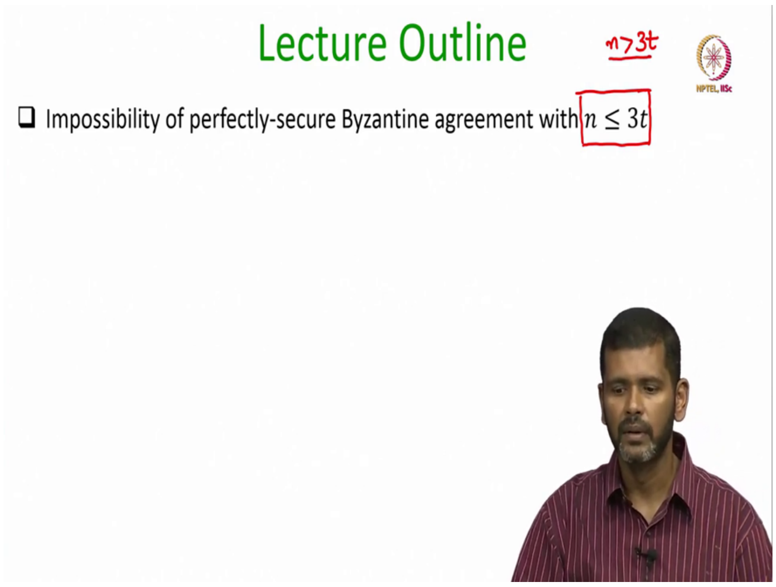
Secure Computation: Part II

Prof. Ashish Choudhury
Department of Computer Science and Engineering
Indian Institute of Science, Bengaluru

Lecture - 16
Lower Bound for Number of Parties for Byzantine Agreement: Part II

(Refer Slide Time: 00:25)



Lecture Outline

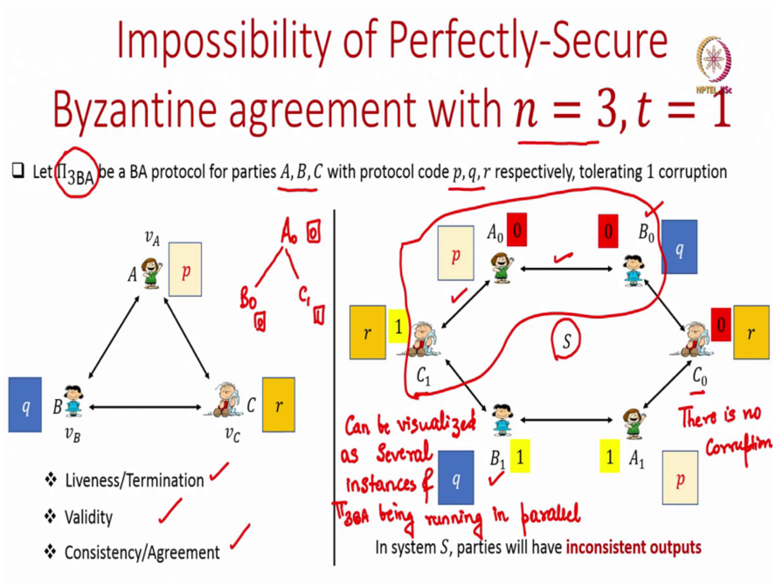
$n > 3t$

- Impossibility of perfectly-secure Byzantine agreement with $\boxed{n \leq 3t}$

Hello everyone welcome to this lecture. So, we will continue our discussion regarding the impossibility proof or the necessary condition for the existence of the perfectly secure Byzantine agreement. So, in this lecture we will complete the proof and we will show that it is not possible to design any perfectly secure Byzantine agreement protocol if this condition n ≤ 3t holds.

That means the condition n > 3t is necessary and recall that all the protocols that we had seen till now, there we assumed that the condition n > 3t holds.

(Refer Slide Time: 01:08)



Impossibility of Perfectly-Secure Byzantine agreement with $n = 3, t = 1$

- Let Π_{3BA} be a BA protocol for parties A, B, C with protocol code p, q, r respectively, tolerating 1 corruption

- Liveness/Termination ✓
- Validity ✓
- Consistency/Agreement ✓

In system S, parties will have inconsistent outputs

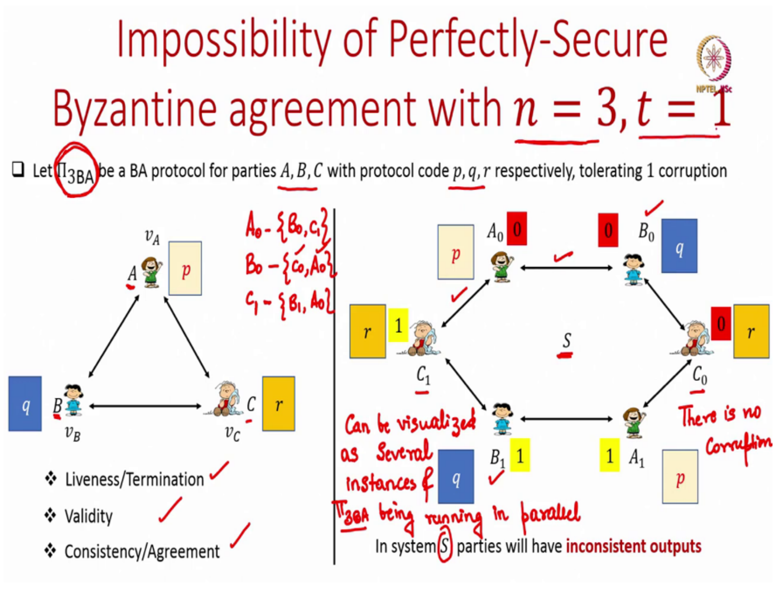
So, let us quickly go through the proof for the impossibility of a perfectly secure Byzantine agreement for 3 parties, tolerating 1 corruption that we had discussed in the last lecture.

So, the proof was through a contradiction, we assumed that suppose there is a protocol say Π_{3BA}, which allows 3 parties A, B, C with protocol codes p, q, r respectively, to solve the Byzantine agreement problem, even if one of the 3 parties A, B, C could get corrupt. Such that the protocol pi 3BA has liveness, validity, and consistency properties. Assuming such a protocol Π_{3BA} exists, we saw that we can compose the protocol Π_{3BA} and we can create a system, a hexagonal system S, where we now have 2 copies of A – A0 and A1 with inputs 0 and 1 respectively, and running the protocol code p, two copies of B – B0 and B1 with inputs 0 and 1 and running the protocol code q respectively, and 2 copies of C namely, C0 and C1 with inputs 0 and 1 respectively, running the code r; such that each participant has knowledge about only its immediate neighbour, but not about the full network.

And in this system, there is no corruption, and every party is just participating with its input as assigned in this system S, and running the protocol code either p or q or r depending upon whether they are the copies of entity A, B or C respectively. And then we saw that the system S can be visualized as several instances of Π_{3BA} being running in parallel. Say, for instance, we can take the triplet of parties A0, B0 and C1.

So, from the viewpoint of A0 its 2 neighbors during the instance of Π_{3BA} are B0 and C1 ok.

(Refer Slide Time: 05:16)



Impossibility of Perfectly-Secure Byzantine agreement with $\underline{n = 3}, \underline{t = 1}$

- Let Π_{3BA} be a BA protocol for parties A, B, C with protocol code p, q, r respectively, tolerating 1 corruption

- Liveness/Termination ✓
- Validity ✓
- Consistency/Agreement ✓

In system S parties will have **inconsistent outputs**

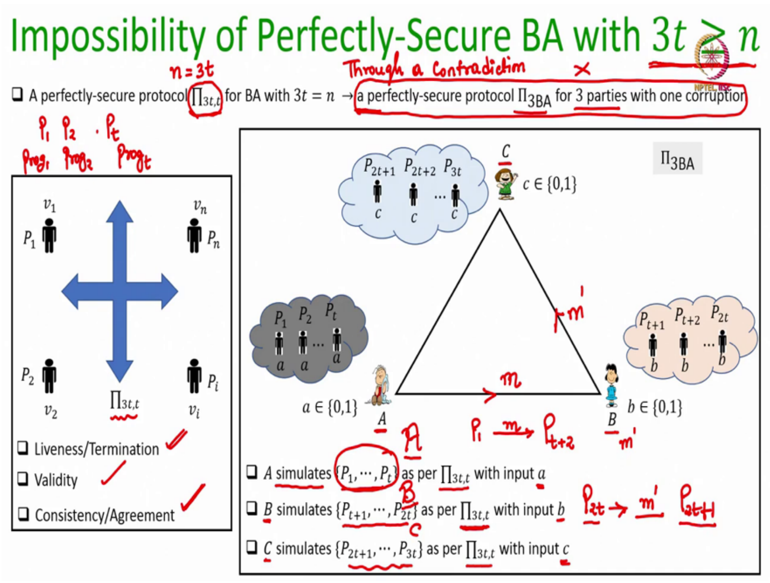
So, A0's neighbor will be B0 and C1 and that matches with what we expect from the protocol Π_{3BA}, because in Π_{3BA} we expect party A to talk with one copy of B, and one copy of C, that is what is happening in the system S, from the view point of B0 ok. We can imagine that its neighbors are C0 and A0. So, this is consistent with what we expect during one execution of Π_{3BA}.

Namely, B is expected to talk with C. So, in this case, B0 talking with C0 and B is expected to talk with A, that corresponds to B0 talking with A0, and we can imagine that we have C1 talking with B1 and A0, that is the way we can imagine. So, like that we had seen in the last lecture that in system S, we can imagine several instances of Π_{3BA} being running in parallel.

And then, we saw various properties, various relationships among the outcomes from those invocations of the instances of Π_{3BA} and the output of the respective parties in the system S and then, we actually end up showing that in the system S parties may have inconsistent outputs; that means, the system S output is not well defined even though that should be the case. Because we are actually running the protocol Π_{3BA} several times that is the way we can imagine the system is.

So, that shows that whatever we assumed regarding the existence of protocol Π_{3BA} is incorrect. No such protocol Π_{3BA} exists, but this impossibility proof is restricted because it is only for the case when we have 3 parties and one of them is corrupt.

(Refer Slide Time: 07:34)



Impossibility of Perfectly-Secure BA with $3t \geq n$

- A perfectly-secure protocol $\Pi_{3t,t}$ for BA with $3t = n$ → a perfectly-secure protocol Π_{3BA} for 3 parties with one corruption
- Liveness/Termination ✓
- Validity ✓
- Consistency/Agreement ✓
- A simulates $\{P_1, \cdots, P_t\}$ as per $\Pi_{3t,t}$ with input a
- B simulates $\{P_{t+1}, \cdots, P_{2t}\}$ as per $\Pi_{3t,t}$ with input b
- C simulates $\{P_{2t+1}, \cdots, P_{3t}\}$ as per $\Pi_{3t,t}$ with input c

Now, we want to generalize this proof and show that there exists no perfectly secure Byzantine agreement protocol, if n ≤ 3t and again the proof will be through a contradiction.

What we will show here is that if at all there exist any protocol, any perfectly secure Byzantine agreement protocol with this condition 3t ≥ n. Let us call such a protocol as $\Pi_{3t,t}$. So, we are taking the case, when n = 3t. So, we assume that suppose there exists a perfectly secure Byzantine agreement protocol with the condition n being equal to 3t.

Then we will design, we will show the existence of a perfectly secure protocol Π_{3BA} for 3 parties with 1 corruption. But we know that the protocol Π_{3BA} does not exist, this does not exist; that means whatever we assumed regarding the existence of $\Pi_{3t,t}$ also is incorrect ok, no such protocol exists. So, imagine we have a protocol $\Pi_{3t,t}$ which solves the Byzantine agreement problem; it gives you the termination guarantee, validity guarantee, and consistency guarantee ok.

We do not care how many rounds of protocol it is, but what we know is that it has the liveness property. So, say it is an r-round protocol. It has the validity property; which means if all the honest parties in this protocol have the same input and that is the output and it has the consistency property; that means, all honest parties have the same output at the end of the r rounds. Now, using this protocol we design our protocol pi sub 3BA, where our 3 parties are A, B, and C respectively.

Suppose the input of party A is a, which is a bit. Similarly, the input of party B is little b, and the input of party C is little c. Now, what party A does is the following it simulates the role of the party's P1, P2, Pt as per the existing protocol $\Pi_{3t,t}$ ok, assuming that all the parties in this collection have the input a.

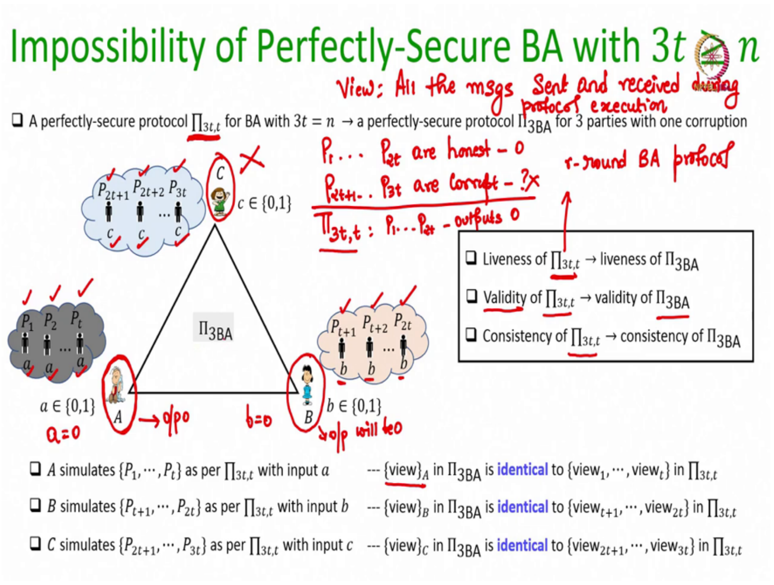
So, basically what I am saying here is that according to the protocol $\Pi_{3t,t}$, there would have been a protocol code for P1. There would be a set of steps, a set of actions for the party P2 and like that there will be a sequence of steps for the party P sub t, depending upon whatever is their input. So, party A in our protocol Π_{3BA} runs the code, which has been assigned for P1. Let us call that code program 1. Whatever code has been assigned in the protocol for P2 in pi sub 3t t, let us call it program 2. And like that whatever set of steps have been prescribed in the protocol $\Pi_{3t,t}$, for the party P sub t, let us call it program sub t. So, party A in the protocol Π_{3BA} will run the program 1, program 2, programs sub t assuming that the inputs of the parties P sub 1, P sub 2, P sub t is little a.

In the same way, in the protocol Π_{3BA} party B will simulate or play the role of the party Pt+1, Pt+2, P2t assuming that their input is b and they are running their respective protocol code as prescribed in the existing protocol $\Pi_{3t,t}$. And in the same way, C simulates the role of the last t parties, P2t+1, P2t+2 like that P3t, assuming that their inputs is c and they are following the protocol code $\Pi_{3t,t}$.

Now, when I say simulate; that means, according to the protocol $\Pi_{3t,t}$, it could be the case that a party in this collection, the collection P, …, Pt want to send a message to any other party in the same subset. So, let us call these 3 subsets as A, B and C. So, during this simulation, whenever any party in the subset A is supposed to send any message within the subset A itself, then that is simulated by the party A sending the message to itself ok.

Whereas, if at all there is a requirement that any party in the subset A has to communicate a message to any other party in the subset , B as per the protocol $\Pi_{3t,t}$, that is emulated that is simulated by A sending the message m to the party B. In the same way if there is a party in the subset B, who has to send a message m prime to a party in C, that is, simulated by B sending the message m prime to the party C.

(Refer Slide Time: 15:05)



Impossibility of Perfectly-Secure BA with $3t \geq n$

View: All the msgs Sent and received during protocol execution

- A perfectly-secure protocol $\prod_{3t,t}$ for BA with $3t = n$ → a perfectly-secure protocol Π_{3BA} for 3 parties with one corruption

$P_1 \dots P_{2t}$ are honest – 0
$P_{2t+1} \dots P_{3t}$ are Corrupt – ?✗
$\prod_{3t,t}$: $P_1 \dots P_{2t}$ – outputs 0

r-round BA protocol

- Liveness of $\prod_{3t,t}$ → liveness of Π_{3BA}
- Validity of $\prod_{3t,t}$ → validity of Π_{3BA}
- Consistency of $\prod_{3t,t}$ → consistency of Π_{3BA}

- A simulates $\{P_1, \cdots, P_t\}$ as per $\prod_{3t,t}$ with input a --- $\{\text{view}\}_A$ in Π_{3BA} is identical to $\{\text{view}_1, \cdots, \text{view}_t\}$ in $\prod_{3t,t}$
- B simulates $\{P_{t+1}, \cdots, P_{2t}\}$ as per $\prod_{3t,t}$ with input b --- $\{\text{view}\}_B$ in Π_{3BA} is identical to $\{\text{view}_{t+1}, \cdots, \text{view}_{2t}\}$ in $\prod_{3t,t}$
- C simulates $\{P_{2t+1}, \cdots, P_{3t}\}$ as per $\prod_{3t,t}$ with input c --- $\{\text{view}\}_C$ in Π_{3BA} is identical to $\{\text{view}_{2t+1}, \cdots, \text{view}_{3t}\}$ in $\prod_{3t,t}$

So, that is the way simulation happens here. Now, due to this simulation what we can say here is the following, if we consider the view of the party A in this protocol Π_{3BA} and what view means, the view means all the messages sent and received during protocol execution. So, if we consider the view of this party A, its view will be identical to the view that parties P1, P2, …, Pt would have in the existing protocol $\Pi_{3t,t}$, if they would have been running the protocol $\Pi_{3t,t}$ with input a. In the same way, the view of the party B in the protocol Π_{3BA} will be identical to what the parties Pt+1, Pt + 2,…,P2t would have produced with their inputs being b during the execution of the protocol $\Pi_{3t,t}$.

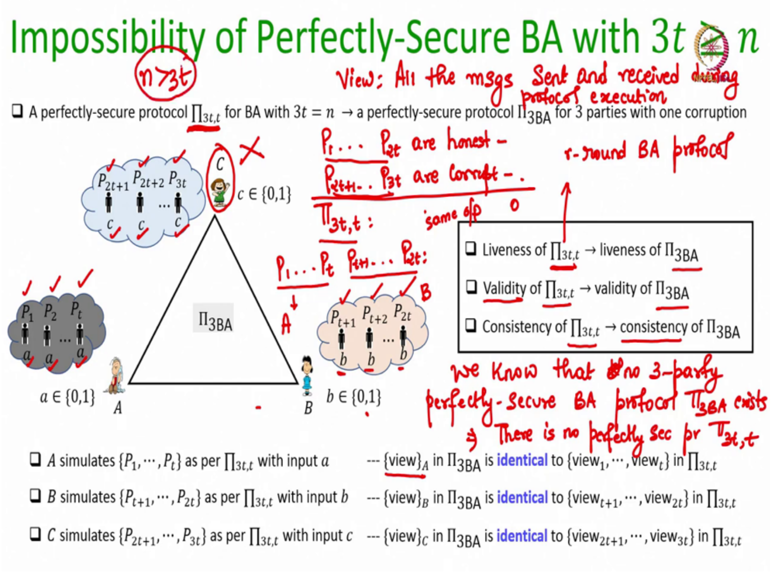
And in the same way the view of the party C will be identical to what parties P2t + 1, P2t+ 2, …, P3t would have, if they would have executed the protocol $\Pi_{3t,t}$ and their inputs being c.

So, now what we can conclude here is that since the protocol $\Pi_{3t,t}$ has the liveness property, protocol Π_{3BA} also would have liveness at the end of the simulation.

So, if $\Pi_{3t,t}$ is an r round, then party A after simulating the role of the first t parties for r rounds output, whatever those t parties would have output during the execution of protocol $\Pi_{3t,t}$. And since, the protocol $\Pi_{3t,t}$ would have produced output after r rounds that, shows that party A also would output something after r rounds. Because each of the parties P1, P2, …, Pt would have output something after r rounds.

So, the liveness property of $\Pi_{3t,t}$ implies the liveness property of this 3-party protocol. Similarly, the validity property of $\Pi_{3t,t}$ implies the validity property of this 3-party protocol. And the consistency property of $\Pi_{3t,t}$ implies the consistency property of this 3 party protocol.

(Refer Slide Time: 21:20)



Impossibility of Perfectly-Secure BA with $3t \geq n$

- A perfectly-secure protocol $\prod_{3t,t}$ for BA with $3t = n$ $\rightarrow$ a perfectly-secure protocol Π_{3BA} for 3 parties with one corruption

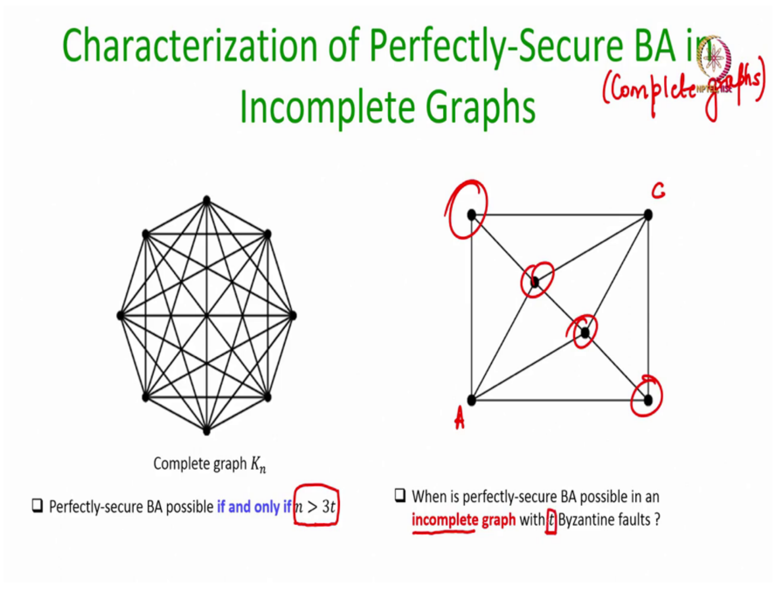
- Liveness of $\prod_{3t,t}$ $\rightarrow$ liveness of Π_{3BA}
- Validity of $\prod_{3t,t}$ $\rightarrow$ validity of Π_{3BA}
- Consistency of $\prod_{3t,t}$ $\rightarrow$ consistency of Π_{3BA}

- A simulates $\{P_1, \cdots, P_t\}$ as per $\prod_{3t,t}$ with input a --- $\{\text{view}\}_A$ in Π_{3BA} is identical to $\{\text{view}_1, \cdots, \text{view}_t\}$ in $\prod_{3t,t}$
- B simulates $\{P_{t+1}, \cdots, P_{2t}\}$ as per $\prod_{3t,t}$ with input b --- $\{\text{view}\}_B$ in Π_{3BA} is identical to $\{\text{view}_{t+1}, \cdots, \text{view}_{2t}\}$ in $\prod_{3t,t}$
- C simulates $\{P_{2t+1}, \cdots, P_{3t}\}$ as per $\prod_{3t,t}$ with input c --- $\{\text{view}\}_C$ in Π_{3BA} is identical to $\{\text{view}_{2t+1}, \cdots, \text{view}_{3t}\}$ in $\prod_{3t,t}$

$n > 3t$
View: All the msgs Sent and received during protocol execution
$P_1 \ldots P_{2t}$ are honest -
$P_{2t+1} \ldots P_{3t}$ are Corrupt - .
$\Pi_{3t,t}$:
same o/p
0
r-round BA protocol
$P_1 \ldots P_t$ $P_{t+1} \ldots P_{2t}$:
A
B
we Know that no 3-party perfectly-Secure BA protocol Π_{3BA} exists
$\Rightarrow$ There is no perfectly sec pr $\Pi_{3t,t}$
P_{2t+1} P_{2t+2} P_{3t}
c c c
C
$c \in \{0,1\}$
Π_{3BA}
P_1 P_2 P_t
a a a
$a \in \{0,1\}$
A
P_{t+1} P_{t+2} P_{2t}
b b b
B
$b \in \{0,1\}$

But, we know that no 3-party perfectly secure BA protocol exists. This implies there is no perfectly secure protocol $\Pi_{3t,t}$ ok. And that shows that the condition n > 3t is necessary for any perfectly secure BA protocol.

(Refer Slide Time: 23:00)



Characterization of Perfectly-Secure BA in Incomplete Graphs (Complete graphs)

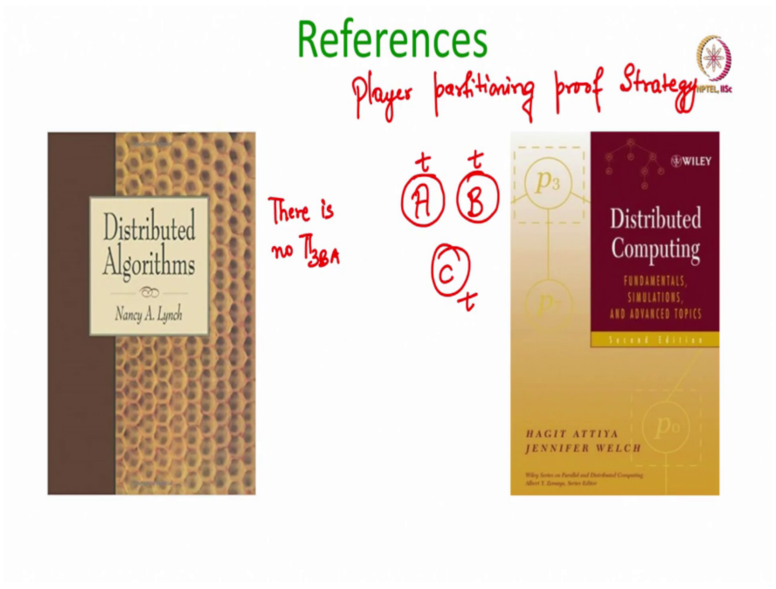
- ❑ Perfectly-secure BA possible **if and only if** $n > 3t$
- ❑ When is perfectly-secure BA possible in an **incomplete graph** with t Byzantine faults ?

Till now, we have seen the characterization of perfectly secure BA in complete graphs ok and when I say complete graphs, I mean we are we considering a network scenario, we considered an underlying network, where every party has a channel or a method to communicate to every other party.

And we have seen, we have just concluded that for designing perfectly secure BA in such networks, the condition n greater than 3t is necessary. Now, we might have an underlying network, which is modelled by a graph which is incomplete. That means, in such a graph there may not be a direct channel, or direct communication link between every pair of parties. So, for instance, if I take this party to be A and this party to be C, then there is no direct link between party A and party C.

Any communication from A to C must go through either this node or this node or one of these 2 nodes ok. So, if in such a network we have up to t Byzantine faults, t Byzantine corruption what would be the necessary condition for the existence of perfectly secure Byzantine agreement ok.

(Refer Slide Time: 24:28)



References

Player partitioning proof Strategy

There is no Π_{3BA}

$\overset{t}{\textcircled{A}}$ $\overset{t}{\textcircled{B}}$ $\textcircled{C}_t$

Distributed Algorithms

Nancy A. Lynch

Distributed Computing

FUNDAMENTALS, SIMULATIONS, AND ADVANCED TOPICS

Second Edition

HAGIT ATTIYA

JENNIFER WELCH

WILEY
p_3
p_2
p_0
NPTEL, IISc

So, we will focus on the answer for this question in our next lecture. These are the references used for today's lecture. So, the proof that I gave today is often called the Player partitioning proof strategy. This is because the way we have completed the proof is we argued that there is no protocol pi 3BA. And then, to show that there exists no protocol for 3t parties we divided the parties into 3 subsets, fancy A, fancy B and fancy C each simulating the role of t parties. So, we get a 3 party protocol. So, that is why that proof strategy is often called as the player partitioning proof strategy.

Thank you.