CS 251: Consensus II

Instructor: Ben Fisch

1

Recall: Consensus characteristics

- Security properties: consistency & liveness
- Network models
  - synchronous, partially synchronous, asynchronous
- Threshold corruption
  - Less than 1/3 in partially synchronous, possible to achieve higher in synchronous
- Permission models
  - Fixed PKI, weighted PKI, dynamic PKI, proof-of-work

2

Recap: Nakamoto consensus

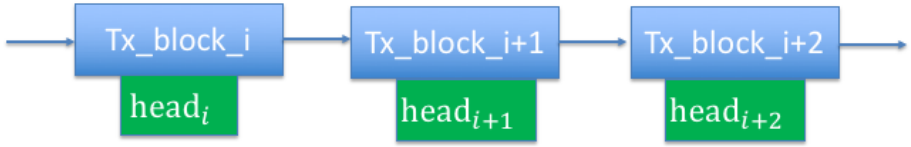
- Follows the "race for leader slot" paradigm
- Originally designed for PoW, later adapted to weighted PKI (PoS) [PassShi'17]
- Is "fully permissionless" in PoW setting:
  - Don't know exact # of nodes participating
  - Nodes come and go, "late joining"
  - No-authentication: anyone can join by solving PoW

3

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

Nakamoto consensus

- State-machine transactions as blockchain



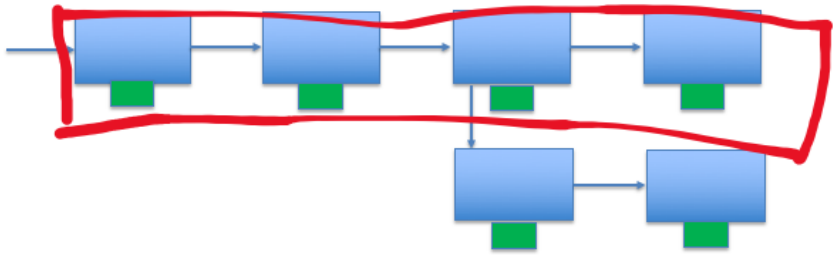
$\text{head}_{i+1} = H(\text{head}_i, \text{Tx\_block\_i+1}, nonce)$

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

4

Nakamoto consensus

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



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

5

Nakamoto consensus

Protocol:

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.

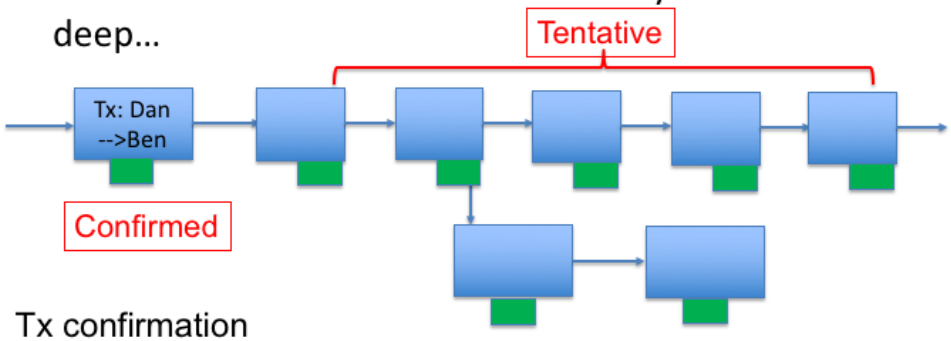
Heaviest by weight ➔ sum of all PoW puzzles in chain weighted by difficulty

6

Beautifully simple!

Nakamoto consensus

- When is transaction "**confirmed**"? Say after 6 blocks deep...



Tx confirmation never truly **final**

7

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: Suppose adversary has 49% power

- Adversary can fork chain by 1 block faster than honest miners extend current chain w/ prob. close to ½, or by 2 with prob. ¼
  - No problem! If adversary broadcasts fork, everyone switches, 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 mines 1
  - Probability $< 8 * 2^{-7}$ it mines 7 blocks before honest mines 2
  - What is probability adversary ever catches up?

8

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

Nakamoto consensus

Consistency intuition: (continued…)

Suppose adversary has $p < 1/2$ fraction of power. What is the probability adversary catches up from 6 blocks behind?

- *Simplified model:* repeated rounds, in every round adversary catches up by 1 block with probability p, and falls behind by 1 block with probability $1 - \mathrm{p}$.
- Biased random walk on number line starting at 0, +1 with probability p and -1 with probability $1 - p$. Probability walk ever reaches 6?
- Probability P_z that walk ever reaches +z is $(\frac{p}{1-p})^z$ (e.g. p = 1/3, then $P_6 < 0.0062$)

9

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

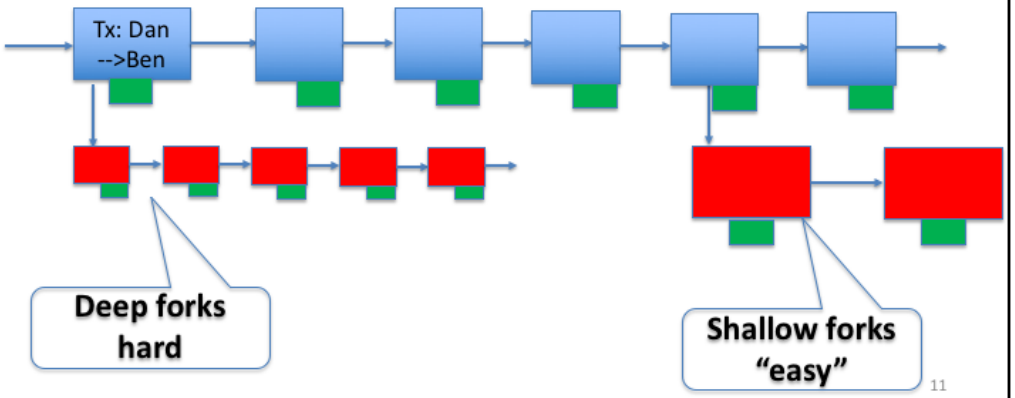
Nakamoto consensus

What goes wrong if adversary has $p > 1/2$ power?

- Adversary's private fork grows at faster rate than honest chain
- For any k, adversary starts k blocks behind, will eventually catch up to length of honest chain

10

Nakamoto consensus



Nakamoto consensus

Network delay & work difficulty

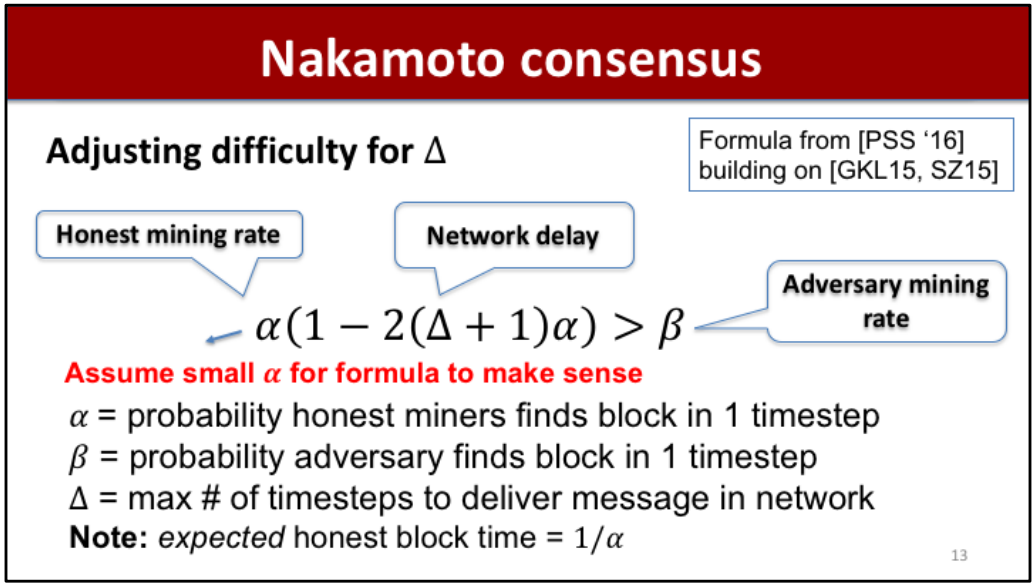
- What happens if miners can solve puzzles faster than they can propagate solutions through network?
- Adversary might receive the next valid block Δ steps ahead of the other honest nodes (Δ = delay)

⇒ *Adversary starts working on next puzzle with a Δ time head start over other honest nodes*

$O(\Delta)$ "free" hash trials

12

Say Δ is greater than the time it takes the adversary to solve puzzles. In worst case, honest nodes only start working on next puzzle every Δ time steps, after they have heard a block from other honest nodes, whereas the adversary hears blocks immediately, solves the next block in time less than Δ, and starts working on the next one, etc. This adversary is now mining blocks at a faster rate than the honest nodes in the network.



Rafael Pass, Lior Seeman, and Abhi Shelat. Analysis of the blockchain protocol in asynchronous networks. In Eurocrypt, 2017.

Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol with chains of variable difficulty. Eurocrypt, 2015.

Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in bitcoin. In Financial Cryptography and Data Security, pages 507–527. Springer, 2015.

Nakamoto consensus

Adjusting difficulty for Δ

Formula from [PSS '16] building on [GKL15, SZ15]

Honest mining rate

Network delay

Adversary mining rate

$$\alpha(1 - 2(\Delta + 1)\alpha) > \beta$$

Small

If $\Delta = 0$, basically says $\alpha > \beta$ (i.e. $\frac{\alpha}{\beta} > \frac{1}{1-2\alpha} \approx \frac{0.51}{0.49}$ for $\alpha = 0.02$)

If $\Delta \geq \frac{1}{2\alpha} - 1$, formula never true because $1 - 2(\Delta + 1)\alpha = 0$

(i.e. when $\Delta > \frac{E[honest\ block\ time]}{2}$)

14

Nakamoto consensus

Adjusting difficulty for Δ

Formula from [PSS '16] building on [GKL15, SZ15]

Honest mining rate

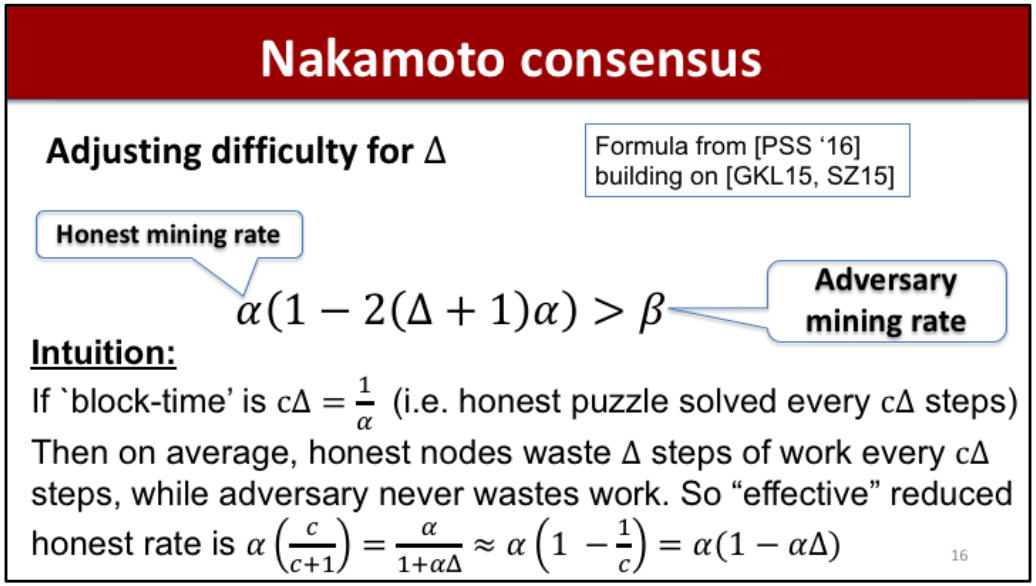
$$\alpha(1 - 2(\Delta + 1)\alpha) > \beta$$

Adversary mining rate

Intuition:

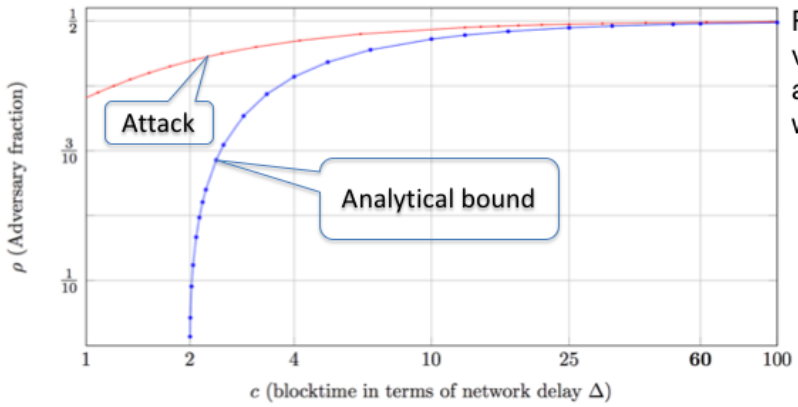
If Δ is larger than the blocktime ($\Delta > \frac{1}{\alpha}$), then the honest nodes in the network only agree at a rate of one block per Δ, in which time the adversary mining at a rate $\beta < \alpha$ might also extend its private chain by one or more blocks, as long as $\Delta > \frac{1}{\beta}$.

15



Every (c + 1)Delta steps, the honest parties only effectively work for c Delta steps, and thus mine only \alpha c \Delta blocks every (c + 1)Delta steps, for a total rate of \alpha c/(c+ 1)

PSS Theorem Graph



Red line = min p value for which attack from PSS works

Nakamoto magically chose c = 60 (10 min blocktime assuming 10s network delay)

Blue line = max value of p s.t. $\frac{\beta}{\alpha} = \frac{p}{1-p}$ and $\frac{\beta}{\alpha} < 1 - 2(\Delta + 1)\alpha$

17

Nakamoto consensus

What about liveness?

- In analyzing consistency, we dismissed shallow forks because the last 6 blocks are *unconfirmed,* and honest nodes simply switch to the adversary's longer fork.
- But could the adversary persistently shallow fork the chain, preventing txs from ever being confirmed? *This is a liveness issue.*

18

Nakamoto properties

1. **Consistency.** Honest nodes agree on all but last **k** blocks (except with prob. exp(-k))
2. **Chain quality.** Any consecutive **k** blocks contain "sufficiently many" honest blocks (except with prob. exp(-k)). *Miners controlling p fraction of power should roughly mine p fraction of blocks.*
3. **Chain growth.** Chain grows at a steady rate.

   g-chain growth: Growth by k blocks every k/g "rounds"

19

Nakamoto properties => SMR

- Consistency implies SMR consistency
- Chain growth + chain quality implies SMR liveness
  - The chain grows by k blocks every k/g periods
  - By chain quality, a high fraction of blocks are contributed by honest miners, and therefore include all transactions they heard so far

20

Nakamoto chain quality

- Say honest mining rate is α and adversary mining rate is β, let $p \leftarrow \frac{\beta}{\beta+\alpha} < ½$
- Ideally honest parties mine a 1 – p fraction
- Can prove they mine at least $1 - \frac{p}{1-p} = 1 - \frac{\beta}{\alpha}$ fraction

If $\beta > \alpha$ then adversary could mine every block in worst case ⇒ chain quality is 0

21

Pass-Seeman-Shelat Theorem

- For every $p < ½$, if mining difficulty is appropriately set as function of network delay Δ then Nakamoto consensus guarantees:
    1. Consistency (for α, β, Δ satisfying formula)
    2. Chain quality: $1 - \frac{p}{1-p}$ fraction blocks honest
    3. O(1/Δ)-Chain growth

22

Nakamoto incentive compatibility

- Participating in Nakamoto is expensive! Need to solve PoW puzzles…
- In real world (especially permissionless setting) important to consider participant incentives.
- Nakamoto's genius **block reward** idea: reward participants for becoming the leader by minting new Bitcoins
- High variance rewards → mining pools

23

A small miner has low probability of successfully mining a block so there is extremely high variance on the reward, even though the miner must continuously put in work..

Miners cooperate in pools to lower this variance. Different ways to distribute rewards among participants in the pool anytime the pool mines a block.

Selfish mining attack

- Nakamoto no longer tolerates 49% corruption in a rational world!
- Surprising attack, Eyal and Sirer 2013
- **Block-withholding strategy:** If you find a valid block, don't broadcast immediately, keep privately working to extend it, causing others to waste effort. Only broadcast privately mined blocks when the rest of the network finds a block.
- Say adversary w/ 30% power finds every third block, so gains advantage every three blocks relative to effort. Other miners incentivized to join the selfish-mining pool.

24

Selfish mining starts to become profitable around 22% control. Analysis via Markov chain models, Bai et. al. 2018.

Also analysis changes for risk-averse miners, which discounts expected reward by risk (seeks to lower variance as well as reward). If a miner infrequently finds a block, selfishly holding onto it instead of immediately broadcasting also increases the risk it will not be rewarded for the block.

Qianlan Bai, Xinyan Zhou, Xing Wang, Yuedong Xu, Xin Wang, and Qingsheng Kong. A deep dive into blockchain selfish mining. arXiv preprint arXiv:1811.08263, 2018.

I. Eyal and E. G. Sirer. "Majority is not enough: Bitcoin mining is vulnerable". J. Comm. ACM, 2018. (Earlier edition in Financial Cryptography, 2014).

Random Beacon

Ideal Random Beacon

At fixed time intervals, a magic random number ”appears in the sky”

25

Weighted PKI Committee Election

Ideal Random Beacon

At fixed time intervals, a magic random number "appears in the sky"

Let $W = \sum_i w_i$ be the total sum of weight on PK_is

In epoch t the random number r_t determines a random "committee" of M public keys. Probability PK_i included in committee is w_i/W

26

Weighted PKI Committee Election

Verifiable Random Function (VRF)

$\mathrm{F}(sk, x) \rightarrow y \in \{0,1\}^{256}$

$\mathrm{Verify}(PK, x, y) \rightarrow 0/1$

y is indistinguishable from random to anyone who doesn't know sk

Set D = W/M

In epoch t there is a "seed" x_t

Each party with (PK_i, sk_i) computes $y_{i,j} \leftarrow \mathrm{F}(\mathrm{sk_i}, \mathrm{x_t}||\mathrm{j})$ for each $j \in [0, w_i)$

Each $y_{i,j} < \frac{2^{256}}{D}$ gives PK_i a committee "slot". Broadcast.

27

If target number of committee slots is M, then want each y_i,j to succeed with probability M/W so that total number of committee members is M is expectation. So set 1/ D = M/ W, … i.e M D = W/M

VRF Committee Election

- In each epoch t, participants evaluate VRF to see how many committee slots they have.
- Participant i broadcasts each eligible $y_{i,j}$ to network, anyone can verify eligibility using: Verify$(PK_i, x_t||i, y_{i,j})$
- Elected members run classical BA protocol to reach agreement on TX block for epoch t, where PK_i gets one "vote" in the BA per slot

28

Random Beacon Techniques

Other techniques for constructing random beacons:

- Verifiable Delay Functions [BBBF'18]
- "Threshold Relay" (Dfinity project)
  - Using deterministic threshold signatures

29

Dan Boneh, Joseph Bonneau, Benedikt Bünz, Ben Fisch. Verifiable Delay Functions. Crypto, 2018.

Timo Hanke, Mahnush Movahedi, and Dominic Williams. DFINITY Technology Overview Series Consensus System. Arxiv, 2018.

Using Beacon for BA

"Byzantine Agreement, Made Trivial" –Micali, 2018

- Synchronous model, binary BA
- All players see fresh magic coin $c \in \{0,1\}$ at start

Round 1: $\forall_i$ server i broadcasts signed vote (b_i, σ_i)

Each server counts votes (including own vote):
- If i sees ≥ 2N/3 votes for b^* then output b^*
- Else, output c

30

Silvio Micali. Very Simple and Efficient Byzantine Agreement. ITCS 2017.

Using Beacon for BA

Round 1: $\forall_i$ server i broadcasts signed vote (b_i, σ_i)

Each server counts votes (including own vote):
- If i sees ≥ 2N/3 votes for b^* then output b^*
- Else, output c

Analysis (consistency w. prob ½ for threshold corruption 1/3)
- If honest servers all start with b^* all honest output b^*
- All honest servers that see ≥ 2N/3 votes output the same b^*
- All servers who don't see ≥ 2N/3 votes output c
- $c = b^*$ with probability ½

31

Using Beacon for BA

- The simple protocol we presented achieves consistency with probability ½ (so?)
  - This might seem trivial…all parties could flip a coin and agree with probability ½, no? But this wouldn't satisfy the first condition: if all honest servers start with same input then they all agree w. prob. 1.
- Micali shows how to amplify this probability by adding more rounds of *correlated* executions.

32

Using Beacon for BA

In round r all players see fresh magic coin $c_r \in \{0,1\}$

Call $\geq$ 2N/3 votes for some b a "quorum vote" for b

When server "HALTS" on output b, it sends the message (b, FINAL)

If server receives (b, FINAL) from ith server, it fixes this server's vote to the value b for all future rounds.

In round r:
All servers repeat for three sub-rounds t = 0, 1, 2:

- Server i broadcasts signed vote (b_i, σ_i)
- If server sees a quorum vote b^* then set $b_i \leftarrow b^*$. If $t = b^*$, HALT.
- Else, if $t \neq 2$ the server sets $b_i \leftarrow t$
- Else, if $t = 2$, set $b_i \leftarrow c_r$. Non-halted servers advance rounds & repeat.

Claim 1: In any round, if no servers halt, they agree at the end of the round with probability ½.

Proof: All servers enter t = 2. All servers that see a quorum for b* agree. All servers that don't see any quorum agree with the others with prob. ½.

Claim 2: If agreement holds at any point (in terms of the current output values of halted servers, and the b_i values of non-halted , it continues to hold.

Proof: All honest servers agree, thus there is a quorum vote for this value, thus they still agree at the end of the round.

Claim 3: If some server HALTs in some round, then all servers reach agreement at the end of that round.

Proof: The halted server saw a quorum, therefore no servers see a quorum on a different value at this same sub-round. If a server halted during t = 0, then all other servers set b_i to 0. If a server halted during t =1, then all other servers set b_i to 1.

Claim 4: If servers reach agreement in round r, then any non-halted servers will halt in round r+1.

Proof: Assume servers agree on b at beginning of round r. If b = 0, there will be an honest quorum vote for 0, and they halt in the first sub-round where t = 0. They don't halt in first sub-round if b = 1, but by by Claim 2 agreement on 1 continues to hold, and there is an honest quorum vote for 1 in the second sub-round where t = 1, upon which all servers halt.

Finally, consider two cases. If the servers all agree at the start then the protocol terminates in 1 round and all servers output the same value.

If the servers do not agree at the start, then by iterating Claim 1, the probability they don't agree by end of round t is 2^{-t}. By Claim 4, they halt in the next round after reaching agreement. In the case that they start with different inputs, the expected number of rounds it takes to reach agreement is 2, and therefore the expected rounds to terminate is 3. Once all servers terminate they know that they have reached agreement. (They know that all servers have halted because of the FINAL messages sent by halted servers.

BA with Partial Synchrony

- Consider previous simple protocol with 1-round voting and ½ probability of consistency: what could go wrong in a partially synchronous model?
- Some honest servers may not hear all votes at same time (long unpredictable network delay). But for liveness need to keep the protocol moving, can't wait forever…

34

One-round voting insufficient

Suppose server A hears 2N/3 votes for b^* and outputs b^*, but all other servers do not hear these votes. They cannot wait forever, eventually enter a phase TIMEOUT.

- Case 1: Timed-out servers do not change votes. Then agreement is impossible starting from distinct inputs.
- Case 2: Timed-out servers can change votes. Then might agree on different output than A.

35

Binary BA with Partial Synchrony

- More advanced protocols…
  - Byzantine Paxos, PBFT, HotStuff, … many others

36

Maofan Yin, Dahlia Malkhi, Michael K. Reiter, Guy Golan-Gueta, Ittai Abraham. HotStuff: BFT Consensus with Linearity and Responsiveness. PODC 2019.

Responsiveness

- In BA protocol above, servers can instantaneously confirm transaction in good case that 2N/3 signatures are received in both rounds
- Consensus protocols that can return an answer immediately under optimistic network conditions are called **responsive** (don't need to wait for time Δ)
- No **responsive** protocol exists tolerating more than 1/3 corruption (Pass-Shi-17)

37

Tight connection between protocols designed for partially synchronous network, which are unaware of Delta, and those for synchronous. Intuitively, the synchronous protocols take advantage of Delta, and therefore wait for Delta time steps, so not responsive.

Rafael Pass and Elaine Shi. Hybrid Consensus: Efficient Consensus in the Permissionless Model. DISC 2017.

Binary BA Reduction

Given: Binary BA tolerating 1/3 corruption of N servers

Round 1: Servers broadcast votes for their input txs

Round 2: Servers count # votes received identical to own. If 1/3 or more of votes are different broadcast "CONFUSED"

Round 3: If server i receives "CONFUSED" from N/3 servers set $b_i \leftarrow 0$, otherwise set $b_i \leftarrow 1$.

Run BA: Run BA on inputs b_i to agree on b.

If b=1, non-confused servers output original tx input, and confused output the most popular tx from of non-confused servers. Otherwise if b = 0, output "Fail".

38

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

Binary BA Reduction

Analysis

If leader honest (all honest servers have same input), no honest servers are confused. All output tx.

Consider two cases if leader dishonest.

Case 1: b = 0**:** Everyone outputs ”Fail”

Case 2: b = 1: Any 2N/3 servers contain majority of honest servers. Non-confused servers inputs agree with majority (all same). There are less than N/3 confused servers. The N/3 non-confused honest votes dominate the malicious votes. So their view of majority agrees with non-confused servers.

39

Summary

- Nakamoto consensus
  - Synchronous model, Slow
  - ½ corruption threshold
  - Fully permissionless, also tolerates "sleepy" nodes
- Classical & proof-of-stake consensus (e.g. PBFT, BA*, HotStuff combined w/ Beacons or VRFs)
  - Partially synchronous model, Responsive (fast confirmation)
  - 1/3 corruption threshold
  - PKI or Weighted PKI

40

Give examples of modern protocols: PBFT, Algorand's BA*, HotStuff designed for partially synchronous networks

References

- [PS'17] Rafael Pass and Elaine Shi. The sleepy model of consensus. In Asiacrypt, 2017.
- [PSS'17] Rafael Pass, Lior Seeman, and Abhi Shelat. Analysis of the blockchain protocol in asynchronous networks. In Eurocrypt, 2017.
- [GKL15] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol with chains of variable difficulty. Eurocrypt, 2015.
- [SZ15] Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in bitcoin. In Financial Cryptography and Data Security, pages 507–527. Springer, 2015.
- [EG14] I. Eyal and E. G. Sirer. "Majority is not enough: Bitcoin mining is vulnerable". J. Comm. ACM, 2018. (Earlier Financial Cryptography, '14).
- [BBBF18] D. Boneh, J. Bonneau, B. Bünz, B. Fisch. Verifiable Delay Functions. Crypto, 2018.
- [HMW18] Timo Hanke, Mahnush Movahedi, and Dominic Williams. DFINITY Technology Overview Series Consensus System. Arxiv, 2018.
- [M'18] Silvio Micali. Very Simple and Efficient Byzantine Agreement. ITCS 2017.
- [YMRGA'19] M. Yin, D. Malkhi, M.K. Reiter, G. Golan-Gueta, I. Abraham. HotStuff: BFT Consensus with Linearity and Responsiveness. PODC 2019.

End