Lecture 9: The Permissionless Setting and Proof-of-Work

(Lecture Series on Foundations of Blockchains)

Permissionless Consensus

Permissioned setting: all nodes known in advance. [PKI: public keys also common knowledge] ("trusted setup")

[aka "private blockchain" or "proof of authority (PoA) blockchain"]

Permissionless setting:
- unknown, ever-changing node set
- clients submit transactions to nodes via broadcast
- honest nodes communicate via broadcast/gossip

Idea: extend permissioned protocols to permissionless setting?

Issue for BFT-type consensus: what's a supermajority? (don't know n)

Issue for longest-chain consensus: how to choose leader sequence? (can't do round-robin or uniformly random selection)

Random Sampling and Sybil Attacks

Solution: (for both BFT-type & longest-chain) random sampling.

- BFT-type: choose a random "committee" of nodes from set of all nodes
- longest-chain: each round, choose a leader at random

Issue: "Sybil" attacks. (a single node can costlessly use large # of distinct IDs)

=> need some sybil-proof way of randomly selecting a node

↳ i.e., probability of a node's selection independent of how many distinct IDs it uses

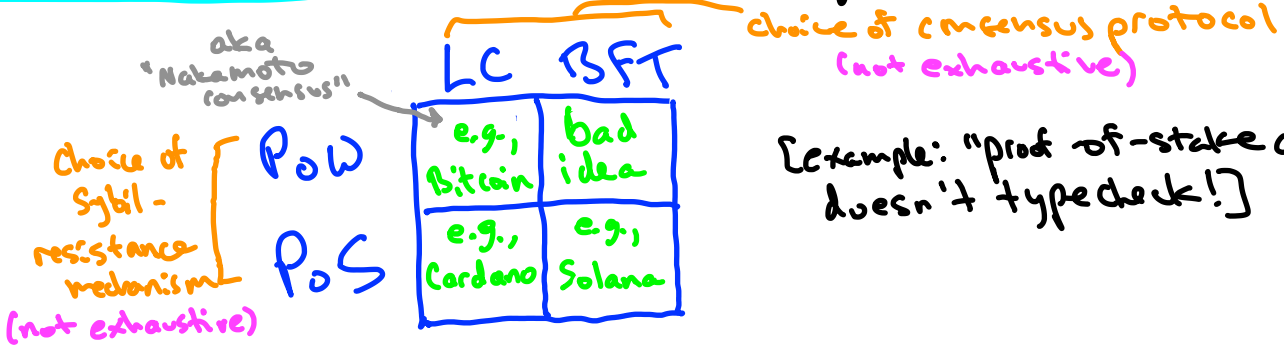
Two solutions:

1. proof-of-work (this lecture). => works best with longest-chain consensus
2. proof-of-stake (Lec #12). => increasingly coupled with BFT-type consensus rather than longest-chain

Consensus vs. Sybil-Resistance

Common point of confusion: consensus protocol ≠ sybil-resistance mechanism.

- Consensus algorithm/protocol: among blocks proposed thus far, decide which ones get committed to honest nodes' local histories (& in what order)
- Sybil-resistance mechanism: decide which nodes get to propose/vote on blocks



[example: "proof-of-stake consensus" doesn't typecheck!]

Proof-of-Work

Upshot: random sampling procedure $r \rightarrow$ [leader selection] $\rightarrow i$ s.t. (under appropriate assumptions) probability of a Byzantine leader is $\leq \alpha < 1/2$.

Caveat: need to re-check assumptions (A1)-(A5) from lec #8

$\Rightarrow$ implicit finality/liveness/chain quality guarantees from lec #8

Ideal#1: next leader = first node to successfully solve a hard puzzle.

proposed by [D work + Naor '92] for spam-fighting!

Assume: existence of cryptographic hash functions (e.g., SHA-256).

Random oracle (RO) assumption: CHF h behaves like a random function, with each $h(x)$ independent + uniformly distributed.

input x → h → 256 bits $h(x)$

Hard puzzle: for a given threshold τ, find x s.t. $h(x) \leq \tau$.

tunable difficulty parameter (smaller τ $\Rightarrow$ harder puzzle)

- e.g., if h = SHA-256, $\tau = 2^{176}$, looking for x s.t. $h(x)$ begins with ≥ 80 zeroes

chosen to target a fixed rate of block production (e.g., 1/10 min in Bitcoin)

What Should Puzzle Solutions Encode?

Hard puzzle: for a given threshold T, find x s.t. $h(x) \leq T$.

Role of random oracle assumption: only way to search for a puzzle solution x is by repeated guessing (with each guess equally likely to be a solution).

- Example: $h = \text{SHA-256}$, $T = 2^{176}$ $\Rightarrow$ expect to need 2^{80} guesses (!) before finding a puzzle solution

Question: which x's are allowed?

Idea #2: x encodes node's choices in step (2b). [Constraint: exactly one block proposal, exactly one predecessor]

I.e., for x to qualify as puzzle solution, in addition to satisfying $h(x) \leq T$, must have the form: $x = B \| pred \| pk \| nonce$ → "number used once" (remaining bits, degrees of freedom to find x s.t. $h(x) \leq T$)

Note: finding one puzzle solution no help in finding any new ones. $\Rightarrow$ Byzantine nodes can't propose >1 block (or pred) in any given round

Proof-of-Work: Sybil-Resistance

e.g., SHA-256. Under RO, indistinguishable from a random function

Proof-of-work: given threshold T, find x s.t. $h(x) \leq T$. [x has form $B||pred||pk||nonce$]

Assume: every node searches for a solution via repeated guessing. [justified by R.O.]

Key property: Suppose nodes $1, 2, \ldots, n$ make $\mu_1, \mu_2, \ldots, \mu_n$ guesses per second. ("hashrates")

$\Rightarrow$ For each i, $\Pr[\text{node } i \text{ is leader of next round}] = \frac{\mu_i}{\sum_{j=1}^{n} \mu_j}$ (independent of # of pks node i uses)

Proof: by Bayes rule,

$$\Pr[x \text{ proposed by node } i \mid x \text{ a puzzle solution}]$$

(same thing)

$$= \frac{\Pr[x \text{ proposed by } i] \cdot \Pr[x \text{ a puzzle soln} \mid \text{proposed by } i]}{\Pr[x \text{ a puzzle soln}]}$$

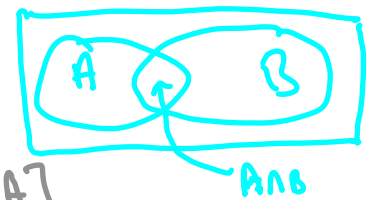
$$= \frac{\mu_i}{\sum_j \mu_j}$$

QED!

Corollary 1: PoW is sybil proof. [elect. prob. depends only on hashrates, not # of IDs]

Bayes rule

$\Pr[A \cap B]$



$= \Pr[A]\Pr[B|A]$

also $= \Pr[B]\Pr[A|B]$

$\Rightarrow \Pr[A|B] = \frac{\Pr[A]\Pr[B|A]}{\Pr[B]}$

Further Properties of Proof-of-Work

Corollary #2: if α fraction of total hashrate is Byzantine, $\Pr[\text{next leader Byzantine}] = \alpha$.

Upshot: if $<50\%$ of hashrate is Byzantine + assumptions (A1)–(A5) hold,
PoW longest-chain consensus ("Nakamoto consensus") satisfies final/liveness/chain quality (with high probability)

Question: do (A1)–(A5) hold?

(A1): nothing to check (trusted setup assumption) [can include unknowable-in-advance data in genesis block as evidence for (A1)]

(A2): node is the leader $\Leftrightarrow$ has a valid x with $h(x) \le T$ (which is trivial to verify)

(A3): (for fixed hashrates $\mu_1, \mu_2, \ldots, \mu_n$) follow from key property/sybil-resistance

(A4): holds because round-r leader must specify blk + predecessor at time of creation (1 block/round)

(A5): still need to relate (see 5th video of Lecture #9) – all good provided Δ is small relative to average length of a round

Note: does not require PKI ("truly permissionless") nor a shared global clock.

Difficulty Adjustment

Hard puzzle: given a threshold T, find x s.t. $h(x) \leq T$. [h = cryptographic hash function, acts like a random function]

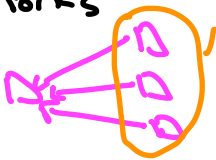
Question: how to set T?

Trade-offs: (i) bigger T => faster rate of block production (Sounds good for both latency + throughput)

(ii) bigger T => more frequent inadvertant forks

(small T)

vs.



(large T)

at least 2 of 3 will be wasted

[to quantify, need to allow $\Delta > 0$ => see next video for details]

[tl;dr - all good provided T set s.t. avg time between solns >> Δ]

Answer: Set T to achieve target rate of block production.

Case study: in Bitcoin, target rate = 1 block/10 minutes. (=> 144 blocks/day)

=> automatically resets T every 2016 blocks (target # of blocks over 14 days)

=> if took $\beta \cdot$ (14 days) to add last 2016 blocks to the longest chain, reset $T := T \cdot \beta$

(ignoring details about timestamp accuracy)

Work-Adjusted Longest-Chain

Issue: with varying puzzle difficulty, can't use vanilla longest-chain rule.

(adversary w/ modest hashrate might be able to quickly generate a long chain of blocks with very easy puzzle difficulty ⇒ could violate finality)

Fix: redefine chain length as total amount of hashrate that went into its construction.

- define work of block B = expected # of guesses to produce a solution at B's difficulty level (e.g., for 256-bit hash outputs, = $2^{256}/\text{target}$)
- work of a chain = sum of work of each block in chain

⇒ honest nodes instructed to extend chain with largest amount of work

Fact: finality, liveness, etc. hold (w.h.p.) provided:

(i) at any given time, < 50% of hashrate is Byzantine

(ii) total hashrate only changes at bounded rate over time

(iii) security parameter k chosen appropriately

Formal analysis: [Garay/Kiayias/Leonardos Cayetano '17]

Extensions to the Synchronous Model

Question: what role(s) did assumption (A5) play in our Lecture #8 analysis?

(1) [for liveness + chain quality] every honest leader increases the length of the longest chain → implies length of longest chain ≥ # honest blocks thus far

(2) [for common prefix property] no two honest blocks at the same height

(3) [for consistency] renders it trivial via instant communication

[assume fixed node hash rates for simplicity]

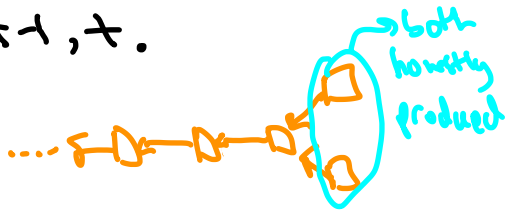
Two key parameters: Δ = max message delay

L = avg amount of time between consecutive puzzle solns

equivalently, $\Pr[\text{puzzle soln at time } t] = 1/L$

Definition: block produced by honest node at time t is safe if no other puzzle solns were found in time steps $t-\Delta+1, t-\Delta+2, \ldots, t-1, t$.

Key point: properties (1) + (2) hold for safe blocks.



Consistency and Liveness (Reprise)

Lemma: expected fraction of blocks that are safe is $\geq (1-\alpha)(1-\frac{\Delta}{L})$.

Proof: $\Pr[\text{a puzzle solution leads to a safe block}] = \Pr[\text{solution found by honest node}] \cdot \Pr[\text{no other puzzle solns in steps } t-\Delta+1, \ldots, t]$

$= \text{fraction of hashrate that is honest} = 1-\alpha$

technically, by Union Bound

$\leq \frac{1}{L}$ per time step $\Rightarrow \leq \frac{\Delta}{L}$ per window of Δ time steps $\Rightarrow$ prob. of is $\geq 1 - \Delta/L$

QED.

Note: unsafe honest blocks no worse than blocks created by Byzantine nodes.

$\Rightarrow$ Lecture #8 analysis carries over provided safe blocks > Byzantine blocks + unsafe blocks

$\Rightarrow$ true if $(1-\alpha)(1-\frac{\Delta}{L}) > \alpha + (1-\alpha)\frac{\Delta}{L}$

$\Leftrightarrow \alpha < \frac{1-2\frac{\Delta}{L}}{2-2\frac{\Delta}{L}}$

Upshot: Nakamoto consensus (PoW + LC) satisfies consistency + liveness in good/synchronous setting with $\leq 49\%$ Byzantine hashrate, provided Δ small relative to L.

PoW Can't Be Consistent Under Attack

Question #1: ever sensible to combine PoW sybil-resistance + BFT-type consensus?

Question #2: can we modify longest-chain consensus to be consistent in partially synchronous setting?

Theorem: [Lewis-Pye + TR '20] No PoW blockchain satisfies both:

1. liveness in the synchronous setting (even with $\Delta = 0$)
2. consistency in the partially synchronous setting.

=> PoW + longest-chain consensus chooses 1, gives up on 2

=> PoW + BFT-type consensus chooses 2, gives up on 1

- even assuming all nodes are honest!

→ would seem to be a deal breaker

Idea: can't distinguish waning hashrate vs. delayed messages:

Suppose node i stops hearing new blocks from other nodes. Maybe:

(i) other nodes turned off their machines

(ii) other nodes' messages all delayed

Catch-22

- if node i finalizes new blocks => it in scenario (ii), violates 2
- if not => it in scenario (i), violates 1

QED!

Where We're Going

Lectures #10 - 13: adding a native (crypto) currency to a blockchain.

- interesting in its own right
- can incentivize nodes to run protocol (e.g. block reward)
  - worry: are nodes too incentivized (=> selfish mining, see lec #10)
- can charge for usage
  - need to design a transaction fee mechanism (see lec #11)
- enables alternative approaches to sybil-resistance
  - proof-of-stake blockchains (see lec #12)
- can help secure blockchain
  - "economic security" (≈ cost of 51% attack, see lec #13)