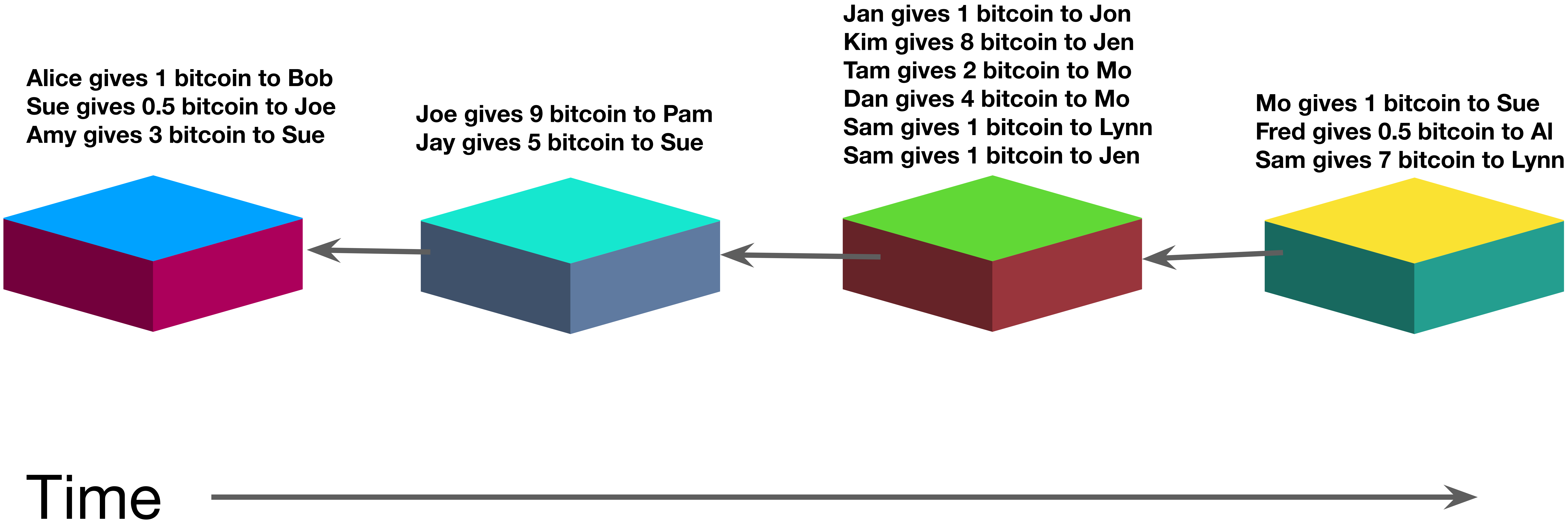
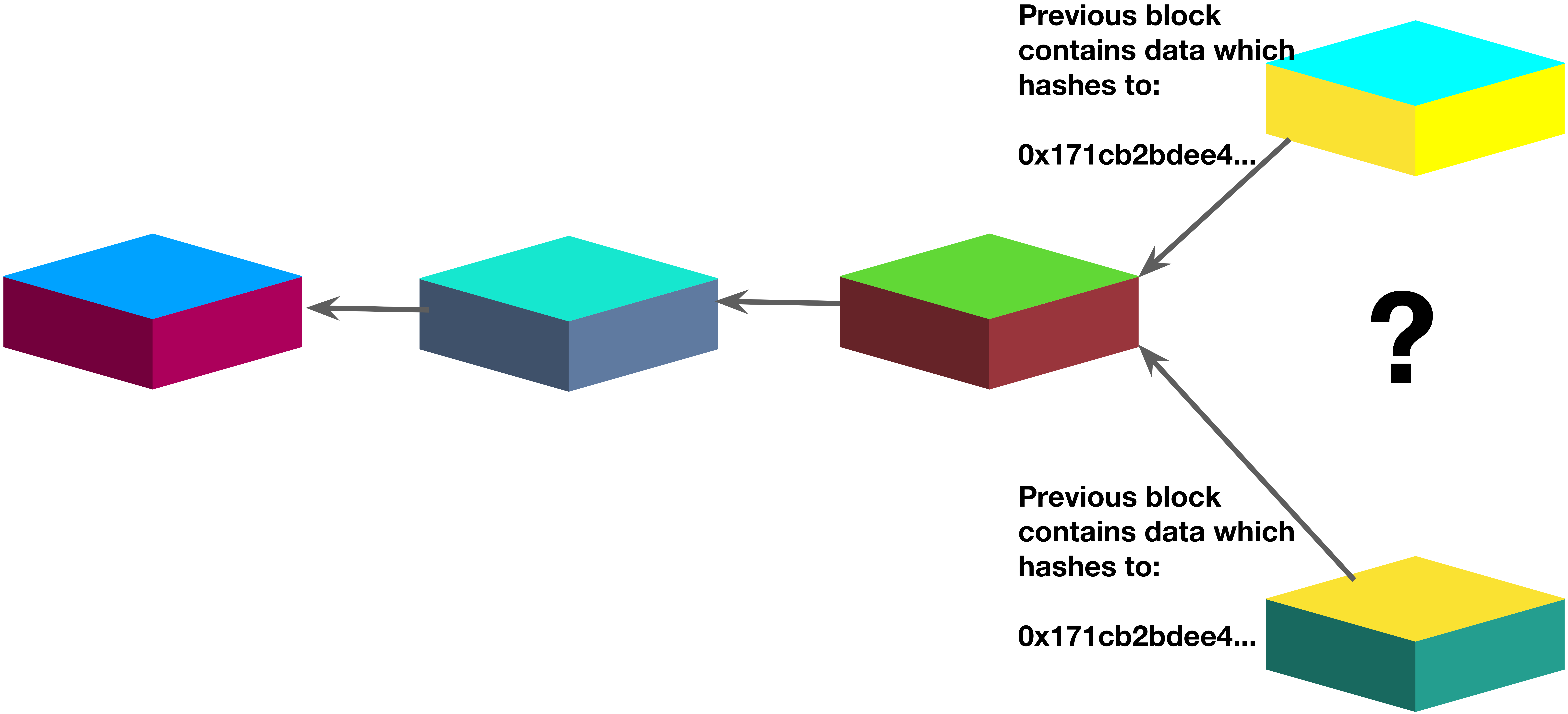
# Intro to Polkadot, Part 4: Proof of Stake and NPoS

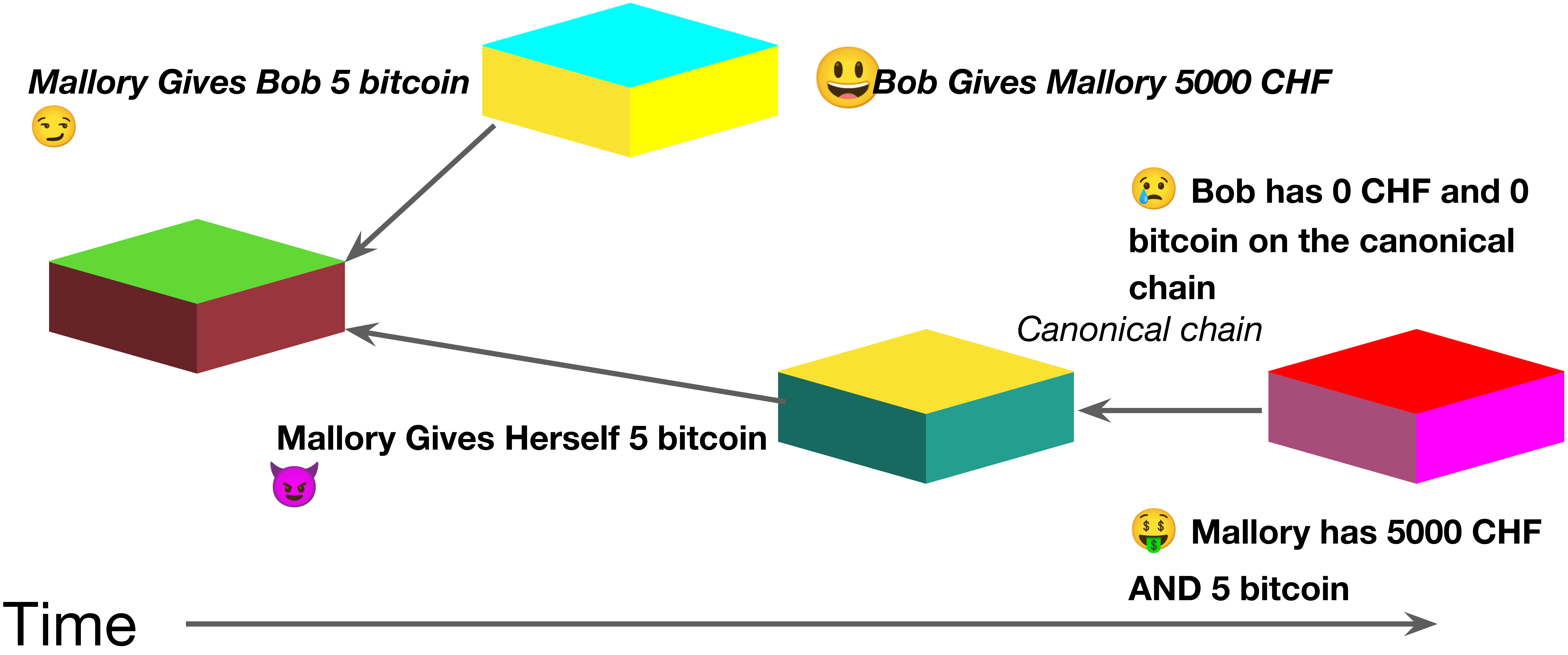
# Agreement on Canonical History



# Block Ordering

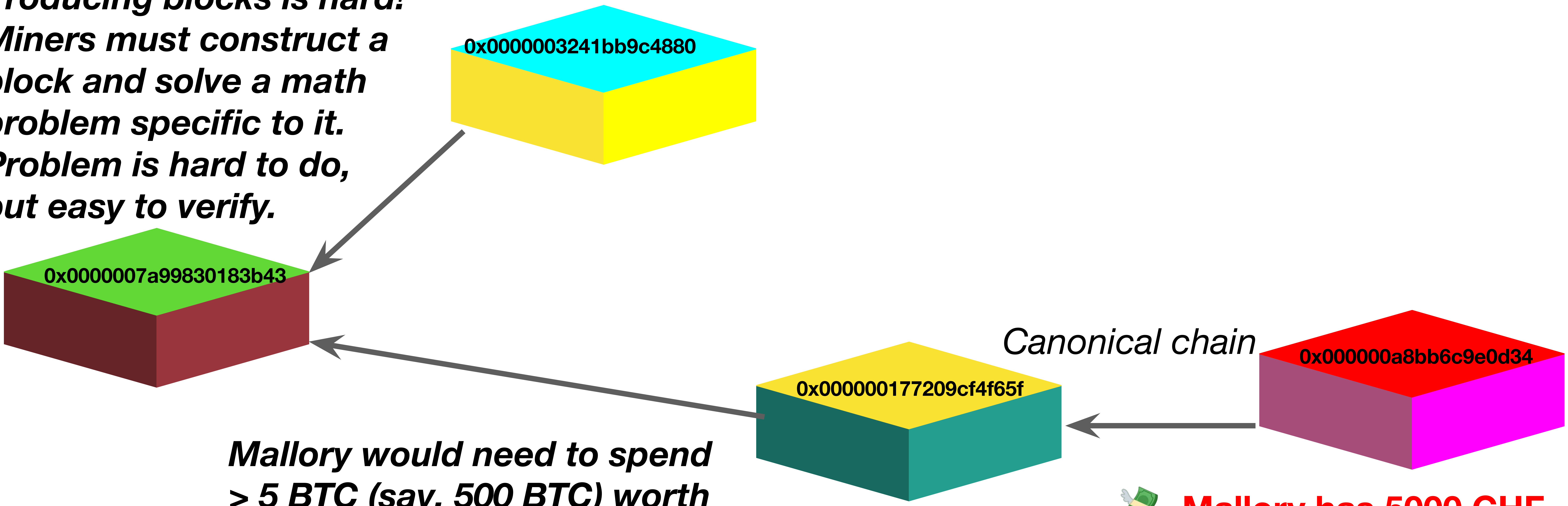


# Double Spend Attack



# The Proof of Work Solution

***Producing blocks is hard! Miners must construct a block and solve a math problem specific to it. Problem is hard to do, but easy to verify.***




***Mallory would need to spend > 5 BTC (say, 500 BTC) worth of computing power to produce multiple valid blocks***

**Mallory has 5000 CHF**

**AND 5 BTC but spent > 500 BTC to do it!**

Time

# Drawbacks to PoW

1. Extremely high electricity usage
2. Electronic waste
3. High variance in block times
4. Economies of scale lead to concentrated mining

# Solutions

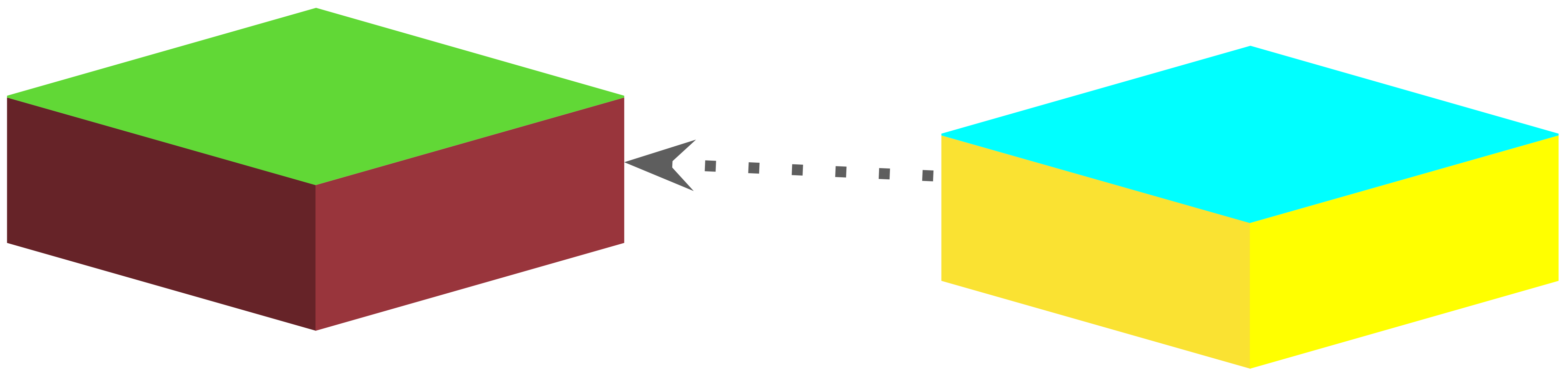
1. Proof-of-useful-work (e.g. Primecoin)
2. Proof-of-storage (e.g. Chia)
3. Proof-of-service (e.g. Dash)
4. Proof-of-authority (e.g. permissioned chains)
5. Proof-of-history (e.g. Solana)
6. **Proof-of-stake**

# Towards Proof of Stake

***Can we develop a canonical ordering and a way to select the next block producer using only on-chain information?***

# Simple Proof of Stake Mechanism

***Hash:***
***0x932AC377***
***= 2’469’053’303***



Potential Producer A (5): 0...4

Potential Producer B (10): 5...14

Potential Producer C (30): 15...44

Potential Producer D (20): 45...64

Potential Producer E (35): 65...99

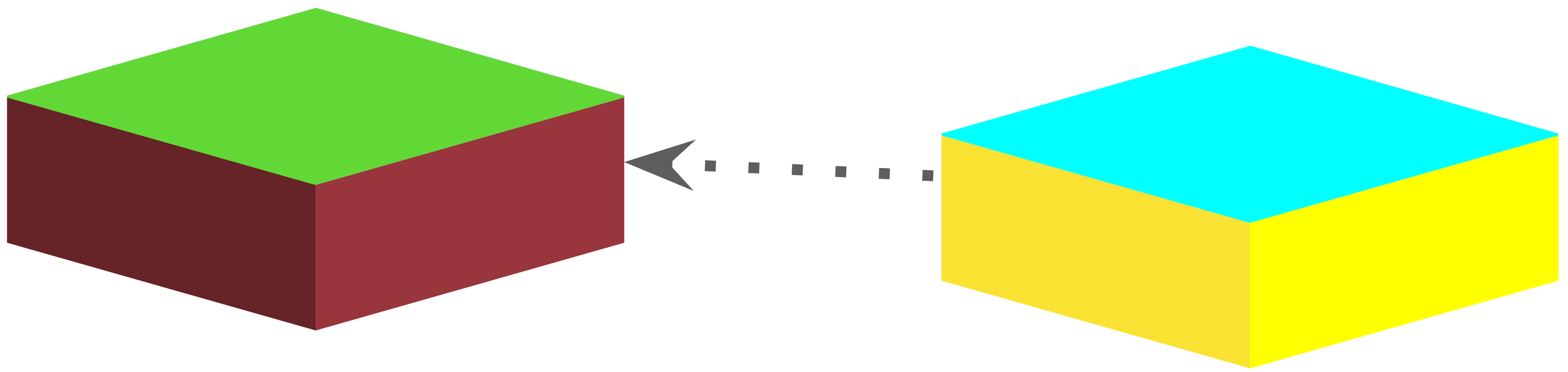
1

next_block_producer = hash_of_previous_block % (A + B + C + D + E)

Time

# Simple Proof of Stake Mechanism

***Hash:***
***0x932AC377***
***= 2’469’053’303***



Potential Producer A (5)

Potential Producer B (10)

Potential Producer C (30)

Potential Producer D (20)

Potential Producer E (35)

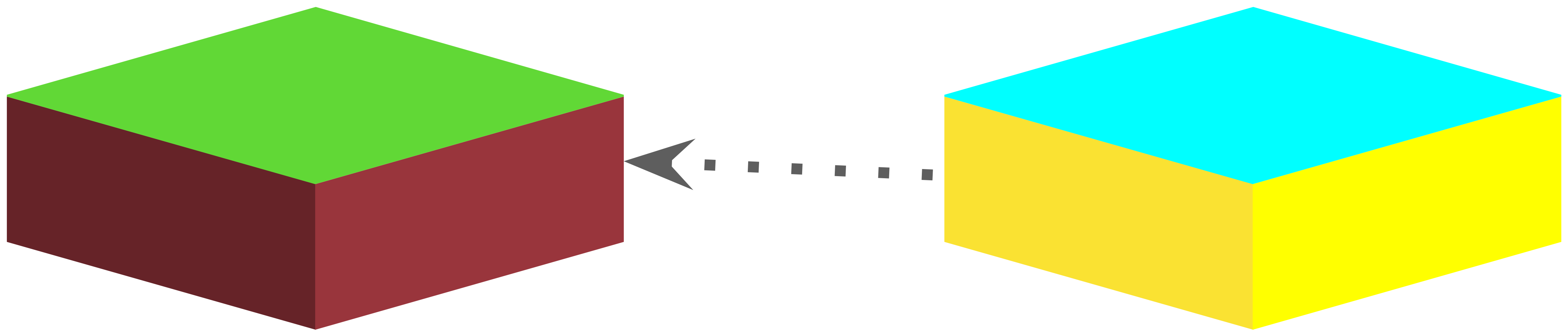
1

next_block_producer = 2’469’053’303 % (100) = 3 -> Producer A

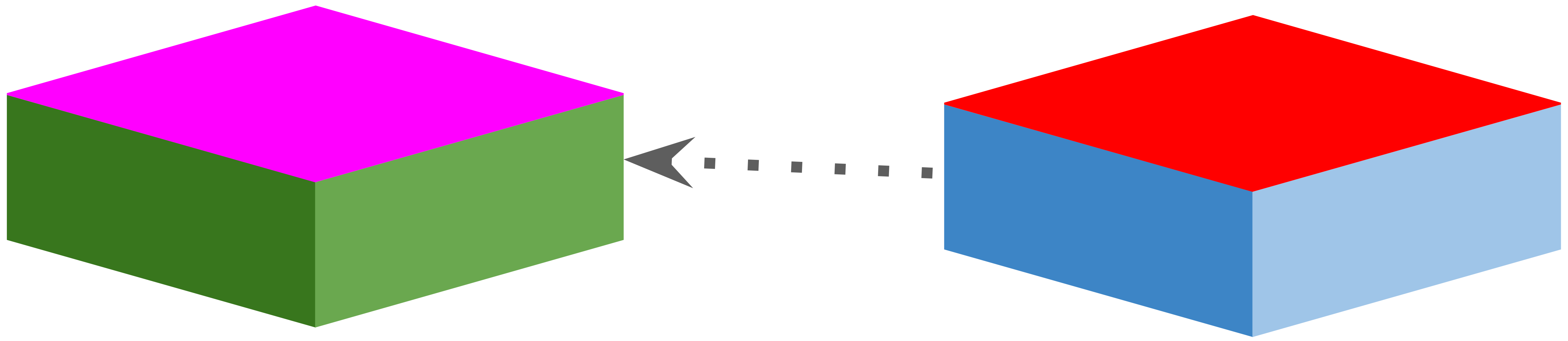
Time

# Problem?

***Hash: 2’469’053’303***



***Hash: 9’322’764’398***



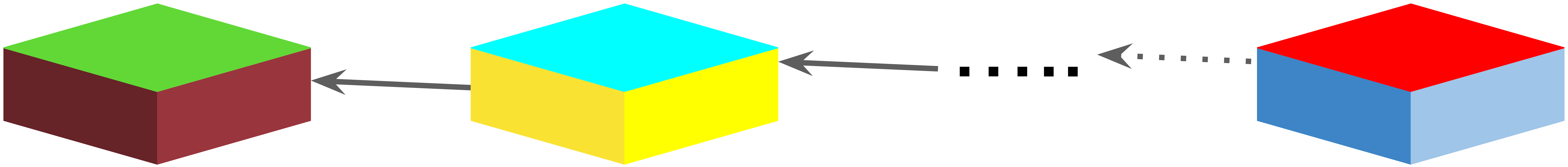
- Potential Producer A (5)
- Potential Producer B (10)
- Potential Producer C (30)
- Potential Producer D (20)
- Potential Producer E (35)

1

next_block_producer = 2’469’053’303 % (100) = 3 -> Producer A
next_block_producer = 9’322’764’398 % (100) = 98 -> Producer E

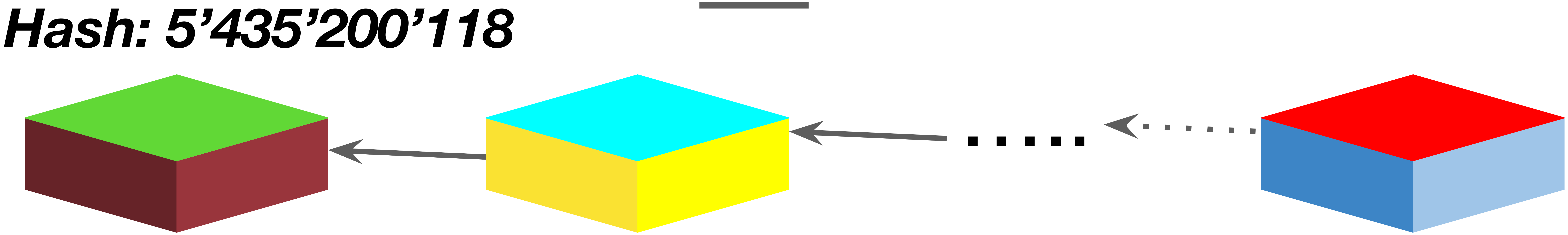
# Look Back Further

***Hash: 5’435’200’118***



next_block_producer = 10_hashes_back % (100) = 18 -> Producer C

# Another Problem: Predictability




If I know Producer C will produce the next block, I can try to DDOS them
If I know Producer E will produce block after that, can move resources to them

Ideally, it should be as difficult as possible to know who is going to produce the next block!

There are solutions to this such as Verifiable Random Functions (VRFs)

# Bonding / Staking

next_block_producer = hash_of_previous_block % (A + B + C + D + E)

***Owners of A, B, C, D, E must be online and ready to produce at any moment***

***Need to know A + B + C + D + E.. cannot be modified***

***(also need to punish bad actors, but let's discuss later..)***

***Thus, tokens usually locked ("bonded" or "staked") for a specific amount of time***

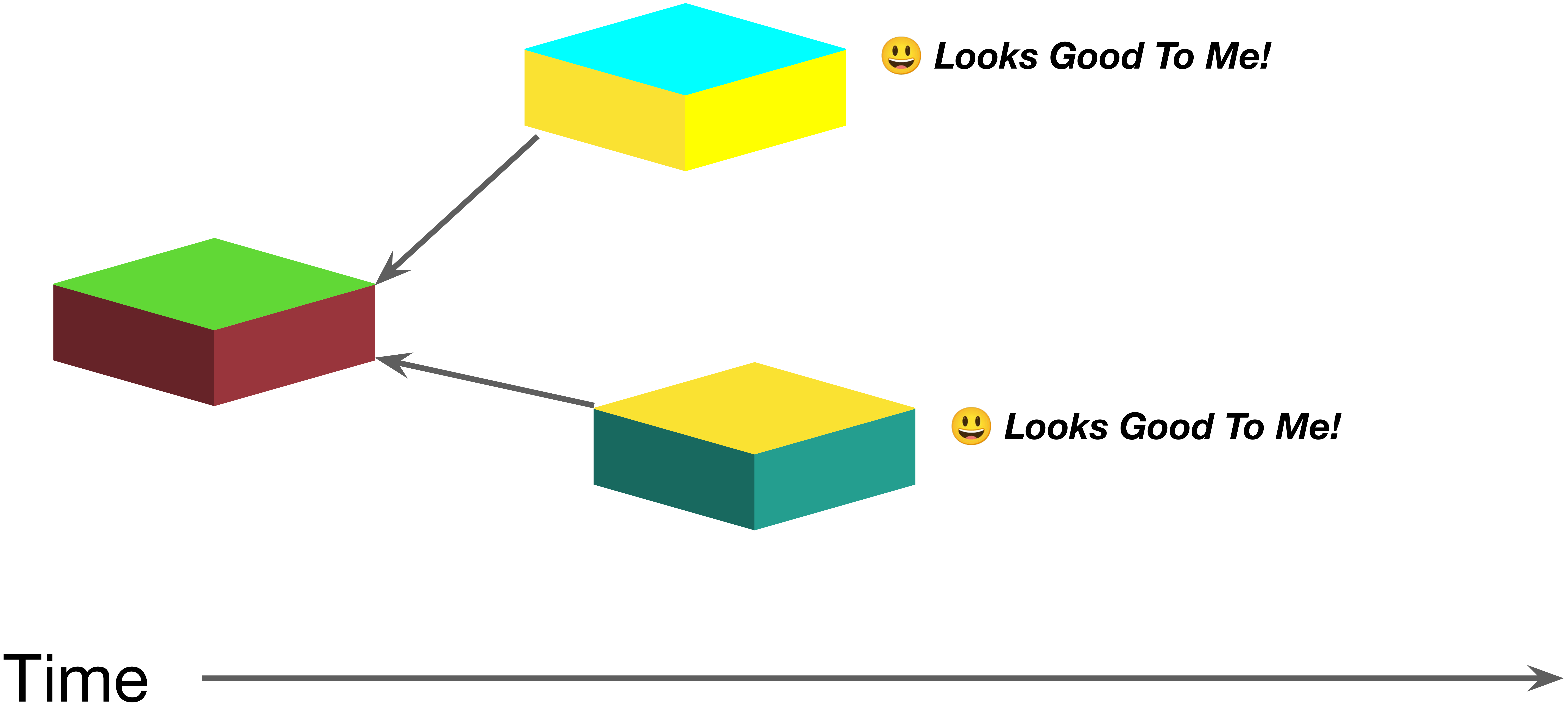
# Towards Proof of Stake

To avoid double-spend, it should be expensive to generate blocks or, fundamentally, to follow a non-canonical chain

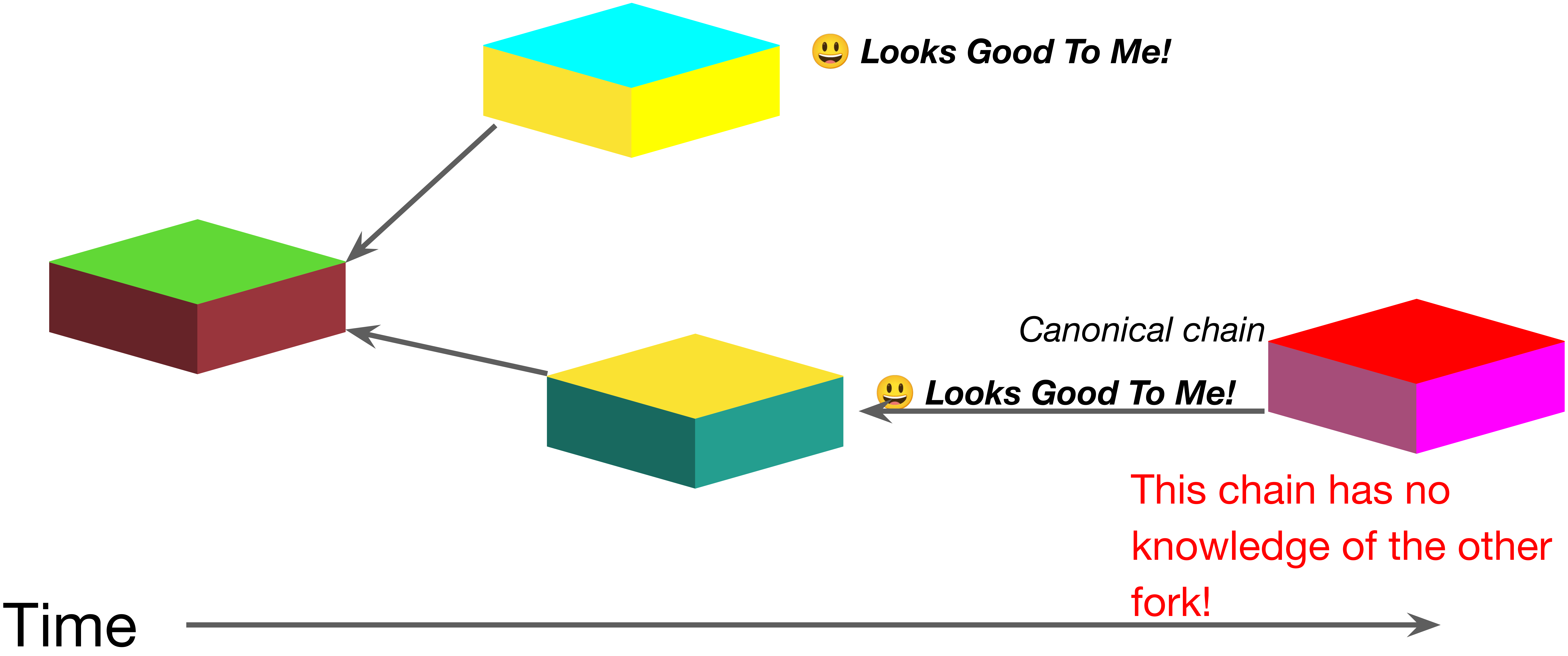
In proof of work systems, there is an external price paid to generate blocks (hardware + electricity)

Can we make it expensive to follow a non-canonical chain without these externalities?

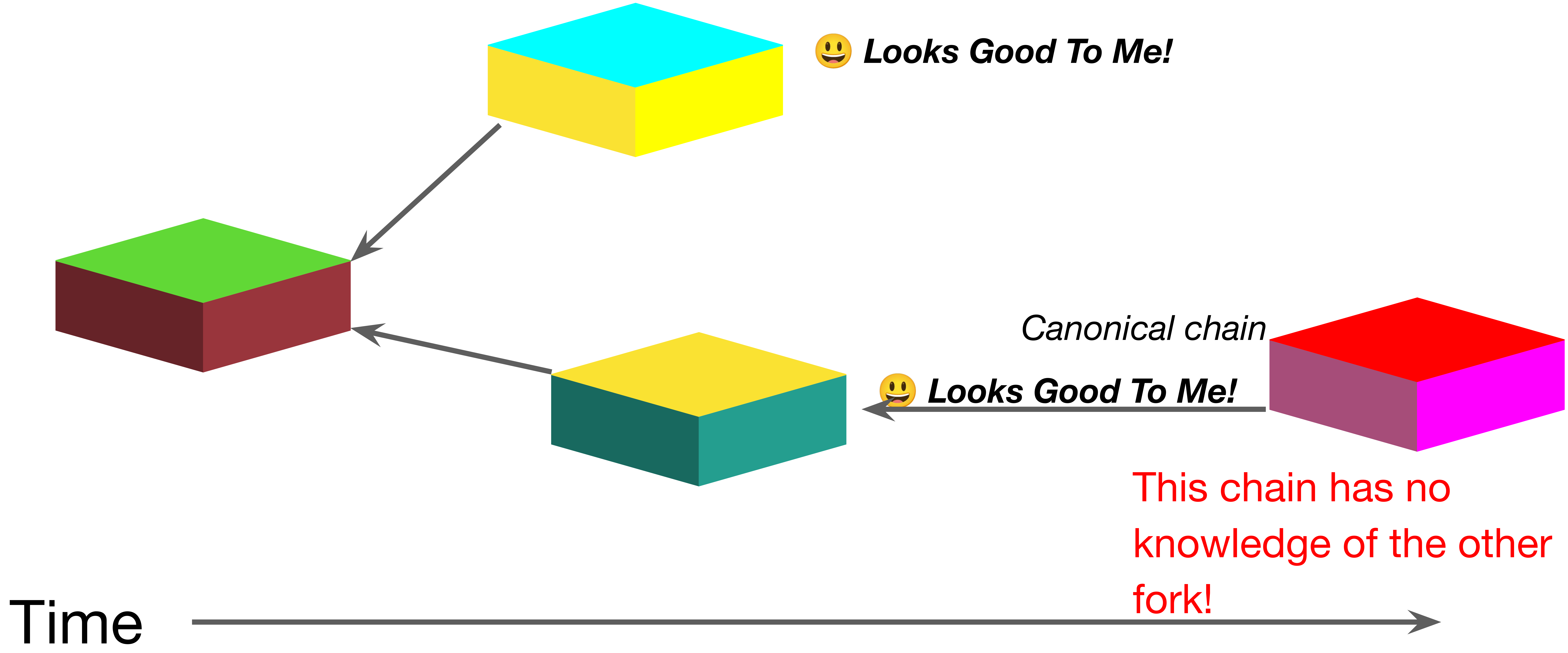
# Proof of Stake - Prevent Double Spend?



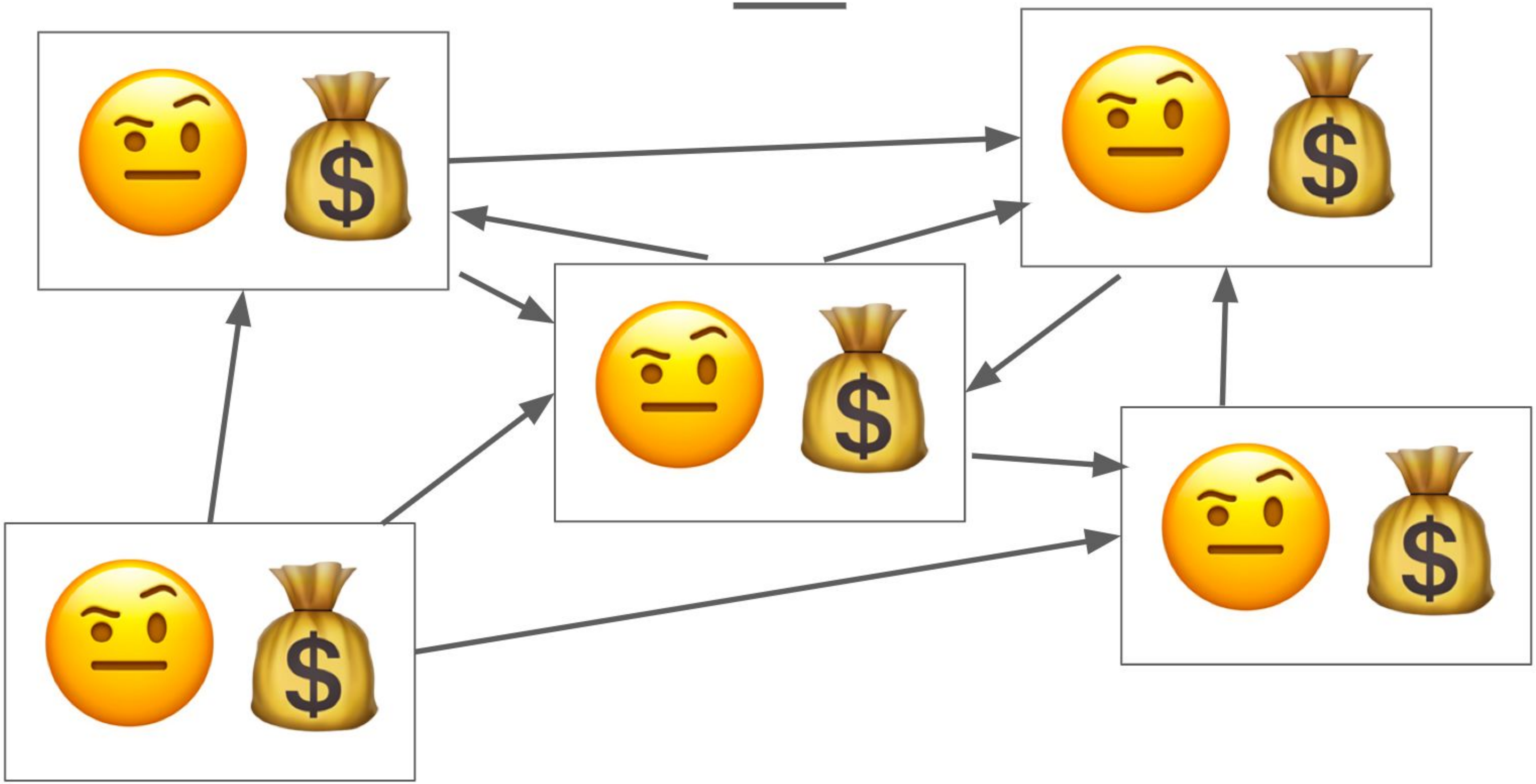
# Proof of Stake - Prevent Double Spend?



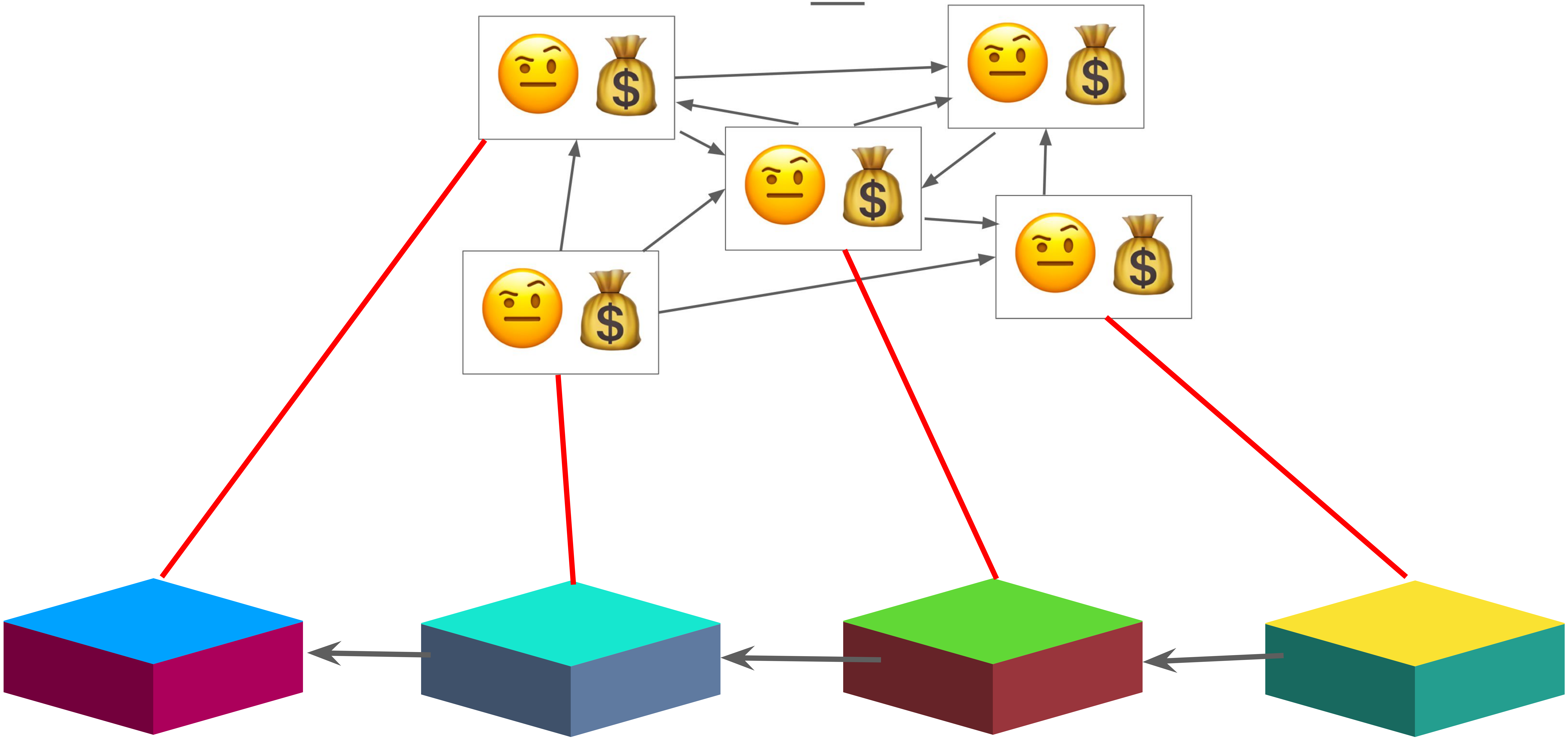
# Nothing At Stake Problem



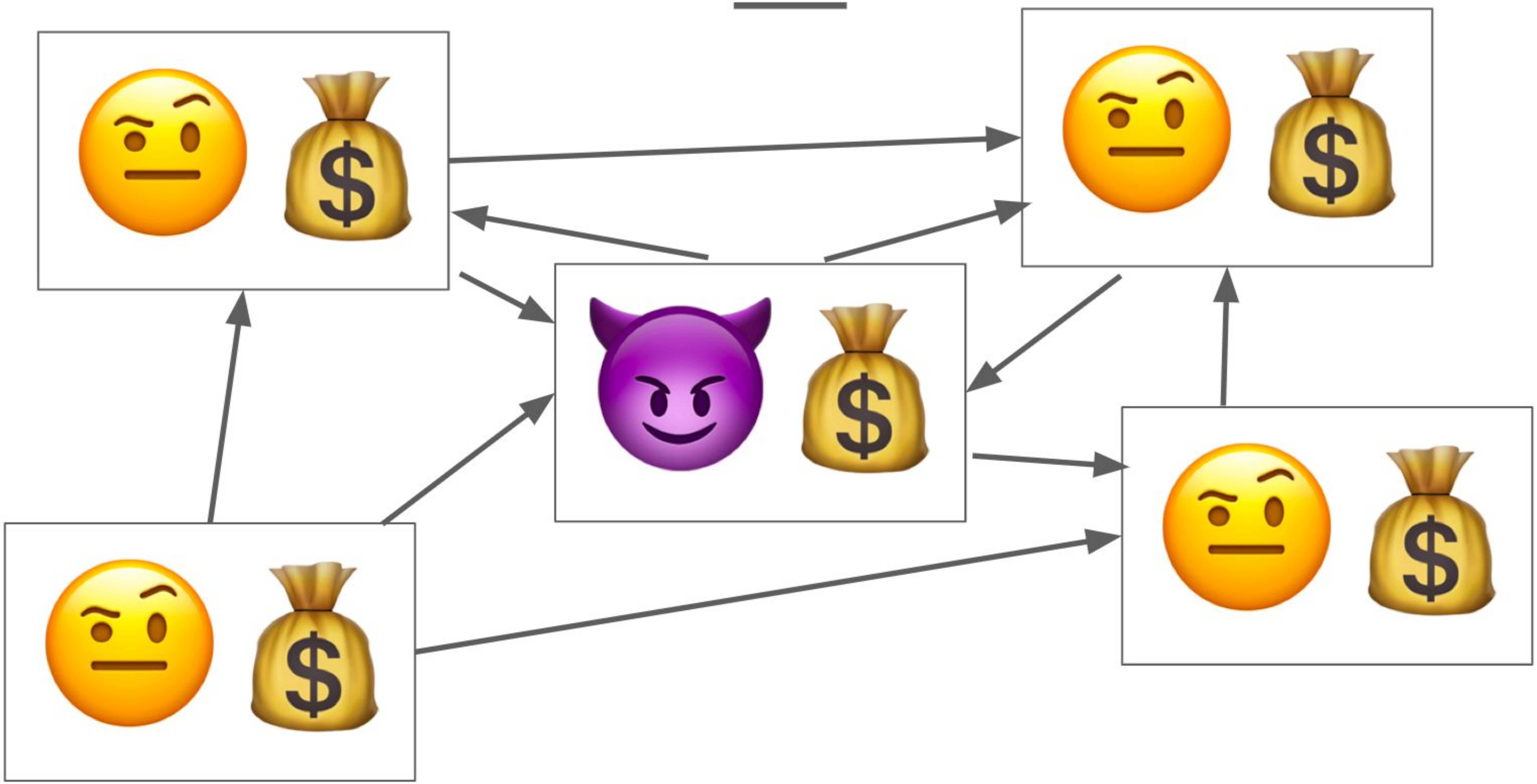
# Proof of Stake



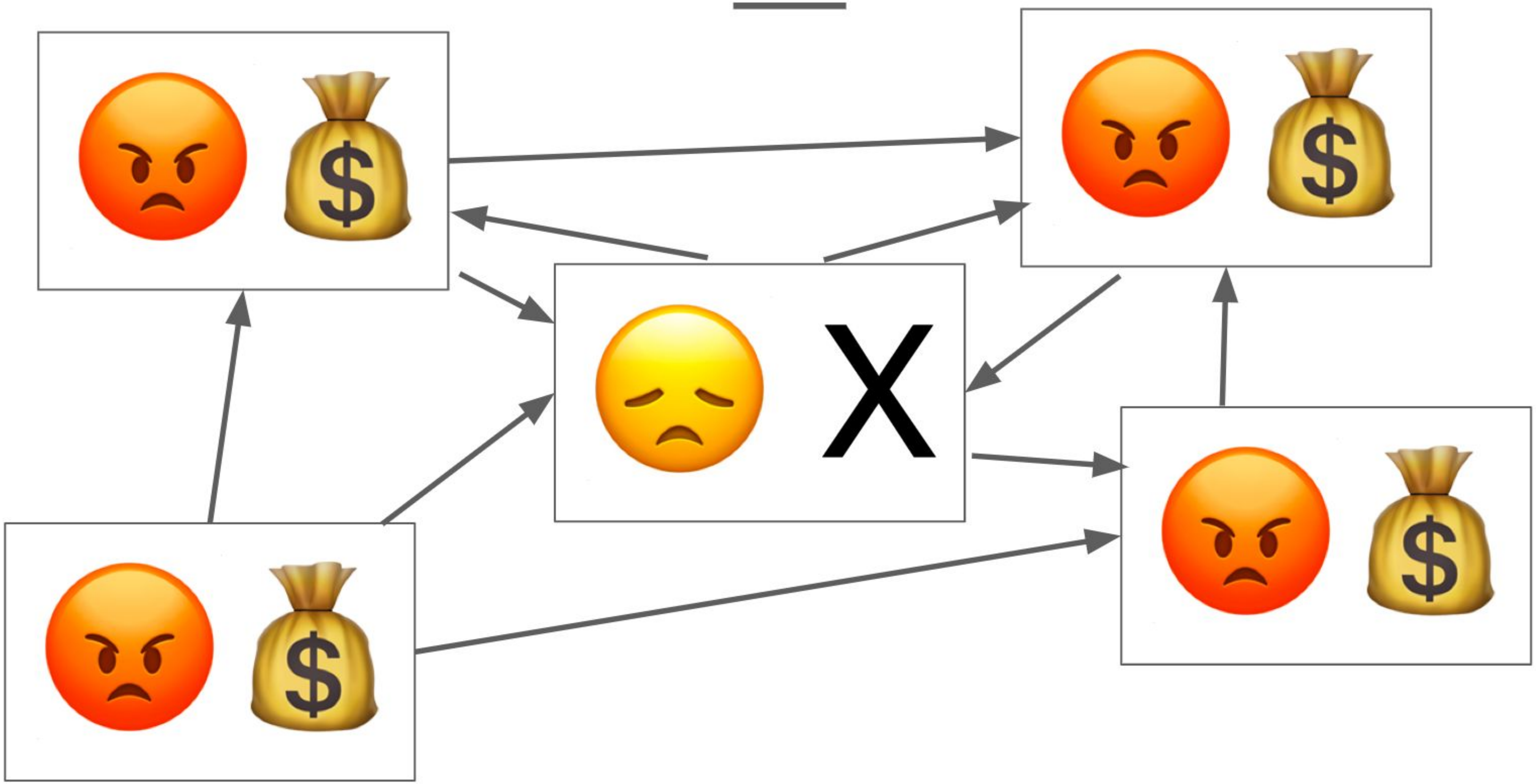
# Proof of Stake



# Bonding and Slashing



# Bonding and Slashing

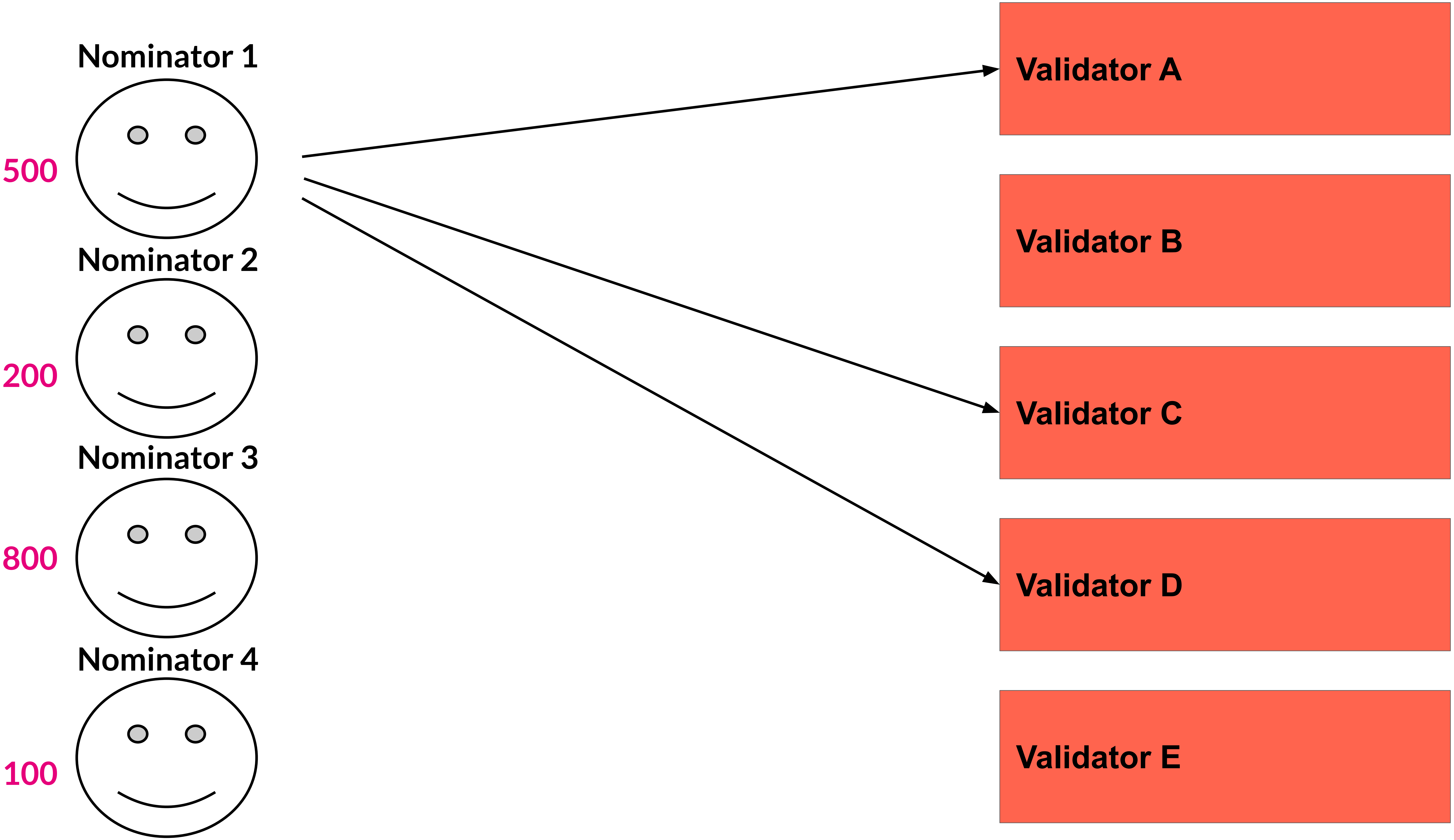


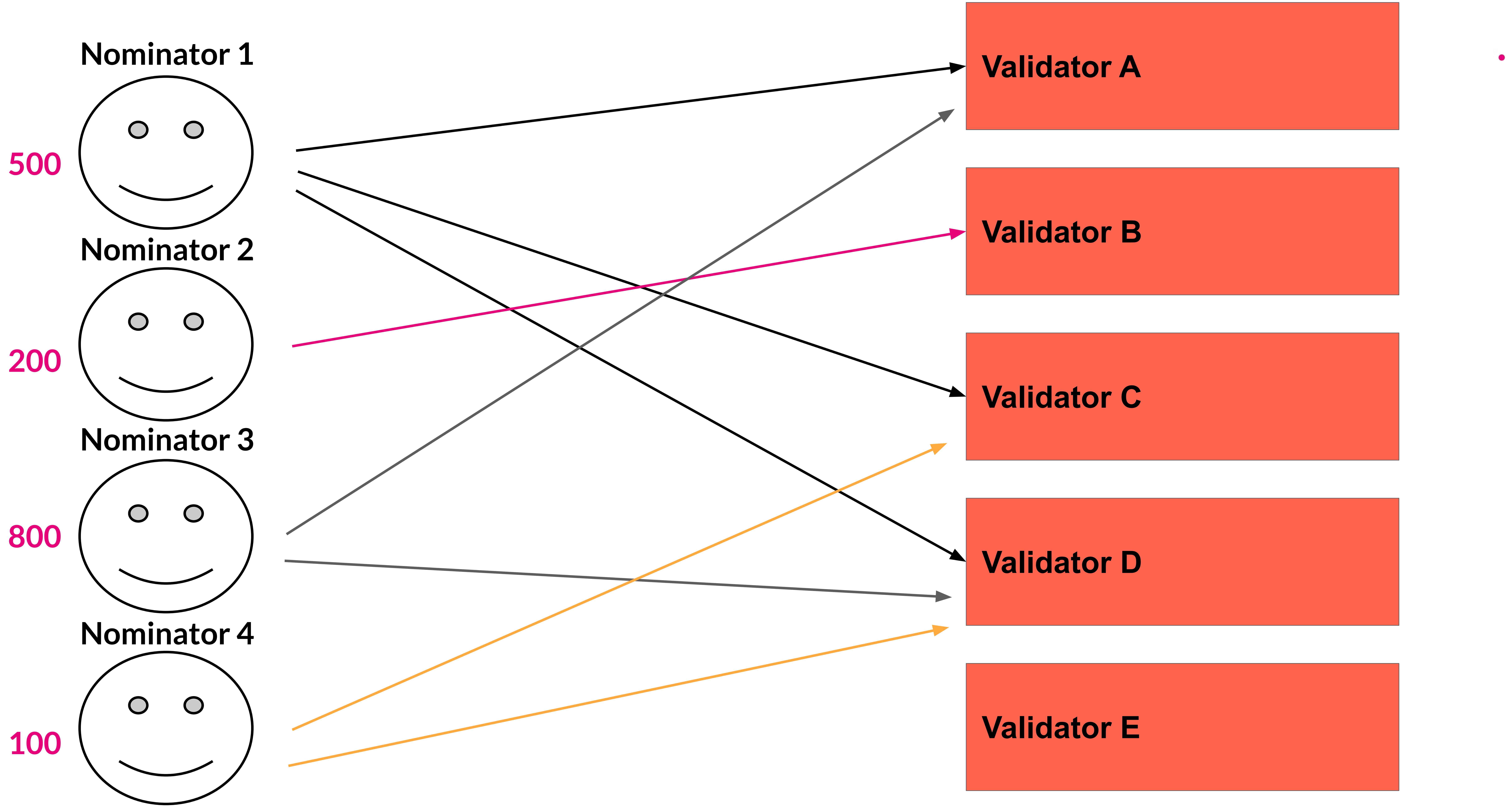
# Proof-of-Stake variations

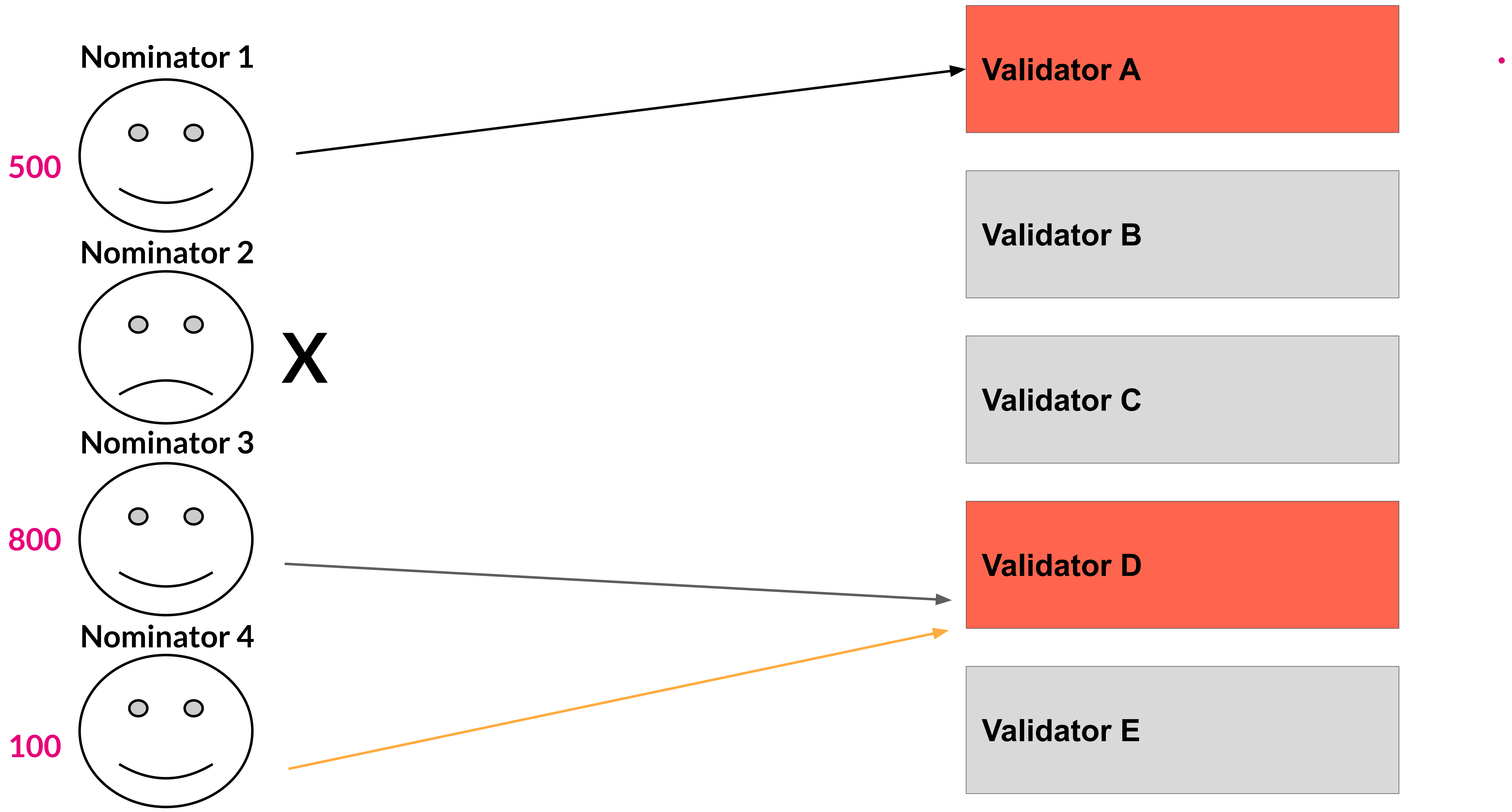
- Original Proof of Stake (e.g. Peercoin)
- Delegated Proof of Stake (e.g. Lisk)
- Leased Proof of Stake (e.g. Waves)
- Pure Proof of Stake (e.g. Algorand)
- Proof of Importance (e.g. NEM)
- Liquid Proof of Stake (e.g. Tezos)
- Proof of Validation (e.g. Cosmos)
- Hybrid Proof of Stake (e.g. Decred)
- Nominated Proof of Stake (Polkadot)

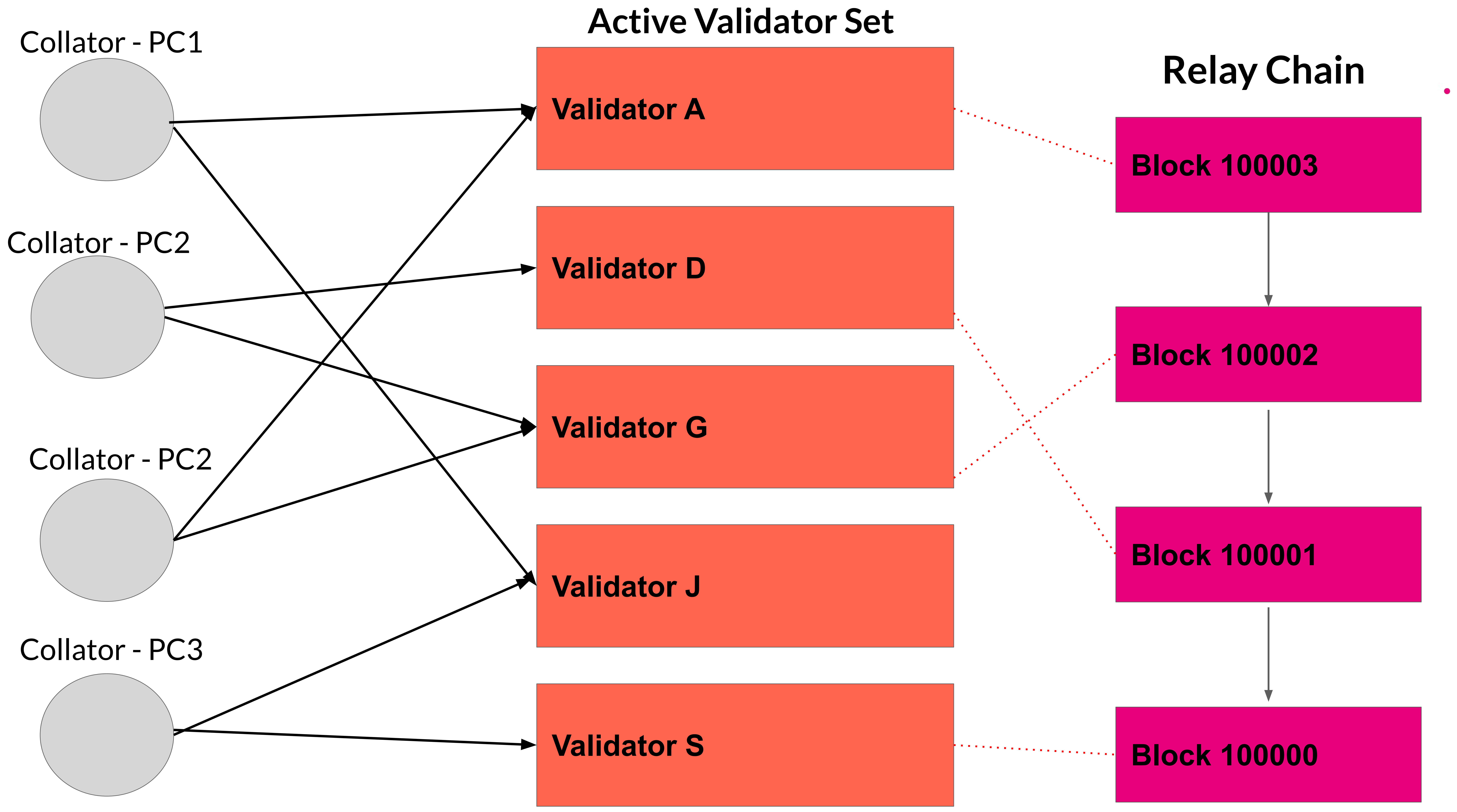
# NPoS

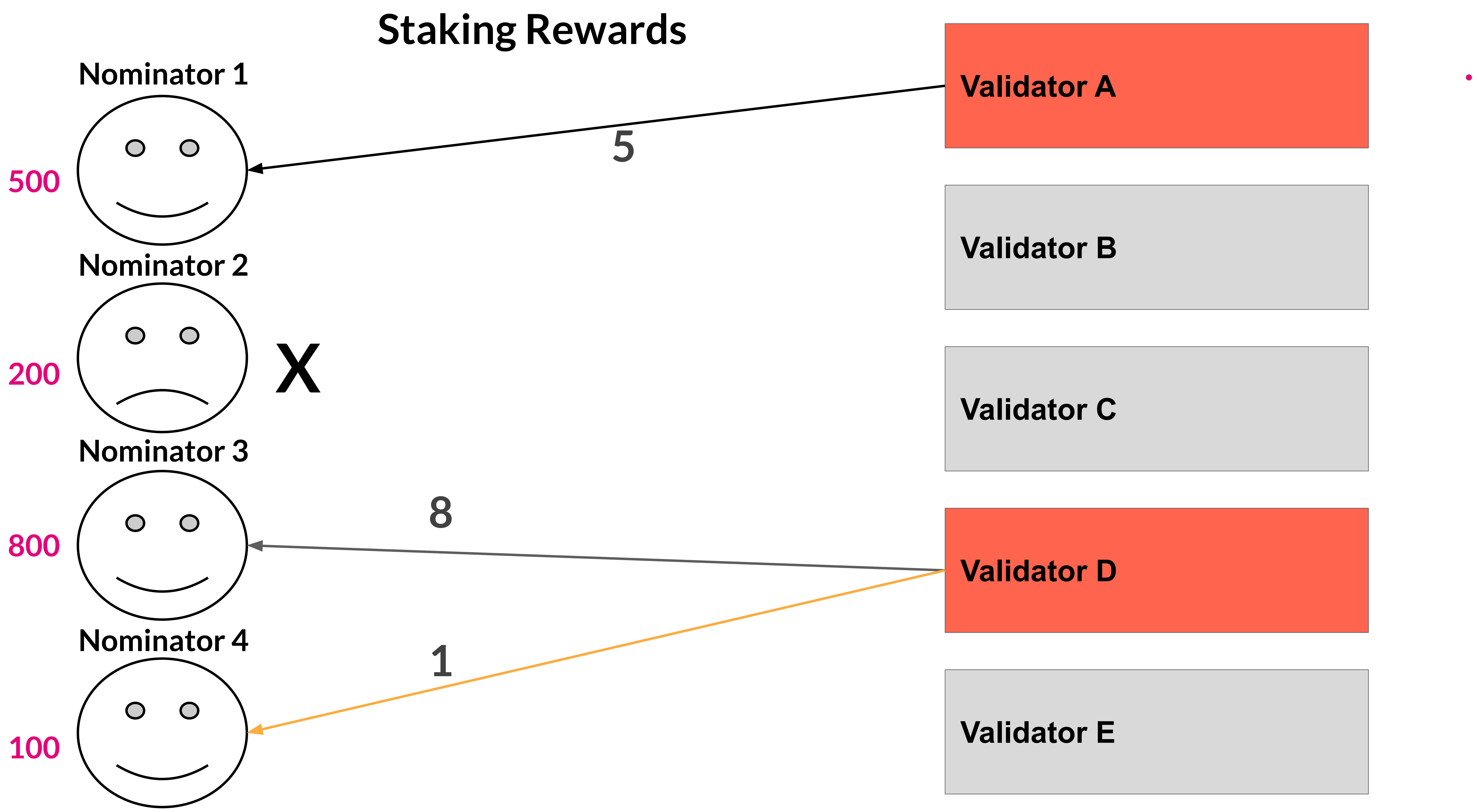
- Can validate with minimal own stake.
- Nominators can help choose validators in exchange for a percentage of rewards, but also punished for their validators’ infractions.
- All validators share essentially (not exactly) equal rewards, as long as they are in the active set
- Slashing increases as the number of validators engaged in the offense increase


Nominator 1
500
Nominator 2
200
Nominator 3
800
Nominator 4
100
Validator A
Validator B
Validator C
Validator D
Validator E


Nominator 1
500
Nominator 2
200
Nominator 3
800
Nominator 4
100
Validator A
Validator B
Validator C
Validator D
Validator E


Nominator 1
500
Nominator 2
X
Nominator 3
800
Nominator 4
100
Validator A
Validator B
Validator C
Validator D
Validator E


Active Validator Set
Collator - PC1
Collator - PC2
Collator - PC2
Collator - PC3
Validator A
Validator D
Validator G
Validator J
Validator S
Relay Chain
Block 100003
Block 100002
Block 100001
Block 100000


Staking Rewards
Nominator 1
500
Nominator 2
200
X
Nominator 3
800
Nominator 4
100
5
8
1
Validator A
Validator B
Validator C
Validator D
Validator E