Blockchain Principles and Applications

Amir Mahdi Sadeghzadeh, PhD

Data and Network Security Lab (DNSL)
Trustworthy and Secure AI Lab (TSAIL)

Most slides have been adapted from ECE/COS 470, Pramod Viswanath, Princeton 2024 and CS251, Dan Boneh, Stanford 2023.

Recap

Bitcoin Security

- **Safety**: A transaction/block confirmed by one user is soon confirmed by all other users and remains confirmed forever after.
- **Liveness**: all (honest) transactions get included into blocks, and further that the blocks feature in the longest chain.

Protocol level attacks

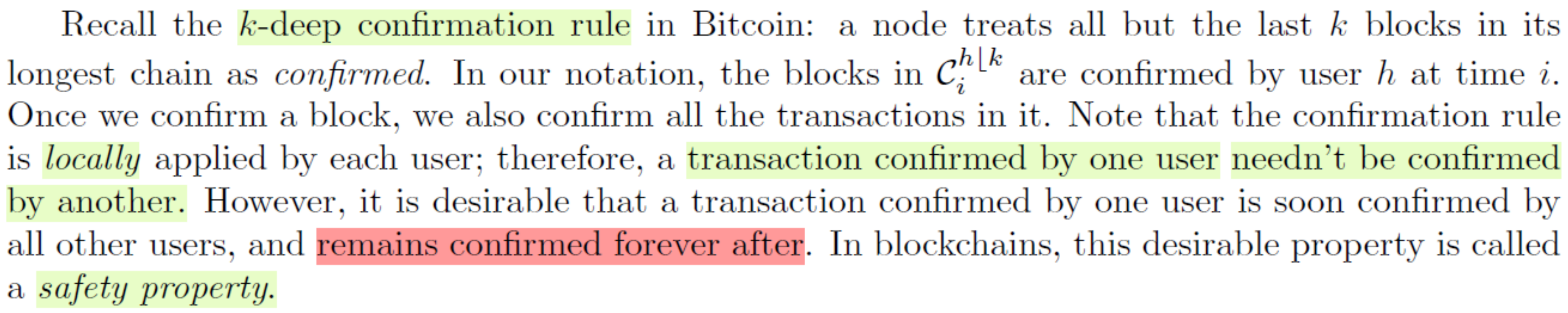
√ Create valid blocks

x Mine on the tip of the longest chain

x Publish the blocks once mined

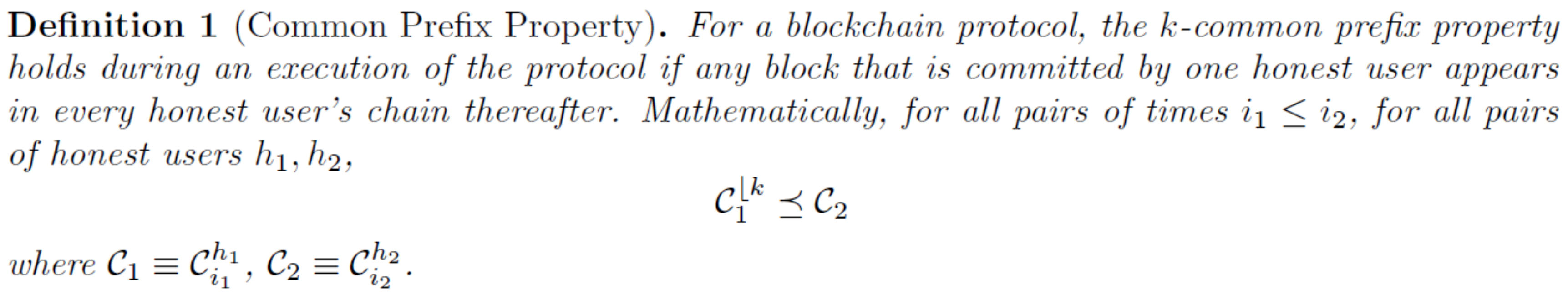
We looked at one strategy called private attack

Nakamoto consensus protocol model



Recall the k-deep confirmation rule in Bitcoin: a node treats all but the last k blocks in its longest chain as *confirmed*. In our notation, the blocks in $\mathcal{C}_i^{h\lfloor k}$ are confirmed by user h at time i. Once we confirm a block, we also confirm all the transactions in it. Note that the confirmation rule is *locally* applied by each user; therefore, a transaction confirmed by one user needn't be confirmed by another. However, it is desirable that a transaction confirmed by one user is soon confirmed by all other users, and remains confirmed forever after. In blockchains, this desirable property is called a *safety property*.

Formal definitions of safety



Definition 1 (Common Prefix Property). *For a blockchain protocol, the k-common prefix property holds during an execution of the protocol if any block that is committed by one honest user appears in every honest user's chain thereafter. Mathematically, for all pairs of times $i_1 \leq i_2$, for all pairs of honest users h_1, h_2,*

$$\mathcal{C}_1^{\lfloor k} \preceq \mathcal{C}_2$$

where $\mathcal{C}_1 \equiv \mathcal{C}_{i_1}^{h_1}$, $\mathcal{C}_2 \equiv \mathcal{C}_{i_2}^{h_2}$.

Formal definitions of safety

Definition 2 (Individual block safety)**.** *In an execution, a block B present in some honest user's chain is safe if, after it has been committed by any honest user, it remains a part of all honest user's chains. Mathematically, if block B is committed by some user h at time t, then for all $t' \geq t$, for all honest users h', $b \in \mathcal{C}_{t'}^{h'}$*

Mining as a Poisson process

Time to a successful mining event is an **exponential** random variable

$$T \sim exp(\lambda) \;\; if \;\; \Pr(T \geq t) = e^{-\lambda t}$$

Memoryless:

$$\Pr(T \geq t + t_0 | T \geq t_0) = \frac{\Pr(T \geq t + t_0)}{\Pr(T \geq t_0)} = \frac{e^{-\lambda(t+t_0)}}{e^{-\lambda t_0}} = e^{-\lambda t} = \Pr(T \geq t)$$

Number of mined blocks in time T is a **Poisson** random variable

$$X \sim Poi(\lambda T) \;\; if \;\; \Pr(X = k) = \frac{(\lambda T)^k e^{-\lambda T}}{k!}$$

The mining process is a Poisson process with rate λ, proportional to hash power

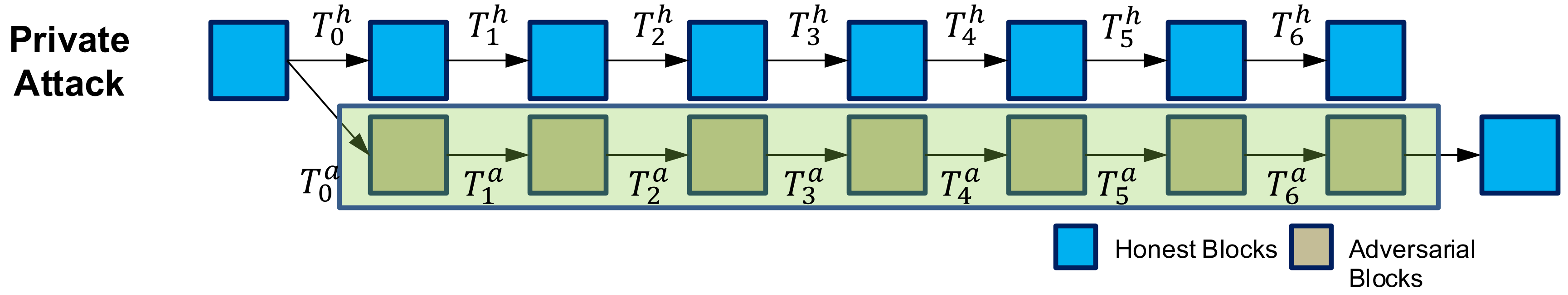
Mining as a Poisson Process

Mathematical fact: The sum of multiple independent Poisson processes is still a Poisson process

Consequence: the honest/adversarial mining processes are independent Poisson processes with constant mining rate

Honest mining:	Poisson process with rate $(1-\beta)\lambda$
Adversarial mining:	Poisson process with rate $\beta\lambda$

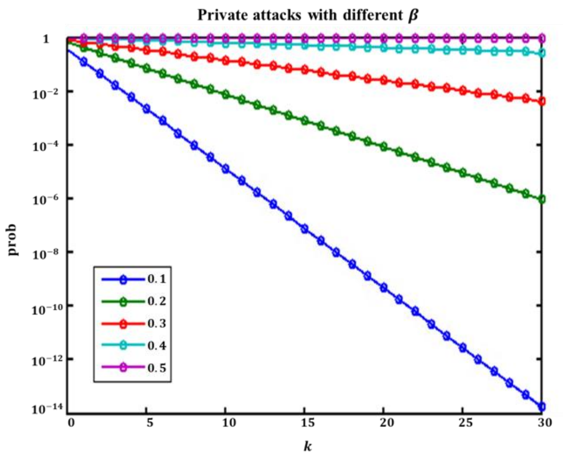
Private attack



Attack Success

$$\xrightarrow{k \to \infty} \sum_{i=0}^{\infty} T_i^h > \sum_{i=0}^{\infty} T_i^a \to \beta\lambda > (1-\beta)\lambda \to \beta > \frac{1}{2}$$

$$\xrightarrow{k < \infty} \quad p_a = e^{-c(k+1)} \quad \longrightarrow$$



Private attack (k>0 and finite)

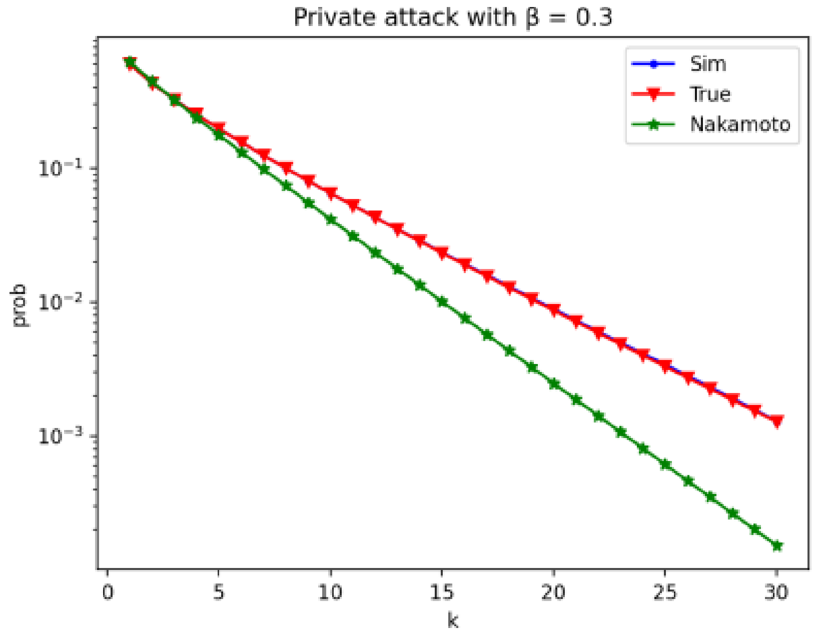


Figure 2: Private attack with $\beta = 0.3$.

Safety analysis under bounded network delay

- Block propagation delay in the Internet is of the order of a few seconds on average.

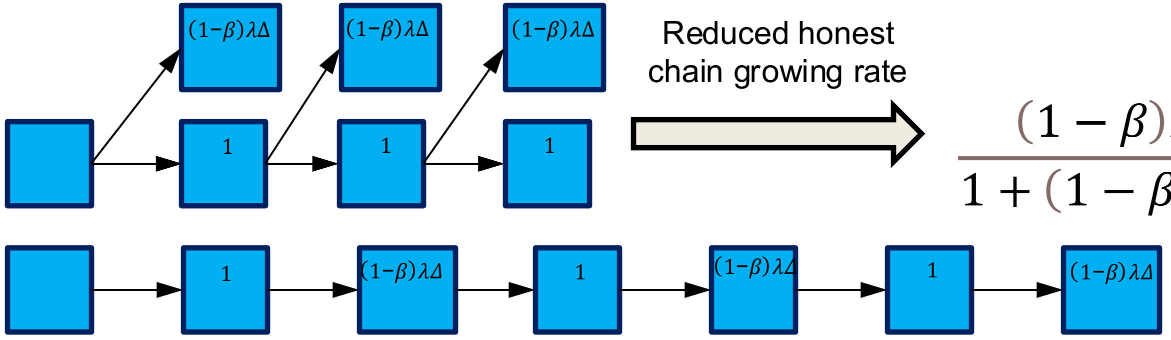
- **Natural Forking**
  - Honest miners may occasionally **fork** naturally due to network delay — they may not see each other's blocks in time.

- **Honest Forking (Top row)**
  - In this model:
    - Network delay causes multiple honest miners to unknowingly mine on different tips.
    - This creates forks among honest blocks.
    - This reduces the **effective chain growth rate** of honest miners, because only one of the forked chains can eventually be accepted.

Private Attack (With Honest Forking)

Δ - synchronous network model: network delays bounded by Δ

Natural forking:



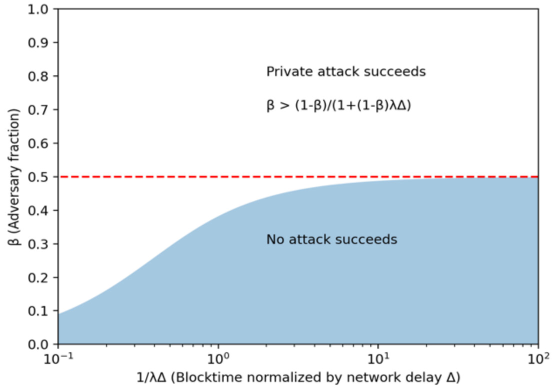
Reduced honest chain growing rate

$$\frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta}$$

Attack Success

Δ network delay

$$\beta\lambda > \frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta}$$



Liveness

Liveness

Last lecture: safety of the longest chain protocol, which means that once a block is confirmed (e.g., k-deep), then the probability of deconfirmation is very small (k large).

Safety is an important security property

But what if no block gets into the ledger, or blocks get in but some honest transactions don't?

Liveness is an important security property: focus of this lecture

Observations

- The longest chain protocol cannot deadlock
  - Mining operation is very democratic and even a single honest miner with tiny hash power will eventually succeed in mining

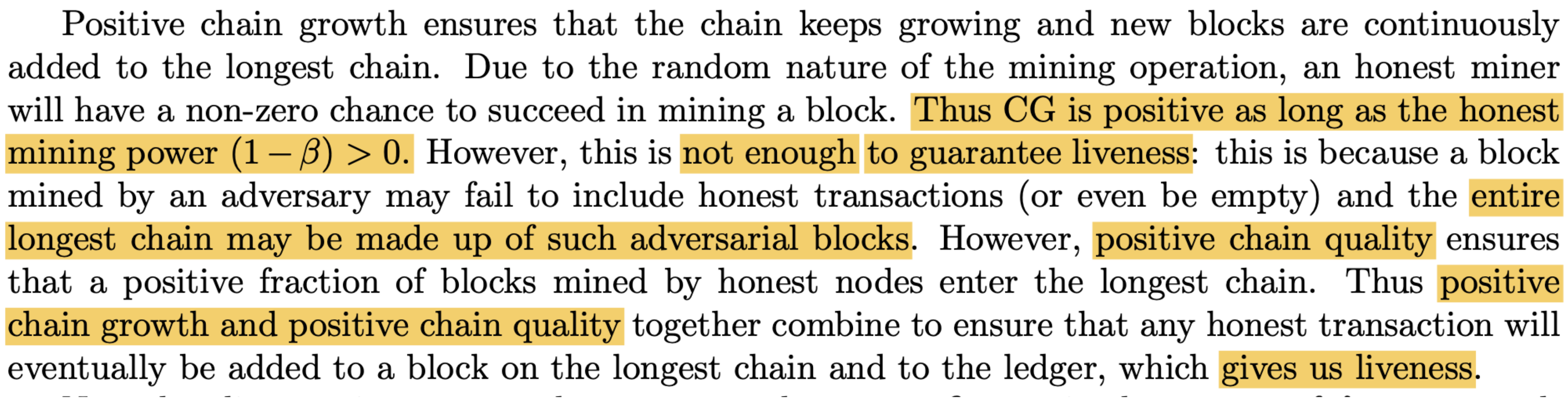
- But that doesn’t guarantee liveness. There are two adversarial events.
  - Notice the ledger is made up of blocks on the longest chain. It could happen that all blocks on the longest chain are adversarial
  - A block mined by an adversary could be simply empty (censoring all transactions) or censor specific transactions

- Both these are fatal liveness attacks.

Liveness

- Liveness ensures that all transactions make their way into the ledger and safety ensures that eventually the transactions stay permanently in the ledger (with high probability).

Liveness



Positive chain growth ensures that the chain keeps growing and new blocks are continuously added to the longest chain. Due to the random nature of the mining operation, an honest miner will have a non-zero chance to succeed in mining a block. Thus CG is positive as long as the honest mining power $(1-\beta) > 0$. However, this is not enough to guarantee liveness: this is because a block mined by an adversary may fail to include honest transactions (or even be empty) and the entire longest chain may be made up of such adversarial blocks. However, positive chain quality ensures that a positive fraction of blocks mined by honest nodes enter the longest chain. Thus positive chain growth and positive chain quality together combine to ensure that any honest transaction will eventually be added to a block on the longest chain and to the ledger, which gives us liveness.

Fairness

- CQ quantifies fairness
  - since block rewards are provided to blocks in the longest chain, having blocks mined by honest nodes in the longest chain ensures that honest miners are rewarded.

Chain Growth and Chain Quality

Definition 1 (Chain Growth)**.** *For a blockchain protocol, we define the chain growth of a longest chain $\mathcal{C}$ as the average growth rate of $\mathcal{C}$ (number of blocks per unit time), denoted as* $\mathrm{CG}(\mathcal{C})$.

Definition 2 (Chain Quality)**.** *For a blockchain protocol, we define the chain quality of a longest chain $\mathcal{C}$ as the fraction of honest blocks in $\mathcal{C}$, denoted as* $\mathrm{CQ}(\mathcal{C})$.

Secure Blockchain

- We require the following properties from a secure blockchain (T=k):

  - *consistency*: with overwhelming probability (in T), at any point, the chains of two honest players can differ only in the last T blocks;
  - *future self-consistence*: with overwhelming probability (in T), at any two points r, s the chains of any honest player at r and s differ only within the last T blocks;
  - *g-chain-growth*: with overwhelming probability (in T), at any point in the execution, the chain of honest players grows by at least T messages in the last $\frac{T}{g}$ rounds; g is called the chain-growth of the protocol.
  - the *μ-chain quality* with overwhelming probability (in T), for any T consecutive messages in any chain held by some honest player, the fraction of messages that were "contributed by honest players" is at least μ.

Chain growth

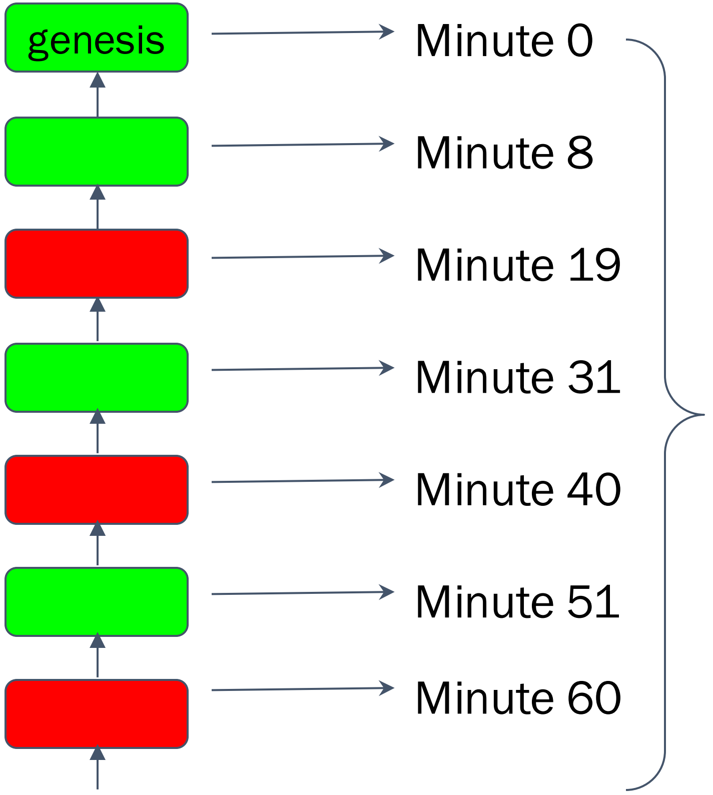
Chain growth (CG): rate of growth of the longest chain

Observation 1: $CG > 0$

- **Adversary stays silent** $CG = \frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta}$
- **Adversary acts honest** $CG = \frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta} + \beta\lambda$

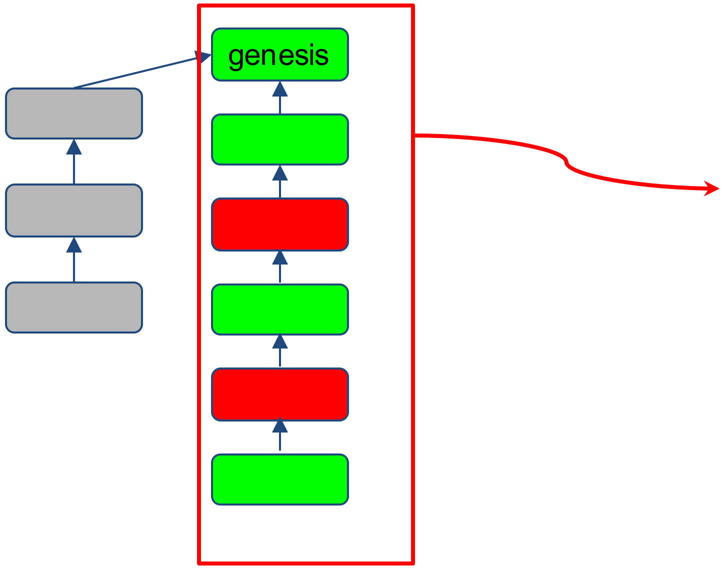
Claim: $CG > \frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta}$

Chain growth



Chain growth is
1 block/10 minutes

Chain quality



4 honest blocks

2 adversarial blocks

In the **longest chain**

=> chain quality is 4/6 = 0.667

Chain quality (CQ): # of honest blocks in the longest chain/# of all blocks in the longest chain

Chain quality

Observation: we need $CQ > 0$ for liveness

$$CQ \geq \frac{CG * T - \beta\lambda T}{CG * T} = \frac{CG - \beta\lambda}{CG}$$

$$CQ > 0 \Leftrightarrow \frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta} > \beta\lambda$$

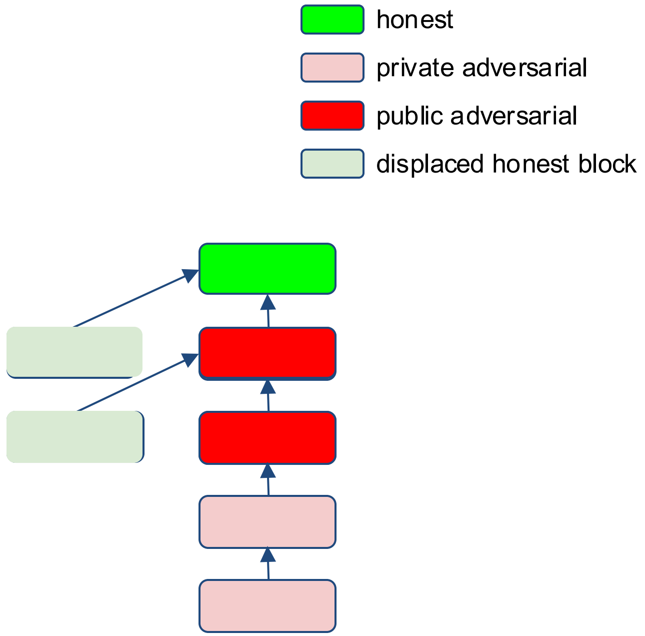
This is exactly the same condition for safety

Turns out the condition above is also **necessary** for liveness $\rightarrow$ Selfish mining

Selfish mining attack

1. The adversary always mines on the block at the tip of the longest chain, whether the chain is private or public. Upon successful mining, the adversary maintains the block in private to release it at an appropriate time.
2. When an honest miner publishes a block the adversary will release a previously mined block at the same level (if it has one).

* Adversary can break ties in its favor, so honest miners will mine on the adversarial block.



Chain quality and liveness

$$1-\beta \geq CQ \geq \frac{\frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta} - \beta\lambda}{\frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta}}$$

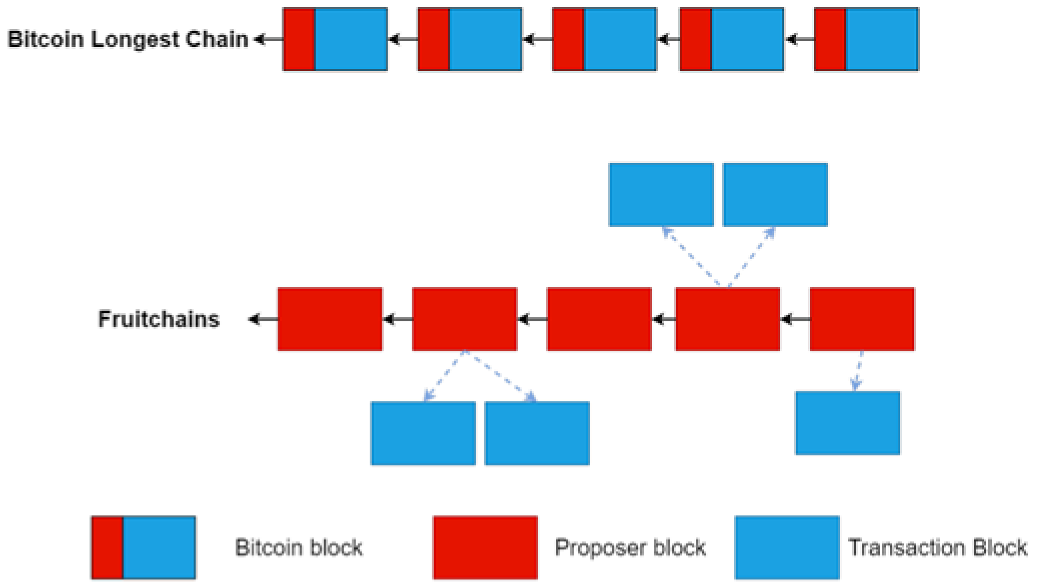
Chain quality quantifies liveness and **fairness**

The most fair reward distribution occurs when number of honest blocks on the longest chain is proportional to honest hash power $(1-\beta)$, i.e., $CQ = 1-\beta$,

This is not happening in the longest chain protocol (due to selfish mining).

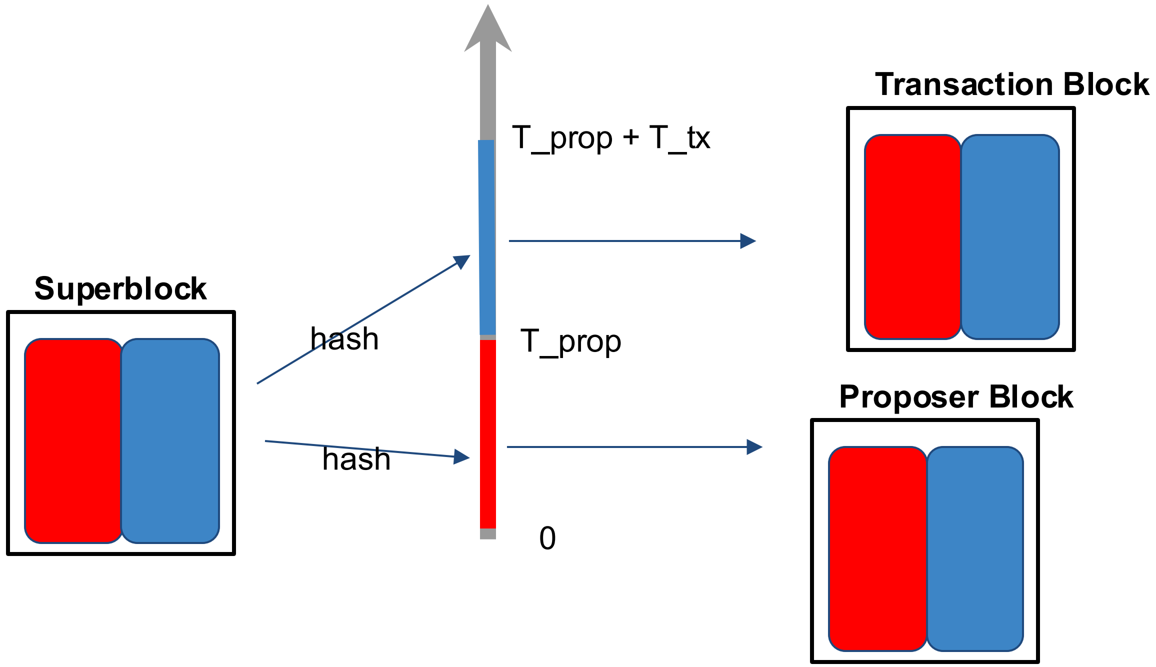
Fruitchains

Main idea: separate transactions (& their rewards) from blocks in the longest chain

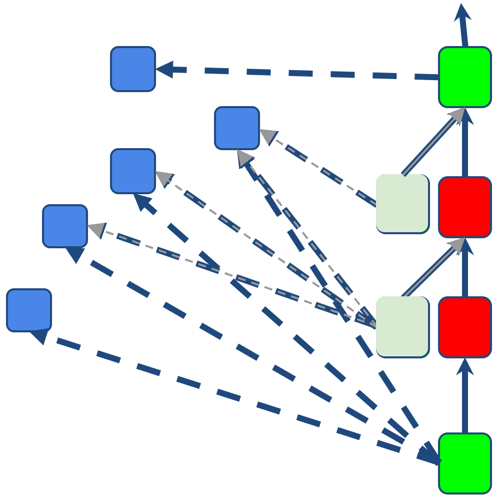


Cryptographic Sortition

How to do PoW for both types of blocks simultaneously?



Optimal chain quality



🟦	Transaction block
⬜	Displaced proposer block
🟩	Honest proposer block
🟥	Adversarial proposer block

Fraction of honest transaction blocks **in the longest chain** $\longrightarrow$ Fraction of honest transaction blocks **overall** $\longrightarrow$

$$CQ \geq \frac{(1-\beta)\lambda_{tx}}{(1-\beta)\lambda_{tx} + \beta\lambda_{tx}} = \boxed{1-\beta.}$$

Short time scale optimal CQ

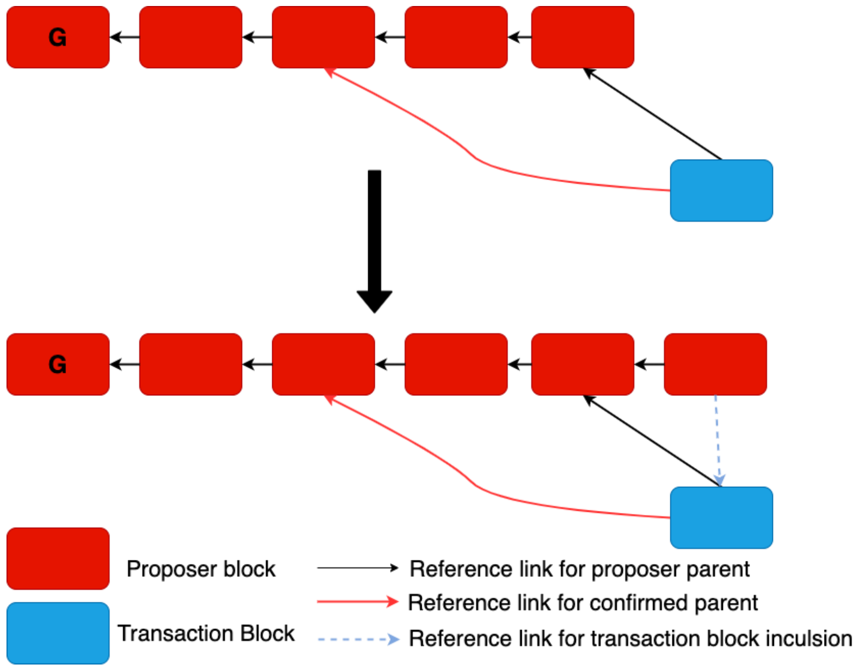
Block withholding attack:

- An attacker keep successfully mined transaction blocks private
- Then it suddenly release a large number of them at the same time, thereby creating a very high fraction of adversarial transaction blocks in some small segment of the proposer chain

Resolution: by requiring that a transaction block should “hang” from a confirmed proposer block which is not too far from the proposer block which includes it.

Short time scale optimal CQ

- During mining, each transaction block refers to a recently stabilized/confirmed proposer block (called confirmed parent)
- Recency condition: a transaction block B is recent with respect to a proposer chain C if the confirmed parent of B is a block that is at most R deep in C, where R is a recency parameter.
- Proposer blocks only include recent transaction blocks


G
G
Proposer block
Transaction Block
Reference link for proposer parent
Reference link for confirmed parent
Reference link for transaction block inculsion

Resources

- ECE/COS 470, Pramod Viswanath, Princeton 2024