# Nakamoto Consensus

Zeta Avarikioti
georgia.avarikioti@tuwien.ac.at

November 19th, 2025

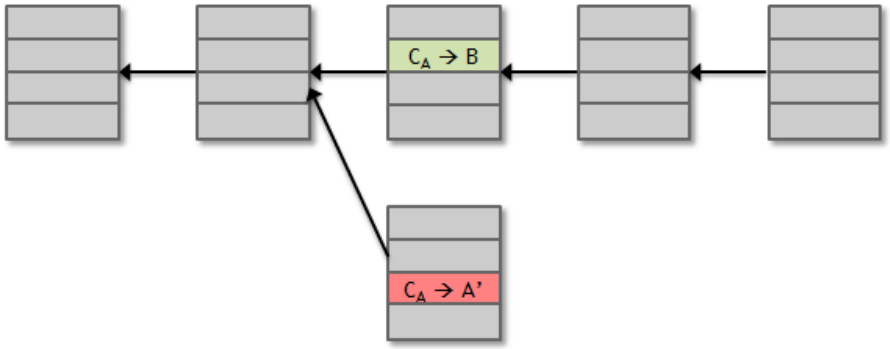
TU WIEN

S&P Security & Privacy

# Overview

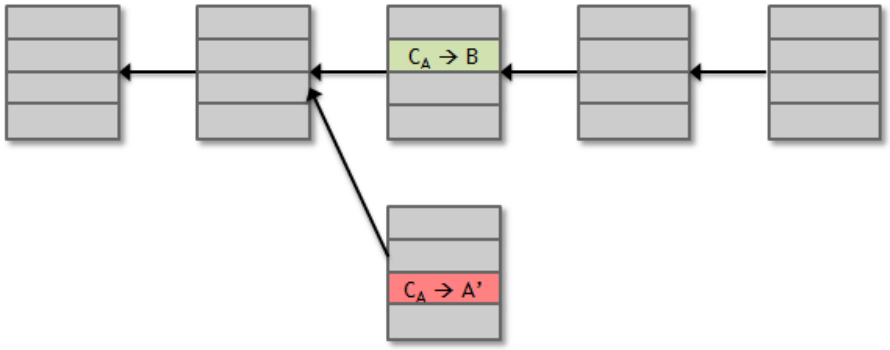
- Nakamoto consensus recap
- Key observations for Bitcoin (assumptions, efficiency, decentralization, etc)
- Security proof (simplified)
- Security under network delays
- Selfish mining attack

# Nakamoto consensus recap




- Mine on longest chain, i.e., repeatedly hash(tx,nonce,previous_hash) until hit the target
- Upon having a longest chain, send to all (via peer-to-peer network)
- Commit blocks buried deep

# Why does it work?




- A unique longest chain keeps growing faster that all other chains
- Hence, once a block is buried deep, it is less likely to be "forked off"
- More rigorous proof will follow

# Key Observations

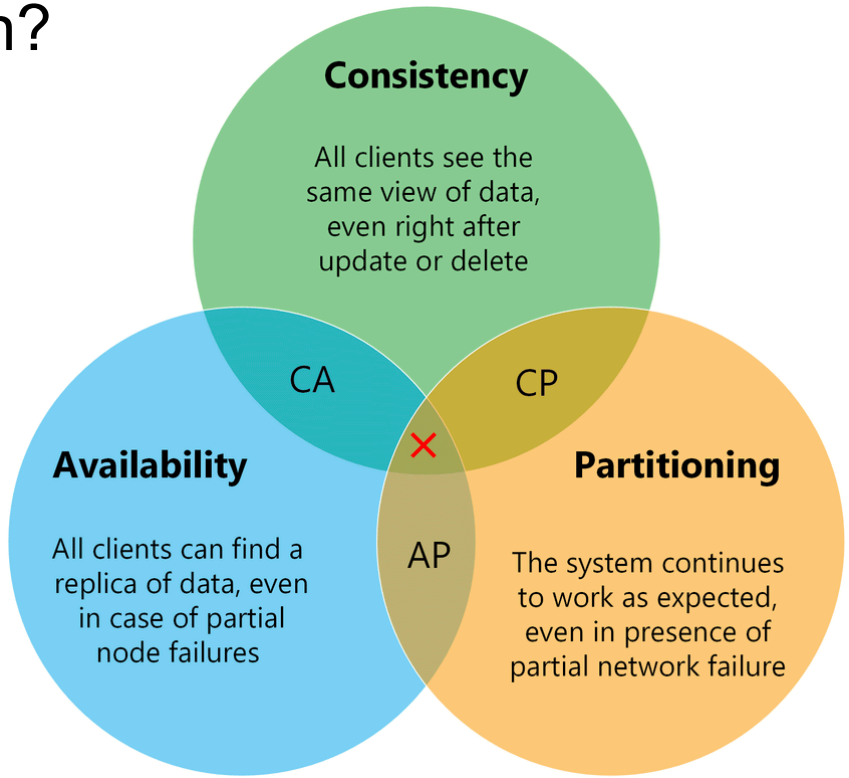
# Interesting questions

- Under which assumptions is Bitcoin secure?
  - Timing model
  - Signatures
- Does peer-to-peer help or hurt?
  - Latency
  - Communication
- What is the role of PoW?
- Is Bitcoin permissionless and decentralised?

# Model of Bitcoin

- Timing model
  - Secure under synchrony
  - Insecure under partial synchrony or asynchrony
  - Some model in between?

The CAP Theorem



# Model of Bitcoin

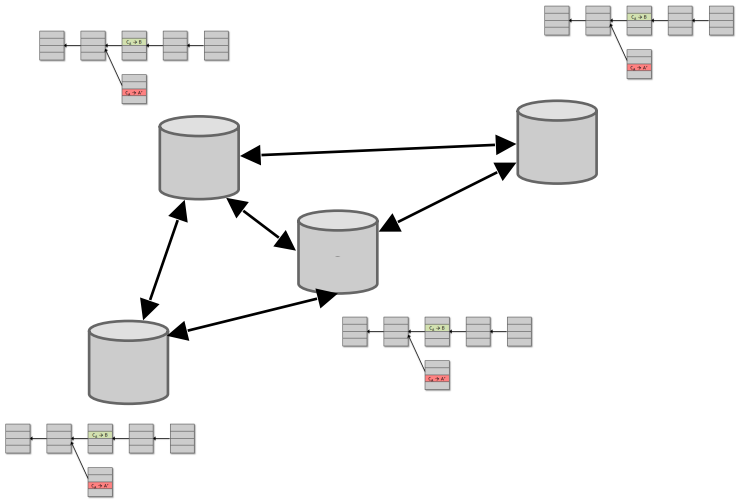
- Timing model
  - Secure under synchrony
  - Insecure under partial synchrony or asynchrony
  - Some model in between?

- Authentication
  - Signatures are not necessary for consensus
  - Yet overcome the 1/3 fault bound (FLM), why?

[PSL80,FLM85] If $f \geq n/3$, no deterministic protocol for Byzantine broadcast satisfies both agreement and validity in the synchronous model.

# Efficiency of Bitcoin

- Very high latency
  - For exp(-k) error probability, wait for k blocks
  - Blocks are slowly arriving
- Communication: O(ndB)
  - n: number of nodes
  - d: number of neighbours
  - B: block size (4MB)
  - Without peer-to-peer: $O(n^2B)$
- Enormous energy consumption

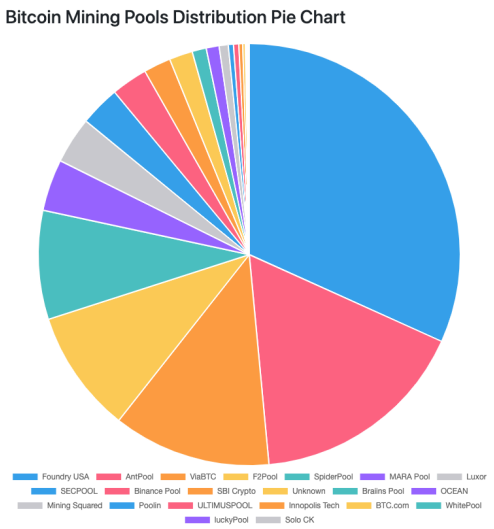


# Roles of PoW

- Sybil resilience
- Leader election
  - Unpredictable
  - Rate limiting
- Make equivocation hard
  - Somewhat resembles signatures

# Bitcoin: permissionless, decentralized?

- Permissionless
  - Dynamic network ✓
  - Anyone can join/no IDs, no permissions ✓




- Decentralized
  - Operated by many nodes ✓
  - Nodes are not controlled by only a few ✗

# Security Proof

(a simplified one)

# Objectives

- Security of Bitcoin
  - **Safety:** A block b is safe if, once it is confirmed it remains confirmed at the view of all honest nodes at all future times, irrespective of the adversary attack.
  - **Liveness:** A non-zero fraction of honest blocks are confirmed.

Safety is:
1) Consistency among honest nodes at any time
2) Self-consistency of each honest node across time

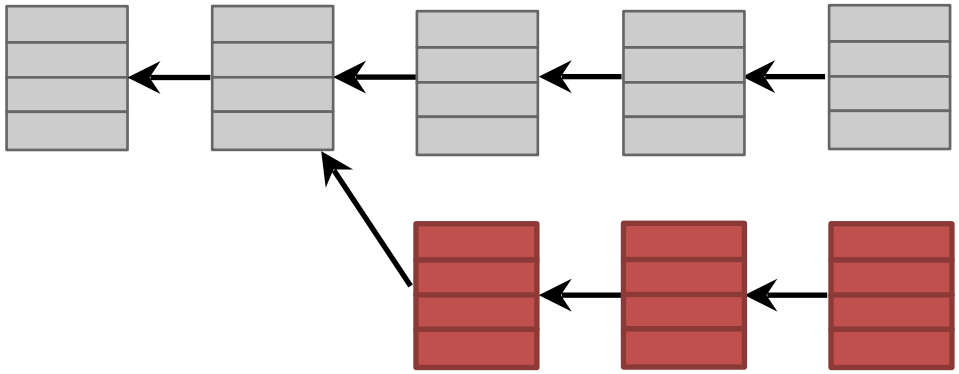
# General Assumptions

- (A1) No node has knowledge of the genesis block prior to the deployment of the protocol (trusted setup).
- (A2) It is easy for all nodes to verify whether a given node is the leader of a given round.
- (A3) No node can influence the probability with which it is selected as the leader of a round.
- (A4) Every block must claim as its predecessor some block that belongs to a previous round.
  (Only PoW) Each leader produces exactly one block per round.
- (A5) At all times, all honest nodes know about the exact same set of blocks and predecessors (telepathy).

# Adversarial Deviations

- Empty blocks
- Delayed block announcements
- Deliberate forking

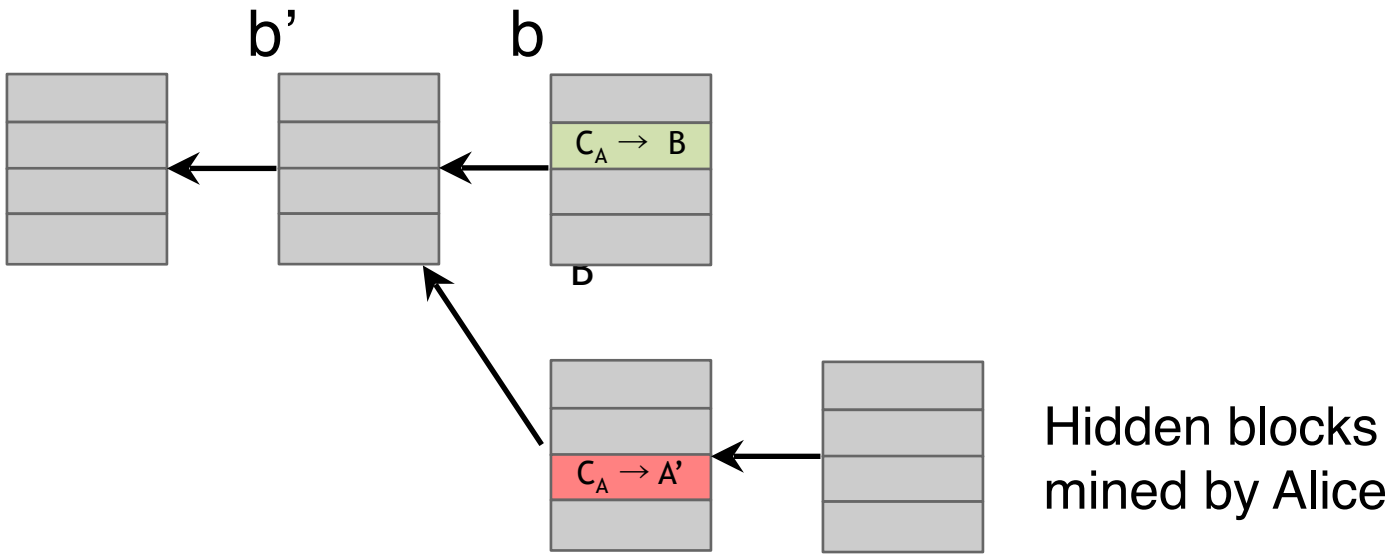
- Combination of the above!
  - e.g. Private double-spent attack



# Safety of Nakamoto

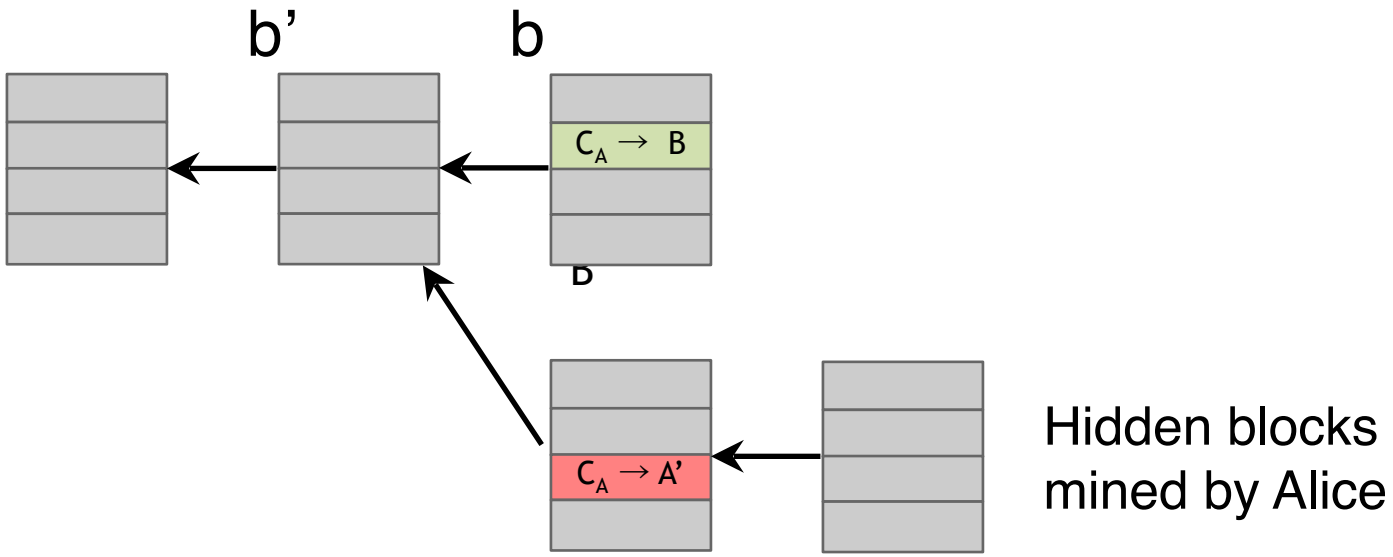
under 0 network delay

# k-deep confirmation rule




- Suppose 1-deep confirmation rule:
  - Alice pays Bob in block b (tx)
  - Alice starts immediately mining in private on top of b' (tx')
  - Bob sends car to Alice
  - Alice reveals her private chain
  - Block b is pushed out of the longest chain

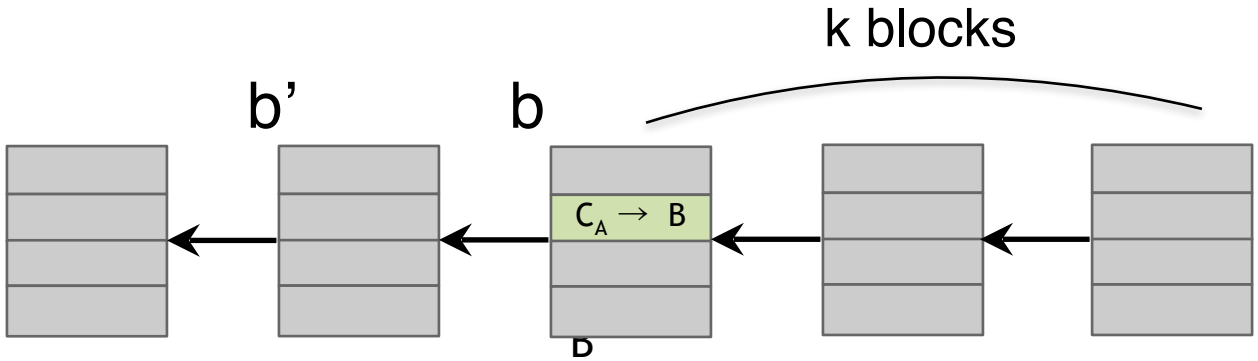
# k-deep confirmation rule




- What if miners keep building on block b?
  - Healability after network partition
  - Safety under network delays
  - Safety for bootstrapping new nodes

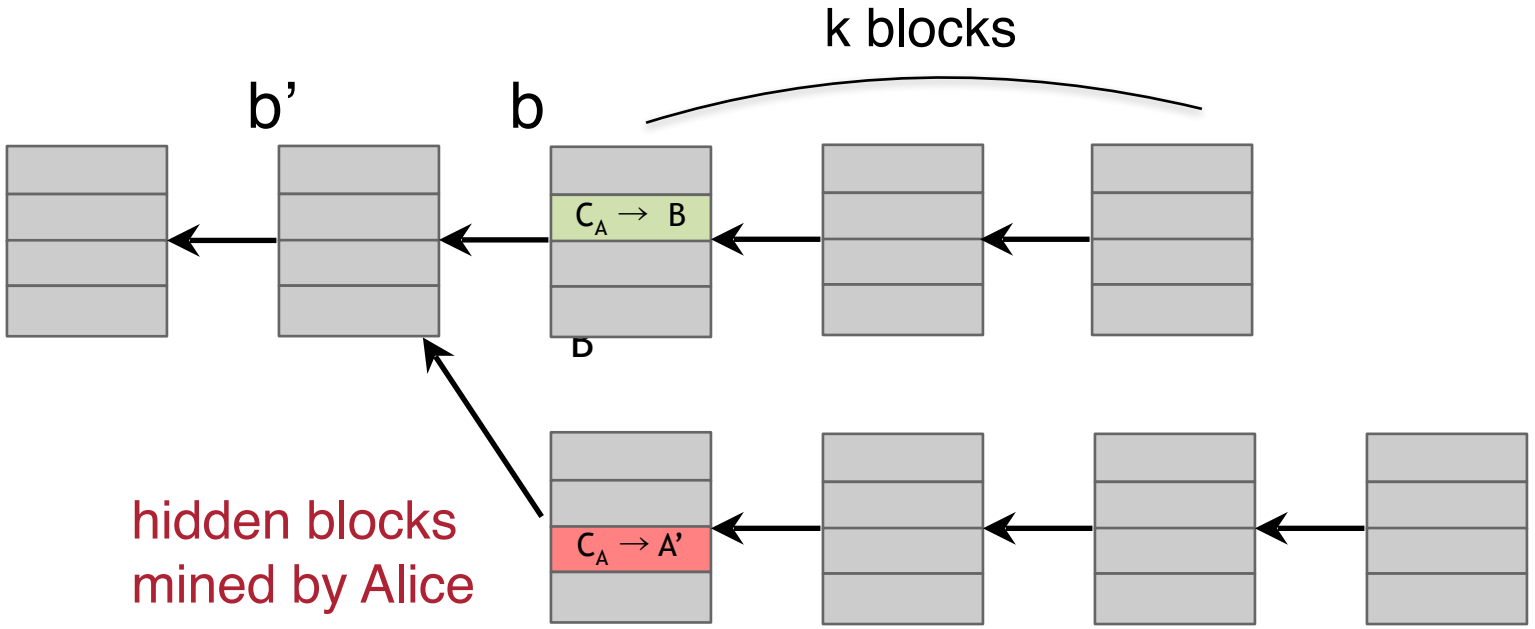
- The attack is feasible even from weak attackers → It would erode people's trust in the system!

# k-deep confirmation rule




- k-deep confirmation rule:
  - Alice pays Bob in block b
  - k-1 blocks are build on top of block b
  - Bob sends car to Alice

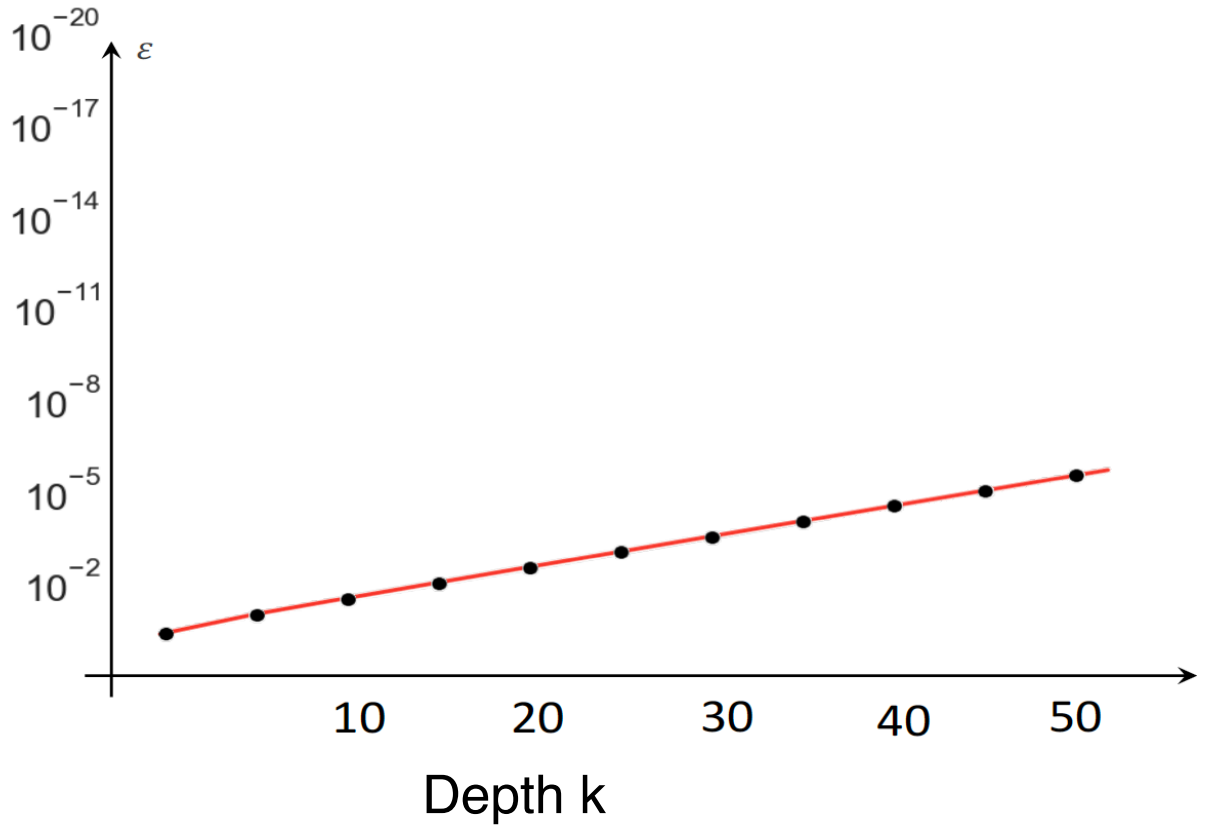
# k-deep confirmation rule




- Why k-deep confirmation rule works?
  - The **probability of tx being pushed out** of the longest chain **decays exponentially as k grows**!
  - The bound holds as long as the adversary holds strictly less than half of the total mining power, $\beta < 1/2$;
    Else, the adversary can always override the honest chain.

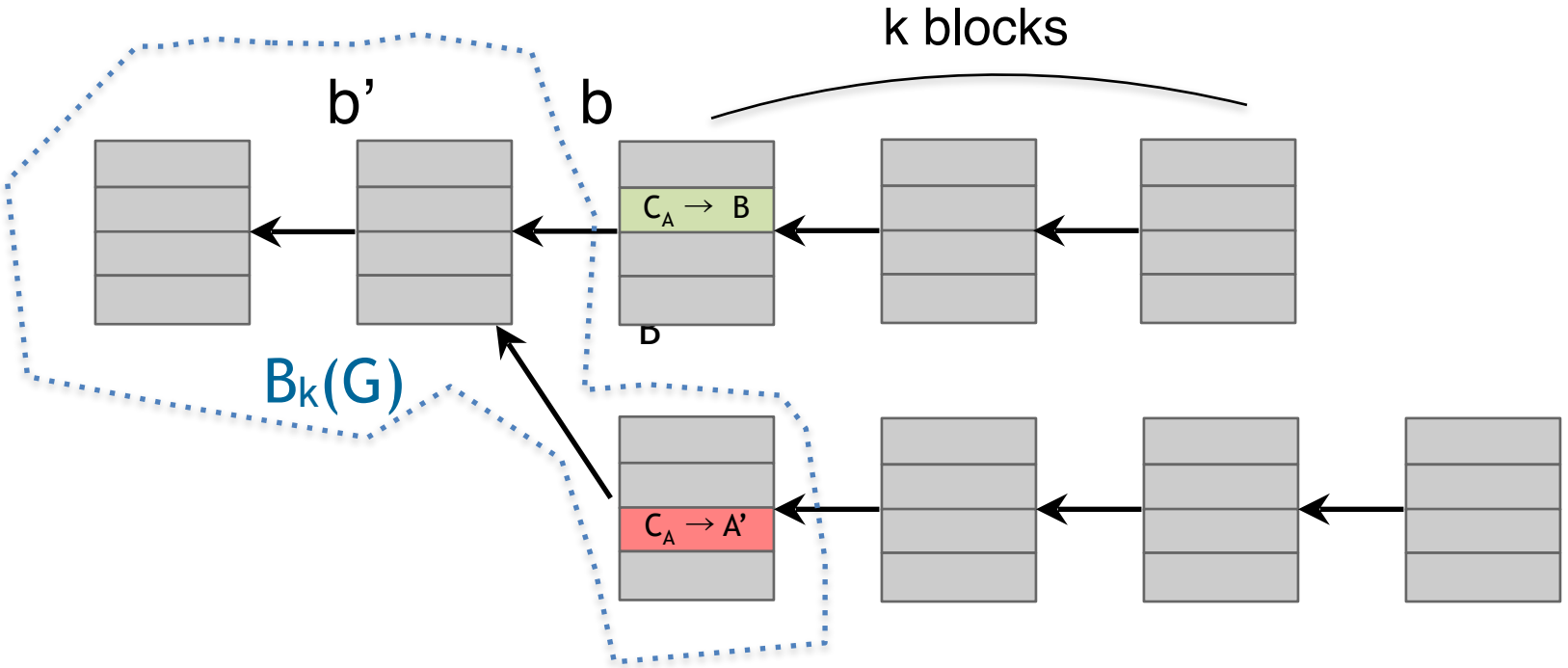
# Confirmation error probability

Confirmation error prob.




Log-linear plot of confirmation error probability as a function of depth k for adversarial hash power 30%

# Definitions




- $B_k(G)$: the longest chain of in-tree G, with the last k blocks removed.

- w-balanced leader sequence: A sequence of leaders is w-balanced when every window of length at least w, honest leaders outnumber adversarial leaders.

# Sketch Proofs

## Methodology

- Random Leader Selection Implies Balanced Sequences (A3 critical)

- Balanced Leader Sequences Imply Security Properties
  - Common Prefix (A1, A4 critical)
  - Chain Quality
  - Chain Growth

# Sketch Proofs

## Random leaders imply balanced sequences

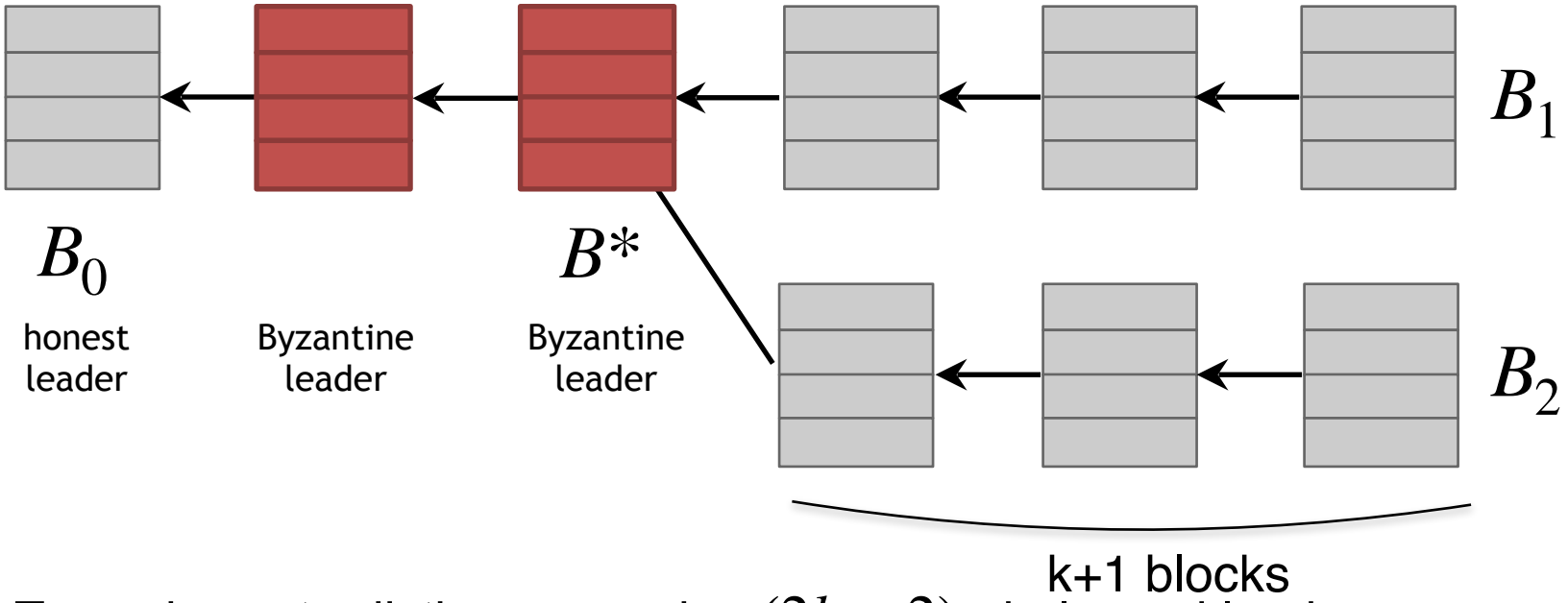
- PR1: Probability a window w is not w-balanced $\leq e^{cw}$
  - $c$ depends on how close is the adversary to 50%

- PR2: Probability any window of length w not being w-balanced is the sum of PR1 in all possible windows in a timeframe (union bound)
  - $\approx T^2$ many windows of at least length w in a timeframe $T$
  - PR2 $\leq T^2 e^{cw}$
  - Hence, the probability all windows are w-balanced is $\geq 1 - T^2 e^{cw}$

Intuition: Law of large numbers

# Sketch Proofs

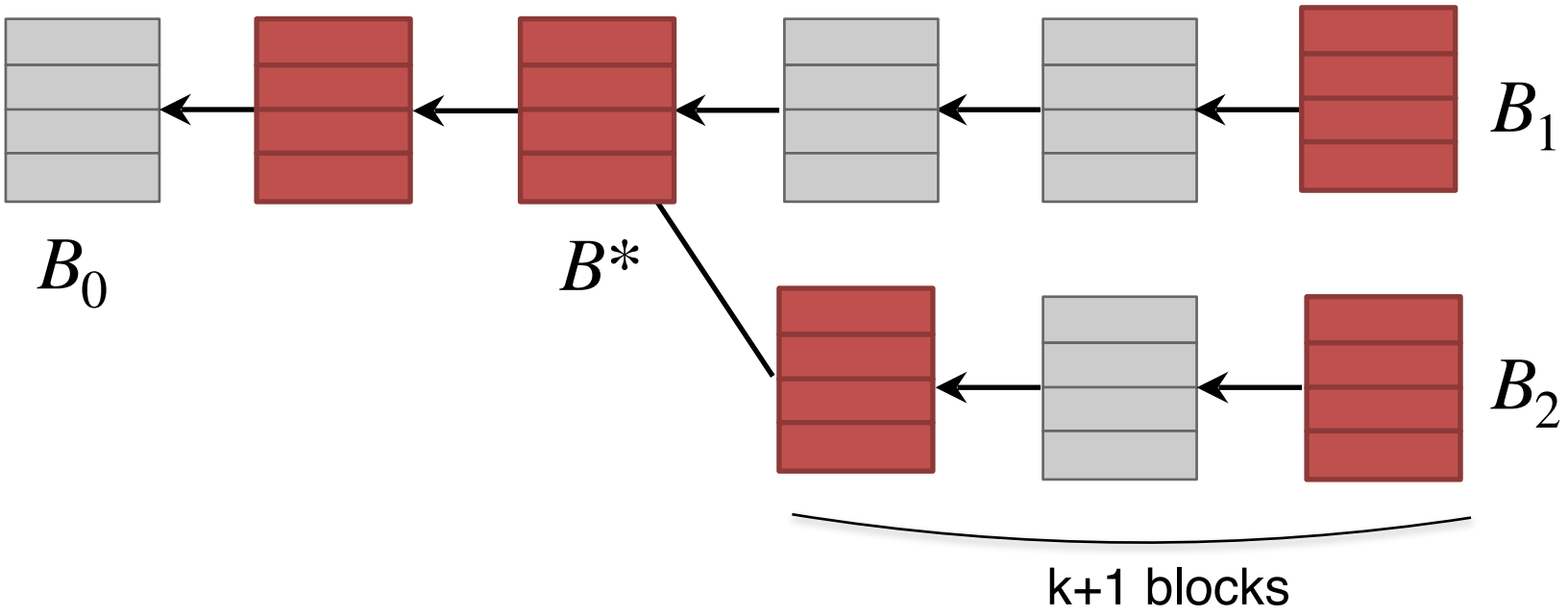
## Balanced leader sequences imply common prefix




- Towards contradiction, assuming $(2k+2)-$balanced leader sequence
  - There are 2 chains that differ for more than $k$ blocks.
  - Let $B_0$ the most recent ancestor of $B^*$ that was produced by an honest leader (it always exists due to A1)
  - We will count leaders and show that there are more Byzantine than honest.

# Sketch Proofs

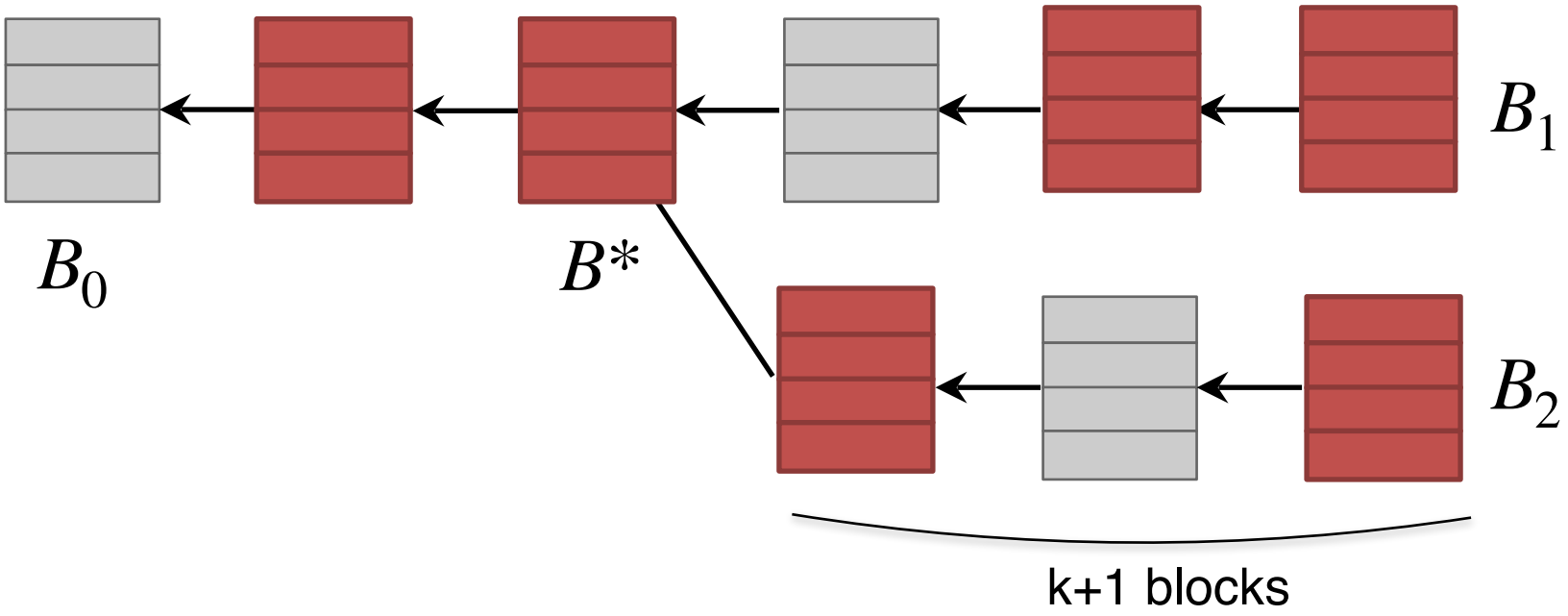
## Balanced leader sequences imply common prefix




- At the same height, we cannot have two honest blocks!
  - Synchrony (A5)
  - Honest follow the Nakamoto consensus (i.e., build on the longest chain)

# Sketch Proofs

## Balanced leader sequences imply common prefix

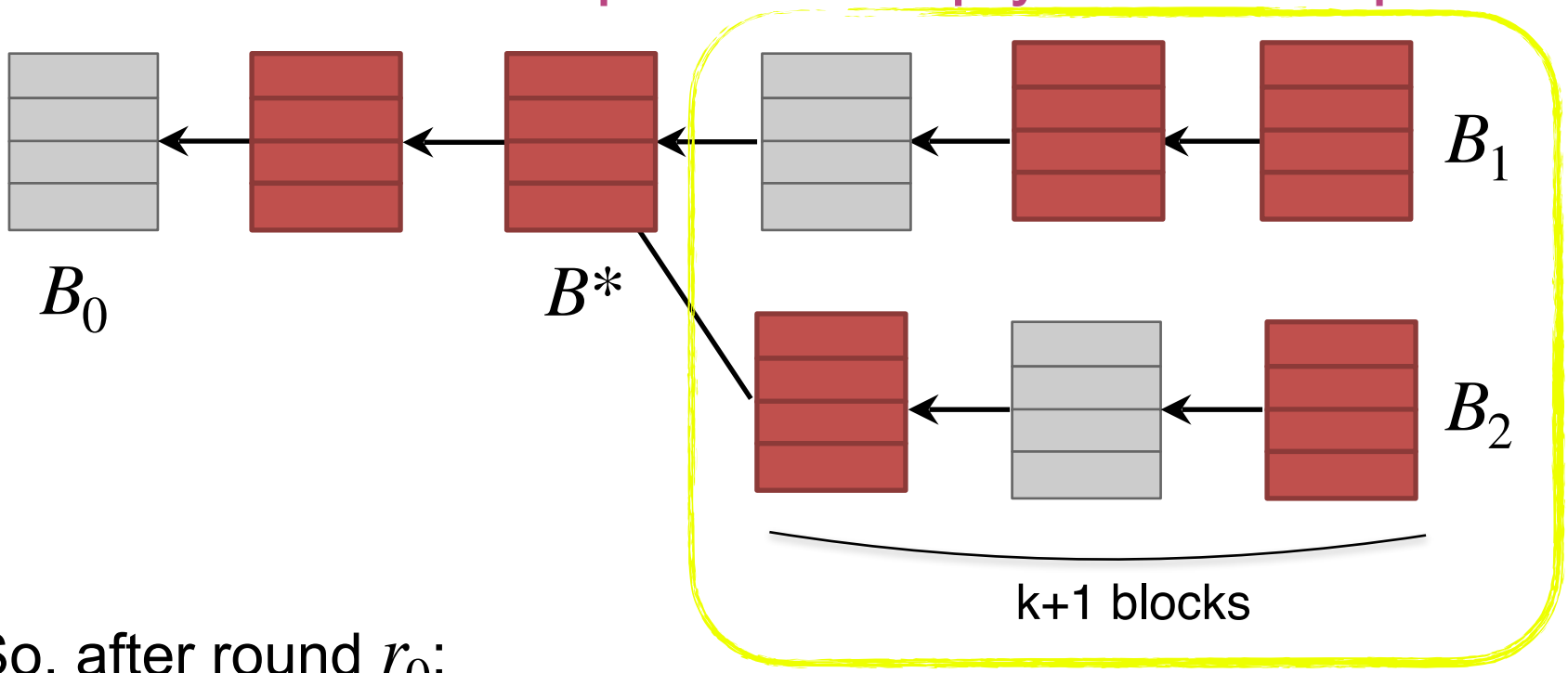



- Hence, after $B^*$, at the same height:
  - One honest block (H) and at least one adversarial (A) (cf. height 1,2)
  - No honest block and multiple adversarial (cf. height 3)

$$A \geq H$$

# Sketch Proofs

## Balanced leader sequences imply common prefix

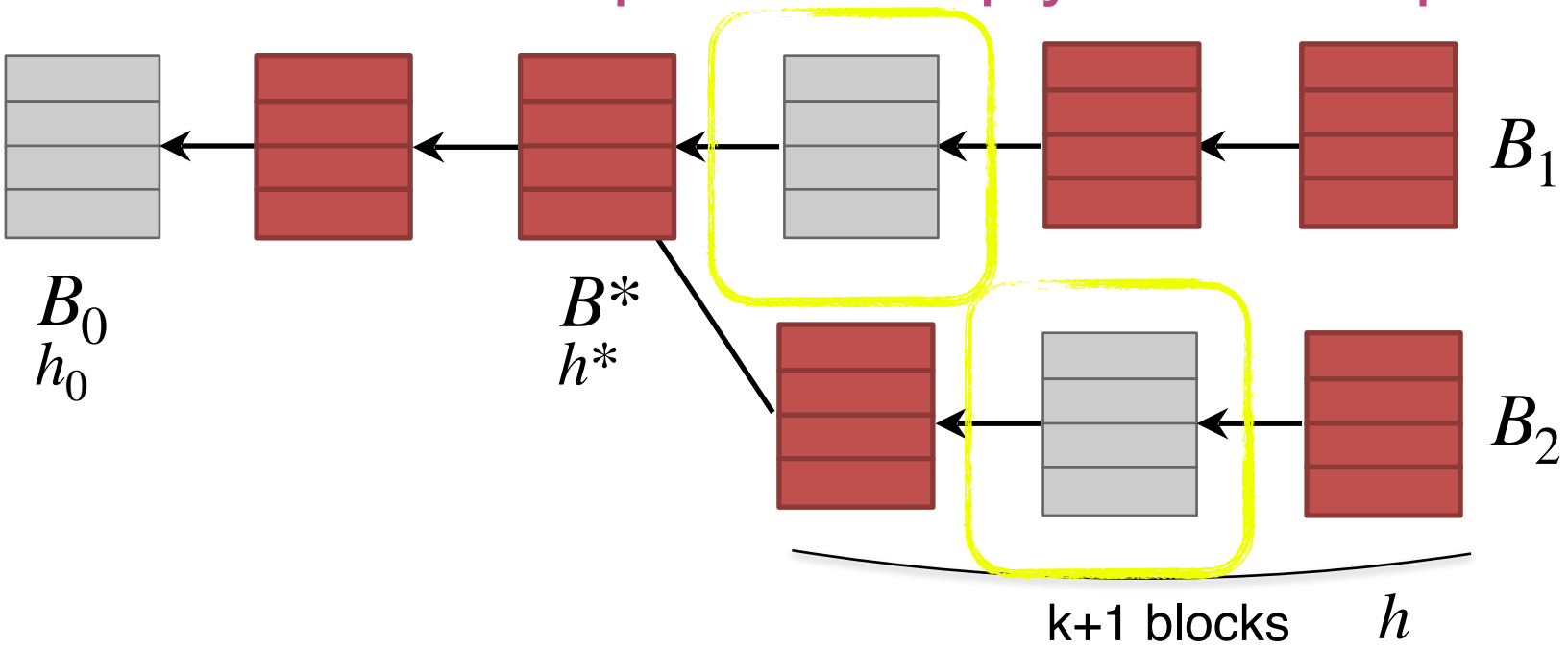



So, after round $r_0$:

1. There have been at least $2k + 2$ leaders

# Sketch Proofs

## Balanced leader sequences imply common prefix

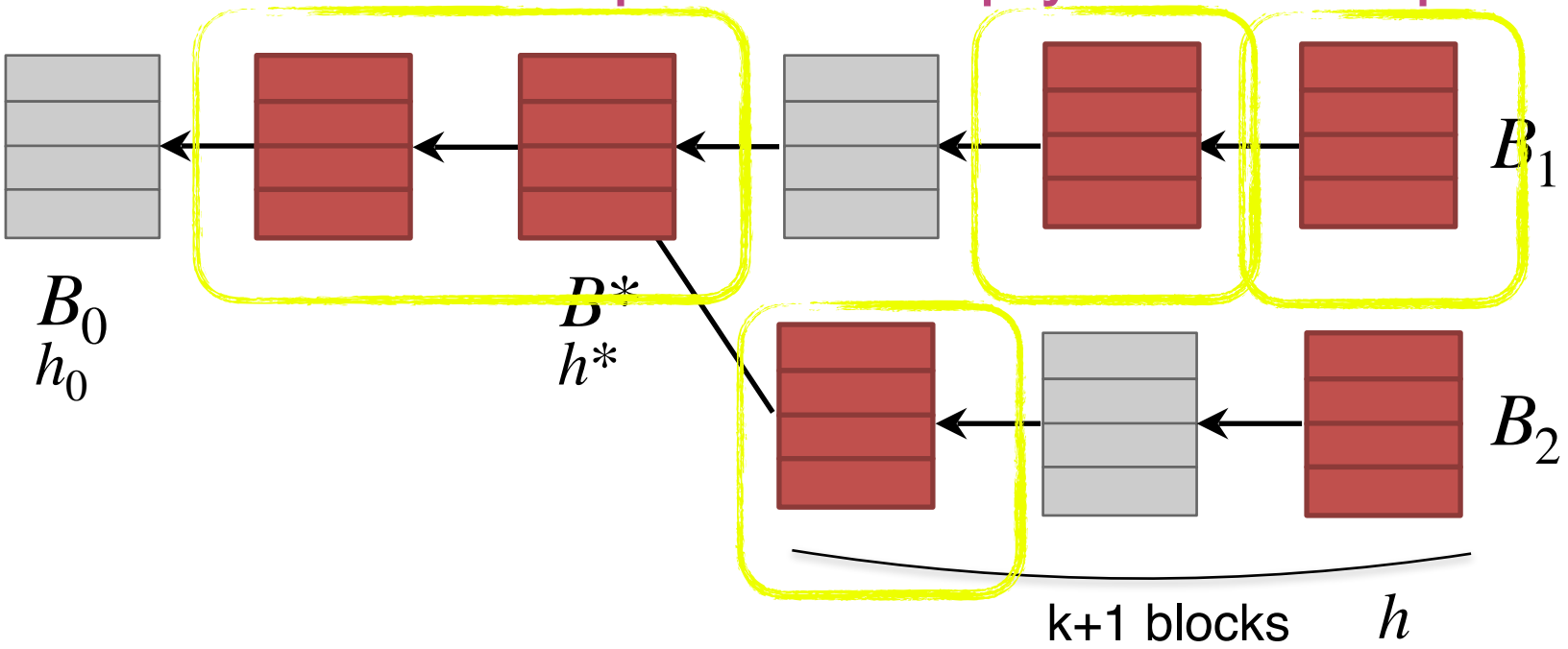



So, after round $r_0$:

1. There have been at least $2k + 2$ leaders
2. At most $h - h_0$ leaders could have been honest

# Sketch Proofs

## Balanced leader sequences imply common prefix

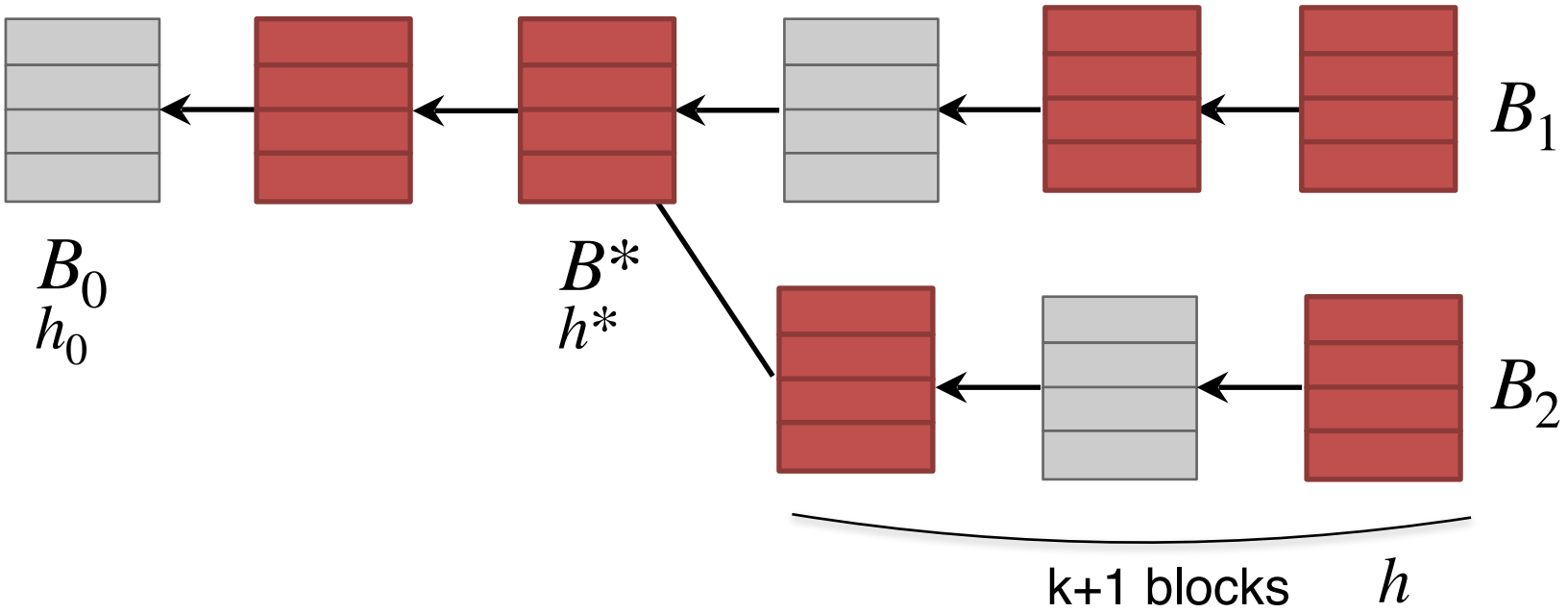



So, after round $r_0$:

1. There have been at least $2k + 2$ leaders
2. At most $h - h_0$ leaders could have been honest
3. At least $h - h_0$ leaders must have been Byzantine
   (all before $h^*$ & at least one per higher after $h^*$)

# Sketch Proofs

## Balanced leader sequences imply common prefix



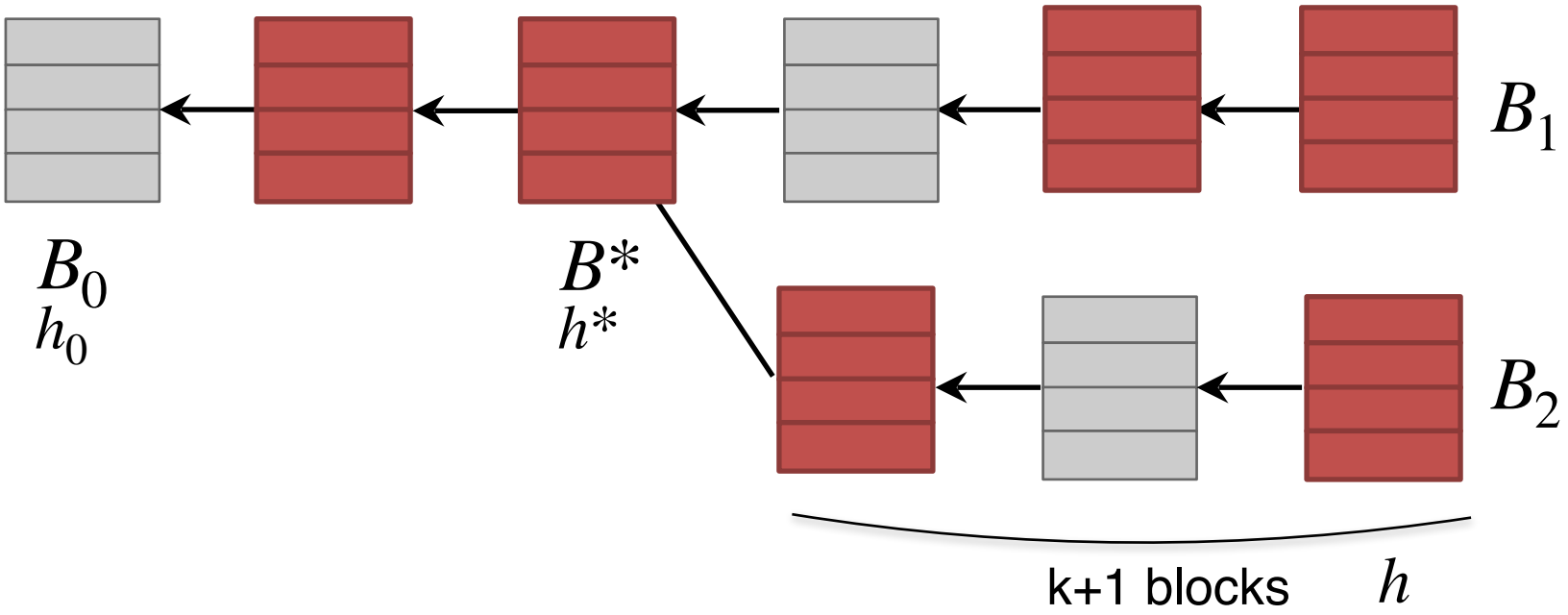

So, after round $r_0$:

1. There have been at least $2k + 2$ leaders
2. Byzantine leaders outnumber the honest leaders

**Not $(2k + 2)-$balanced leader sequence (contradiction).**

# Sketch Proofs

## Finality




Common prefix implies that if I crop the last k blocks on my local chain, the remaining **chain is final** (i.e., common among all honest nodes across time).

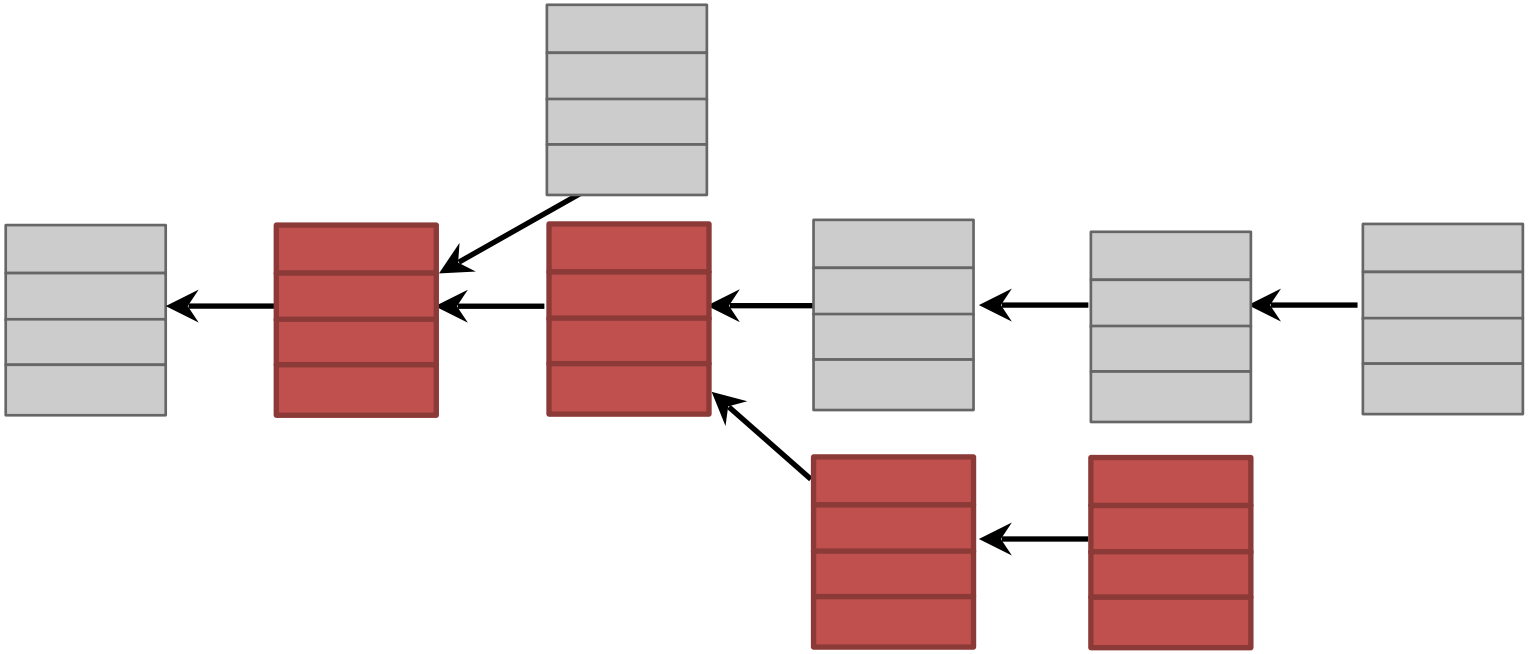
# Liveness

# Liveness

- Recall **liveness** definition: *A non-zero fraction of honest blocks are confirmed.*

- Observations:
  - Non honest blocks can be empty or junk
  - Non zero fraction guarantees that any honest transaction will be eventually included

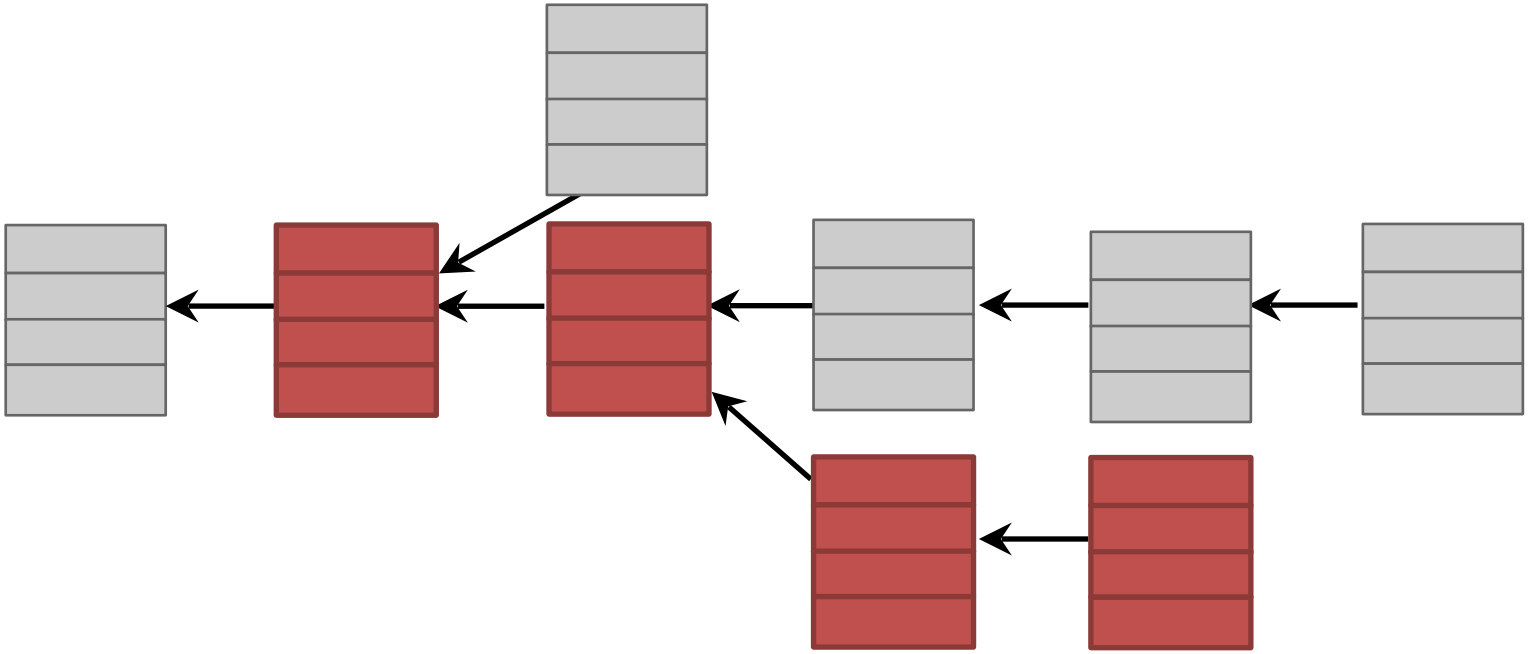
# Chain quality



**Chain quality ($CQ$):** The long term average fraction of honest blocks on the longest chain.

Example: $CQ = 4/6$

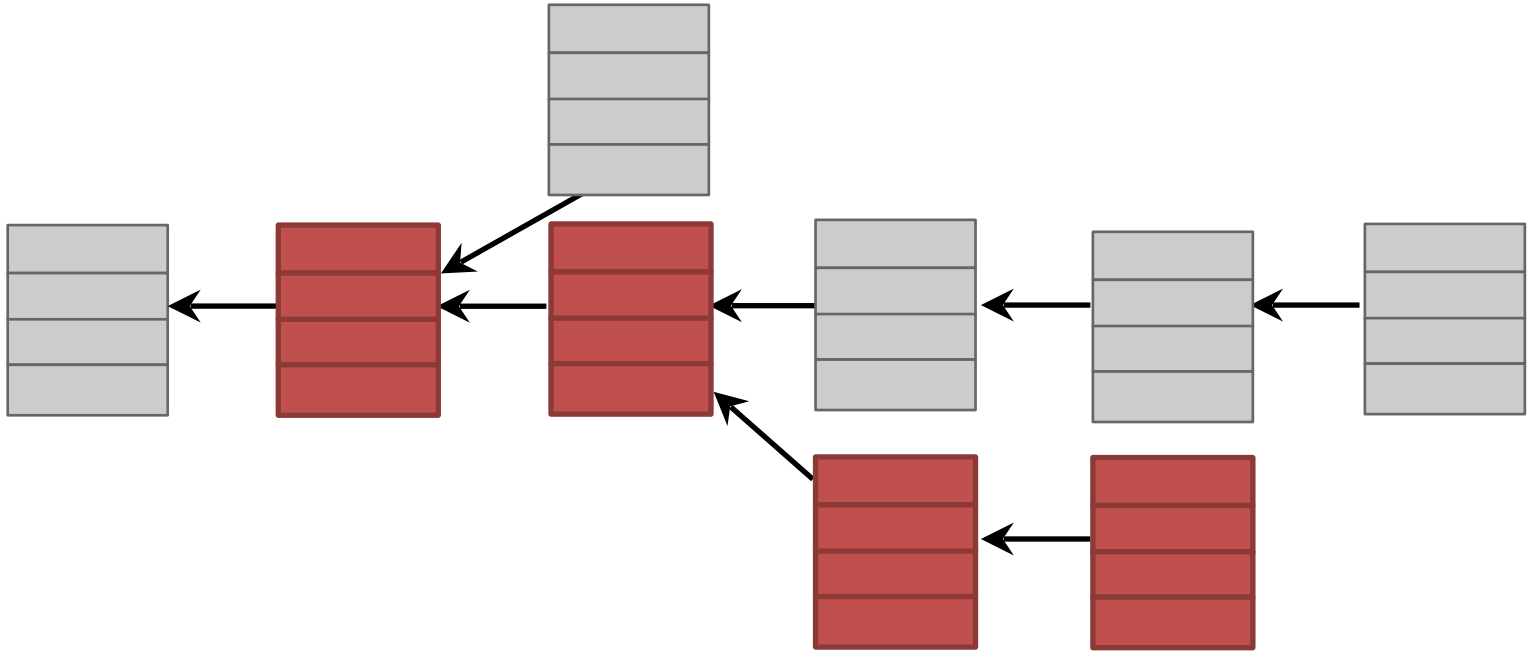
# Chain quality



**Chain quality ($CQ$):** The long term average fraction of honest blocks on the longest chain.

**Liveness:** $CQ > 0$

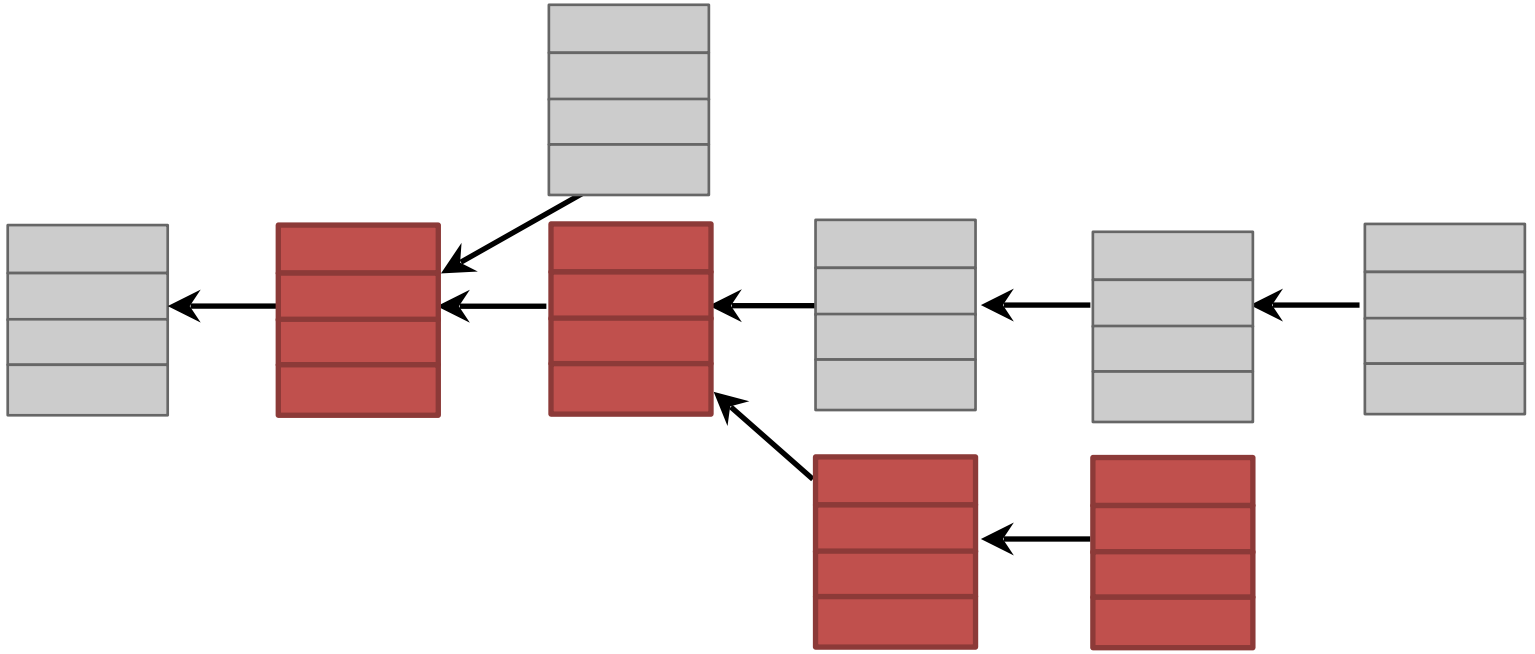
# Chain growth



**Chain growth ($g$):** The long term average growth rate of the longest chain (blocks/sec).

- Let $m_a, m_h$ be the mining rate of adversary and honest (blocks/sec)
- If the adversary inserts all his blocks in the longest chain, $g \geq m_a$

# Chain quality lower bound



- We have: $CQ \geq \min_{g} \frac{g - m_a}{g}$
  - $g$ can be manipulated by the adversary
  - If the adversary is following the protocol: $g = m_a + m_h$
  - The minimum growth though depends on the honest mining $g \geq m_h$

$$CQ \geq \min_{g \geq m_h} \frac{g - m_a}{g} = 1 - \frac{m_a}{m_h} \quad \Rightarrow \quad CQ > 0 \; if \; m_h > m_a$$

# Security conclusions

Bitcoin is live if $\alpha < 1/2$

Bitcoin is safe if $\alpha < 1/2$

Bitcoin is secure if $\alpha < 1/2$

Safety under…

bounded network delay

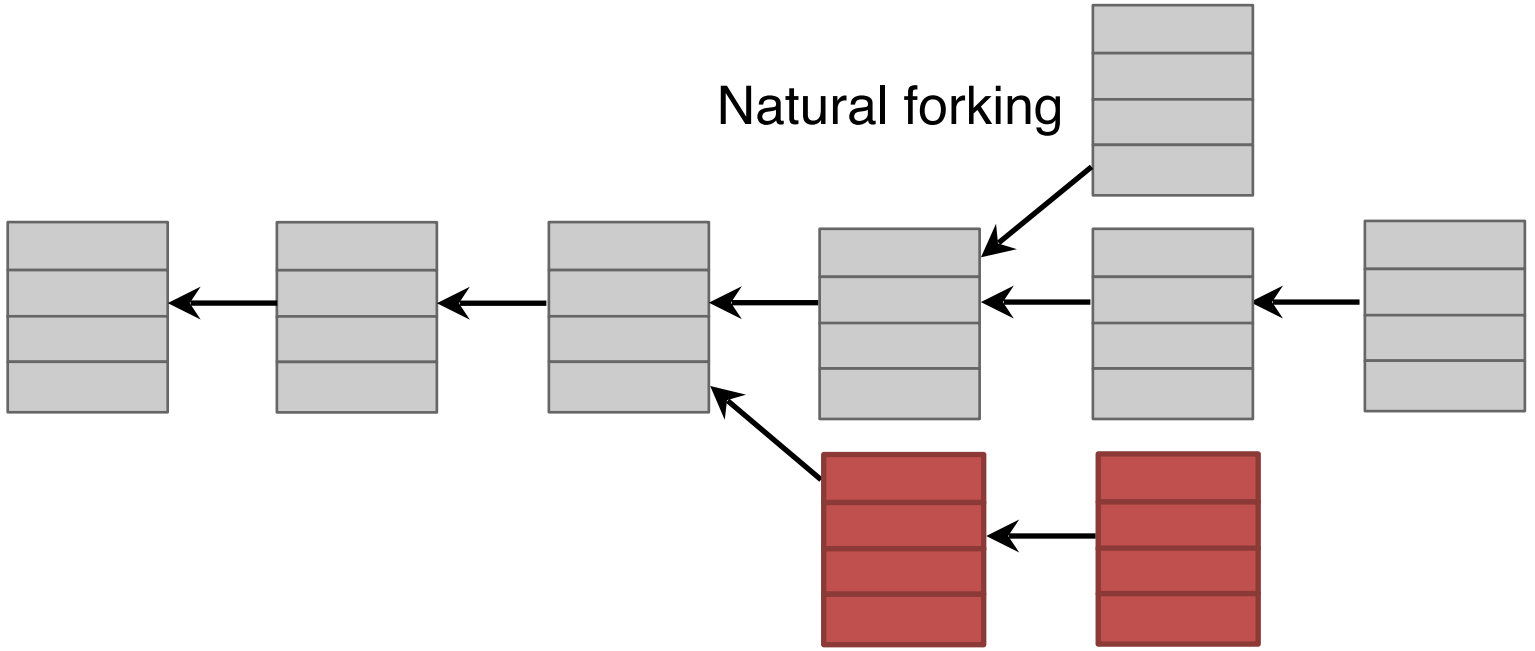
# Network delays

- Bitcoin block creation time: 10min
- Bitcoin propagation delay: ~12sec

- $\Delta$-synchronous network model: every node receives the data (block) within the bounded delay, denoted $\Delta$

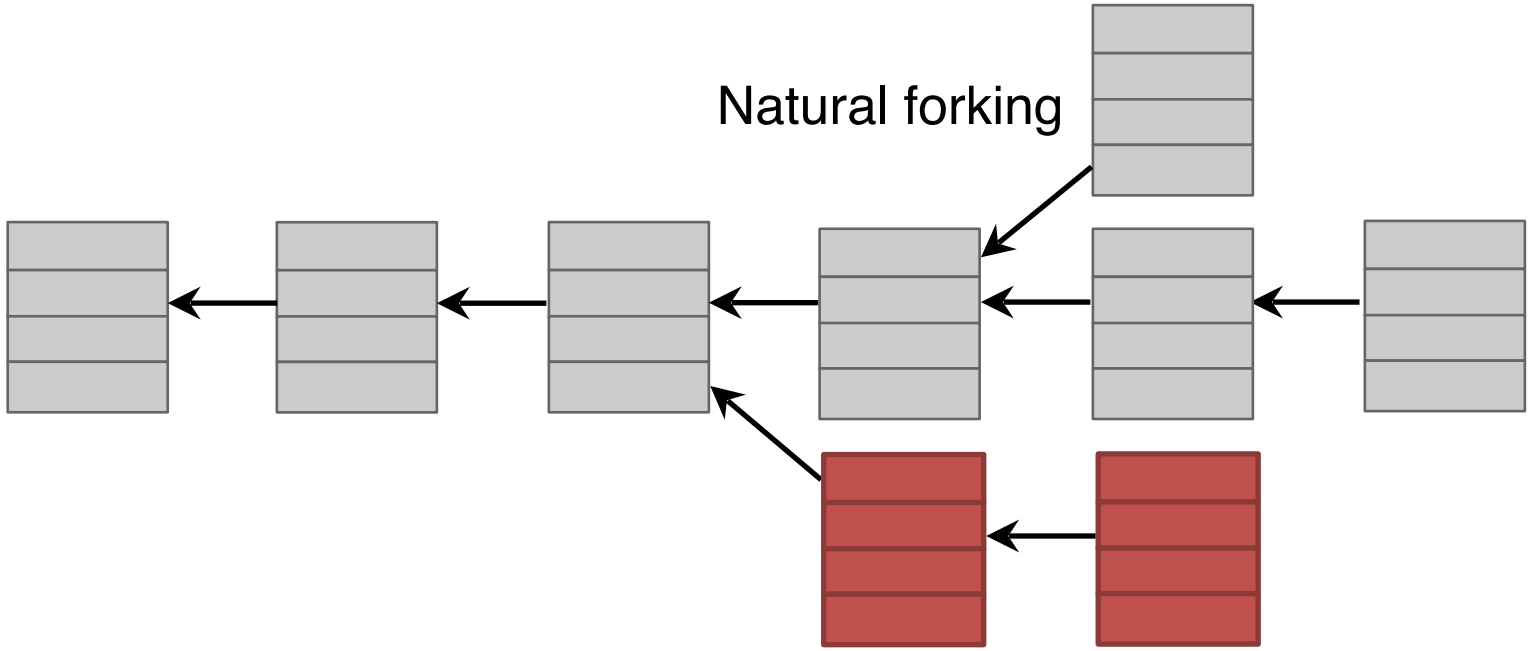
- The adversary can *reorder* blocks, always respecting the $\Delta$ bound

# Network delays




- Why do we care?
  - Honest nodes mine on *stale* blocks
  - **Natural forking** - without adversarial intervention
  - Waste of honest mining power!

# Network delays




- Ratio of block generation time and network delay is critical for security
- Bitcoin has low ratio (2%)
- Ethereum had 12sec block generation time, thus many *orphan* blocks (10-15%)

- Scalability: we cannot increase arbitrarily the block size or reduce the block generation time without compromising safety!

# Network delays

What fraction of honest mining power is wasted due to natural forking?

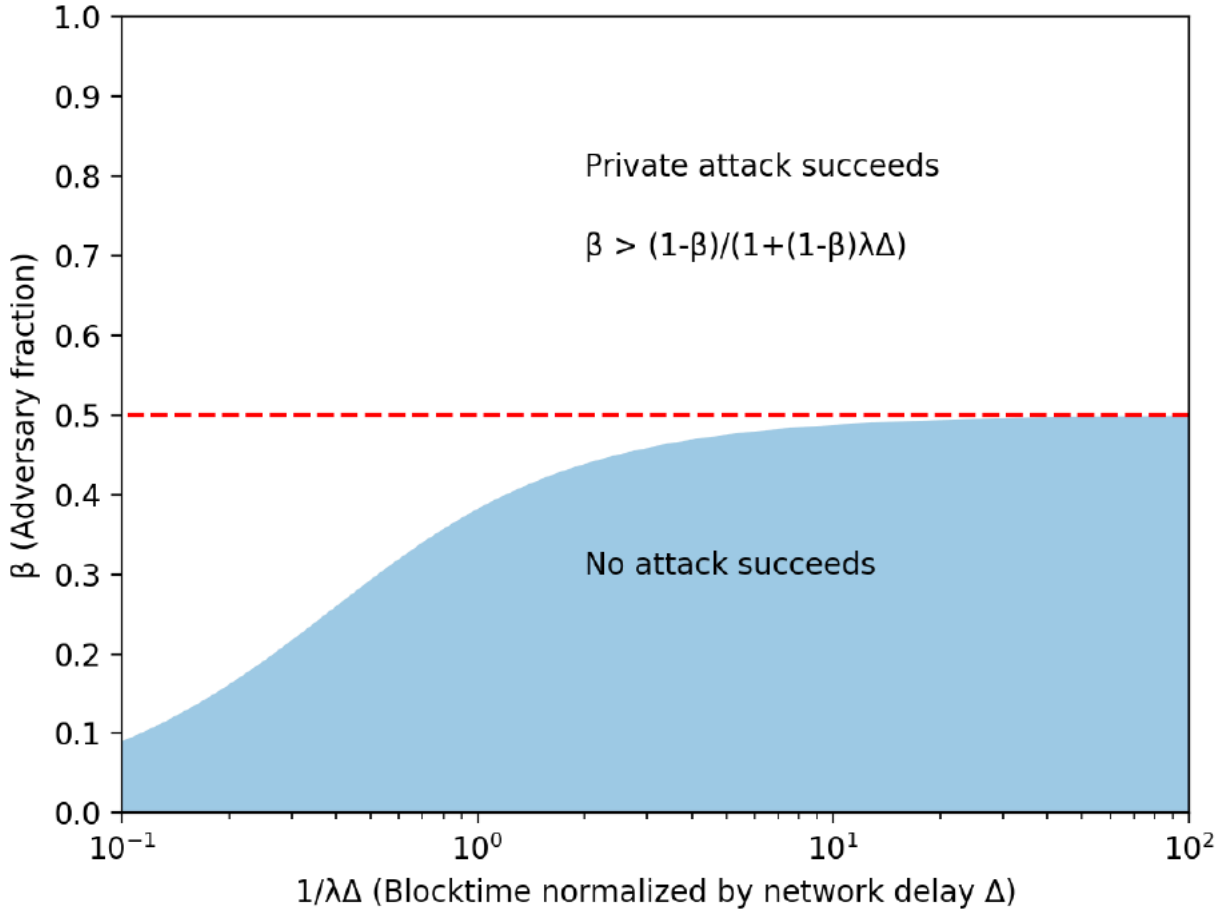
- Every time an honest block is created, it takes $\Delta$ time to propagate it
- Expected # blocks mined in $\Delta$ is $m_h\Delta$, where $m_h$ is the honest mining rate
- Only one of these blocks will extend the longest chain
- Honest miners produce blocks at rate $m_h$ , but only a $\frac{1}{1+m_h\Delta}$ fraction survives
- Thus, the rate growth of longest chain is $\frac{m_h}{1+m_h\Delta}$

# Network delays

- Nakamoto consensus is now secure iff

$$m_a < \frac{m_h}{1 + m_h \Delta}$$

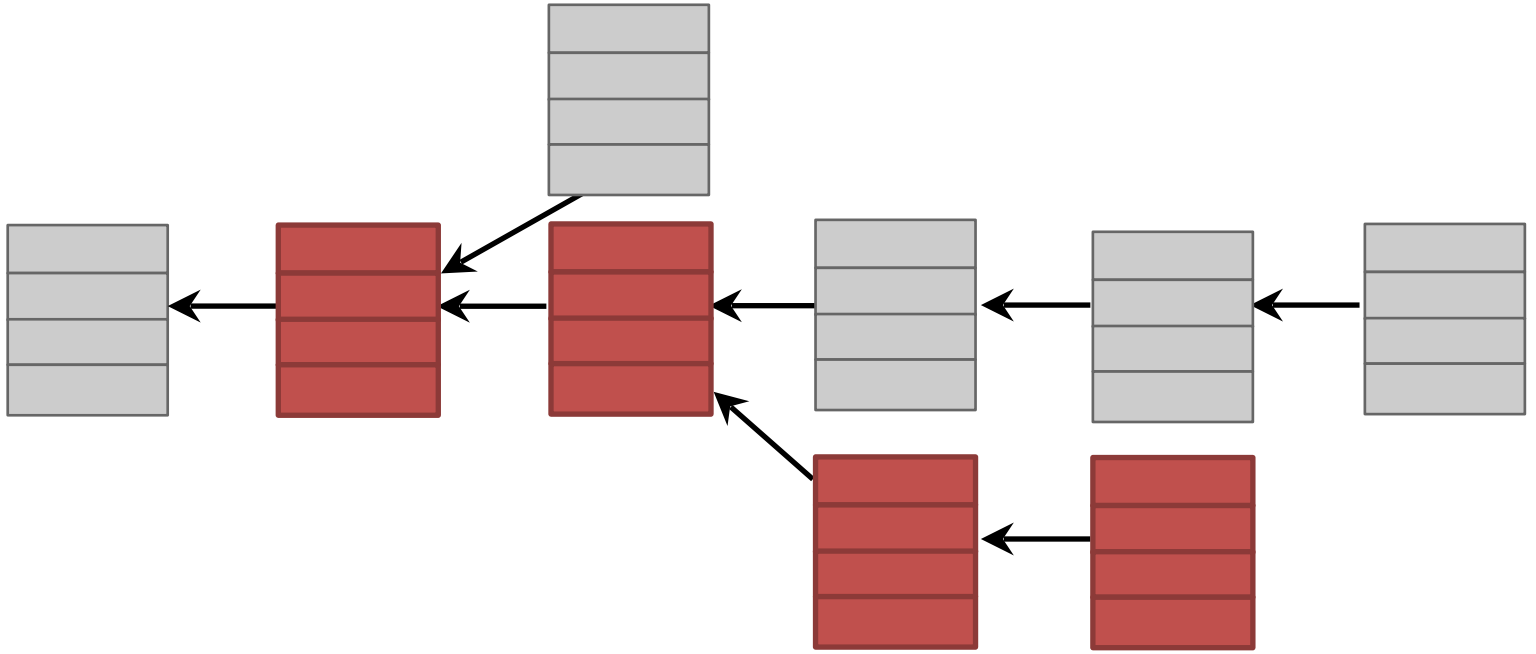
# Network delays




Minimum hash power needed by the adversary to succeed in its private attack (security threshold for any successful safety attack).

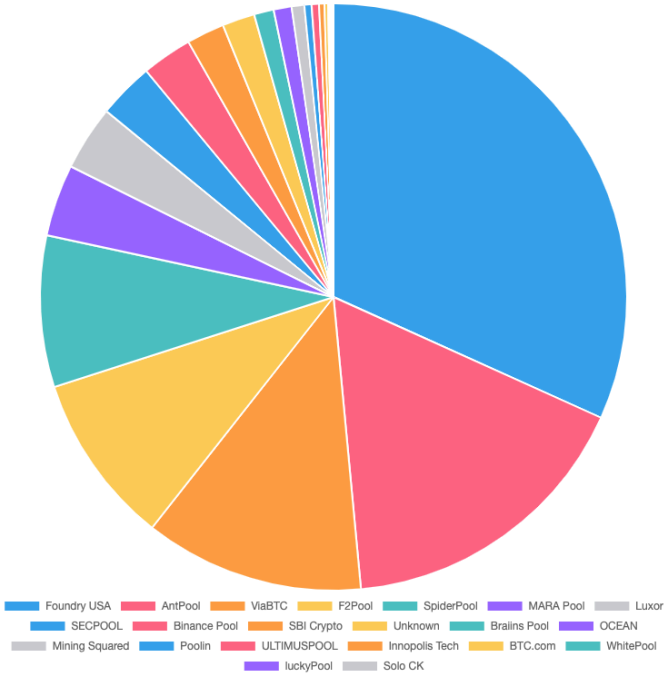
# Selfish mining

# Chain quality



- Chain quality (CQ) can be seen as a measure of *fairness*
- Fairness *motivates* nodes to participate in the protocol and follow it faithfully
- Block rewards are awarded accordingly

# Chain quality

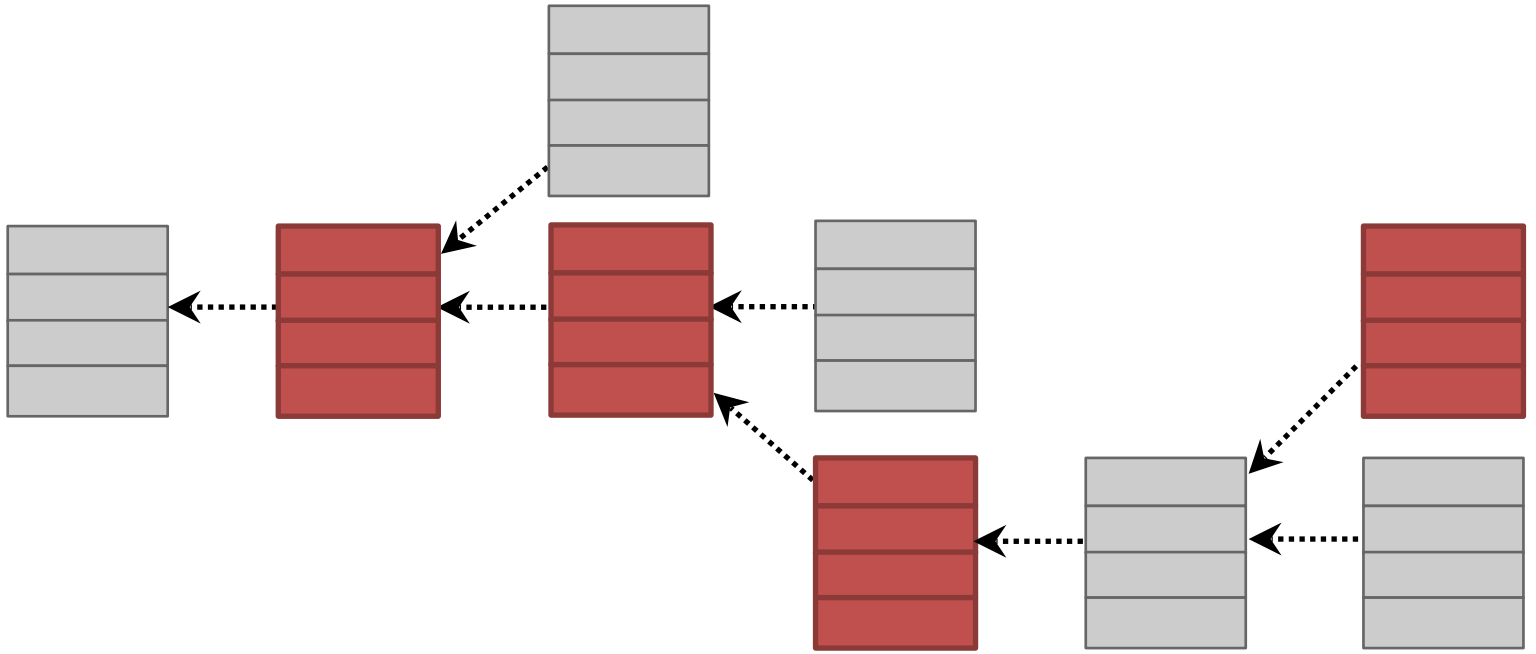



We showed $CQ \geq 1 - \frac{m_a}{m_h}$, but is this bound tight?

**Yes :(**

- Example:
  - AntPool has 20% of global hash rate
  - They can gain up to $\frac{20\,\%}{80\,\%} = 0.25$ block rewards. Is this possible?

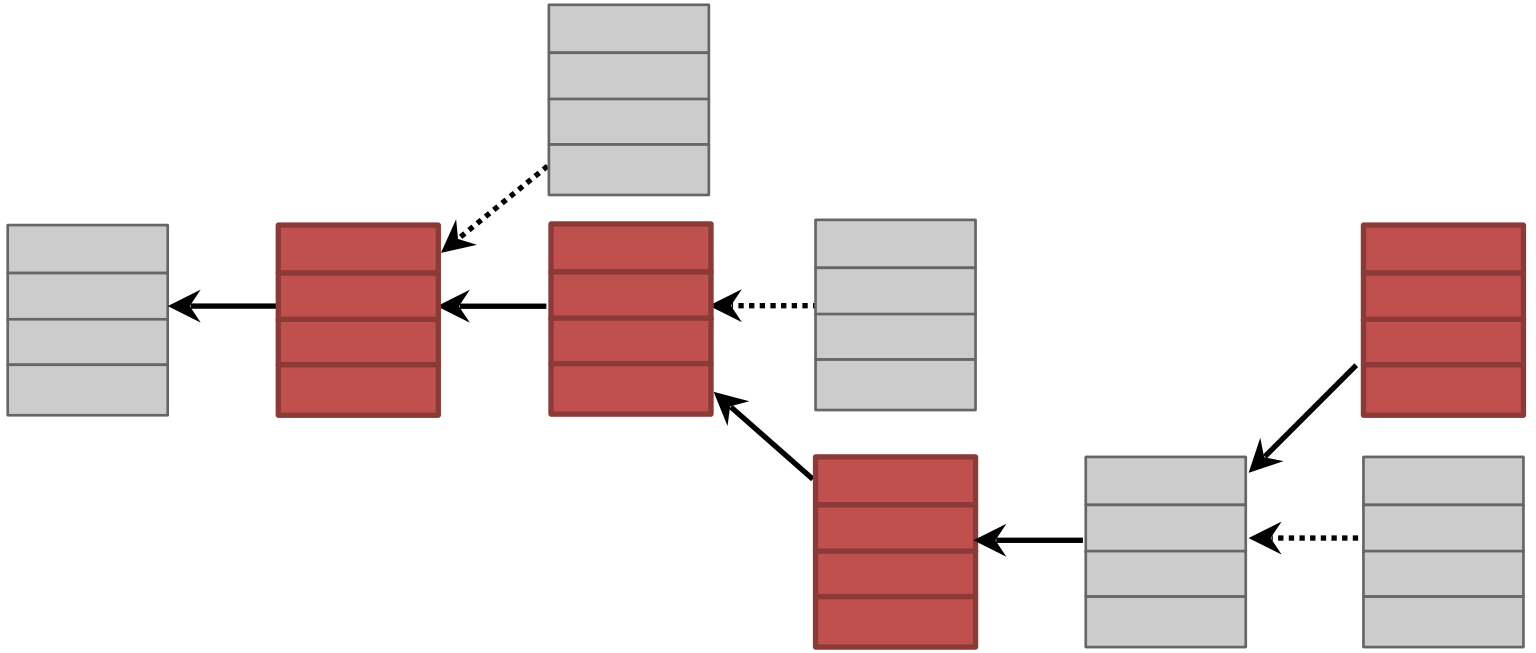
# Selfish mining



1. If latest block is H: try to orphan it alone

2. If latest block is A: mine to extend it, announce only once an honest node is produced at the same height.

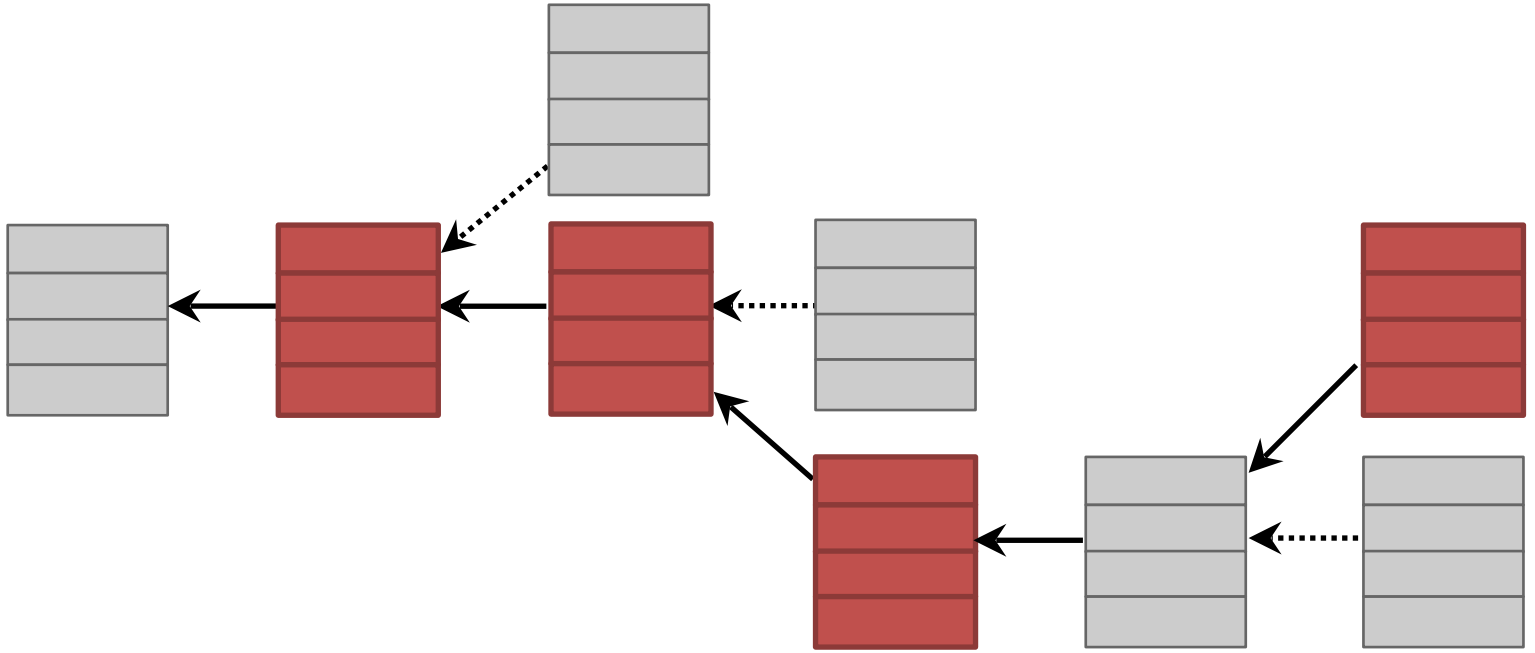
# Selfish mining



1. If latest block is H: try to orphan it alone

2. If latest block is A: mine to extend it, announce only once an honest node is produced at the same height.

# Selfish mining



- Honest nodes and adversary not symmetric
- Adversary has an advantage by deviating, can reduce fairness
- Incentives to honestly follow the protocol are not clear
- Still safety and liveness hold

# Selfish mining

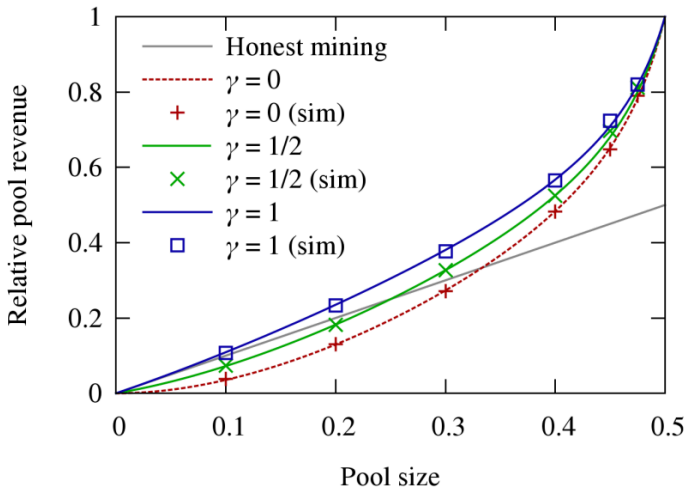



**Fig. 2:** Pool revenue using the Selfish-Mine strategy for different propagation factors $\gamma$, compared to the honest Bitcoin mining protocol. Simulation matches the theoretical analysis, and both show that Selfish-Mine results in higher revenues than the honest protocol above a threshold, which depends on $\gamma$.

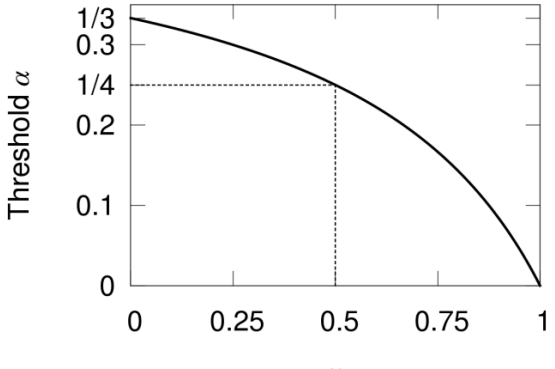



**Fig. 3:** For a given $\gamma$, the threshold $\alpha$ shows the minimum power selfish mining pool that will trump the honest protocol. The current Bitcoin protocol allows $\gamma = 1$, where Selfish-Mine is always superior. Even under unrealistically favorable assumptions, the threshold is never below 1/3.

**Majority is not Enough: Bitcoin Mining is Vulnerable***

Ittay Eyal and Emin Gün Sirer

Computer Science, Cornell University
cs.cornell.edu

# A sneak peek into the future

More on the economics of blockchains on Lecture 9!

# Recap

- Bitcoin is live but not safe in network partitions
- Bitcoin is permissionless but not decentralized
- PoW bypasses FLM: randomization & adversary cannot simulate
- Nakamoto consensus is secure for minority adversaries assuming 0 network delays in synchrony
- Network delays affect security by increasing the power of the adversary via natural forking - latency matters
- Security does not imply Nakamoto consensus is also fair
- There are attacks (e.g., selfish mining) that give an adversary more block rewards than their fair share

# Nakamoto Consensus

## Questions?

Bitcoin: A peer-to-peer electronic cash system. Satoshi Nakamoto. 2008.

Majority is not Enough:Bitcoin mining is Vulnerable. Eyal and Shirer. Communications of ACM, 2018.

November 19th, 2025

TU WIEN

S&P Security & Privacy