Blockchains Architecture, Design and Use cases

Prof. Sandip Chakraborthy
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 53
Research Aspects - VI (Algorand - II)

Welcome back to the course on Blockchain.

(Refer Slide Time: 00:21)

Gilad, Y., Hemo, R., Micali, S., Vlachos, G., & Zeldovich, N. (2017, October). *Algorand: Scaling byzantine agreements for cryptocurrencies.* In *Proceedings of the 26th Symposium on Operating Systems Principles* (pp. 51-68). ACM.

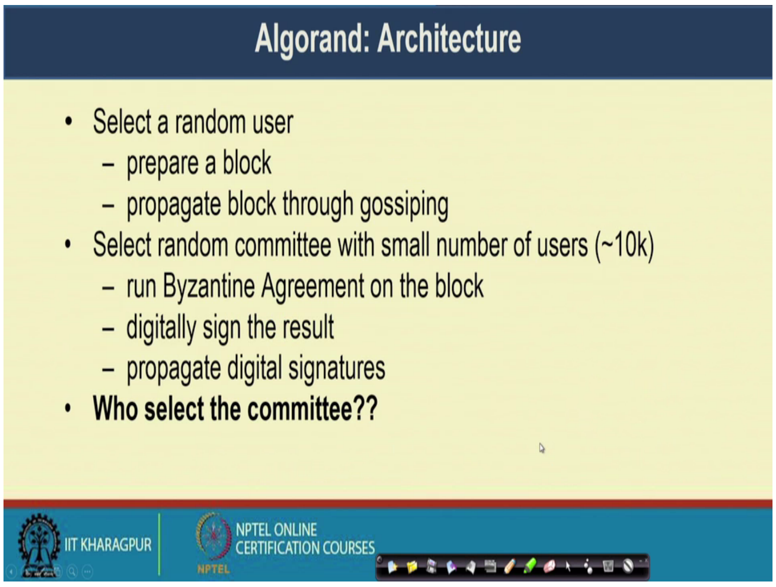
Algorand: Scaling Byzantine Agreements for Cryptocurrencies

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, in the last class, we were discussing about this Algorand protocol in details which scales of byzantine agreement for Scyoto currencies. So, they develop a scalable protocol for ensuring consensus in a byzantine agreement based architecture. So, we have looked into the basic features of Algorand.

So, these basic features of Algorand that comprises of having byzantine agreement based protocol for the consensus by utilising the gossip base communication protocol. And we have looked into the objectives of Algorand which states that well the protocol need to be fork free. So, you want to avoid the number of forks which are there so that the protocol finality can be ensured. The another requirement for Algorand is to ensure transaction scalability in the presence of a million number of nodes and at the same time supporting large number of transactions per second in terms of transaction throughput.

(Refer Slide Time: 01:30)



Algorand: Architecture

- Select a random user
  - prepare a block
  - propagate block through gossiping
- Select random committee with small number of users (~10k)
  - run Byzantine Agreement on the block
  - digitally sign the result
  - propagate digital signatures
- **Who select the committee??**

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

So, with that objective we have seen that Algorand has 3 basic steps that it executes one after another. First it is selects a random user from a group of users which can malicious or normal. So, as we have mentioned earlier that in a typical network the majority of the users cannot be adversary that way we have assumed that majority of the nodes in the network honest nodes and some nodes are adversary. So, whenever you are selecting a user among a set up user randomly, then there is a possibility that the user is either an honest user or a adversary. So, with this particular modelling mind Algorand proposes some steps to ensure consensus finality.

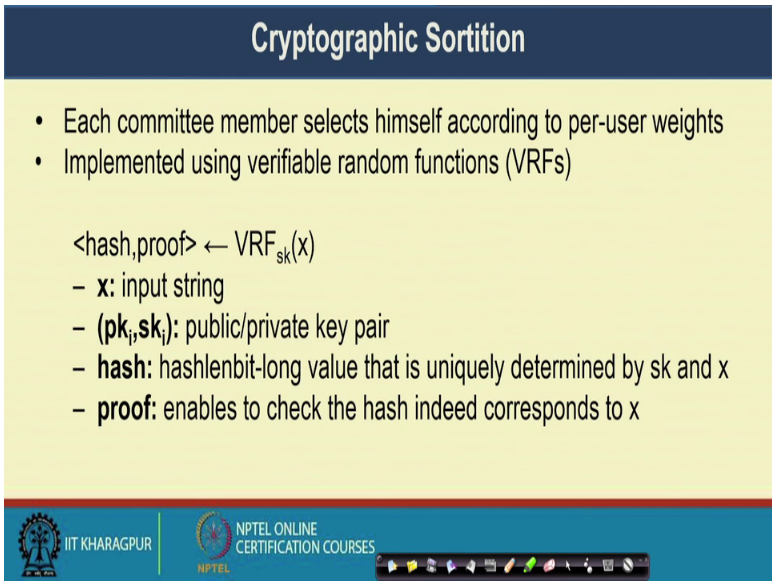
So, it selects random user whose task is to prepare a new block and then, propagate the new block in the form of gossiping and then, from the remaining set of user we select random committee members and the task of this random committee members is to valid that particular block by running the byzantine agreement protocol and then, digitally signed the result and propagate the digital signature in the network with the help of gossiping. Again, if you can see that for a particular block that have been proposed in the network, there are more than certain number of a signatures floating in the network. Then, you can be assured that the block has been reached into the consensus and you can adopt block to your existing blockchain.

And that way you have the advantage that well at every individual round of the protocol which select different set of nodes for the committee member. So, that is the beauty or that is the typical feature inside Algorand mechanism. So, we use something called a cryptographic shorting mechanism that select different number of users even the total number of users in a committee may be different at different round of the protocol.

So, whenever you are selecting a single user for block proposal and block propagation that varies from different number different routes and at the same time the committee member also differ as different routes. So, that way the problem which was there with Bitcoin based protocol that time at the entire consensus mechanism can be focused on some pre define number of block chain miners; so, a Bitcoin miners. So, we have seen that around 5 number of Bitcoin miners, the entire consensus mechanism at the entire mining procedure gets concentrated among those a handful number of minor.

And it is if it is easier to find out those mining pools, the location of these mining pools and tempered the entire protocol. So, that way from this centrality which was the major drawback for bit kind of protocols, where the entire mining procedure or get concentrated on 5 or 6 different pools, there are we are coming out with the help of this Algorand procedure, where it is a truly random protocol and had different rounds, a different set of committee will get selected. Now the question that we have asked and we have tried to find out that who select this committee?

(Refer Slide Time: 04:58)



Cryptographic Sortition

- Each committee member selects himself according to per-user weights
- Implemented using verifiable random functions (VRFs)

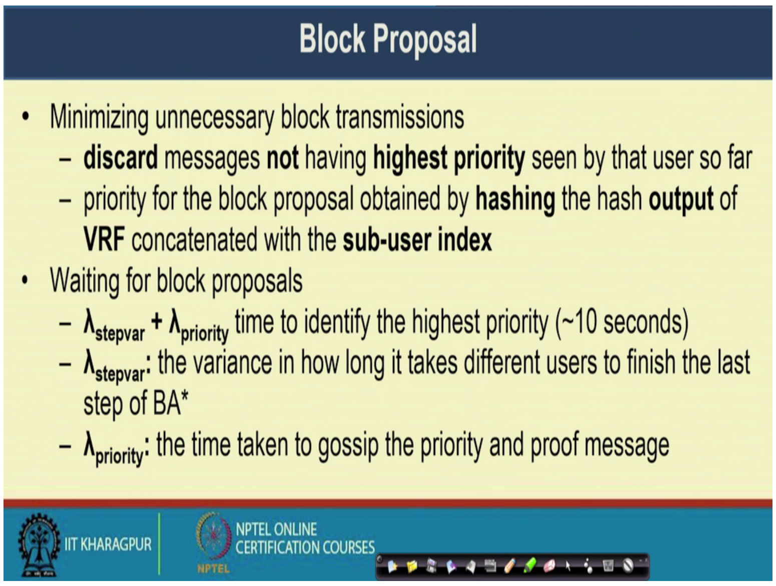
  <hash,proof> $\leftarrow$ $VRF_{sk}(x)$
  - **x:** input string
  - **(pk_i,sk_i):** public/private key pair
  - **hash:** hashlenbit-long value that is uniquely determined by sk and x
  - **proof:** enables to check the hash indeed corresponds to x

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, we have seen that we can apply certain algorithm called cryptographic sortition were each committee member selects himself according to some per-user weights. So, for that we apply something called the verifiable random function. So, this verifiable random function is a function by instituting that every member is electing himself or herself to be a part of the committee members. So, this verifiable random function, it returns a hash and the corresponding proof that the hash has been actually generated by the user who is claiming to be a committee member.

So, it is just like you are electing yourself to be a committee member and whenever you are successful in electing yourself, during that time you are presenting a proof that you have correctly executed a verifiable random function. And by presenting that proof you are proving to the environment of proving to the world that, you have actually won the election and you are successful to become a member of the committee for validating a new block or to propose a new block.

(Refer Slide Time: 06:06)



Block Proposal

- Minimizing unnecessary block transmissions
  - **discard** messages **not** having **highest priority** seen by that user so far
  - priority for the block proposal obtained by **hashing** the hash **output** of **VRF** concatenated with the **sub-user index**
- Waiting for block proposals
  - $\lambda_{stepvar} + \lambda_{priority}$ time to identify the highest priority (~10 seconds)
  - $\lambda_{stepvar}$: the variance in how long it takes different users to finish the last step of BA*
  - $\lambda_{priority}$: the time taken to gossip the priority and proof message

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Then, we have looked into the Block Proposal mechanism. So, during the Block Proposal because we are using this cryptographic sortition mechanism, where every individual nodes elects himself or herself for proposing an new block. So, that is why at different around, it may happen that more than 1 number of members, they win the race and they are able to elect them self for proposing a new block.

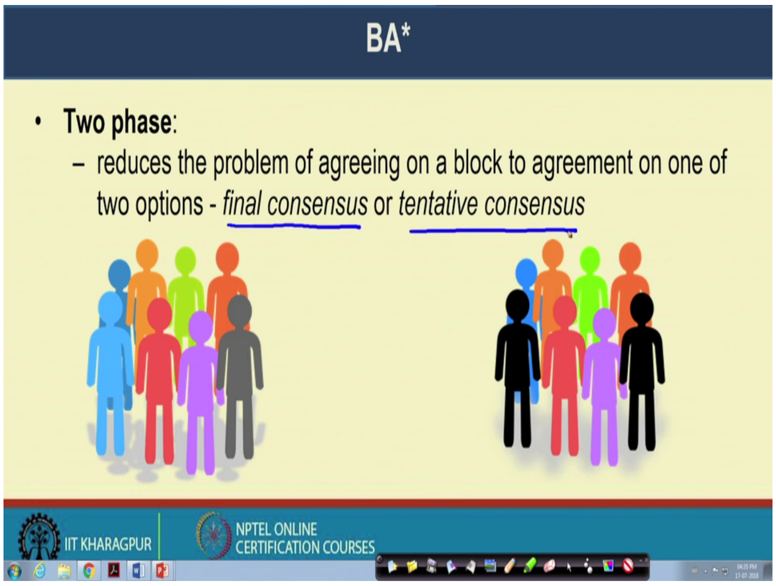
Now, to ensure a unnecessary that to ensure that we are able to avoid unnecessary block propagation in the network, we discuss the messages not having the highest priority seen by the users so far. So, this priority value for block proposal it is obtained by hashing the hash output of the VRF and concatenate it with the sap user index. So, this total number it ensures the priority of a user.

So, whenever you are ready to propose a new block during that time you wait for certain duration. Within that duration, if you are hearing a new block coming from another user were that user is having highest priority or higher priority compared to you, then you do not propagate your block any further.

So, that way, we minimise unnecessary block propagation in the network. Now, the waiting time that was calculated equal to lambda that STP variance and lambda priorities. So, lambda step variance is the variance in how long it takes different user to finish the last step of the byzantine agreement and lambda priority is the time taken to gossip the priority and the proof messages among individuals.

So, we have seen that on average it is approximately 10 second. So, if you wait for approximately 10 second, if there is any new block which is floating in the network, you are sure to hear that blocked; otherwise, you can propose your new block if you have been selected as an new member for blocked proposal by the cryptographic sorting algorithm.

(Refer Slide Time: 08:11)



BA*

- Two phase:
  - reduces the problem of agreeing on a block to agreement on one of two options - *final consensus* or *tentative consensus*

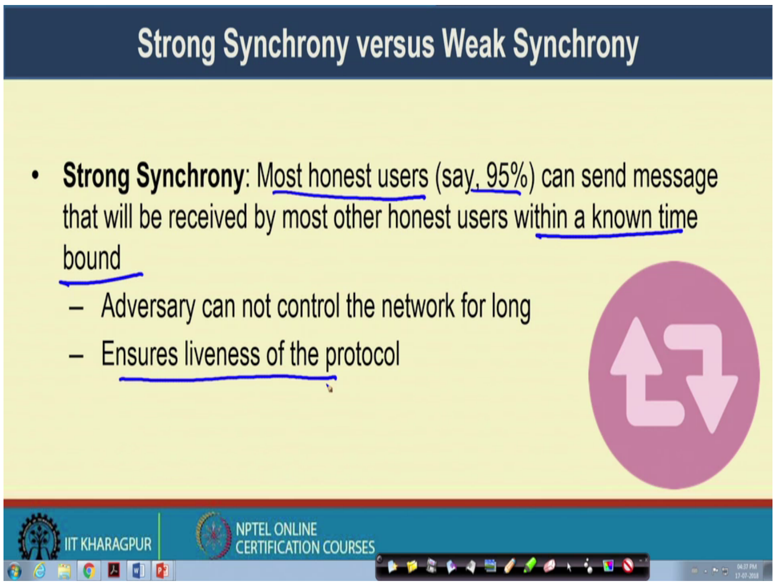
Now, comes to the byzantine agreement protocol that we will discuss in a today's lecture in details. So, this byzantine assignment protocol, it is a 2 face protocol. So, in the first phase of byzantine agreement, it reduces the problem of agreeing on a block to agreement on 1 or the 2 options. So, the objective of the byzantine agreement protocol is to come to an agreement on a block. So, the idea is there that if multiple blocks are flowing in the network and all the blocks are kind of valid block.

Then, you select the block where the user is having the highest priority, the priority is defined as I mentioned earlier with the hash value concatenated with the users of index and then, select that block and make sure that all the nodes in the committee, they are agreeing on the same block of highest priority; otherwise if a single user has a proposed a new block. Then, all the members in the committee, they will agree on the same block. Now, in byzantine agreement mechanism in Algorand, they proposed to different notions of consensus.

So, one notion of consensus is called as final consensus and the second notion of consensus is called as a tentative consensus. So, we have 2 different notions of consensus; final consensus and tentative consensus. So, if the committee member reaches to a tentative consensus; that means, that all the members they have not reached to the consensus. It is just like that the value is a local optimum value or a local consensus value, where a part of the committee member has reached to a consensus.

So, you need to execute the protocol in further steps, if you are having a tentative consensus. Otherwise, if you have reached to a final consensus that means, the block has been reached to the consensus and the block is ready to be added in the existing blockchain. So, let us look into this notion of final consensus and tentative consensus in little detail.

(Refer Slide Time: 10:18)



Strong Synchrony versus Weak Synchrony

- **Strong Synchrony**: Most honest users (say, 95%) can send message that will be received by most other honest users within a known time bound
  - Adversary can not control the network for long
  - Ensures liveness of the protocol

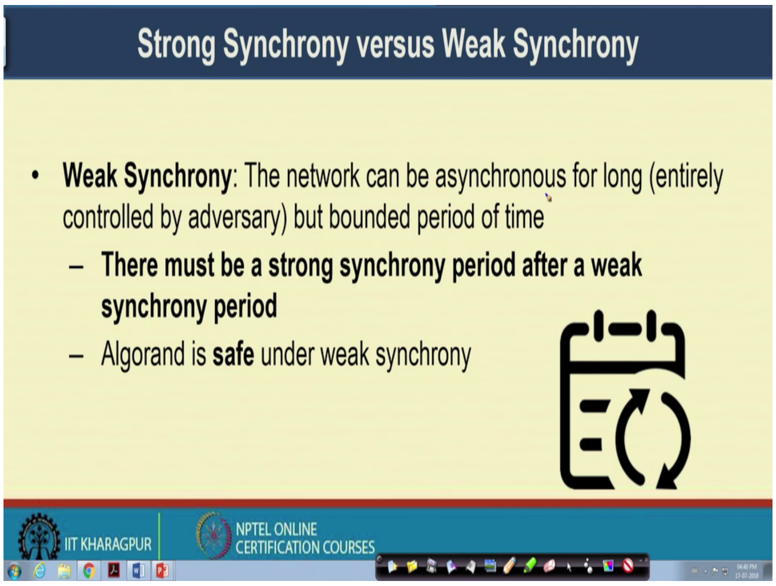
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, before going to the notion of final consensus and tentative consensus, we will look into the synchronous model assumptions which is done in Algorand. So, Algorand assumes to 2 different type of synchrony; Strong Synchrony and Weak Synchrony. So, Strong Synchrony says that most honest users. So, as we assume that around 95 percent of the honest user in the network, can send message that will be received by most of the honest users within a known time bound.

So, the idea of the strong synchrony is similar to the synchronous network assumption, where if you are honest user if you are sending a message. Then, all the honest users in the network or rather than having a that is 100 percent synchrony. So, the strong synchronous says that the majority of the honesty is users. So, around the 95 percent of the honest user in the network, they will receive that message within a predefined timed bound. So, in that case when the network is in the strong synchrony, then the adversary cannot control the network for long duration because these honesty users they will be able to receive the massage within a predefined time bound. And this strong synchrony assumption, it requires to ensure the liveness of the protocol.

So, whenever that adversary takes control of the network, we assume that it is synchrony assumption. Periodically, the network comes into this strong synchrony period when all the honest users are ore majority of the honest users will be able to exchange the messages among themselves and during that time these entire network will come out of liveness due to certain attacks from adversaries.

(Refer Slide Time: 12:02)


Strong Synchrony versus Weak Synchrony

- **Weak Synchrony**: The network can be asynchronous for long (entirely controlled by adversary) but bounded period of time
  - **There must be a strong synchrony period after a weak synchrony period**
  - Algorand is **safe** under weak synchrony

Then, comes the notion of weak synchrony weak synchronous says that the network can be as synchronous for long. So, whenever we are saying in Algorand that the network can be as synchronous for long we mean that the internet work is controlled by the adversaries, when the network is controlled by the adversary this indicates that the notes and not been able to send the message within a predefined time bound.

So, if you just think about our normal computer network, if the network is running fine or if the network is not been attacked; during that time if you are sending certain message we have certain level of quality of service from the network because of which the messages guaranteed to reach at the destination within in a predefined time intervals. So, if seldom happens that the network is fine, there is no attack to the network, but still the messages stuck somewhere and it is not been able to reach the destination. There can be congestion in the network in normal IP based network, but when the network comes out of congestion because of the TCP congestion control mechanism that we have.

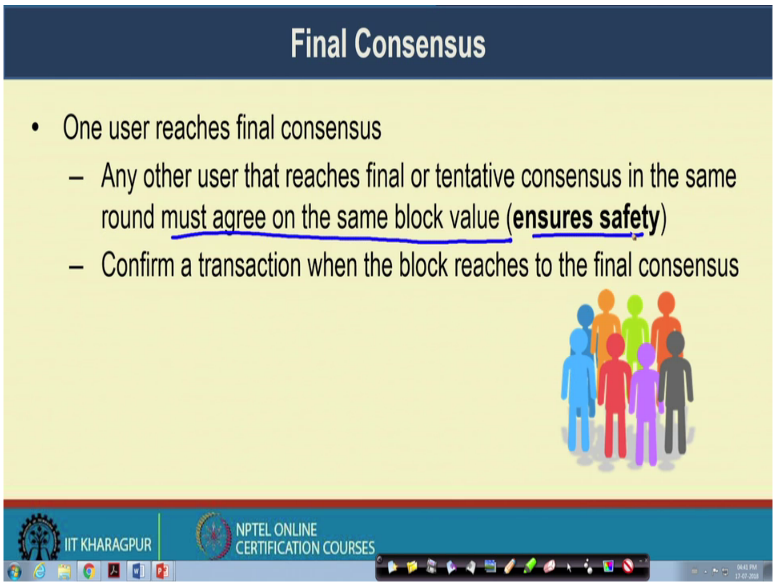
So, that way if you define some kind of realistic time bound within this realistic time bound it is guarantee that if the network is not under any kind of attack like a denial of service setback or something like this. Then, if a honest user send some message then all the honest user in the network or at least the majority of the honest users in the network, they will receive that message.

But that is not true, when the network is under the control of an adversary. So, certain attacks has been made on top of that network. So, either that can be a denial of service attack or similar kind of network attack, where the honest users are not able to send the messages any further. So, we consider that kind of scenario as a weak synchrony scenario. So, in case of a weak synchrony scenario the network can be asynchronous for long, where it is entirely controlled by the adversary. But bounded period of time it is not like that.

This kind of attack network, it is going for infinite duration. It can go for 10 hours or it can go for 10 days; but after some finite duration, it will come out of that attack and it will again reach to today's strong synchrony behaviour. It will show the strong synchrony behaviour. So, when the network is under weak synchrony, Algorand ensures that the protocol is safe the protocol will never accept invalid block. But the liveness may get hampered; the liveness will get resumed, once the network moves to the strong synchrony phase after a weak synchrony phase.

So, we must have a strong synchrony period after a weak synchrony period. So, Algorand is safe under weak synchrony. So, that is the guarantee we are providing under these weak synchrony assumptions. So, this is the difference between strong synchrony and weak synchrony and Algorand assumes that this entire network comes periodically in these 2 phases; the strong synchrony phases followed by a weak synchrony phases. Then, again a strong synchrony phase; again a weak synchrony phase and that way it goes on.

(Refer Slide Time: 15:13)



Final Consensus

- One user reaches final consensus
  - Any other user that reaches final or tentative consensus in the same round must agree on the same block value (**ensures safety**)
  - Confirm a transaction when the block reaches to the final consensus

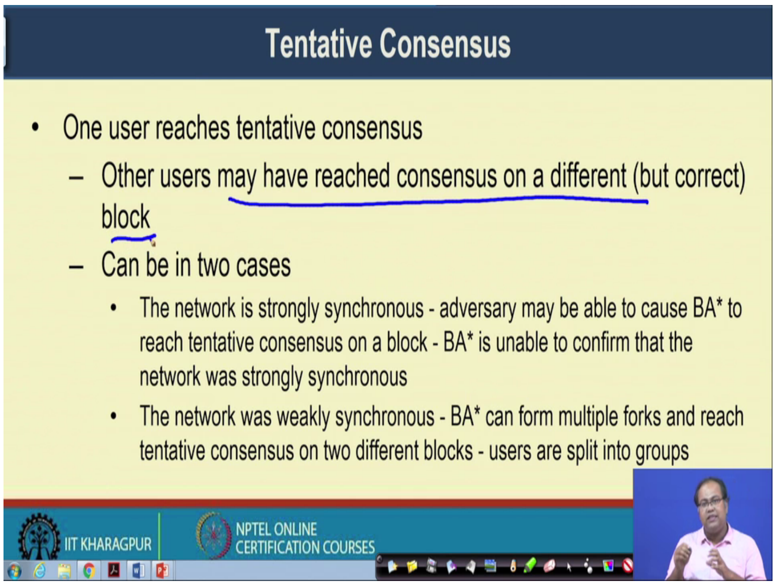
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, let us now look into this concept of final consensus and tentative consensus in the context of strong synchrony and weak synchrony network. So, in case of final consensus, one user reaches final consensus when any other use that reaches final or tentative consensus in the same round must agree on the same block values. So, this particular property of final consensus, it ensures the safety property of the protocol.

So, we say that any user who reaches to the final consensus; then, any other user that reaches either consensus or tentative consensus, they agree on the same block value. So, this ensures that all the members in the committee which has been selected by the cryptographic sortition algorithm, all the members in the committee has reached to a specific block value. So, the consensus has been reached and if we go for byzantine agreement kind of thing; so, we say that at least two-third of the committee members has reached to this final agreement on that particular block value.

And if two-third number of nodes has been reached to this final agreement based on this byzantine agreement protocol, then we say that the protocol is safe and the block that has been reached into consensus it is in the final consensus. So, it confirms a transaction when a block reaches to the final consensus. So, when a block reaches to the final consensus; that means, the transaction which has been proposed inside the block that transaction is final and that transaction can be executed.

(Refer Slide Time: 16:55)

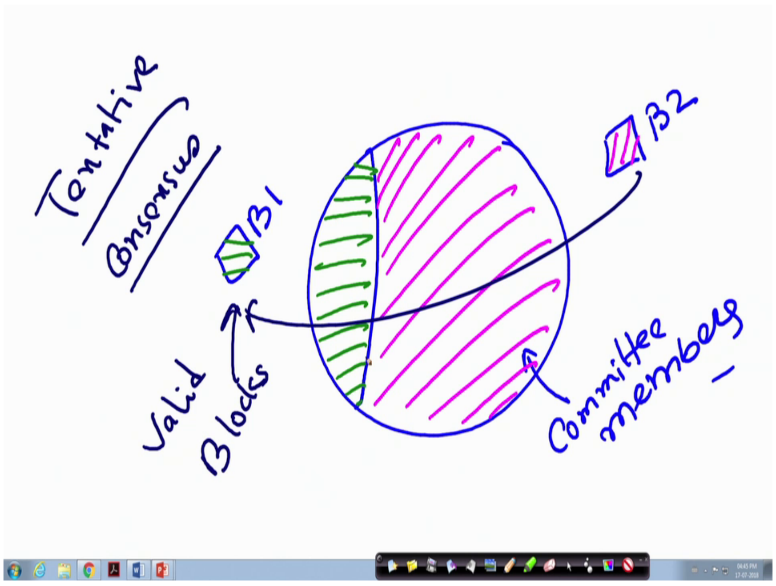


Tentative Consensus

- One user reaches tentative consensus
  - Other users may have reached consensus on a different (but correct) block
  - Can be in two cases
    - The network is strongly synchronous - adversary may be able to cause BA* to reach tentative consensus on a block - BA* is unable to confirm that the network was strongly synchronous
    - The network was weakly synchronous - BA* can form multiple forks and reach tentative consensus on two different blocks - users are split into groups

So, next we look the concept of Tentative Consensus. Tentative Consensus says that one user reaches Tentative Consensus when other users may have reached consensus on a different; but correct block.

(Refer Slide Time: 17:13)



So, it is just like that if I look into this entire committee member, entire set of committee members. So, these is my entire set of committee members and say 2 blocks has been proposed. So, one block has been proposed as B 1 another block has proposed as B 2. Now say half of the or some numbers of the committee members, they have reached consensus. So, this set of committee members they have reached consensus on block B 1 and the other set of nodes, they have reached consensus on block B 2. Now interestingly in this case both block B 1 and block B 2 are valid blocks. So, both of themes are Bm's are valid blocks.

So, both the blocks are valid blocks, but interestingly not all the members has reached to a particular block value and 50 percent of the nodes they have reached to one block value or it may not be 50 percent. It may be like half of the members, they have reached to a particular block value and remaining half or remaining members they have reached to a different block value. But both of them are the final blocks. So, the network has reached two consensus; but this consensus is not final because all the nodes are not reached to the same consensus.

So, this kind of consensus we call it as a Tentative Consensus. So, it says that well we have reached to a consensus, but in this case part of the members, they have reached to one they have reached consensus over one block and the part of the members, they have reached consensus to a different block. So, that is the notion of tentative consensus. So, if the protocol if the byzantine agreement protocol reaches to a tentative consensus, then we ensure that after a certain number of rounds.

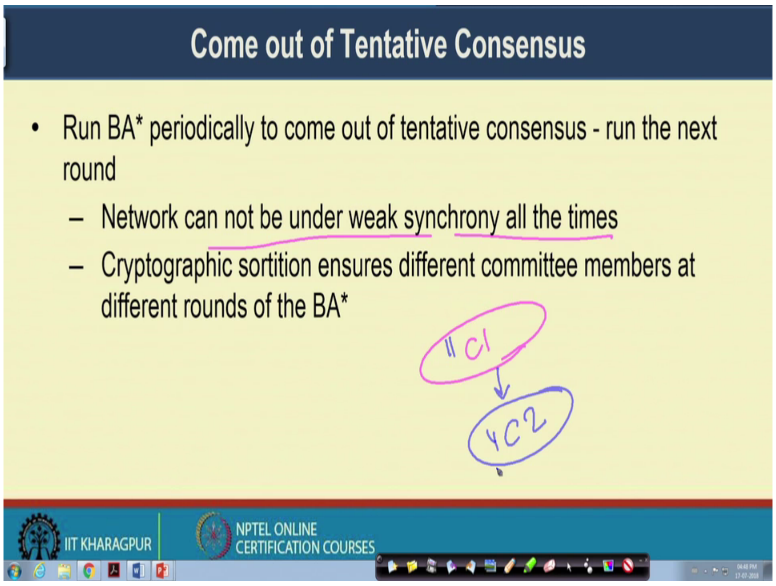
So, the protocol is not finaled here. We have to execute another route and after in the next round the entire protocol will be executed with a different set of committee members elected by the cryptographic sortition mechanism ok. So, tentative consensus; so, we can have tentative consensus in 2 cases; the first case is the network is strongly synchronous. When the network is strongly synchronous during that time adversary may be able to cause byzantine agreement to reach tentative consensus on a block.

So, byzantine agreement is unable to confirm that the network was strongly synchronous. So, if we look into the previous example in that case it may be happened that during a strongly synchrony period the adversary has taken control of the network. In that case, the adversary has made somehow these set of nodes to reach on the consensus and block B 1 that may be the reason because the adversary has launched certain kind of denial of service attack because of which block B 2 has not been propagated to the users which are there in this set of committee members. And because of which this set of committee members, they have not heard of block B 2.

So, they have reached to the consensus on block B 1. And the second case can be when the network was weakly synchronous, if the network is weakly synchronous; that means, the network is anyway under the control of the adversary. So, BA star can form multiple forks and reach tentative consensus on 2 different blocks so, because the network is the under adversary. So, it may happens that the adversary has launched certain kind of denial of service attack, where the network got partition because the network got partition we have multiple forks.

So, these multiple forks ensures that well 1 block has been propagated in one part of the node; another block has been propagated in another part of the node. And that way, the committee members who are spreaded in different part of the nodes they have reached to consensus on 2 different blocks. So, that can be the reason of a Tentative Consensus. Now, we ensure that whenever a consensus is reached in byzantine agreement protocol of Algorand that consensus is not a final that consensus is not a tentative consensus. If the consensus is a final consensus, then only the block is added to the existing block chain ok.

(Refer Slide Time: 22:03)



Come out of Tentative Consensus

- Run BA* periodically to come out of tentative consensus - run the next round
  - Network can not be under weak synchrony all the times
  - Cryptographic sortition ensures different committee members at different rounds of the BA*

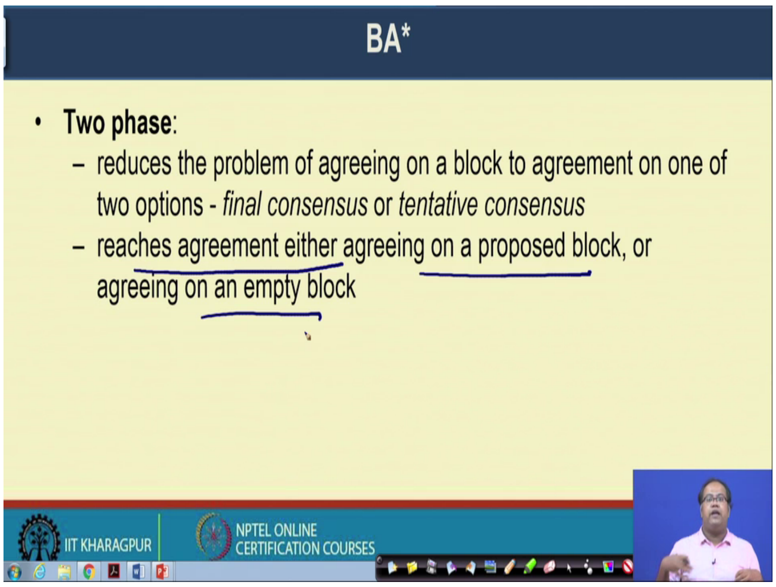
$$\begin{tikzcd} \boxed{\| C1} \arrow[d] \\ \boxed{\forall C2} \end{tikzcd}$$

So, to come out of that entity consensus, we run byzantine agreement periodically and in the next round the byzantine agreement is executed again. So, the assumption here is that the network cannot be under weak synchrony all the times. So, after sometime there would be strong synchrony and the cryptography sortition ensures that different committee members are selected at different round of the byzantine agreement protocol.

So, if I have 1 round of byzantine agreement protocol and in that round of byzantine agreement protocol say committee members C 1 was the set of committee members. Then, in the next round, it may happen that well a different set of nodes has been selected as a committee member and they are may not be any overlap between C 1 and C 2. So, that way we ensure that well if even if the adversary takes control of certain nodes certain nodes in C 1, there would be in the nodes in C 2 where the adversary cannot control.

So, that way, it will come out of tentative consensus and eventually reach to the final consensus and Algorand gives a value or gives a bound on the number of rounds that is required to reach into the final consonants. So, if the network is perfect, then in a single round of the byzantine agreement protocol, you will reach in the final consensus; otherwise you will require around 8 rounds to reach into the a final consensus.

(Refer Slide Time: 23:34)



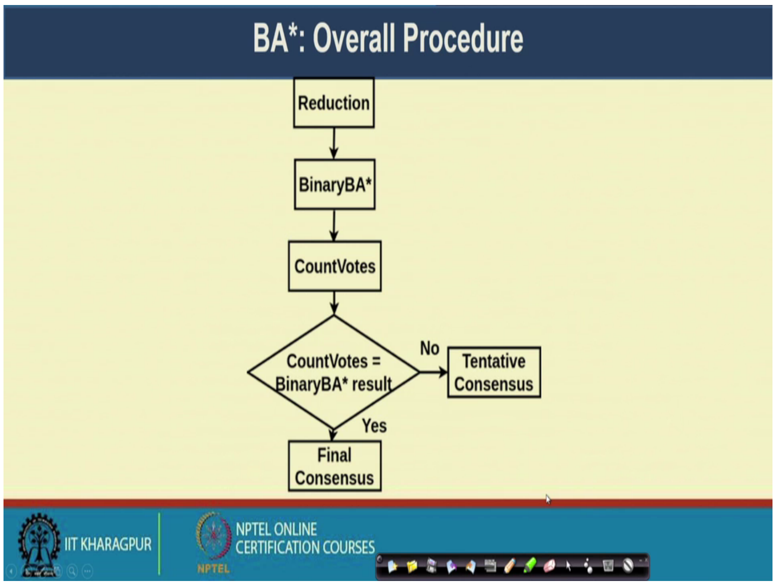
BA*

- **Two phase**:
  - reduces the problem of agreeing on a block to agreement on one of two options - *final consensus* or *tentative consensus*
  - reaches agreement either agreeing on a proposed block, or agreeing on an empty block

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, this second phase of the byzantine agreement protocol says that every node reaches agreement; either agreeing on a proposed block or agreeing on an empty block. Either they agree on one of the proposed blocks or they agree on some empty block; that means, none of the proposed block can be added to the existing blockchain.

(Refer Slide Time: 24:00)

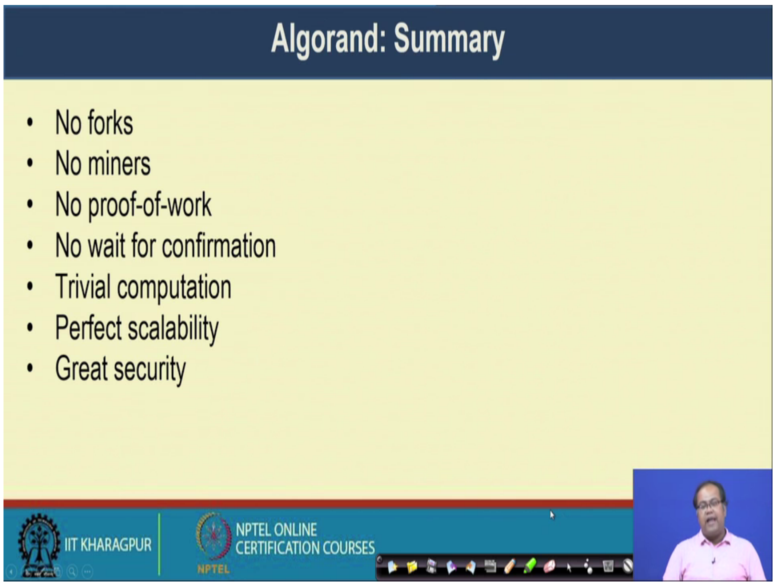


BA*: Overall Procedure

So, this is the overall procedure of the byzantine agreement. So, initially we have a reduction phase. So, this reduction phase applies cryptography partition to find out the committee members the members will be there in the committee.

Then, we execute the binary byzantine agreement protocol. The byzantine binary agreement protocol says that we agree on a block or not, then you count the votes; the number of votes that you have from this committee members. So, if this count vote is equal to the binary byzantine agreement places; that means, everyone has agreed on the same block. Then, it is a final consensus; otherwise, it is a tentative consensus if it a tentative consensus. Then, again you done the same thing in the next round by selecting a new set of committee members.

(Refer Slide Time: 24:53)



Algorand: Summary

- No forks
- No miners
- No proof-of-work
- No wait for confirmation
- Trivial computation
- Perfect scalability
- Great security

So, that is the broad idea of Algorand and in summary Algorand ensures that there is no fork because we are executing byzantine agreement protocol. So, after the byzantine agreement execution is over, you have only a single copy of the block that will get added to the blockchain. So, you do not have any fork or rather they have shown that the probability of having a fork is and it extremely low in Algorand.

We do not have any miner anyone can participate in the in the Algorand procedure and we have trivial competition. So, you do not require huge amount of resources to participate in the procedure. Then, you have no proof of work. So, you have no wastage of resources, you do not need to wait for confirmation.

So, whenever you get sufficient number of signatures from the network, you are sure that the block has been added. The computations are trivial we have a good scalability of the entire protocol and it provides nice security architecture because, of this cryptographic mechanism that different number of committee members and different members are selected as a part of the committee at different round.

So, it is difficult for the adversary to create a control of the majority of the members in a committee. So, that is about Algorands. So, I have tried to give you a very high level picture of how Algorand works and if you are interested you are you can read the original paper from the author and there are nice videos in the way from Sylvia Mikkeli, one of the designer of Algorand. So, you can look into that videos it that is a nice video to look into.

And there are certain theoretical aspects of the entire Algorand protocol, like the theoretical proof about how many number of different rounds you require for the byzantine agreement to converts to the final consensus or what is the probability of having a fork in Algorand. So, all these details you can find in the original paper. So, if you are interested you can a certainly look into that. So, that would be a interesting read for you. So, that is all about this particular lecture.

Thank you all for attending the class today.