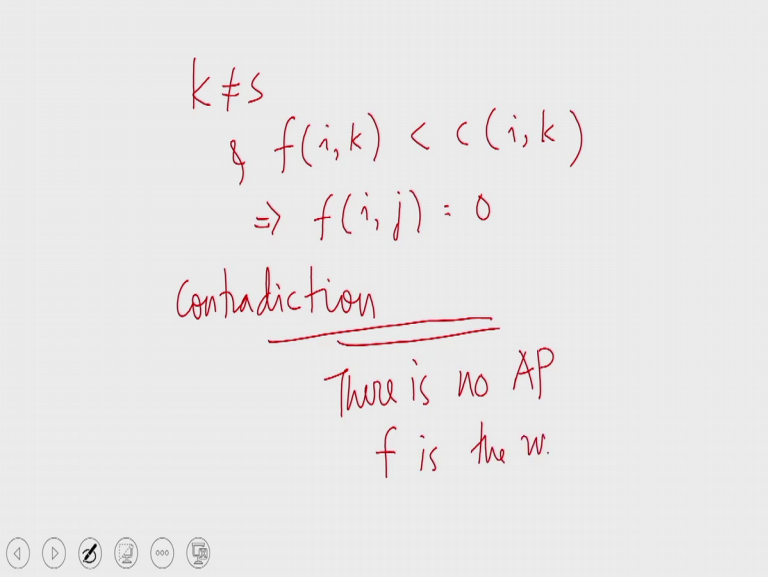
So, g i is an input to g j f of i j is greater than 0 means that, f of i j is 2 power i it can have only two values 0 and 2 power i. So, in this case it is 2 power i which means that g i evaluates to 1, that is when f of i j is 2 power i. If g i evaluates to 1, then f of i k equal to c of i k from the definition of the flows; whether k equal to t or not which means the edge i k is saturated, but that is what we explicitly assumed otherwise we assumed that f of i k is less than c of i, k. So, this is a contradiction again. So, in this case also we get a contradiction.

(Refer Slide Time: 28:24)



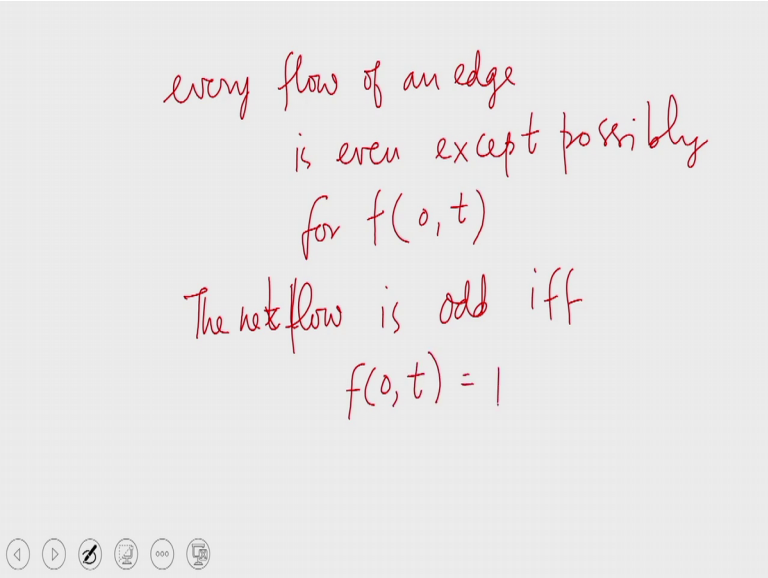
Finally in the third case when g i is an OR gate we see that f is either 0 on all arcs from g i that is when g i evaluates to 0 or all arcs from g i are saturated as we saw earlier except possibly i s.

(Refer Slide Time: 28:48)



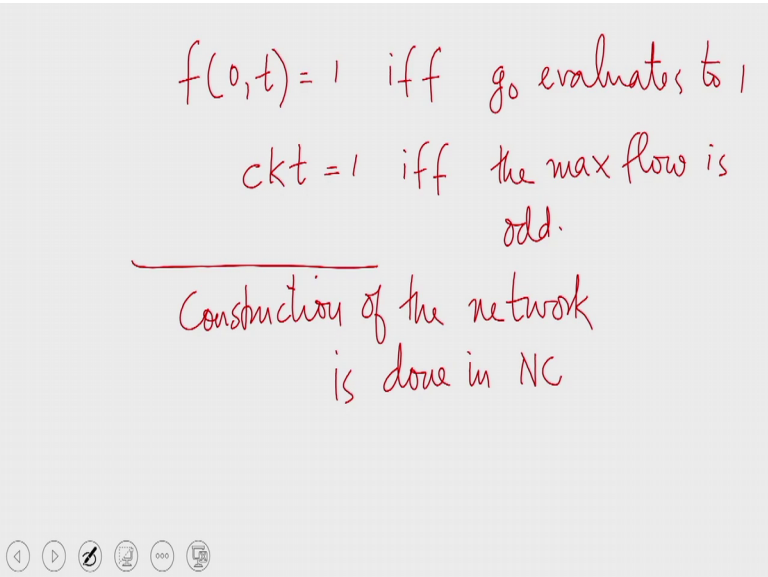
Now, k is not equal to s because k is on the side of t in the augmenting path and f of i, k is less than c of i, k thus we explicitly assumed that the edge i k is unsaturated. But then this means that f of i, j has to be 0 which is a contradiction again because the other assumption that we made is that f of i j is greater than 0. So, either way we get a contradiction that is once you assume that there is an augmenting path in the flow network, there is a contradiction which means there is no augmenting path. And if there is no augmenting path then f is the max flow.

(Refer Slide Time: 29:33)



Now, if you look at f you find that every flow of an edge, that is the f value for every single edge this even except possibly for the edge 0, t; f of 0 t could be 1 or 0 every other flow is a power of 2 or 0. Therefore, the net flow is odd the value of the flow is odd if and only if f of 0 t is 1, but then when is f of 0 t 1.

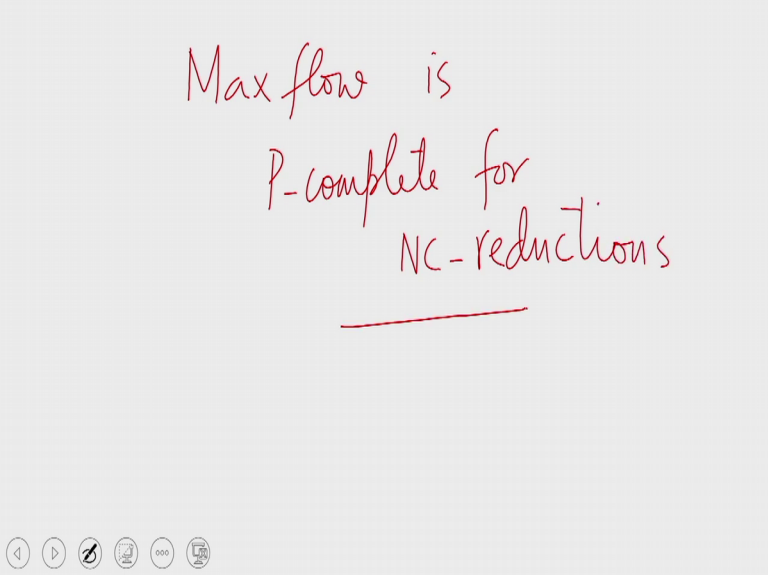
(Refer Slide Time: 30:06)



F of 0 t is 1 if and only if g 0 evaluates to 1 or in other words the given circuit evaluates to 1, if and only if the max flow of the network that we constructed is odd.

Now, we can easily see that the construction of the network it is done in NC, that is because to construct the network all that we have to do is to take the circuit create one vertex for every single gate; create the new vertices s and t and then connect up and define the capacities this can all be done in NC.

(Refer Slide Time: 30:47)



Therefore, putting it all together now we have shown that the max flow problem is a P complete for NC reductions that is it from this lecture.

Thank you.