


Secure Computation: Part II
Prof. Ashish Choudhury
Department of Computer Science and Engineering
Indian Institute of Science, Bengaluru

Lecture - 05
EIG Protocol for Perfectly - Secure Byzantine Agreement: Analysis Part I

Hello, everyone. Welcome to this lecture. So, this will be Part I of our analysis of the EIG protocol for perfectly – secure Byzantine Agreement.

(Refer Slide Time: 00:30)

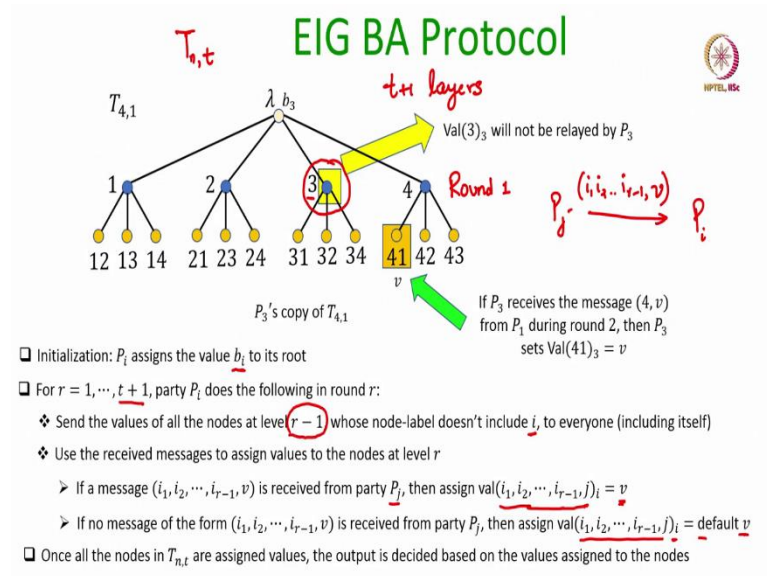
Lecture Outline



- ❑ Analysis of the EIG BA protocol
 - ❖ Proof of the validity property

So, the lecture outline is as follows we will focus on the proof of the validity property, the rest of the properties, we will discuss in the next lecture.

(Refer Slide Time: 00:42)



So, just let us quickly go through the EIG protocol. So, we have the EIG data structure $T_{n,t}$ which has $t + 1$ layers, and each node will have a label associated with it or a tag associated with it. The idea behind the labelling is that as we go from the root to the leaf node then the labels that we encounter it consist of the index of distinct parties. That means, it should not happen that the index i_t gets repeated.

Each party maintains a local copy of its EIG tree and in the protocol, the parties exchange messages for $t + 1$ communication rounds and assign values to their respective copies of the EIG tree. And, based on the values which they have assigned they recompute the values assign new values and based on the new values which are computed the overall decision is done taken.

So, the initialization is as follows. Each party assigns the bit b_i , the input for the byzantine agreement protocol, to the root. And, then for the next $t + 1$ rounds each party P_i does the following in round number r . It sends the values of all the nodes in its local copy of the EIG tree at level $r - 1$ except for those nodes whose label has the index i somewhere.

So, for instance during round 1 if we consider party number 1, then it will never send the value which it has assigned to this node to anyone because the label of this node has the index 3 appearing in it. Now, based on the messages which the parties receive during round r they assign values to the nodes at level r in their respective copies of the EIG tree. This is done as follows.

So, if P_i has received a message of the form that a node with label i_1, i_2, \dots, i_{r-1} has the value v in P_j 's local copy of the EIG tree then what P_i does is that it goes to the child node in its local copy of the EIG tree which has the label i_1, i_2, \dots, i_{r-1} followed by j and it's in its local copy of the EIG tree that the value v is assigned.

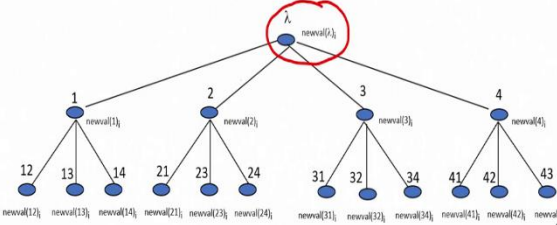
However, it could be possible that party P_j is corrupt and it does not send any message and remember that we do not have separate instructions for messages which do not arrive within a specific round.

The nomenclature or the strategy that we follow is that if some expected message does not arrive in a particular round and substitute it with some default message and proceed. So, if the message comes fine well and good; if the message does not come then the node with this label $i_1, i_2, \dots, i_{r-1}, j$ in P_i 's copy of the local EIG tree is assigned a value v .

And, once all the nodes in EIG tree have been assigned values, the output is decided based on the values assigned to the respective nodes.

(Refer Slide Time: 04:43)

EIG BA Protocol: Output Decision



P_i 's copy of $T_{n,t}$

- P_i re-assigns new values to its nodes, based on old values in a **bottom-up** fashion:
 - ❖ $\text{newval}(x)_i = \text{val}(x)_i$, for every **leaf node** with label x
 - ❖ $\text{newval}(x)_i = \text{Strict majority}$ of the newval values of the children of x , if majority exists
 - $\text{newval}(x)_i = \text{default value } v_i$ if no majority exists
- P_i outputs $\text{newval}(\lambda)_i$ as its decision value

Namely parties now go to their respective copies of the EIG tree and start from the leaf nodes and go in a bottom-up fashion. So, they reassign values to the nodes and reassignment happens as follows. For the leaf nodes, the reassigned values remain the same as they were earlier, i.e., whatever values the parties have assigned are assigned during the $t + 1$ th round.

But as we go up the new values are computed based on the majority strict majority of the new values which have been computed for the child nodes if at all majority exist. Of course, it could be possible that majority does not exist in which case some default bit v is taken and assigned as the new value.

And, once the new value has been assigned to the root node in the respective copies of the EIG tree the i th party outputs the value which has been assigned as the new value to the root node. That is the decision that is the output for the byzantine agreement protocol for the party P_i , ok.

(Refer Slide Time: 06:02)

EIG BA Protocol: Helping Lemmas

liveness, validity

Lemma 1: If P_i, P_j, P_k are honest, then $\text{val}(x)_i = \text{val}(x)_j$, for every node label x ending with k .

- ❖ Let $x = (i_1 i_2 \dots i_{r-1} k)$
- ❖ Let $\text{val}((i_1 i_2 \dots i_{r-1})_k) = v$

And, in the last lecture we have also seen a demonstration. So, I hope that the working of the protocol is clear. Now, let us prove that this protocol satisfies the liveness requirement and validity requirement. The proof for the consistency we will see in the next lecture. So, the liveness of the termination requirement is that every party should terminate the protocol and should have an output by some specific time.

And that is very trivial to verify because there are $t + 1$ communication rounds and each round will take Δ clock cycle if once the party sends a message in a round, it is received at the end of the Δ clock cycles. So, therefore, at the end of time $(t + 1) \cdot \Delta$ every party will have the new value ready for the root node and that is the value which they are going to output.

So, termination is trivial to verify, I am not separately going through the proof of the termination property. It will never happen that the parties keep on running the protocol forever even if the corrupt parties decide not to cooperate, not sending any message and so on.

So, let us go to the validity property the proof of the validity property and to recall the validity property demands that if in the EIG protocol all the honest parties start with the same input, say all the parties start with the input 0 then the output decision of all the honest parties should be 0 and vice versa.

All the honest parties start with the input 1, then all the honest parties should output 1 as the decision at the end of the protocol. Of course, we do not care anything regarding the output of the corrupt parties because the corrupt parties can output whatever they feel like. So, to prove the validity property we will require help of several lemmas.

So, let us start with lemma 1 which states the following that if you have a triplet of parties P_i, P_j, P_k and say all of them are honest, then if we take any node whose label is x and suppose the label x has the last indexes k corresponding to party P_k and say P_k is honest, then the claim is that the value which is assigned to the node with label x in the i th party's EIG tree and the j th party's EIG tree will be the same.

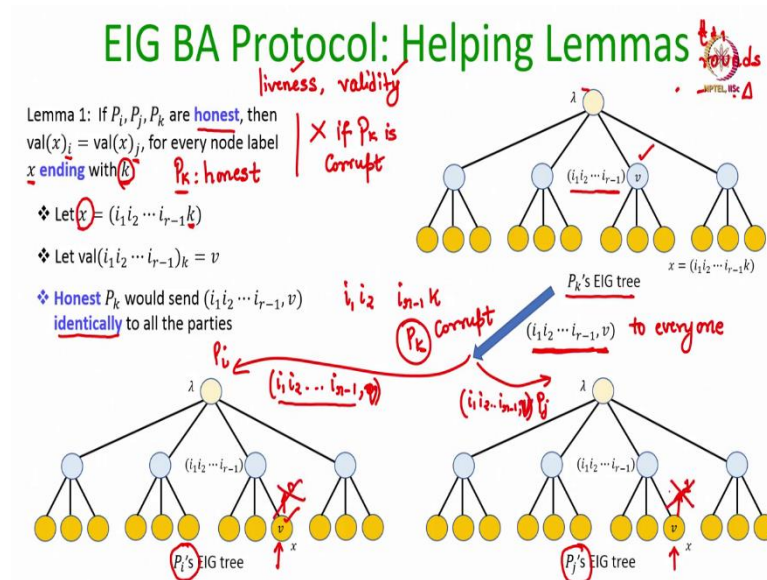
So, it is like saying the following that you take any node x whose label ends with the identity or the index of an honest party and there will be a node with this label x in the copy of the EIG tree of all the parties. So, what I am saying is what this lemma says is that if we take the node with label x across the EIG tree of all the honest parties, then the value which will be assigned by the respective honest parties during the protocol to this node will be the same.

It will not be the case that P_i assigns a value say 0 to the node with label x in its local copy of the EIG tree and P_j assigns the value 1 in its local copy of the EIG tree for the same node if. It will never happen provided we are talking of a node whose label ends with the index of an honest party and the proof is very trivial, ok.

So, imagine what will be happening in the P_k 's copy of the EIG tree. So, I am assuming here that x has a label of the form i_1, i_2, \dots, i_{r-1} followed by k . Now, if we consider the P_k 's copy of the EIG tree during round $r - 1$ P_k would have assigned some value say v to

the node with this label, right and during the same round r it would have sent the following message to everyone.

(Refer Slide Time: 11:04)



It would have informed everyone that the node with the label i_1, i_2, \dots, i_{r-1} has the value v in P_k 's copy of the EIG tree and it would have sent the same message identically to all the parties because we are assuming that P_k is an honest party. This lemma is with respect to a label which ends with the index of an honest party. So, P_k would not do the following that to one set of parties it sends v being 0 and to another set of parties it sends v being 1, it would not be the case.

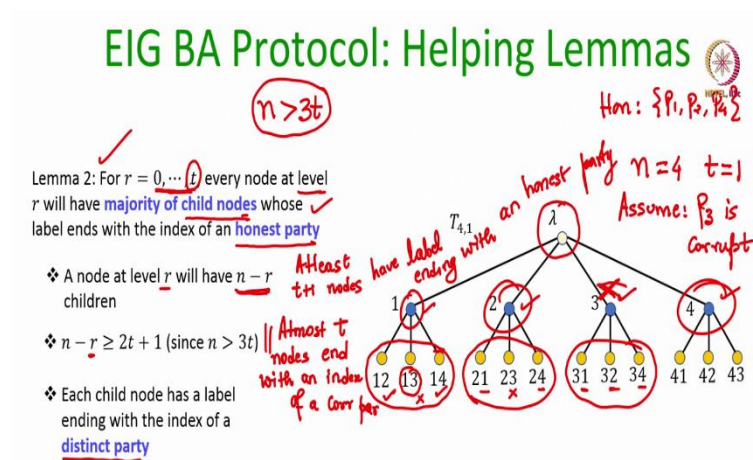
Now, consider what happens in P_i 's copy of the EIG tree and P_j 's copy of the EIG tree when P_i and P_j receives this message from P_k during round r . They will go to their respective copies of their EIG tree and focus on the node labelled x and x is i_1, i_2, \dots, i_{r-1} followed by k .

Now, since P_i has received this message from P_k ok what P_i will do P_i will assign the value v to the node labelled x and since P_j also would have received the same message from P_k it will assign the value v to the node with label x in its copy of the EIG tree and this holds for other part honest parties as well. So, if you take any honest party, in their respective copies of the EIG tree, the value v will be assigned to the node whose label ends with the index k , where k is an honest party.

This lemma does not hold if P_k is corrupt because it might send the value $i_1, i_2 \dots, i_{r-1}$ and say 0 to P_i ; that means, it is telling that 0 is the value in my copy of the EIG tree with the $i_1, i_2 \dots, i_{r-1}$ whereas, to P_j it might send that the node with label $i_1, i_2 \dots, i_{r-1}$ has the value 1 if P_k is corrupt.

Due to which P_i may end up assigning the value 0 to the node with label x and P_j may end up assigning the value to the node with label x , but we are talking about the case when P_k is honest. If P_k is honest it will send the same v due to which this would not happen.

(Refer Slide Time: 14:30)



Now, helping lemma number 2, which is also very straight forward. The lemma states that if I focus on the EIG tree then at every level r where r ranges from 0 to t , majority of the child nodes will have a label whose index corresponds to an honest party. So, again before going into the proof you can verify that the lemma is trivially true for this example. So, here $n = 4$ and $t = 1$ and assume P_3 is corrupt.

So, if I focus on this node its children are this node this node this node and this node. So, if P_3 is corrupt honest parties are P_1, P_2 and P_4 . So, you can see those three circled nodes at level one end with the index of an honest party and it's only one node whose label ends with a corrupt party.

Similarly, if I go one layer down right then among these three children of the node with label 1, it is only one node with label ending with the index of a corrupt party, but majority

of them are ending with the index of an honest party everywhere you can see here. So, 1 is honest here, 2 is honest here and 4 is honest. In fact, all the three children end with the index of an honest party.

If I focus on these three children then 1 corresponds to the index of an honest party, 4 corresponds to the index of an honest party and only 3 corresponds to the index of a corrupt party. So, majority of the children have a label ending with the index of an honest party. So, why this is true? In general, at any node at level r will have how many children? It will have $n - r$ children because of the nomenclature because of the way we have labelled the nodes assigned the labels to the various nodes.

And remember the EIG protocol assumes that $n > 3t$ holds. So, if $n > 3t$ holds for every r in the range 0 to t the condition $n - r \geq 2t$ holds. That means out among the $n - r$ children at most t can be corrupt at most at most t nodes end with an index of a corrupt party.

And at least $t + 1$ nodes have labels ending with an honest party. Remember that each child node has a label ending with the index of a distinct party. So, among these $n - r$ children it would not be the case that if we focus on the last index repetition occurs, no. So, that is why this lemma is trivially true.

If I go one layer further down; that means, if I make $r = t + 1$, then this lemma need not hold because then there will be only $2t$ children and among those $2t$ children there might be t children whose labels end with the index of a corrupt party and t children whose labels end with the index of an honest party.

In that case strict majority may not be there, but in the protocol, we do not go beyond level t because during the $t + 1$ th round only the values of the nodes at level t are communicated.

(Refer Slide Time: 19:33)

EIG BA Protocol: Helping Lemmas

Lemma 3: If x ends with the index of an honest party, for every honest P_i :
 $\text{newval}(x)_i = \text{val}(x)_i = v$, for some value v

Proof by backward induction on $|x|$ Size of x

❖ Base case: if x is the label of a leaf node

- From Lemma 1: $\text{val}(x)_i = \text{val}(x)_j$ for every honest P_i and P_j
- For leaf nodes: $\text{newval}(x)_i = \text{val}(x)_i$

Now, comes the very crucial lemma which states the following that if I focus on any node with label x and it ends with the index of an honest party. Then in every local copy of the EIG tree corresponding to the honest parties; that means, if I consider the party P_i and if party P_i is honest, then the old value of the node with label x will be the same as the new value for the same node.

It would not be the case that the old value was different and the new value which is computed based on the majority rule is different and this holds with respect to every honest party's local copy of the EIG tree provided the node x ends with the index of an honest party. Now, this proof will be done through a backward induction; backward induction on the size of x why size of x because this lemma is for any x .

So, x could be of size 1; that means, it could be as simple as that x is the label of a layer 1 node or x could be a string of size 2; that means, this lemma even holds for the nodes x which are occurring at layer number 2 or x could be a string of size $t + 1$; that means, lemma even holds for a node x occurring at layer number $t + 1$.

So, we must prove that this lemma holds irrespective of the position of the layer number of x . So, that is why to prove it for any x whose label ends with the index of a corrupt party we are going to use a proof by induction and the induction will be a backward induction; that means, we will first start with the leaf nodes, and we will keep on going from leaf to root and prove it is true for all the layers.

So, our base case will be when x is the label of a leaf node. So, assume that x is a leaf node and it ends with the index of an honest party. So, we want to prove that across all the copies of EIG trees corresponding to the honest parties, the new value which is assigned to x will be the same as the old value.

So, for that we will trigger the lemma number 1 first and lemma number 1 states that the old value which is assigned to the node x across all the copies of the EIG tree by the respective party respective honest parties is same. Say a value v is the old value which is assigned to all the in all the EIG trees.

Now, recall that the way new values are computed for the leaf nodes the leaf value of x the for the leaf nodes x the new value is same as the old value. That is the way new values are computed. We do not apply the majority rule for the nodes labelled x , where x corresponds to a leaf node because since it is a leaf node it does not have any children. So, how can we apply the majority rule?

So, the new value for x will remain the same as the old value of x if x corresponds to a leaf node and from lemma one the value which is assigned, or the old value of the node labelled x is same across all the EIG trees that also implies that the new value for the x also remains the same across all the copies of the EIG tree. So, the base case is done.

(Refer Slide Time: 23:42)

EIG BA Protocol: Helping Lemmas

Lemma 3: If x ends with the index of an honest party, for every honest P_i :
 $\text{newval}(x)_i = \text{val}(x)_i = v$, for some value v

❖ **Base case:** if x is the label of a leaf node

- From Lemma 1: $\text{val}(x)_i = \text{val}(x)_j$ for every honest P_i and P_j
- For leaf nodes: $\text{newval}(x)_i = \text{val}(x)_i$

❖ **Inductive hypothesis:** let the statement be true for $|x| = k + 1$

❖ **Inductive step:** let $x = (i_1 i_2 \dots i_k)$, where P_{i_k} is honest

- From Lemma 1: $\text{val}(x)_j$ is a common v , for every honest P_j
- Every honest P_j sends (x, v) to everyone in round k
- $\text{Val}(x_j)_i = v$, for each honest P_j and honest P_i
- By induction hypothesis, $\text{newval}(x_j)_i = v$ for each honest P_j and honest P_i
- **Lemma 2:** label of majority of children of x ends with the index of honest parties P_i

By majority, $\text{newval}(x)_i = v$, for every honest P_i

Proof by backward induction on $|x|$

During Round k

P_i 's EIG tree

(x, v)

P_j 's EIG tree

(x, v)

Now, assume the inductive hypothesis assume that the statement is true; that means, the lemma is true for a node any node x occurring at layer $k + 1$. And now we go one layer further up; that means, now we are taking the case when the label of x is a string of size k ; that means, x is occurring at layer k .

And say the label of x is i_1, i_2, \dots, i_k which are all distinct and P_k is an honest party; that means, the index k corresponds to the party P_k and P_k is honest. Why honest? Because we want to prove the lemma for all the x whose label ends with the index of an honest party not corrupt party. So, what do we know? What is going to happen in the protocol and what exactly are the things which we already know?

So, again let me trigger lemma number 1 for this node x occurring at layer k and what we know is that across all the EIG trees corresponding to the honest parties the old value which is assigned to x is a common value. That means, in P_i 's copy of the EIG tree the old value of x will be v and in P_j 's copy of the EIG tree the old value of x also will be v and if there are other honest parties in their respective copies of the EIG tree the old value of x will be v that comes from lemma number 1.

Now, we want to prove what is our goal? Our goal is to prove that even the new value which is computed for the node labelled x across all the honest parties EIG tree based on the majority rule also remains the same namely v . So, for that the first thing to observe is that during round k what every honest P_j would have done? Every honest P_j would have communicated the message that in my copy of the EIG tree the value of the node labelled x is v and it would have sent this message identically to everyone.

So, P_j would have sent the message (x, v) to everyone. P_i would have sent the message (x, v) to everyone and so on.

And all other honest parties would have sent the message (x, v) to everyone of course, the corrupt parties might send that in their respective copies of the EIG tree x has the value v' where v' is the complement of v . Corrupt parties can behave arbitrarily.

So, before going into the before proceeding further what we are going to see here is that because every honest party P_j sends this message (x, v) identically to everyone what is going to happen is that if we focus on the children of the node labelled x there will be

children which will have labels of the form x followed by the index of an honest party. Those indexes could be j, i and other honest parties.

The old value which will be assigned to the nodes with or to the children with label xj, xi and x followed by the index of an honest party will be v because as I said all honest parties will be sending the message x followed by v to everyone. And, now let us apply the inductive hypothesis.

Let us now apply the inductive hypothesis and as per the inductive hypothesis since the children are now appearing at level $k + 1$ and we have assumed that the statement is true the lemma statement is true even for nodes which have labels of size $k + 1$. So, if we focus on all the children of x whose label ends with the index of an honest party the new value and the old value for such nodes will be v only.

That comes from the inductive hypothesis. And now, we can apply we can trigger the lemma 2 which states that if we focus on the node with label x , majority of it is children will have the last index corresponding to an honest party. And such nodes whose last index corresponds to an honest party, their new values will be the common v that comes from the inductive hypothesis.

Due to this when the parties respectively apply the majority rule in their respective copies of the EIG tree to compute the new value for the node x , it will turn out to be value v only. So, the old value also remains the v and the new value also remains v . It would not change for all the nodes x irrespective of at which layer they occur provided x ends with the index of an honest party.

(Refer Slide Time: 29:50)

EIG BA Protocol: Validity Proof

Lemma 2: For $r = 0, \dots, t$, every node at level r will have majority of child nodes whose label ends with the index of an honest party

Theorem (Validity): In the EIG protocol, if all honest parties have same input bit v then all honest parties output v

- ❖ Say P_1, \dots, P_{n-t} are the honest parties with input v
 - $n - t > 2t$, as $n > 3t$
- ❖ During Round 1, $\text{val}(1)_i = \dots = \text{val}(n-t)_i = v$ holds, for every honest P_i
- ❖ From Lemma 3: $\text{newval}(1)_i = \dots = \text{newval}(n-t)_i = v$ holds, for every honest P_i
- ❖ From Lemma 2: majority of child nodes of root have labels ending with the index of an honest party in EIG tree
- ❖ By majority rule, every honest P_i will set $\text{newval}(n-t)_i = v$

Lemma 3: If x ends with the index of an honest party, for every honest P_i : $\text{newval}(x)_i = \text{val}(x)_i = v$, for some value v

At least $n-t$ honest parties

$P_1 - (\lambda, v) \leftarrow$
 $P_2 - (\lambda, v) \leftarrow$
 $P_{n-t} - (\lambda, v) \leftarrow$

P_1 's EIG tree

So, now we will use the lemma number 2 and lemma number 3 which we have proved to conclude our validity proof. So, the theorem statement is the validity statement is that in the EIG protocol, if all the honest parties start the protocol with the same input bit say v where v could be either 0 or it could be 1, then after exchanging messages assigning old values computing new values each party will have the decision output v only; that means, the old state v remains the same it does not get changed.

So, how do we prove this? So, without loss of generality assume that the parties P_1, P_2, \dots, P_{n-t} are the honest parties. Remember there are at least $n - t$ honest parties and at most t corrupt parties and their exact identities will not be known. But what we will be knowing is that there are at least $n - t$ honest parties.

To do the analysis of the validity statement I am making a simplified assumption. Imagine that the first $n - t$ parties are the honest parties, but this is without loss of generality because whatever we are stating here as the proof carries over for the general case where the $n - t$ parties are scattered means their indices need not be consecutive.

Also note that $n - t$ is strictly greater than $2t$ because $n > 3t$. So, during round one what would have happened in the EIG protocol? During round 1 P_1 would have reported to everyone that its input is v P_2 would have sent so, P_1 would have sent the message (λ, v) to everyone P_2 would have sent the message (λ, v) to everyone because we are assuming

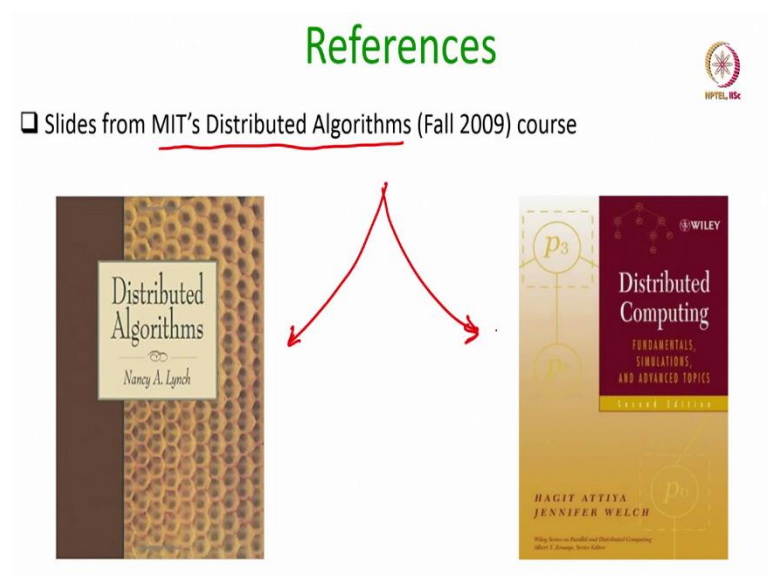
that the inputs of all the honest parties are same and P_{n-t} would have send the message (λ, v) to everyone else.

Of course, corrupt parties can send different versions of their input to the respective parties. Now, due to this at the end of the round one what would have happened? Every honest party P_i would have assigned the value v, v, v, v, v to the nodes with label $1, 2, \dots, n - t$. I do not care what values it assigned to the nodes whose label ends with the corrupt party.

Now, what does lemma 3 states? Lemma 3 states that the old value which has been assigned to the nodes with label $1, 2, \dots, n - t$ will be same as the new value. That means, the re computed values will also remain the same namely v, v, v, v, v and from lemma 2 majority of the child nodes of this root of the EIG tree will end with the index of an honest party.

Namely, there are $n - t$ children whose labels end with the index of an honest party and $n - t \geq 2t$; that means, at least $2t + 1$; that means, the majority of them are ending with the index of an honest party. So, by the majority rule every honest party will assign new value as v to their respective root nodes and that will be the overall output of the EIG protocol. So, that shows that completes the validity proof and that also concludes today's lecture.

(Refer Slide Time: 34:05)



So, these are the references which I have been used. So, I have used heavily the material from MITs Distributed Algorithm course and of course, the full details of the EIG protocol and proof you can find from any of these two textbooks.

Thank you.