CS 251: Consensus I

Instructor: Ben Fisch

Recap: The Consensus Problem

- ***"State Machine Replication"*** on N servers
- Stream of transactions: $tx_1, tx_2, \ldots$
- For $i = 1, \ldots, n$: $L_i(t)$ is a list of confirmed Tx at server i at time t
- Goal: Protocol that satisfies two properties:
  - ✓Consistency
  - ✓Liveness

2

Recap: The Consensus Problem

Consistency

For all servers $i, j \in [N]$ and times t, t':

Either list $L_i(t)$ is a prefix of $L_j(t)$ or vice versa

If $t < t'$ then $L_i(t) < L_i(t')$

Liveness

Exists function T such that: if *any* honest server receives tx at time t then $\forall i \;\; L_i(t + T(\Delta))$ contains tx

Δ = *maximum network delay*

T is a polynomial

Authentication: PKI Model

- Each server $i \in [N]$ identified by public key PK_i
- Digital signature scheme $\{Setup, Sign, Verify\}$
- Server i signs message m as $sig \leftarrow Sign(SK_i, m)$
- Other servers verify signature by running $Verify(PK_i, sig, m)$

Network Synchrony Models

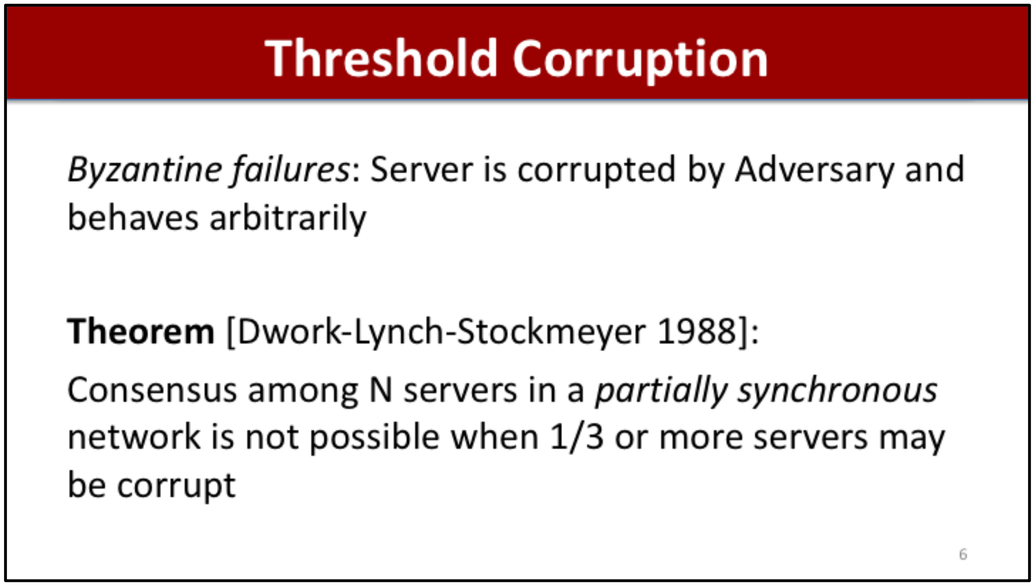
- **Synchronous:** There is a <u>known</u> maximum delay Δ such that any message sent from one server to another is delivered within Δ time.
  - Protocol *can* use Δ as a parameter
- **Partially Synchronous:** Δ exists but is <u>unknown</u>
  - Same protocol must work for any Δ
- **Asynchronous:** Network experiences arbitrary failure
  - Consensus problem unsolvable

5

Will revisit in more detail in next lecture.

Fact that arbitrary network failures prevents consensus is trivial. Network can split into two isolated parts that each must reach agreement by liveness, but can be on different values because they are isolated, so either consistency or liveness is violated.

More subtle impossibility is the FLP impossibility, which says that in a network where at most one server fails and every message between non-failed servers are eventually delivered, it is still impossible to solve the consensus problem. Therefore we cannot hope to solve it for some bound on the number of server failures.



Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the presence of partial synchrony. J. ACM, 1988.

One-shot consensus (Byzantine Agreement)

- A designated server is called the proposer
- Proposer broadcasts tx to every other server

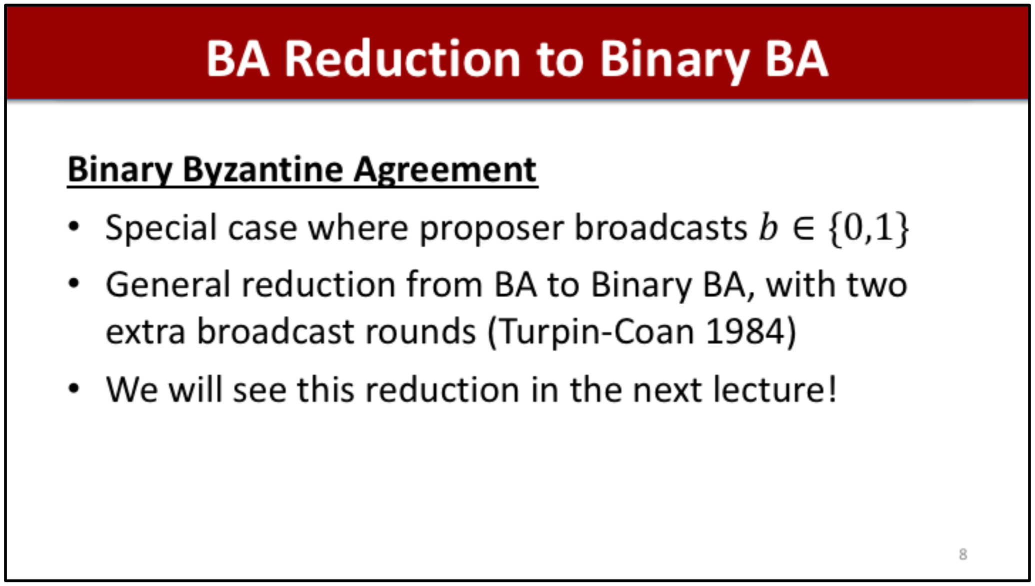
Goal: Each server outputs $tx_i \in \{0,1\}^*$. Protocol guarantees:

- $\forall i, j$ if servers i and j are honest then $tx_i = tx_j$
- If proposer is honest then $\forall i \; tx_i = tx$

Challenge: malicious proposer might send different tx values to different servers

7

Problem is that a malicious proposer could send different tx values to different severs!



R. Turpin and B. Coan. Extending binary Byzantine agreement to multivalued Byzantine agreement. Inform. Process. Lett., 18 (1984), pp. 73-76.

Basic Binary BA

Assume synchronous model

Servers broadcast signed vote b_i

Each server counts votes: if ≥ 2N/3 votes seen for then output b^*

If a server sees < 2N/3 votes, enter TIMEOUT. This initiates a more complex recovery path… we'll see one in the next lecture.

Basic Binary BA

- No honest servers i, j output $b_i \neq b_j$
  - Implies ≥ 2N/3 votes for both b_i and b_j.
  - *Let $S_i \subseteq [N]$ be the set of servers that i saw vote for b_i*
  - Let $S_j \subseteq [N]$ bet the set of servers that j saw vote for b_j.
  - $|S_i \cap S_j| \geq N/3$, hence the sets intersect on at least one honest server as less than N/3 servers are corrupt
  - This implies an honest server voted for distinct values, a contradiction.

10

From BA to SMR

- Leader election (e.g. round robin)
  - Synchronous clocks → time epochs
  - New **leader** elected each epoch
- Leader acts as **proposer** in BA protocol
  - In epoch r leader broadcasts tx_r
  - Servers run BA on tx_r

11

PKI Permission Models

- **Fixed participation:** N servers in PKI model
- **Weighted PKI:** List of (PK_i, w_i). Votes signed by PK_i are weighted by w_i.
- **Dynamic Weighted PKI:** Weights on public keys are updated through consensus

  Equivalent to "proof-of-stake", where weights are balances in an account

Non-PKI permission models

- Without PKI, each participant could pretend to control arbitrarily many servers, called the *Sybil attack*
- Other forms of assigning weights to participants known as “Sybil resistance”.
- **Proof-of-work**: Participant voting power is proportional to energy *consumed* (solving computational puzzle)
- **Proof-of-space**: Participant voting power proportional to storage space dedicated to system

Proof of work

Hash function $H: X \times Y \rightarrow \{0,1\}^{256}$ (e.g. SHA256)

Puzzle Difficulty: Integer D

Puzzle input: Challenge $c \in X$

Puzzle solution: Value $r \in Y$ s.t $H(c,r) < 2^{256}/D$

<u>Question:</u> If H modeled as *random function* such that for any x,r,z probability H(x,r) = z is 2^{-256}, then what is the expected number of trials to find a solution r?

14

Probability 1/D that H(c, r) falls in range [0, 2^256/D). Expected number of trials is D.

Proof of work

- $\text{PuzzleSetup}(\lambda, D) \rightarrow pp$
- $\text{PuzzleSolve}(pp, c) \rightarrow r$
- $\text{PuzzleVerify}(pp, c, r) \rightarrow 0/1$

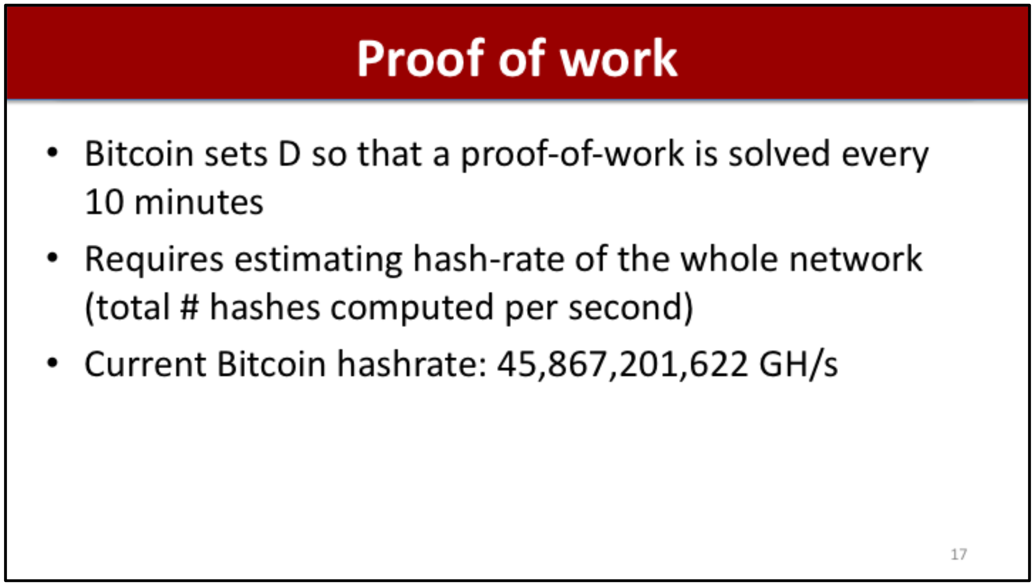
Proof of work

Definition: H is **proof of work** secure if runtime of H is T_H and for every algorithm A and $\epsilon > \frac{1}{D}$ such that RunTime(A) < $\epsilon D \mathrm{T_H}$, for random challenge x:

- $\Pr[H(x, A(x)) < 2^{256}/D] < \epsilon$

16

Probability 1/D that H(c, r) falls in range [0, 2^256/D). Expected number of trials is D.



45 * 10^9 * 10^9 hashes/second

= 45 * 10^18 hashes/second

= 2.7 * 10^21 hashes / ten minutes

D ~ 2^{74.5}

Consensus for large scale participation

- Classical permission model expects N to be a small number of participants
- Classical BA & SMR protocols don't scale for large N (too much communication)
- *Next:* Committee/leader election deigned to reduce network communication with large scale participation

Two paradigms for leader election

1. In each epoch, **sample “random” committee** from total # of participants (or weighted by w_i in weighted PKI model). Run classical BA in the committee.
2. **Race for the next leader slot**, leader immediately proposes transactions. Probability of winning the race is proportional to Sybil power (e.g. first to solve PoW puzzle).

Nakamoto consensus

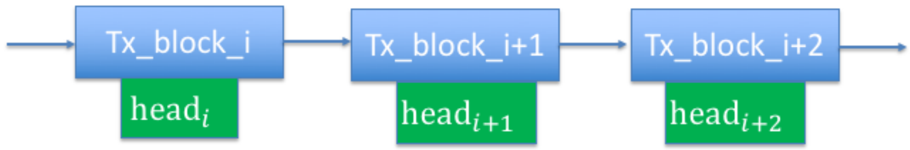
- Follows the second paradigm
- Originally presented in PoW permission model, later adapted to fit with other permission models [PassShi'17]
- Is "fully permissionless" in PoW setting:
  - Don't know exact # of nodes participating
  - Nodes come and go, i.e. "late joining"
  - No-authentication: anyone can join by solving PoW

20

Rafael Pass and Elaine Shi. The sleepy model of consensus. In Asiacrypt, 2017.

Nakamoto consensus

- State-machine transactions as blockchain



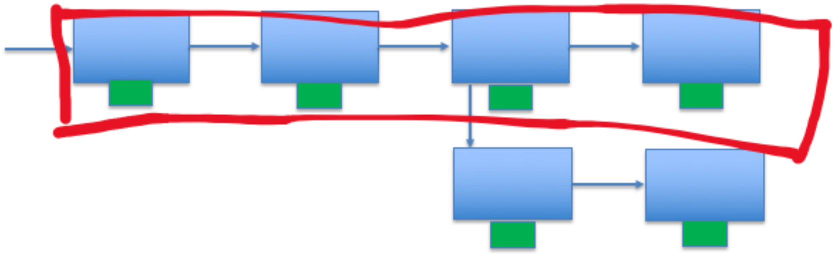
$$\text{head}_{i+1} = H(\text{head}_i, \text{Tx\_block\_i+1}, nonce)$$

$nonce \leftarrow$ PuzzleSolve(D, [head_i, tx_block_i+1])

21

Nakamoto consensus

- Fork rule (fixed difficulty): **extend longest chain**



Question: what happens when puzzle difficulty varies over time? ⇒ follow “**heaviest” chain**

22

Nakamoto consensus

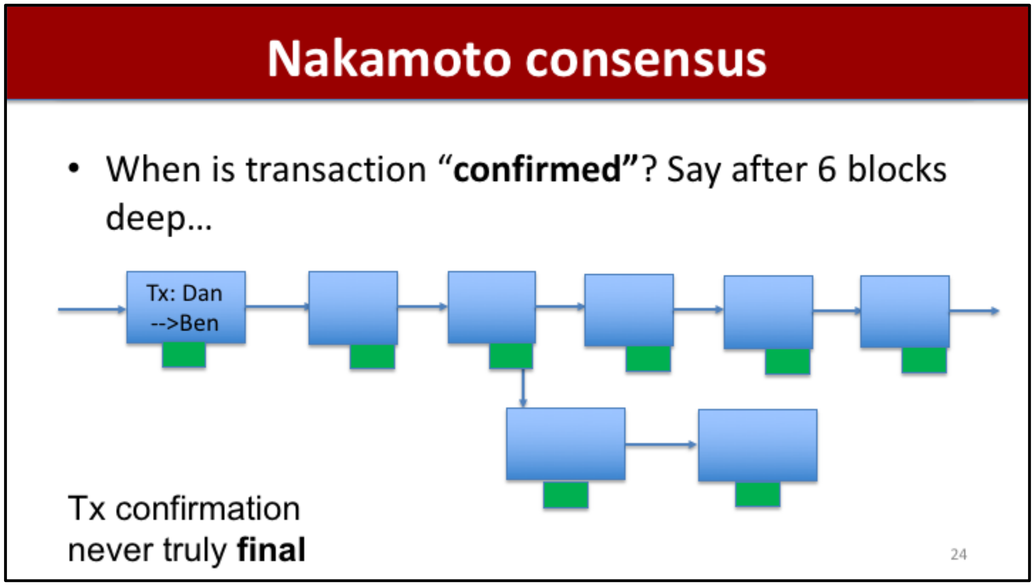
<u>Protocol:</u>

Every consensus participant (aka miner) works (i.e. solves PoW puzzle) to find a **valid** block head extending *heaviest* **valid** chain in its local view. Broadcast extension immediately upon discovery.

Heaviest by weight ➔ sum of all PoW puzzles in chain weighted by difficulty

23

Beautifully simple!



Probabilistic reasoning that after sufficiently deep transaction will not be reversed, as long as majority of work performed by honest miners

Nakamoto consensus

Consistency intuition:

- Miner controlling 49% power can immediately fork chain by 1 block and beat all other miners w/ prob. close to ½, or by 2 blocks with prob. close to ¼
  - No problem! If miner broadcasts fork, other miners switch as this is now the longest chain
- What if miner forks chain 6 blocks deep and doesn’t broadcast until it has a longer chain than honest?
  - Probability 1/64 it mines 6 blocks before honest

25

Probability of privately mining longest chain faster than honest portion of network degrades exponentially

Nakamoto consensus

Next lecture

- Relationship between network delay Δ and work difficulty. *What happens if miners can solve puzzles faster than they can propagate solutions through network?*
- What happens to Nakamoto in a*synchronous* (or *partially synchronous)* network?
- Mining incentive compatibility

References

- [DLS88] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the presence of partial synchrony. J. ACM, 1988.
- [TC84] R. Turpin and B. Coan. Extending binary Byzantine agreement to multivalued Byzantine agreement. Inform. Process. Lett., 18 (1984), pp. 73-76.

End