Permisionless Consensus

“Anyone Can Join” Consensus

Rafael Pass

TAU, Cornell Tech

CPIIS - CHECK POINT INSTITUTE FOR INFORMATION SECURITY

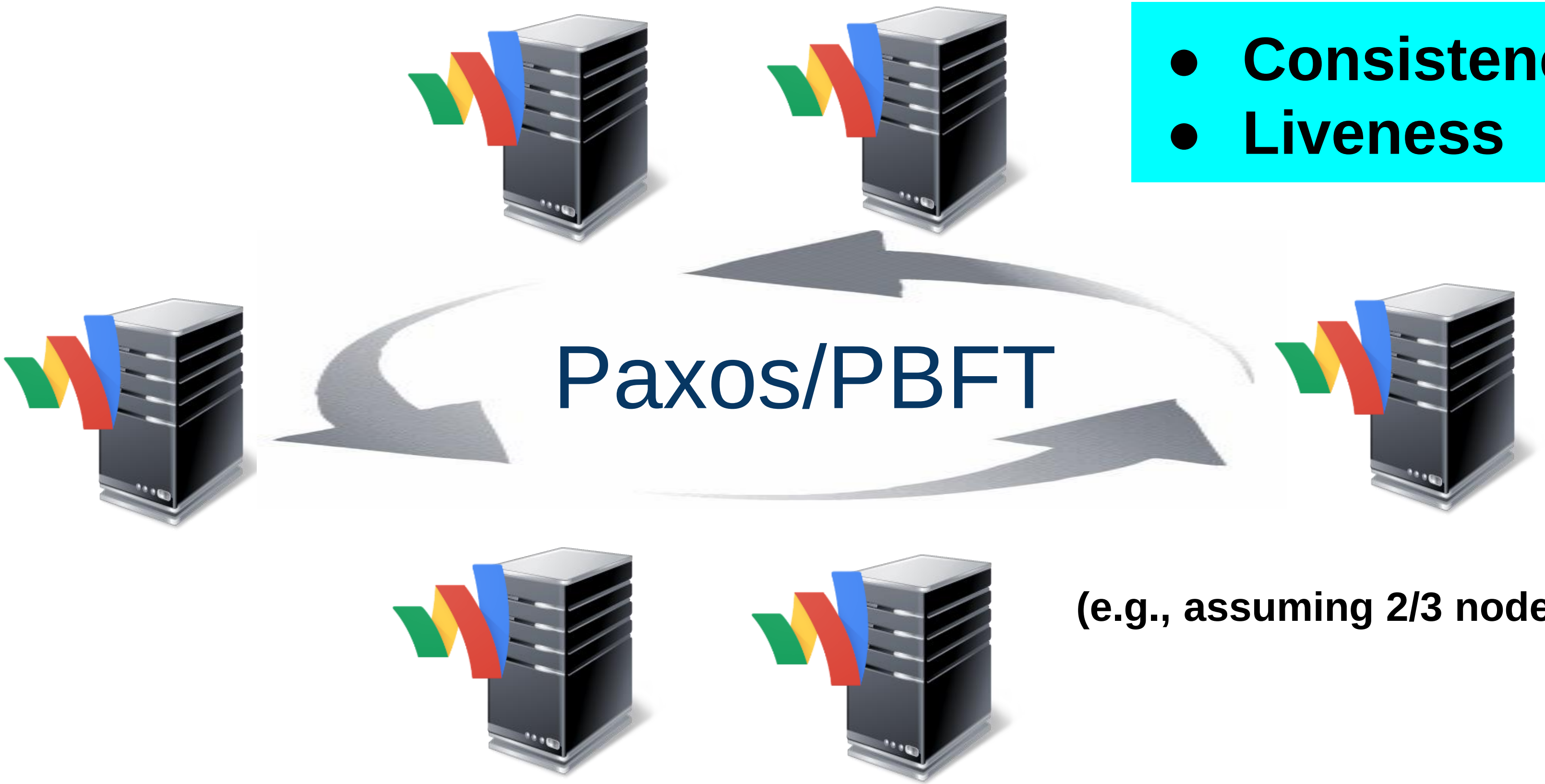
Blavatnik School of Computer Science

Tel Aviv University

Consensus

(a.k.a. state-machine replication, public ledger, permissioned blockchain)

- **Consistency**
- **Liveness**



(e.g., assuming 2/3 nodes honest)

Consistency:

At any point, my ledger is a prefix of yours or vice versa.

At any point, my ledger is a prefix of my **future ledger**.

Liveness:

There exists some polynomial **Confirm**, such that if any honest player sees a transaction, w.h.p. it will be added to everyone's ledger within time **Confirm**(Δ), where **Δ = max network delay**.

Synchronous model: Protocol may be parametrized by Δ

Partially synchronous model: Same protocol works for any Δ

Consistency + Liveness

=

Trusted Public Ledger

"a trusted party that maintains ledger"

(e.g., think of Facebook wall)

The Traditional “Permissioned” Model

- **number** and **identities** of nodes is common knowledge
- nodes **stick around** for the whole execution.
- authenticated channels/PKI

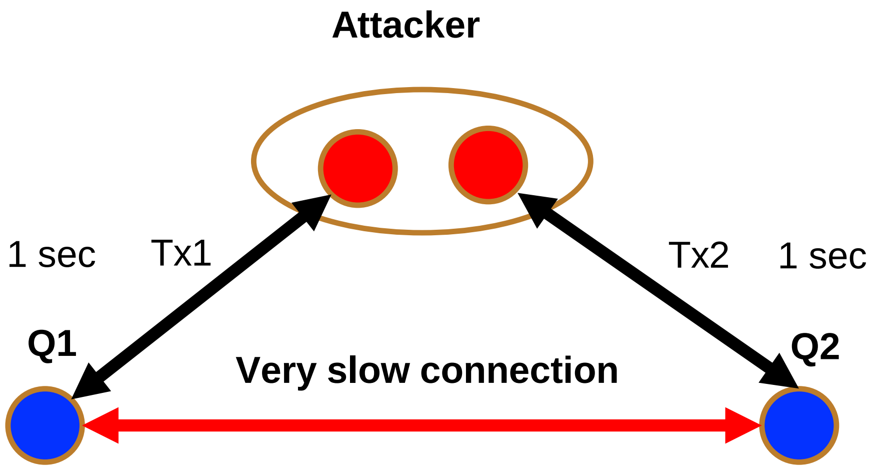
The Traditional "Permissioned" Model

- **number** and **identities** of nodes is common knowledge
- nodes **stick around** for the whole execution.
- authenticated channels/PKI

Thm: **In Sync:** possible **iff** 2/3 honest with **auth channels**.
possible **with just 1 honest**, in **PKI model** (+ OWF)

Thm: In Part-sync: possible **iff** 2/3 honest with **auth channels**
PKI doesn't help

Impossibility of 1/3 corruption with partial synchrony

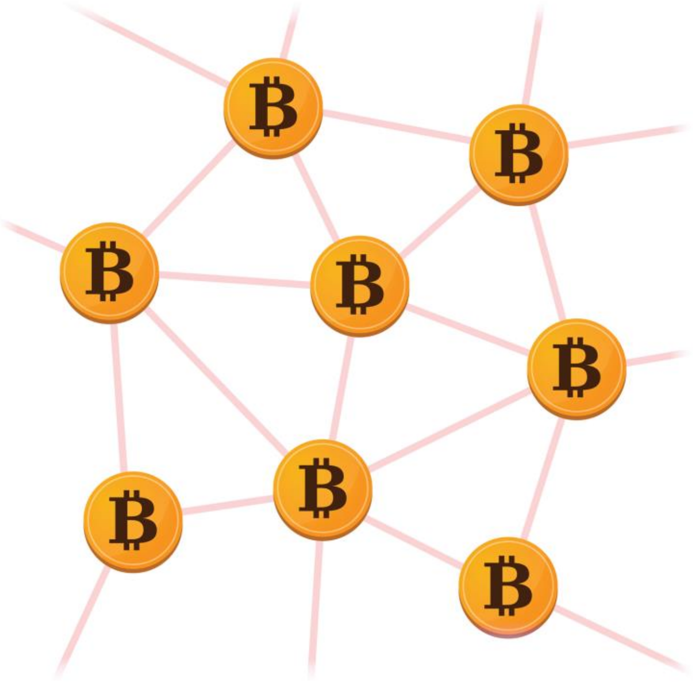


Must output Tx1 within Confirm(1 sec)
But not Tx2

Must output Tx2 within Confirm(1 sec)
But not Tx1

The “Permissionless” Model: Bitcoin/Blockchain

The Times 03/Jan/2009
Chancellor on brink of second bailout for banks.



The “Permissionless” Model

Axiom: Computation polylog(# nodes)

- Nodes **don’t know the exact # of nodes**
- Nodes come and go: **“late joining”**
- No authentication mechanisms: **“anyone can join”**
- **“economic robustness”**

The "Permissionless" Model

Axiom: Computation polylog(# nodes)

- Nodes **don't know the exact # of nodes**
- Nodes come and go: **"late joining"**
- No authentication mechanisms: **"anyone can join"**

We are still at the beginning of understanding even this model…

The Unauthenticated Model [BCLPR05]

Thm [BCLPR’05]: Consensus impossible without authentication in **partially synchronous** model.

Thm [PS’17]: Consensus impossible without authentication even in **synchronous** model.

Proof: **the “Sybil” attack**...but a bit delicate to formalize

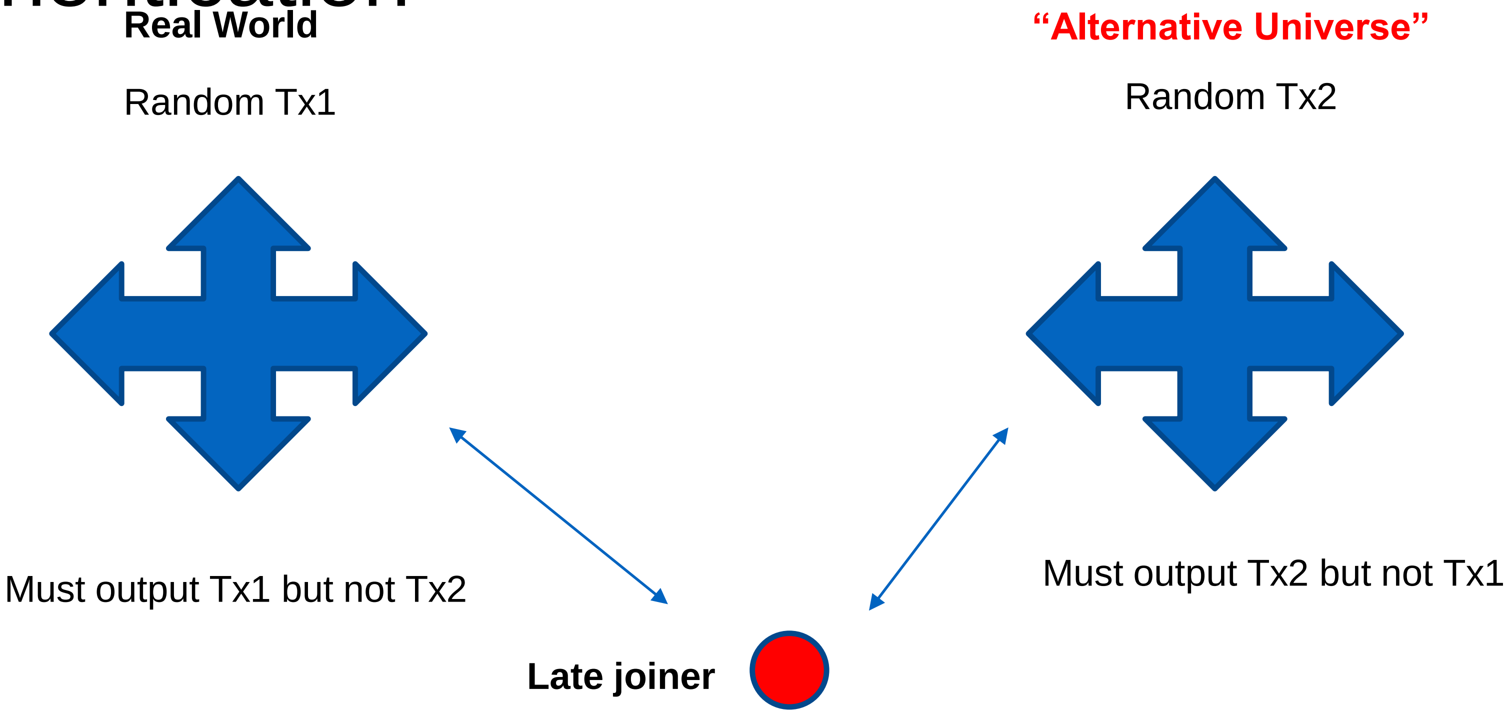
Impossibility of Consensus without Authentication

Real protocol running with a random Tx1

Alternative Universe: Attacker "honestly" runs a different execution with a random Tx2

Which transaction should a **late joiner** output?

Impossibility of Consensus without Authentication



Must agree with Real by security.

But also with attacker produced alternative universe

Nakamoto’s Blockchain [Nak’08]

Prevents Sybil attacks with Proofs-of-Work Puzzles [DN’92]

Claims protocol achieves “public ledger” assuming **“honest majority of computing power”:**

- **Consistency**: everyone sees the same history
- **Liveness**: everyone can add new transactions

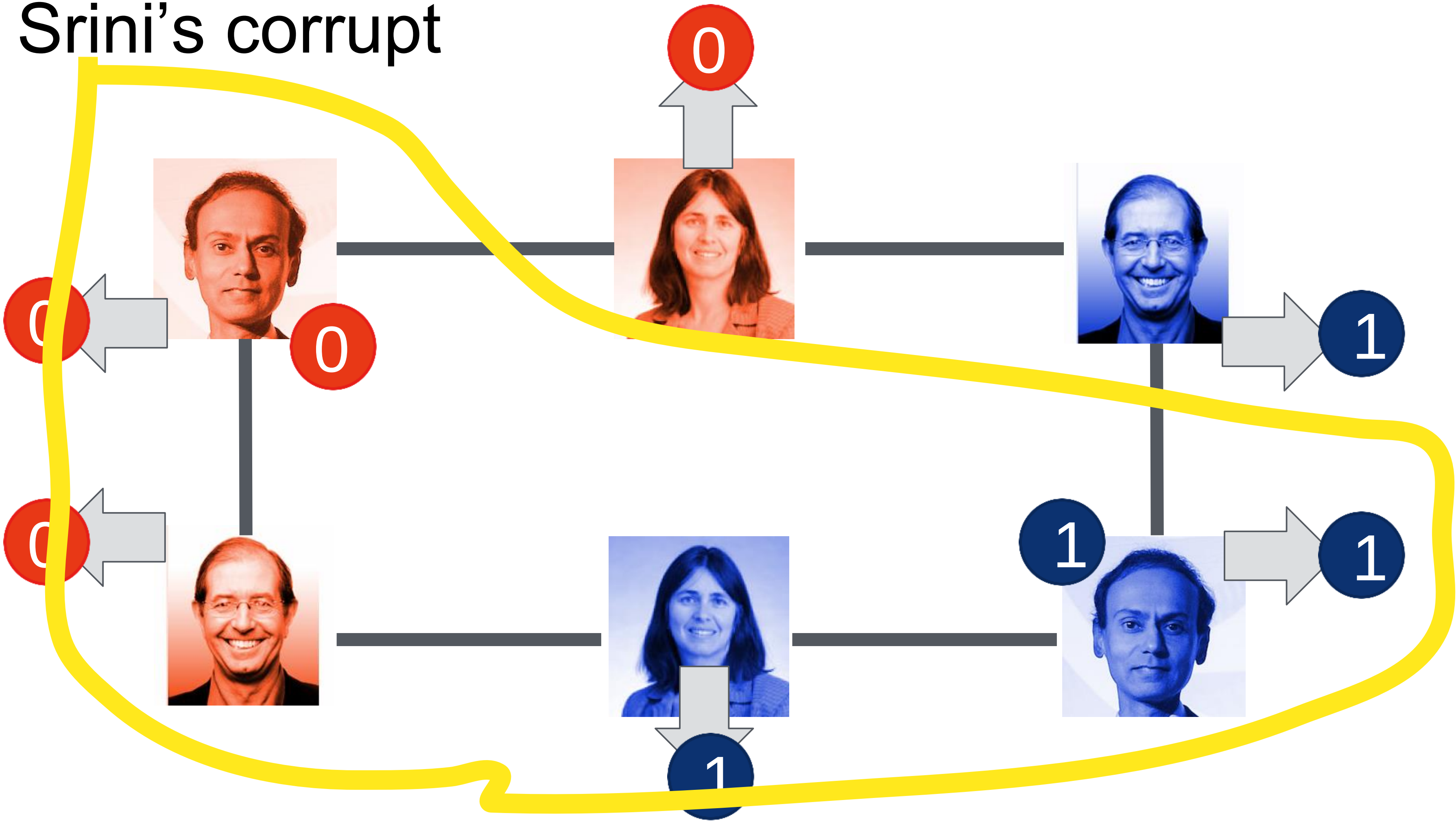
Nakamoto's Blockchain [Nak'08]

Prevents Sybil attacks with Proofs-of-Work Puzzles [DN'92]

2 amazing aspects:

- Overcomes "unauthetication barrier"
- Overcomes ⅓ barrier even in permissioned setting

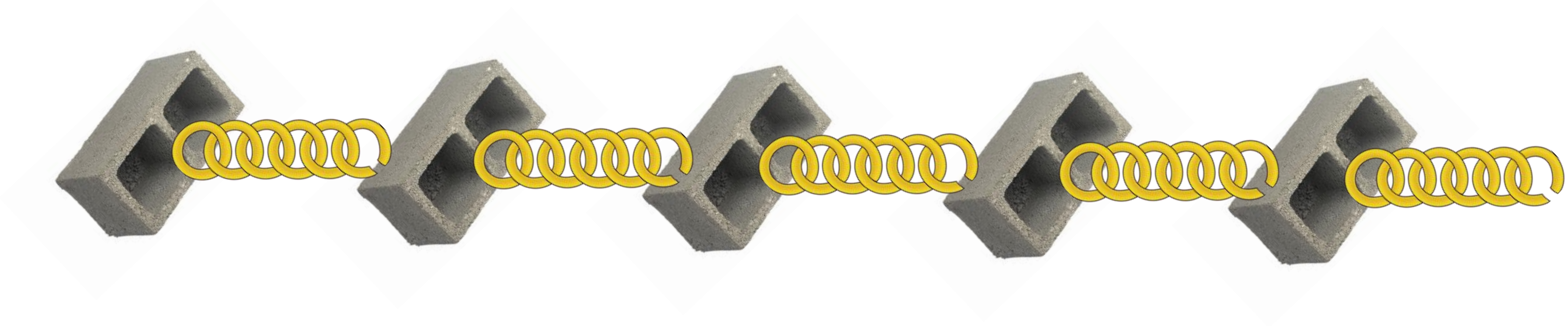
Srini's corrupt

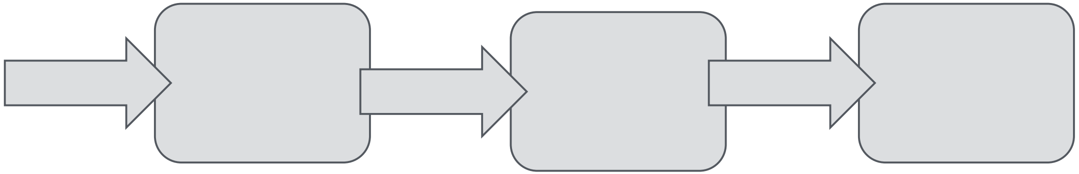


- An **abstract** notion of a blockchain;
  - *how it compares to "consensus";*
  - *why a new definition? (hint: incentives)*

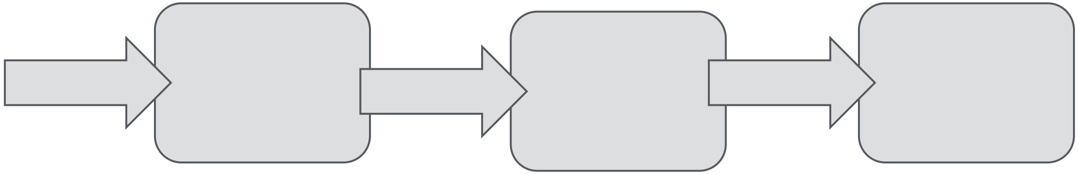
- Does Nakamoto's protocol achieve **CONSISTENCY**?
  - Classes of attacks don't work [N'08,**GKL'15**, SZ'15]
  - 49.1% attack (with 10s network delays) claimed [DW'14]

What is a blockchain?



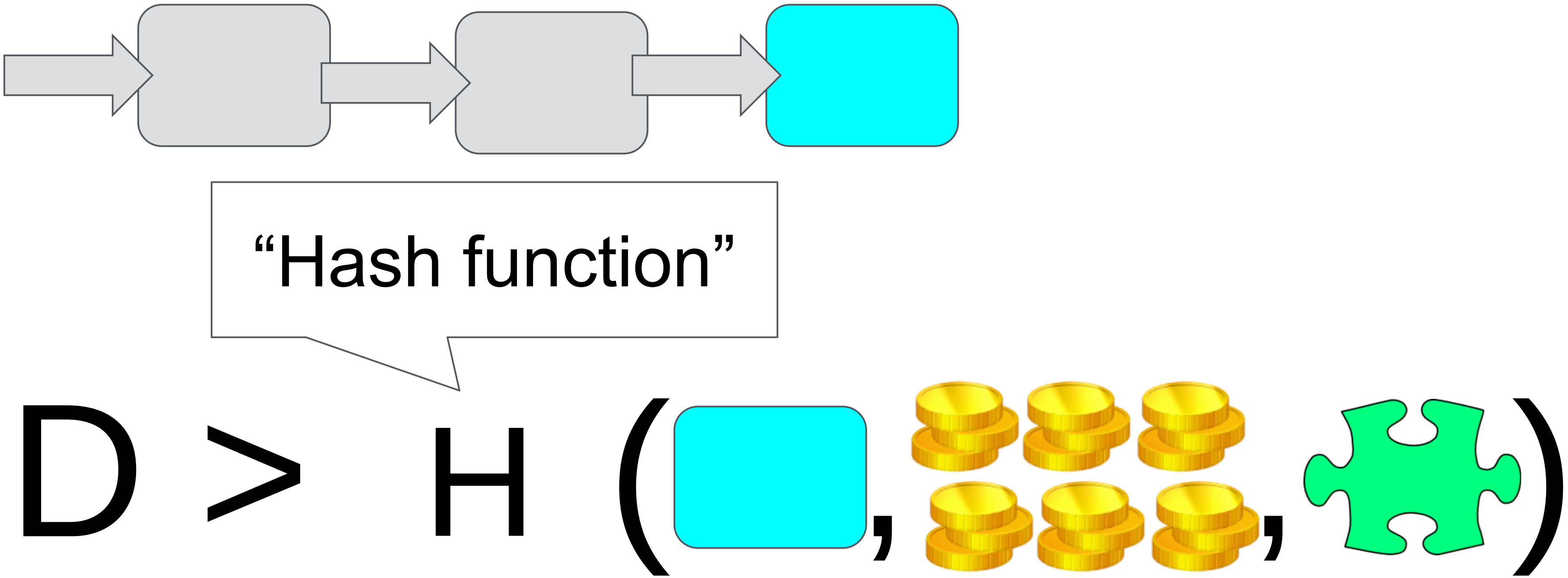


How to build a “blockchain”

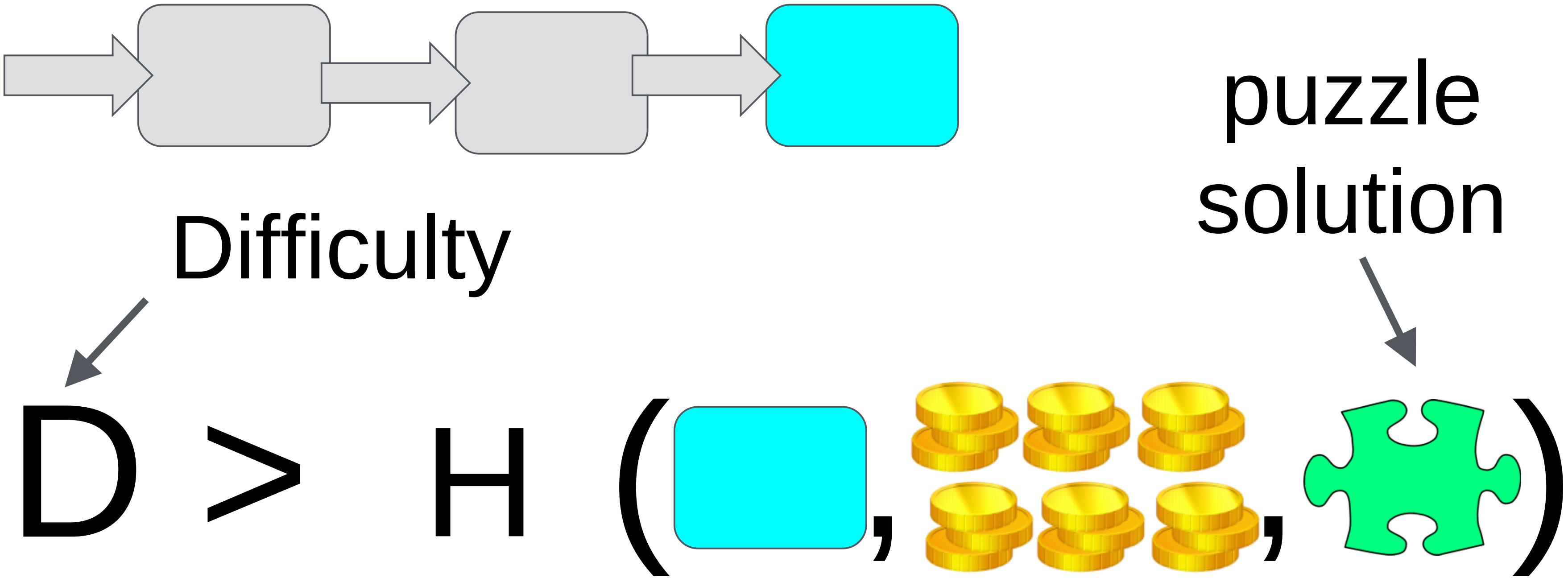


Eli➔ Dan: Ƀ10

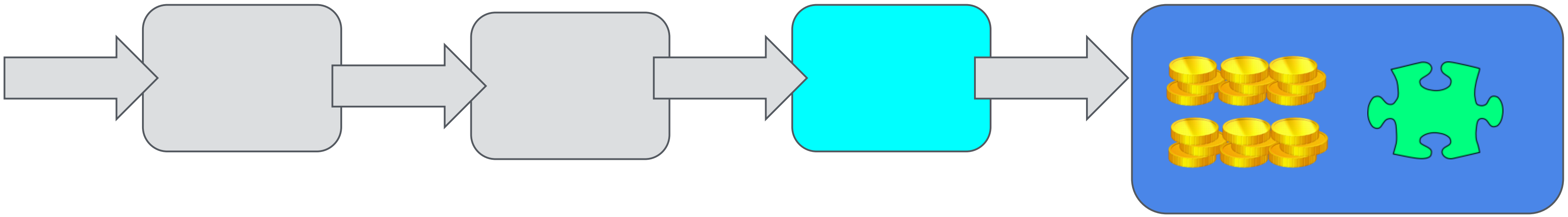
How to build a “blockchain”

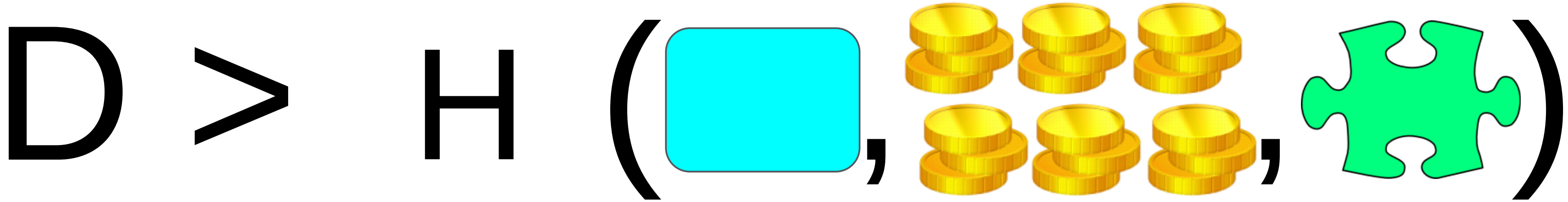


How to build a "blockchain"

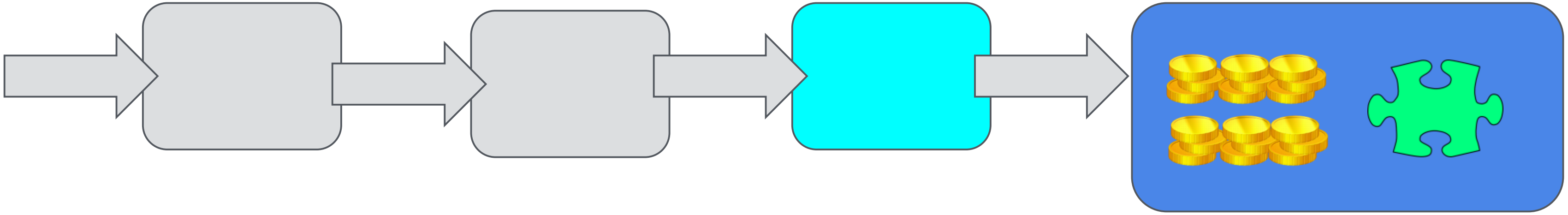


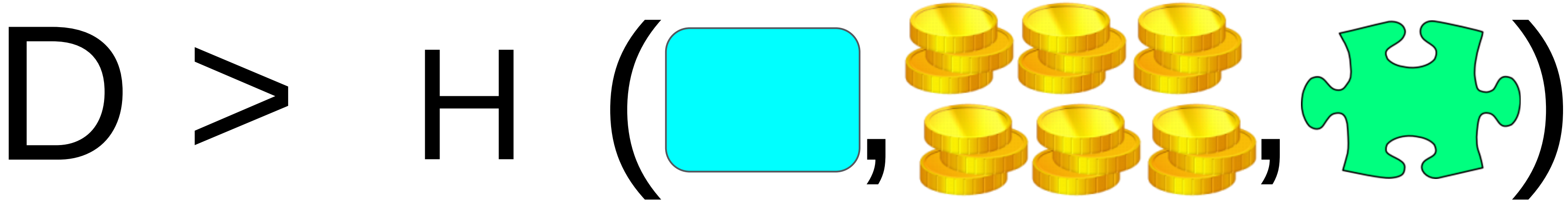
Search for a puzzle solution



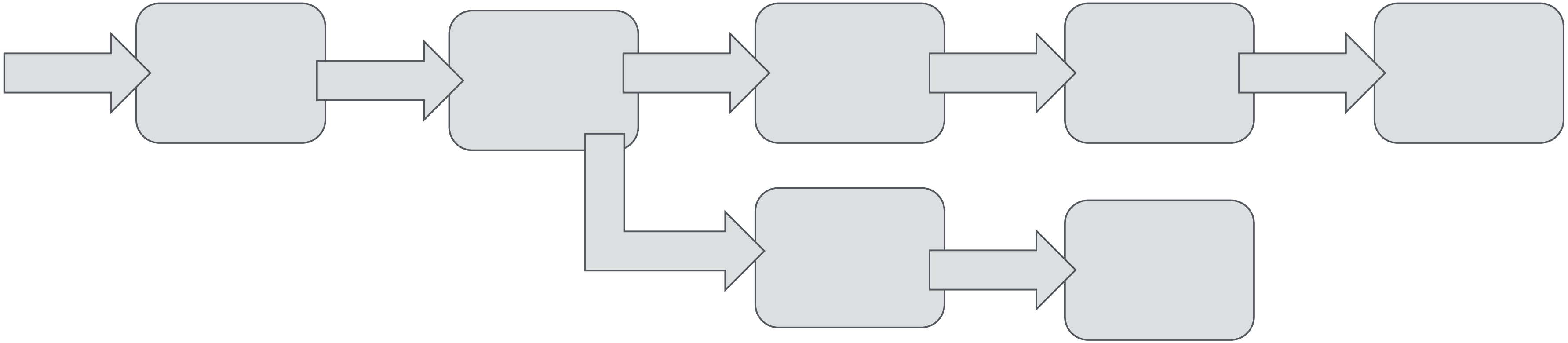


We found a new block



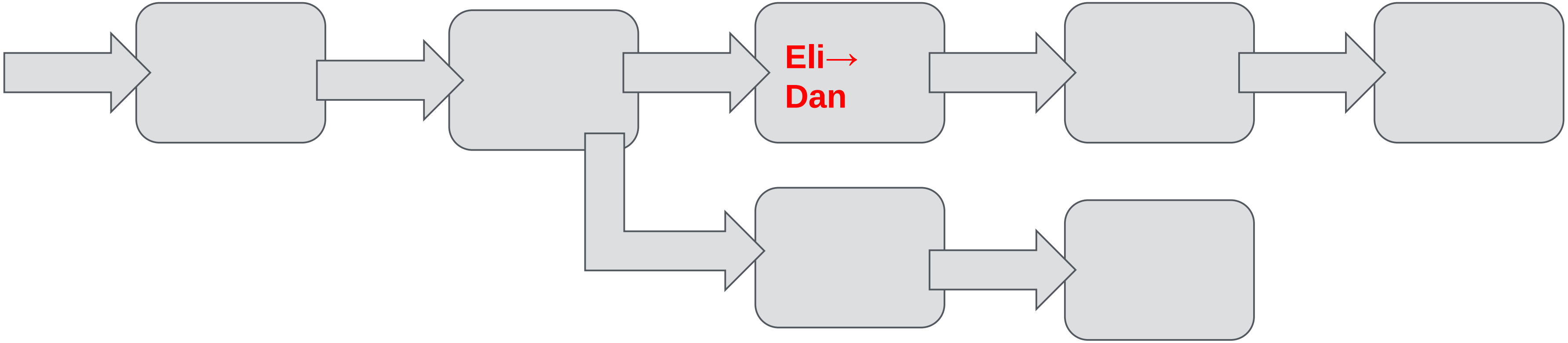


Best way to find a solution is brute-force search: model H as RO

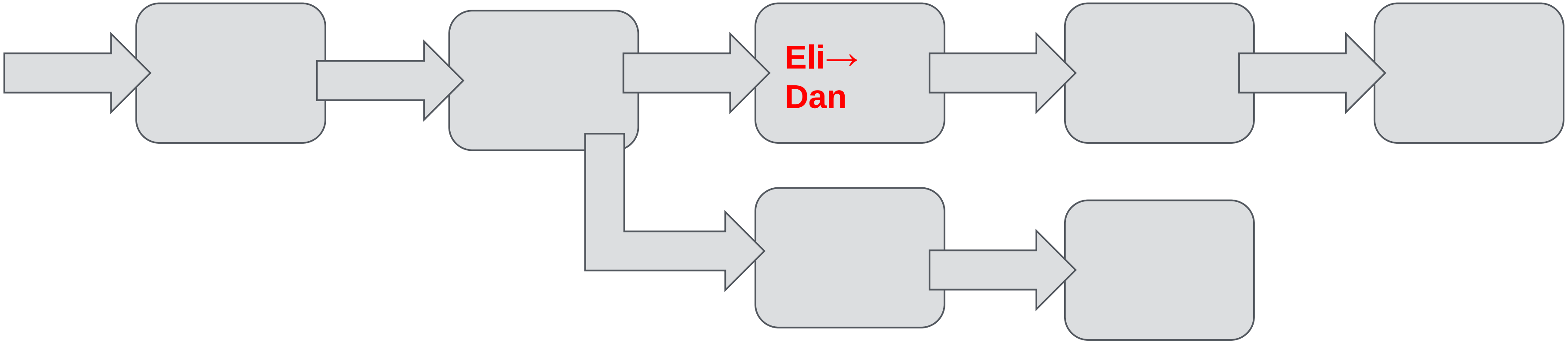


Honest nodes only “believe”

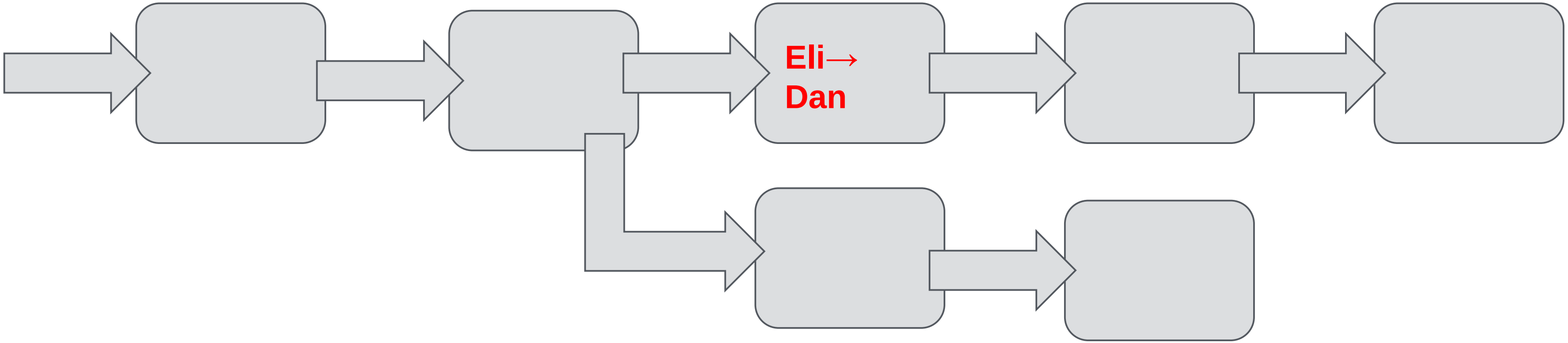
longest chain



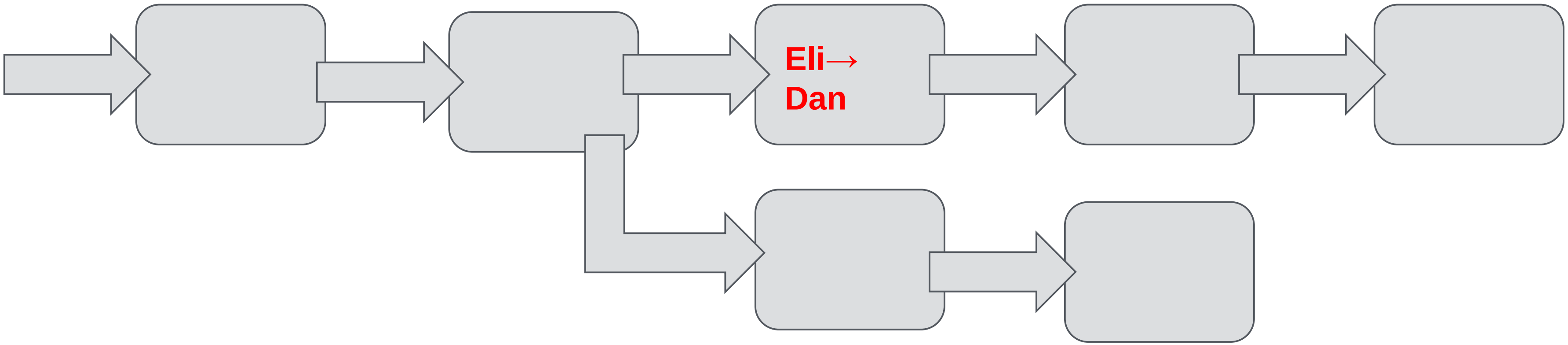
Eli wants to erase this transaction



For Eli to erase his transaction, he has to find a longer chain



“If transaction is sufficiently deep, he cannot do this unless he has majority hashpower”



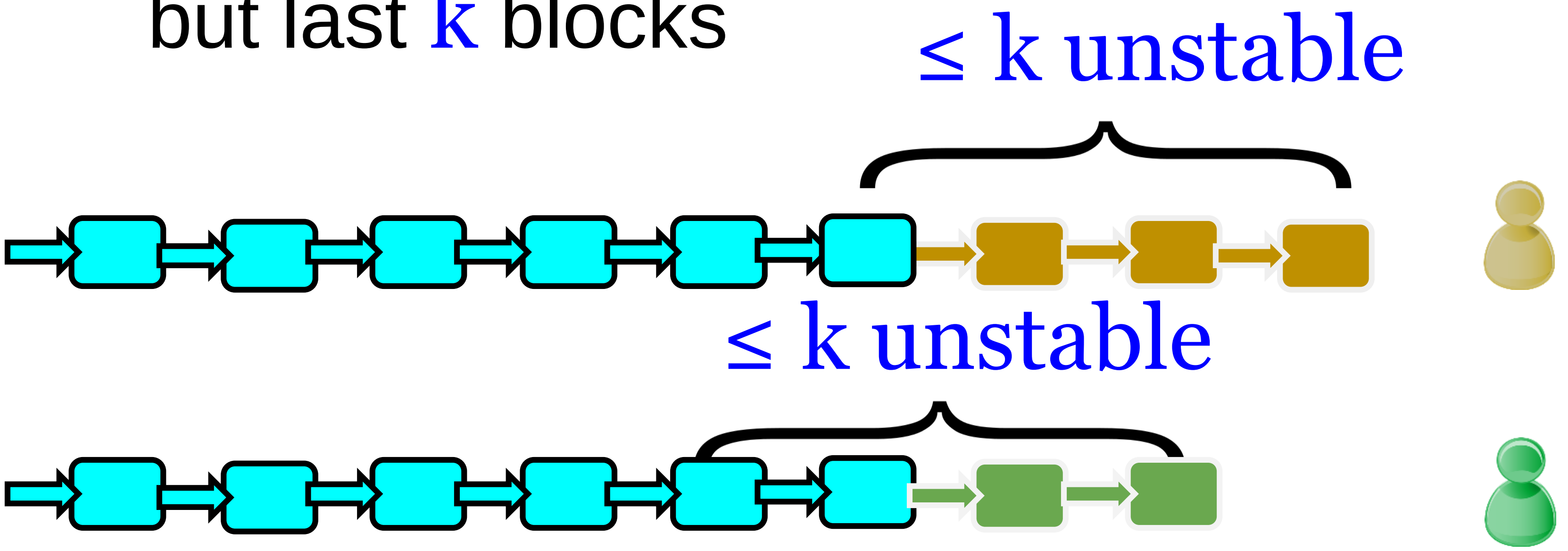
“If transaction is sufficiently deep, he cannot do this unless he has majority hashpower”

- [Nak’08]: “trying to mine alternative chain fails”
- [**GKL’15**]: **no attack**, as long as Δ = 1
- [SZ’15]: “non-withholding attacks” fail also with Δ-delays

Blockchain abstraction (a la GKL,SZ,KL,PSS)

w/ prob exp(-k)

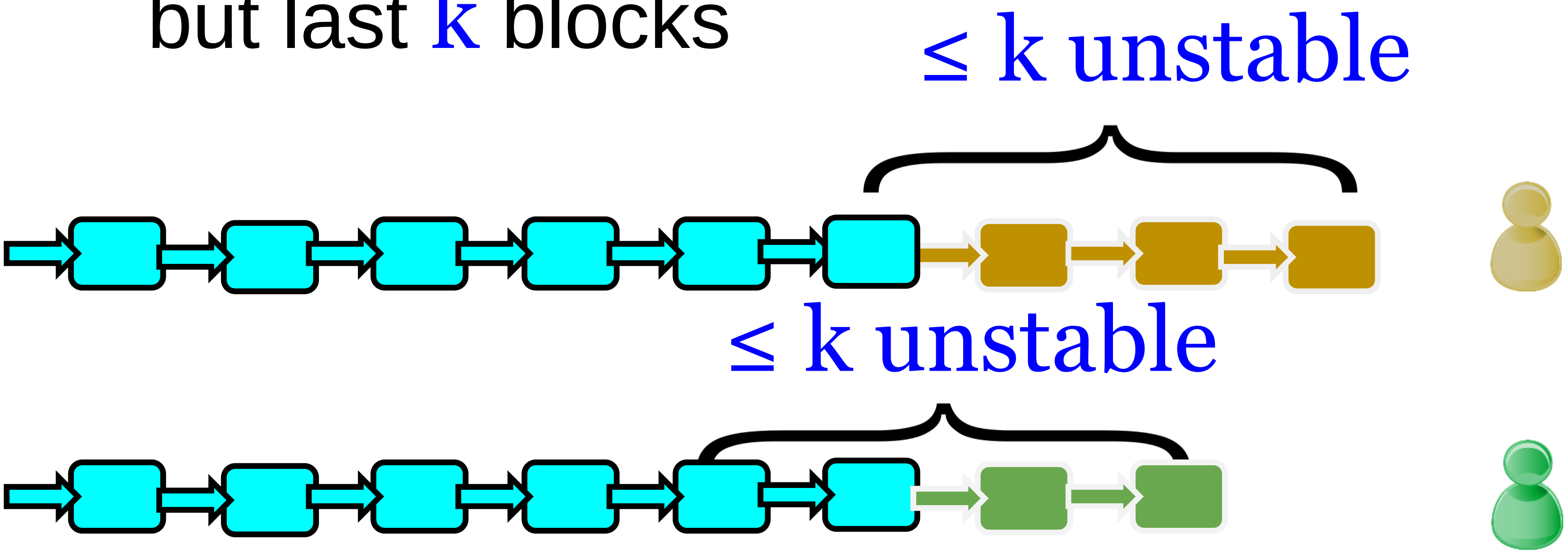
1 Consistency: Honest nodes agree on all but last k blocks



Blockchain abst… w/ prob $exp(-k)$

Future-self consistency

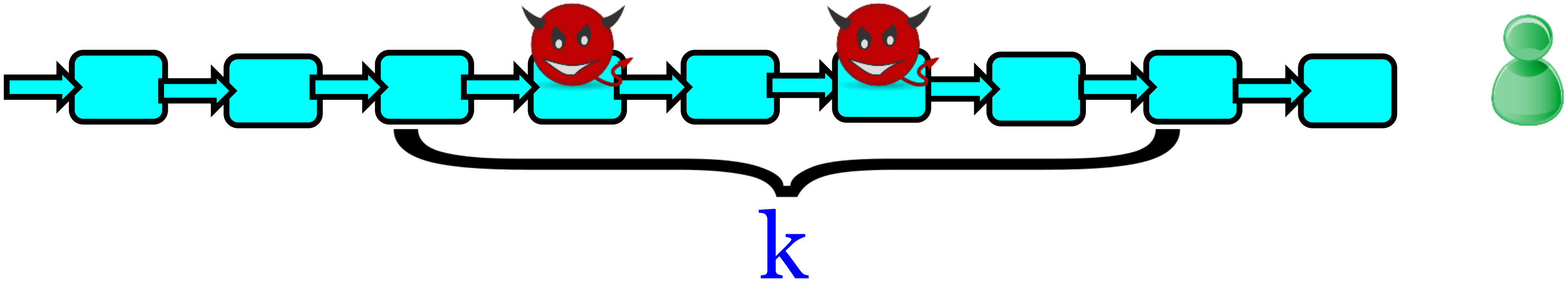
1. Consistency: Honest nodes agree on all but last k blocks



Blockchain abstraction

w/ prob exp(-k)

1. Consistency: Honest nodes agree on all but last k blocks
2. Chain quality: Any consecutive k blocks contain "sufficiently many" honest blocks



Blockchain abstraction

w/ prob exp(-k)

1. Consistency: Honest nodes agree on all but last k blocks
2. Chain quality: Any consecutive k blocks contain “sufficiently many” honest blocks
3. Chain growth: Chain grows at a steady rate

Blockchain implies "state machine replication" in the permissionless model

1. Consistency
2. Chain quality
3. Chain growth

Traditional
"state machine replication"

1. Consistency
2. Liveness

Theorem [PSS'16]:

For every $\rho<1/2$, if "mining difficulty" is appropriately set (as a function of the network delay Δ, and total mining power), Nakamoto's blockchain guarantees:

- Consistency
- Chain quality: $1 - \rho/(1-\rho)$
- Chain growth: $O(1/\Delta)$

where ρ adv's fraction of hashpower, and **adv controls the network**

Theorem [PSS'16]:

For every $\rho<1/3$, if "mining difficulty" is appropriately set (as a function of the network delay Δ, and total mining power), Nakamoto's blockchain guarantees:

- Consistency
- Chain quality: $1 - (1/3)/(2/3) = 1/2$
- Chain growth: $O(1/\Delta)$

where ρ adv's fraction of hashpower, and **adv controls the network**

Theorem [PSS'16]:

For every $\rho<1/2$, if "mining difficulty" is appropriately set (as a function of the network delay Δ, and total mining power), Nakamoto's blockchain guarantees:

- Consistency
- Chain quality: $1 - \rho/(1-\rho)$
- Chain growth: $O(1/\Delta)$

where ρ adv's fraction of hashpower, and **adv controls the network**

Theorem [PSS'16]:

For every $\rho<1/2$, if "mining difficulty" is appropriately set (as a function of the network delay Δ, and total mining power), Nakamoto's blockchain guarantees:

- Consistency
- Chain quality: $1 - \rho/(1-\rho)$
- Chain growth: $O(1/\Delta)$

"Blocks are found SLOWER than Δ"

where ρ adv's fraction of hashpower, and **adv controls the network**

Theorem [PSS'16]:

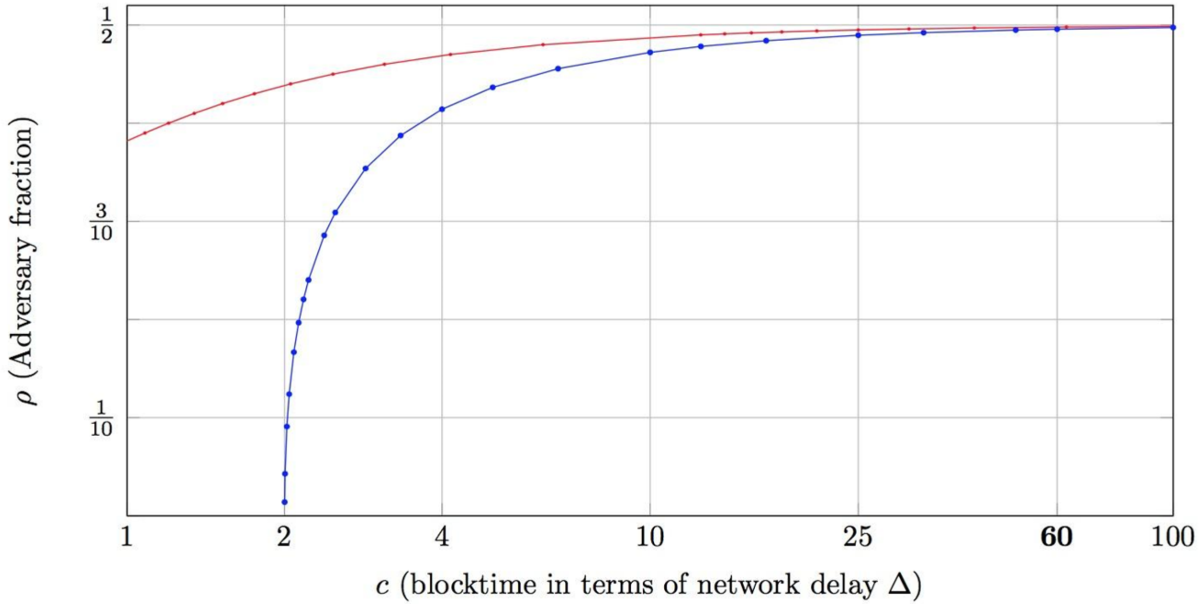
For every $\rho<1/2$, if "mining difficulty" is appropriately set (as a function of the network delay Δ, and total mining power), Nakamoto's blockchain guarantees:

- Consistency
- Chain quality: $1 - \rho/(1-\rho)$
- Chain growth: $O(1/\Delta)$

"Blocktime" >> Δ

where ρ adv's fraction of hashpower, and **adv controls the network**

“Appropriately set”



When c = 60 (10 min blocktime, 10s network delays)

Secure: ρ < 49.57

Attack: ρ > 49.79

“Appropriately set”

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

Mining rate of honest players

Network Delay

Mining rate of Adv

Proof Intuition:

Attack: When honest node mines a block, delay it by Δ.
Gives attacker Δ “free time”.
If **Blocktime** = cΔ, average advantage is 1/c

Proof Overview:

1. Replace RO with ideal **F_mine** func.

2. **Identity a “good pattern” for honest nodes.**
Convergence opportunity: “silence” for Δ time, a **single** guy mines, then “silence” again for Δ time.

3: Use **convergence opportunity growth rate** to argue chain growth and consistency; chain quality follows as easy consequence

Convergence opportunity: "silence" for Δ, a **single** guy mines, then "silence" for Δ

Chain growth: whenever we have a convergence opportunity =>
ALL honest guys' chains increase by 1!

Convergence opportunity: "silence" for Δ, a **single** guy mines, then "silence" for Δ

Chain growth: whenever we have a convergence opportunity => ALL honest guys' chains increase by 1!

Consistency: whenever we have a convergence opportunity **for length l**, unless attacker can mine a block for **length l**, the honestly mined block at **length l** can never be changed.

in fact, to ruin convergence, attacker must mine a block for length l, **close to the time** of the conv opportunity.

so, as long as # conv opps in any "long" interval >> # adv blocks in a "slightly longer" interval, we are guaranteed convergence in that interval.

Convergence opportunity: "silence" for Δ, a **single** guy mines, then "silence" for Δ

How to analyze convergence opportunity growth:

Easy! This is just a markov chain, lets use concentration bounds for markov chains…

[PSS'17]:

- Use concentration to bound # of successful mines.
- Look at **distances between successful mines.**.
- On average, they should be longer that Δ
- Use concentration to bound the number of short distances.
- **Each such short distance, can ruin at most 2 successful mines.**

Today: better bounds now know when c is small [LRS'18] [DKT'19][Ren'20] by directly analyzing Markov chain

Theorem [Security of Nakamoto]

For every $\rho<1/2$, if mining difficulty is appropriately set (as a function of the network delay, and total mining power), Nakamoto's blockchain guarantees a) consistency, b) chain quality $1 - \rho/(1-\rho)$, and c) Chain growth: $O(1/\Delta)$

Theorem [Blatant attack]:

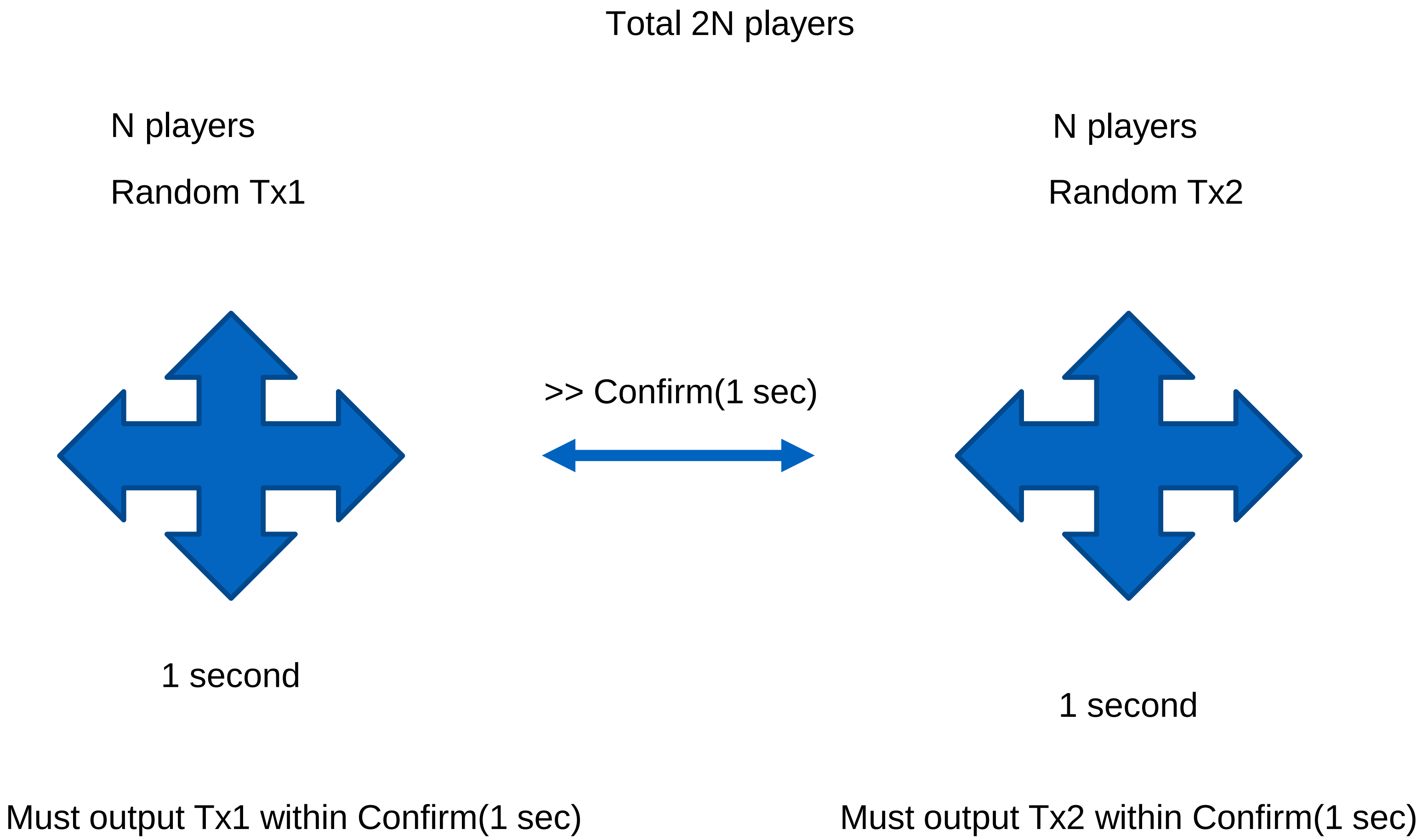
For every $\rho>0$, for every mining difficulty, there exists a network delay such that Nakamoto's blockchain is inconsistent and has 0 chain quality

Theorem [Security of Nakamoto]

In the ROM, assuming attacker controls < 0.49 fraction of computational resources, there exists a **synchronous** state-machine replication protocol.

Theorem [Impossibility of partially-sync]:

Even with POW, there does not exist a **partially synchronous** state-machine replication protocol **if players only know a 2 approx of the # of nodes**, even if assuming attacker controls less than < .0001 of the computational resources.


Total 2N players
N players
Random Tx1
N players
Random Tx2
>> Confirm(1 sec)
1 second
1 second
Must output Tx1 within Confirm(1 sec)
Must output Tx2 within Confirm(1 sec)

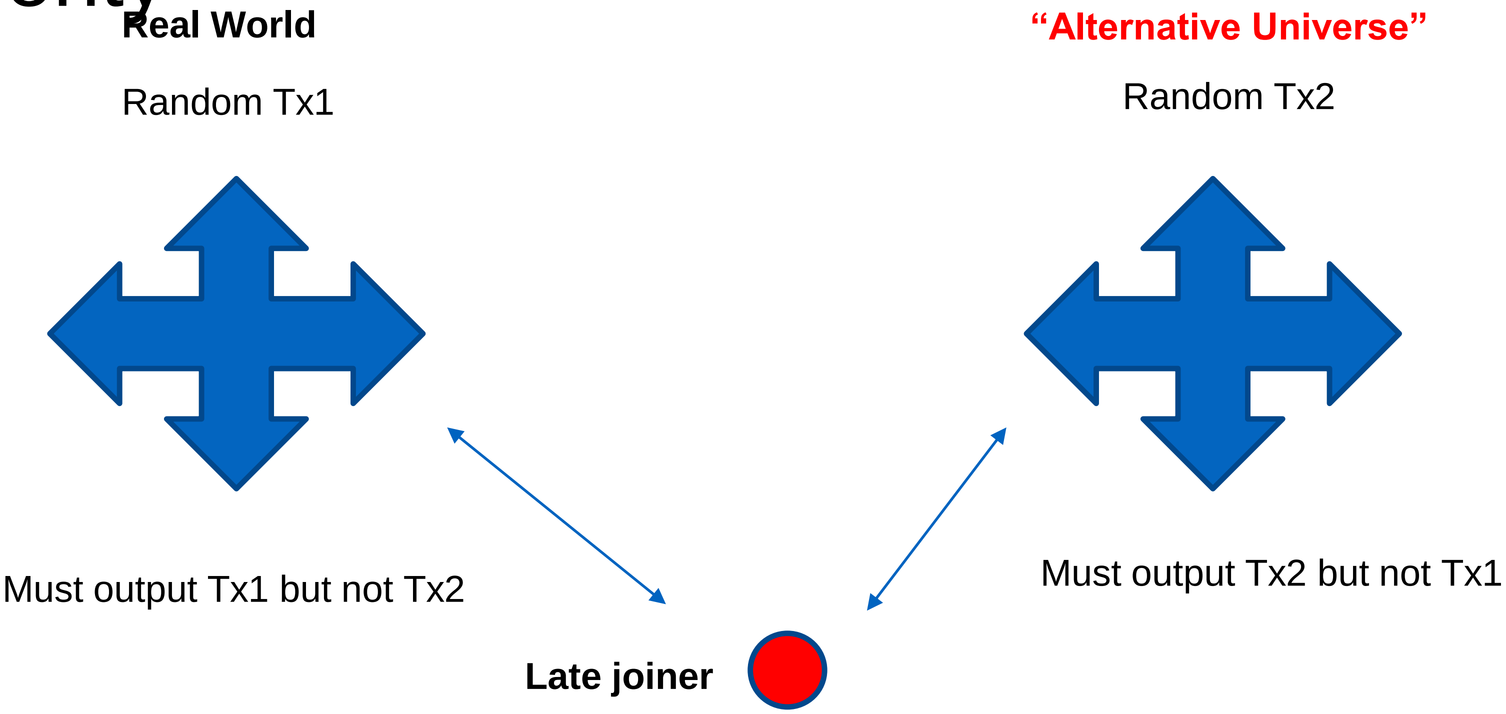
The “Permissionless” Model

.>

- Nodes don’t know the exact # of nodes => **synchronous**
- Nodes come and go: “late joining” => **½ honesty**
- No authentication => **need POW**

Under all those assumptions, Naka works!

Impossibility of Consensus without Honest Majority



Must agree with Real by security.

But also with attacker produced alternative universe

Nakamoto's protocol achieves **strong robustness properties**:

- assuming "honest majority of computational power"
- assuming puzzle difficulty is appropriately set as a function of network delay Δ (i.e., synchrony)

BUT 1: Blocktime needs to be roughly 10 * Δ to handle ρ> 0.45 ; thus, **slow confirmation times**

BUT 2: low throughput : 10 Tx/sec…

BUT 3: wasteful proof of work…

BUT 4: not fair, not incentive compatible!

Do we need to waste energy?

Permissioned Blockchain

- Instead of voting based on **computing power**, have a fixed set of voting authorities (e.g., banks)

- 1 vote per authority

- **High throughput & Fast Confirmation!**
- But **not “open”**

Proof of Stake

- Instead of voting based on **computing power**, vote based on amount of currency in the systems (a.k.a. **stake**)
  - Note: needs a blockchain with a **cryptocurrency** for this
  - similar thing actually true also for Naka: how to incentivize mining

- 1 coin = 1 vote

- A greener alternative to Bitcoin
  But: large account holders get more votes

- **Main Take-away:**

"Anyone can join" ≠ no authentication

Consensus for Proof of Stake Blockchains

Two approaches:

1. Variants of Nakamoto consensus that remove proof of work [PS’17,GKL’17]

Pro: handle **dynamic participation:**
we don’t know how may people show up; security holds
(assuming that ½ of **online nodes** are intact).

Con: roughly as **slow confirmation** as Nakamoto consensus

2. **Sortition to Elect a Committee** and next use **Byzantine Fault Tolerance (BFT)**
[Micali’17, Chen-Micali’17, TenderMint]

Pro: has been researched since 1970s;
fast confirmation, partial synchrony

Con: requires all **honest/intact nodes to be online**
(security relies on > 2/3 of all players being online and intact)

Proof of Stake with Dynamic Participation

The “Permissionless” Model

.>

- Nodes don’t know the exact # of nodes => **synchronous**
- Nodes come and go: “late joining” => **½ honesty**
- No authentication: “anyone can join” => **need POW**

The “**Sleepy**” Model (a.k.a. dynamic participation) [PS’17]

.>

- Nodes don’t know the exact # of nodes => **synchronous**
- Nodes come and go: “late joining” => **½ honesty**
- ~~No authentication: “anyone can join” => **need POW**~~

will assume PKI

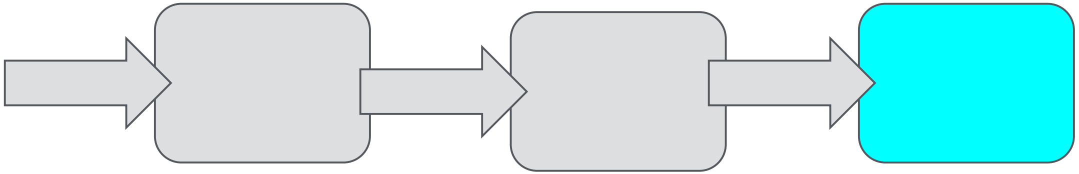
Can we Remove

“proof-of-work”

from Nakamoto Consensus

in PKI model?

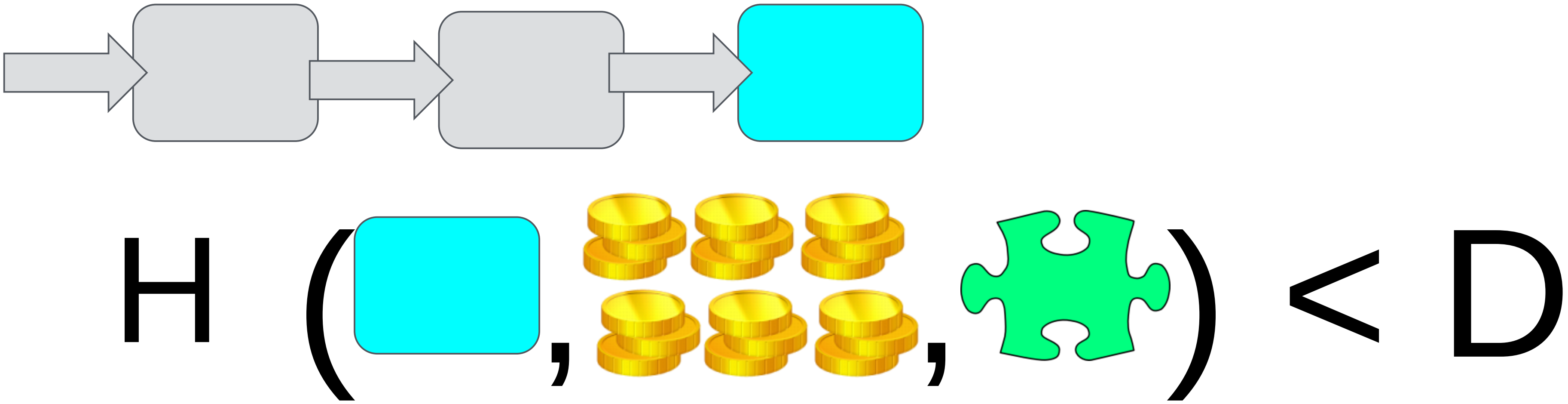
(dynamic participation)

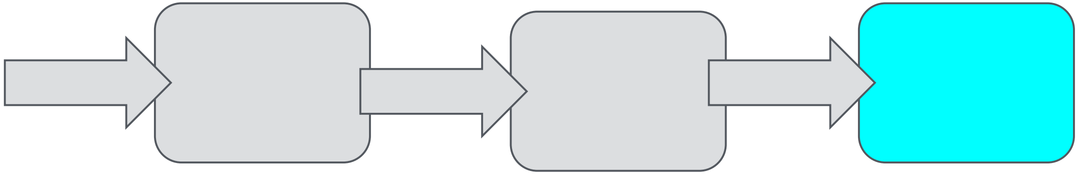


Proof-of-work = “Leader election”

Key idea: restrict the puzzle space

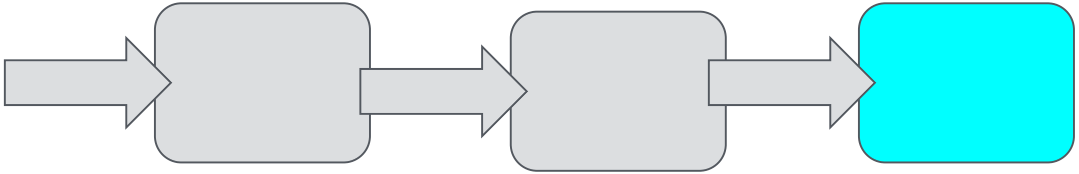
(possible since we have a fixed set of players and a PKI)


H
< D

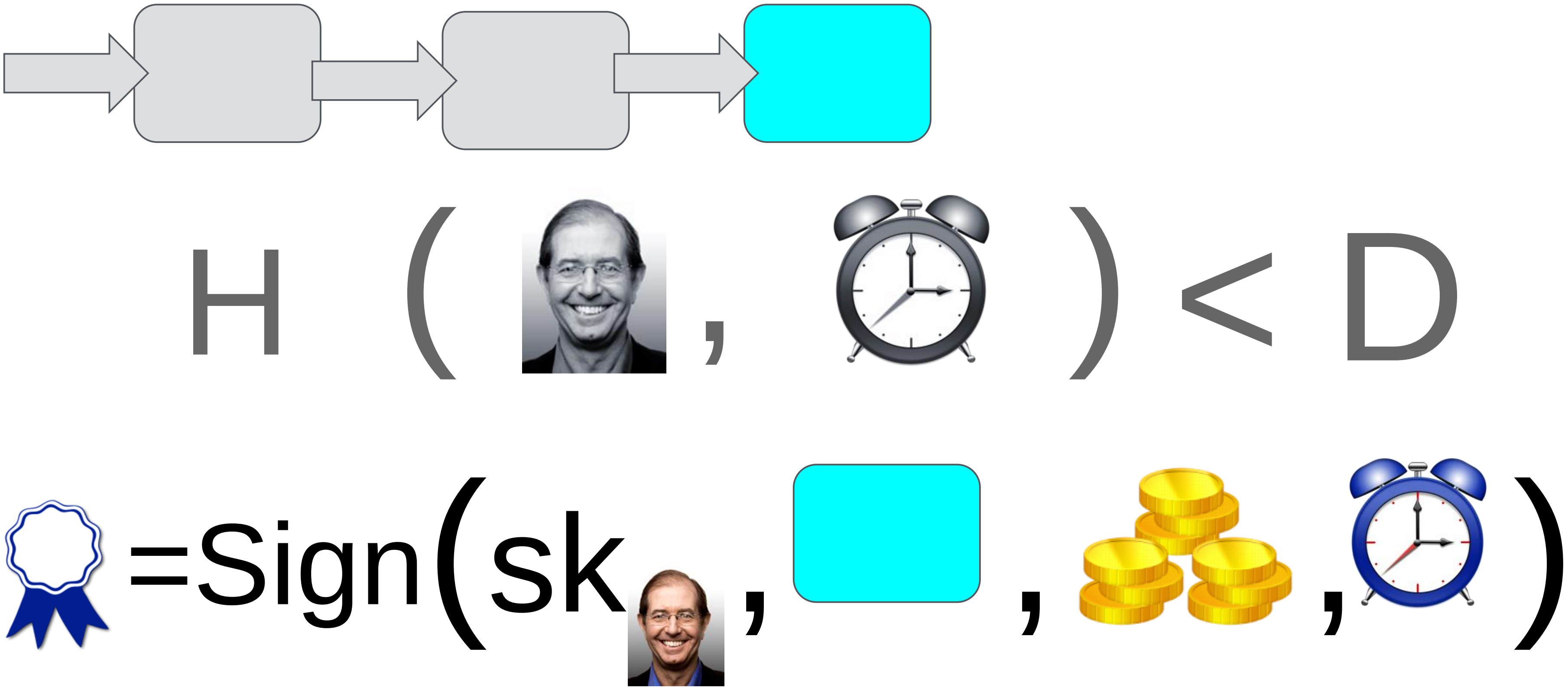


H ( , ) < D

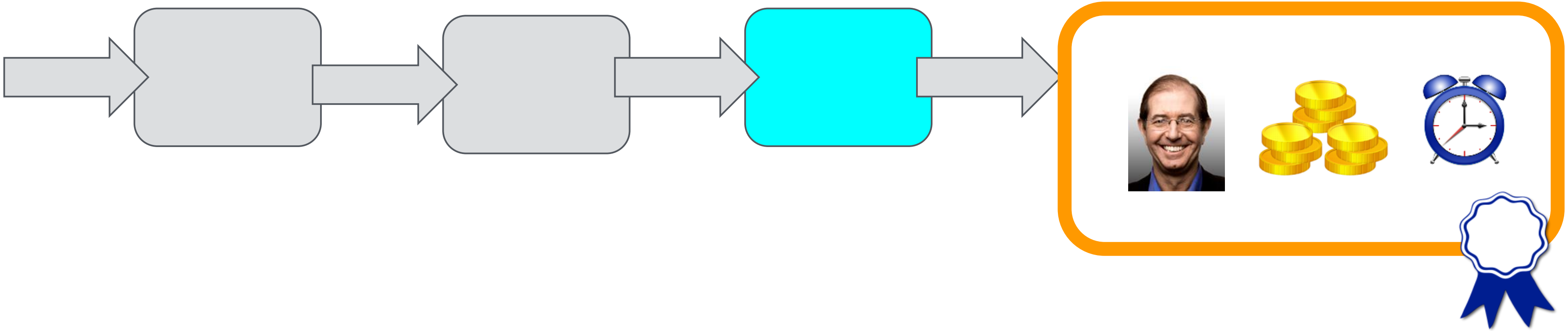
Time-Based Leader Election

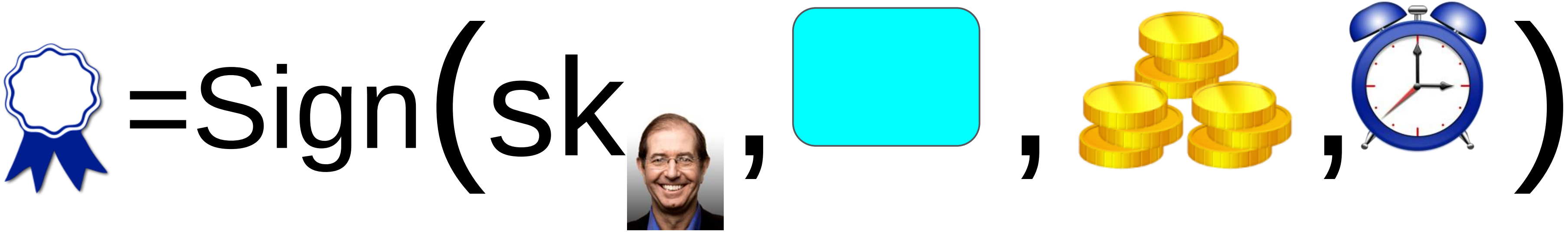


H ( , ) < D



Sign a new block as a leader

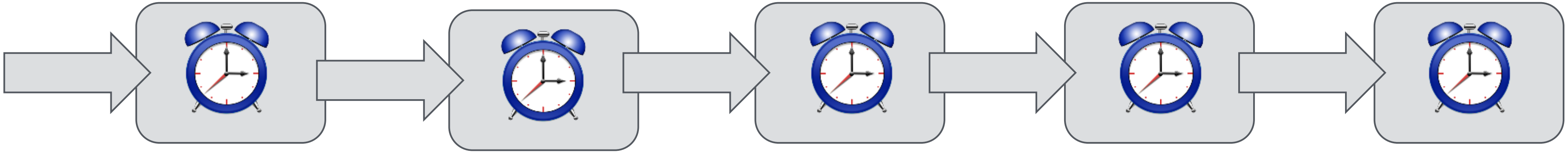




Sign a new block as a leader

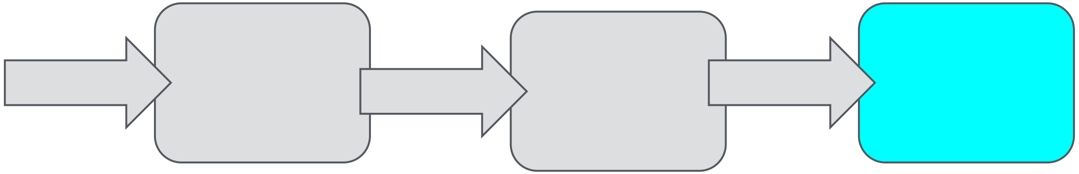
Also:

- Time steps in blocks strictly increasing
- Honest nodes reject blocks “in the future”



Thm [PS'17]: Assuming OWF + CRS+PKI, there exists a secure blockchain in the **synchronous model**, handling **dynamic participation** and < ½ **static corruption**

Problem: Can predict who will be a leader in advance. Corrupt them!



VRF( , ) < D

“Cryptographic Sortition” [Micali’17]

Thm [PS’17]: Using stronger Crypto, there exists a secure blockchain in the **synchronous model**, handling **dynamic participation** and < ½ **adaptive corruption**

Dynamic participation => Synchronous

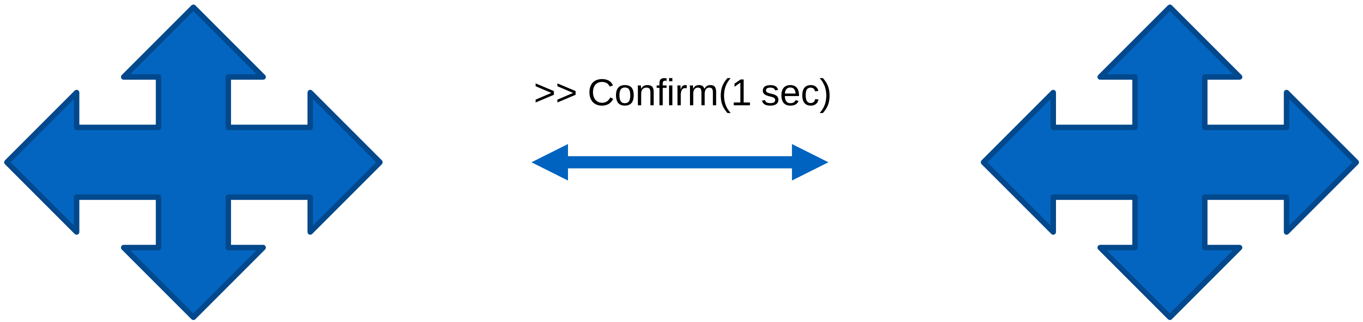
N players

Random Tx1

Total 2N players

N players

Random Tx2



1 second

1 second

Must output Tx1 within Confirm(1 sec)

Must output Tx2 within Confirm(1 sec)

Axiom: Computation polylog(# nodes)

Proof of Stake
From Partially-Synchronous BFT
(known # participants)

Sortition + BFT [Micali’17, Algorand]

VRF( , )

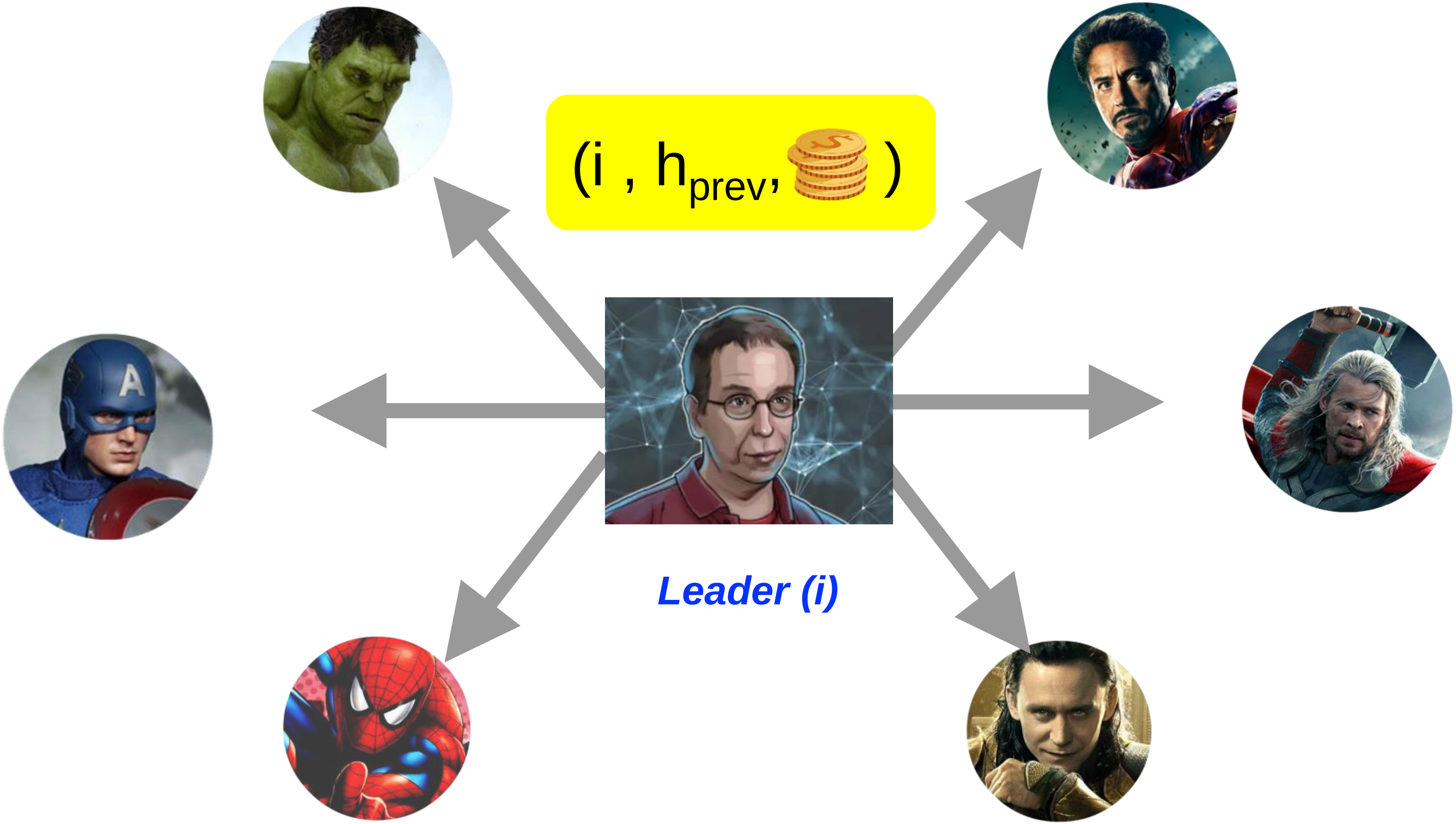
Use sortition to elect a committee; use BFT on the committee

Need an underlying BFT protocol with “speak once property”:
YOSO = “You only speak once” [GHKMNRY’21]

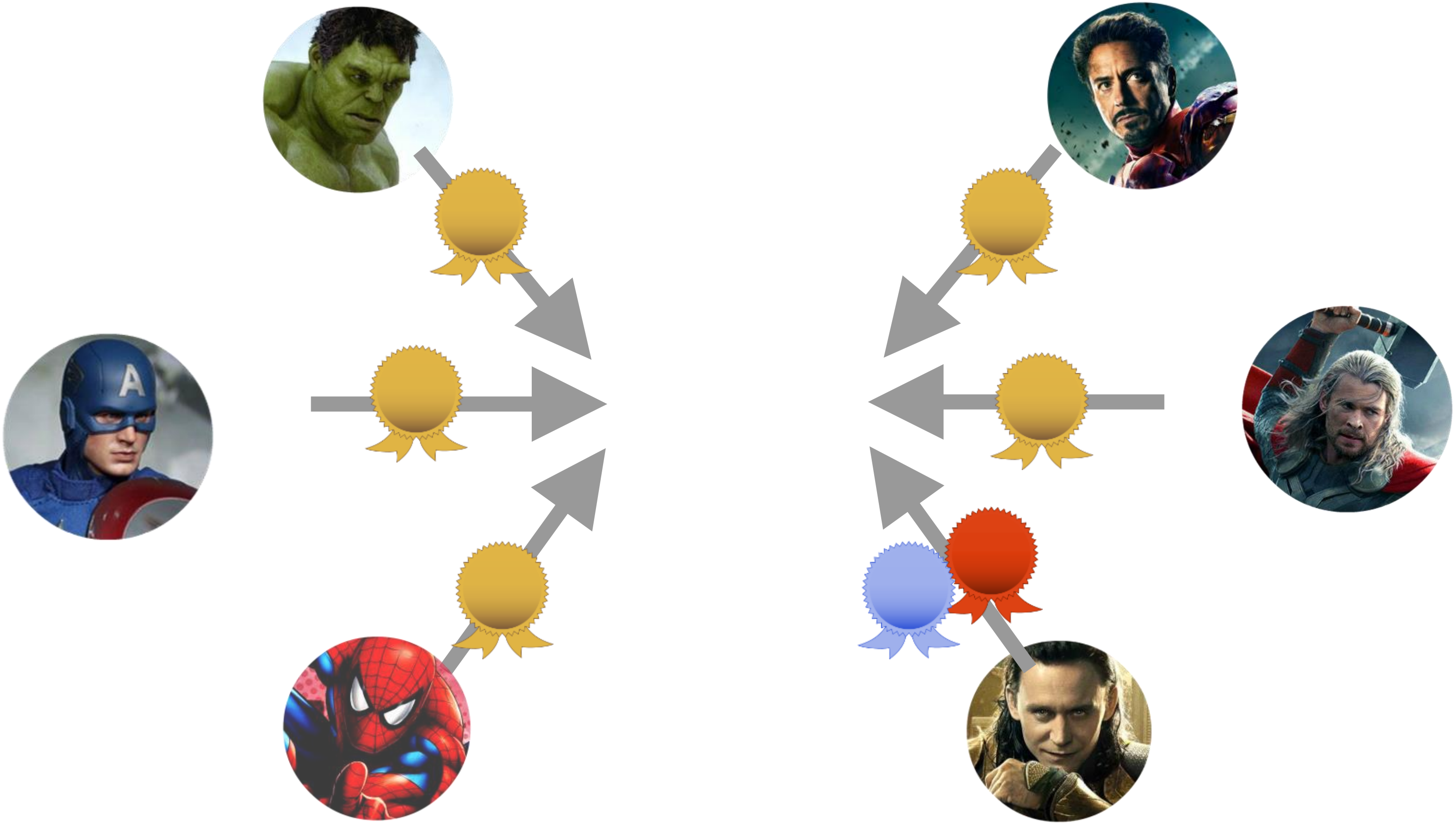
Proceed in iterations i

Leader (i)

❶ Leader(i) proposes "block"

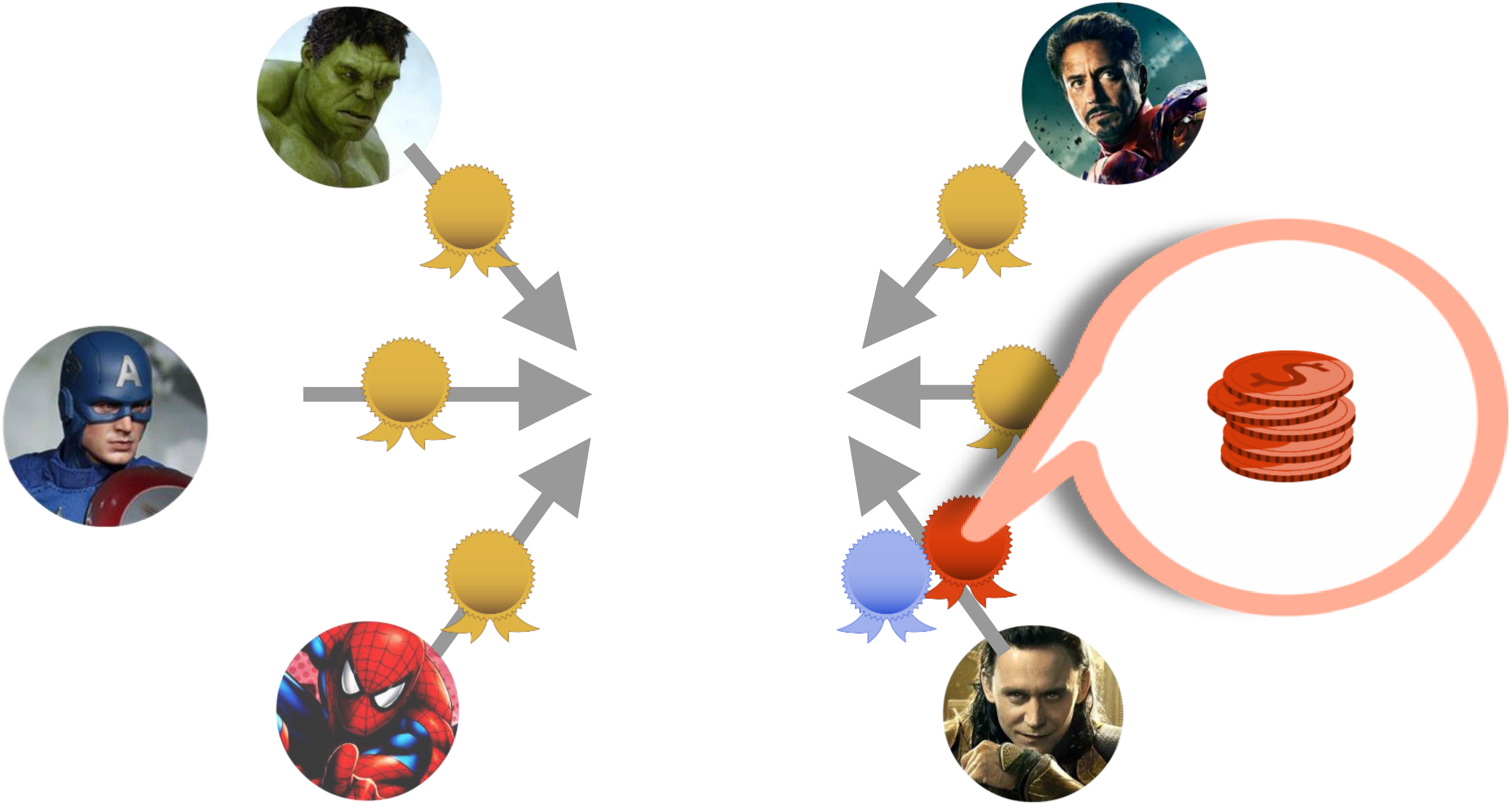


2 Everyone votes

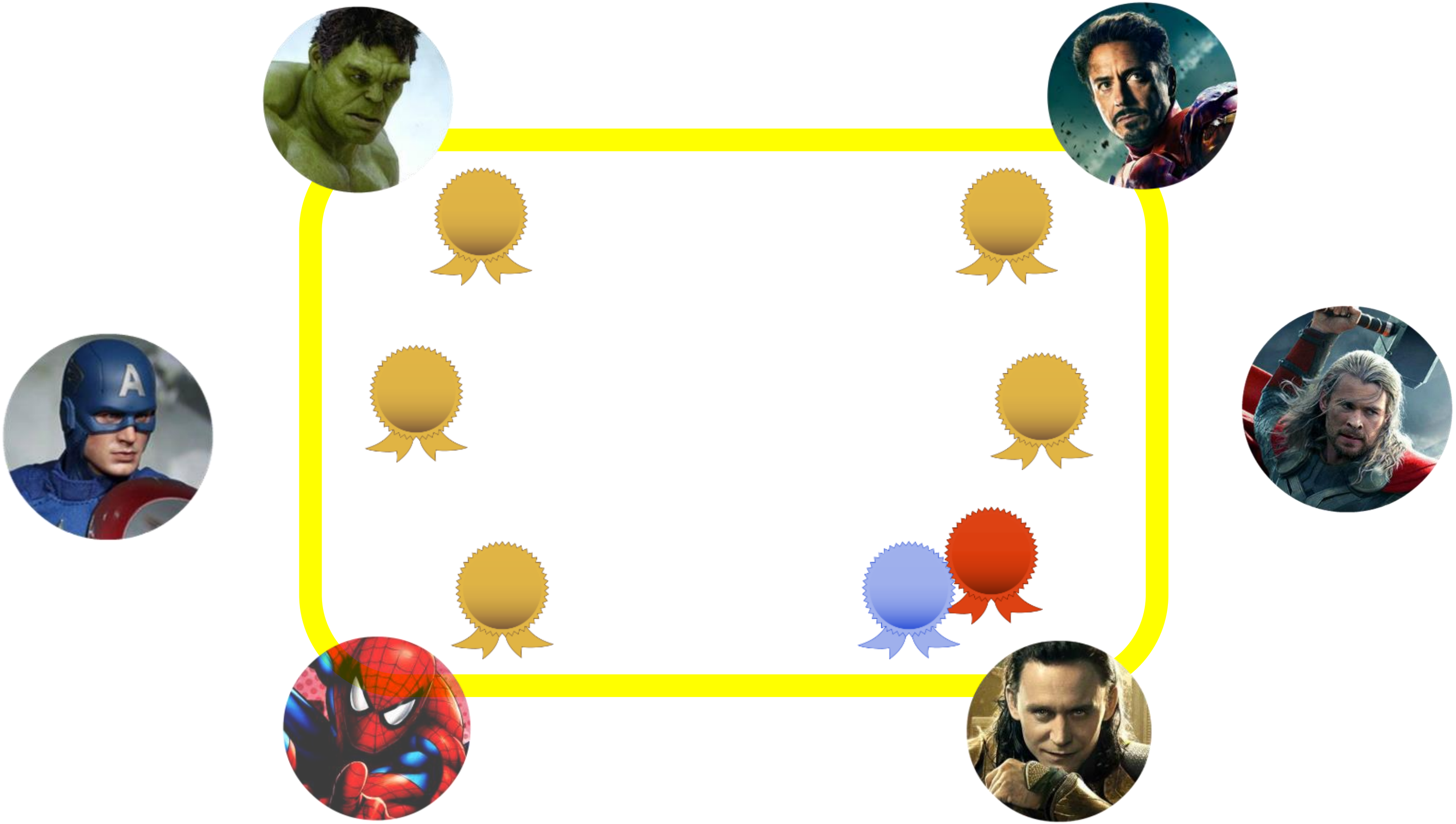


2

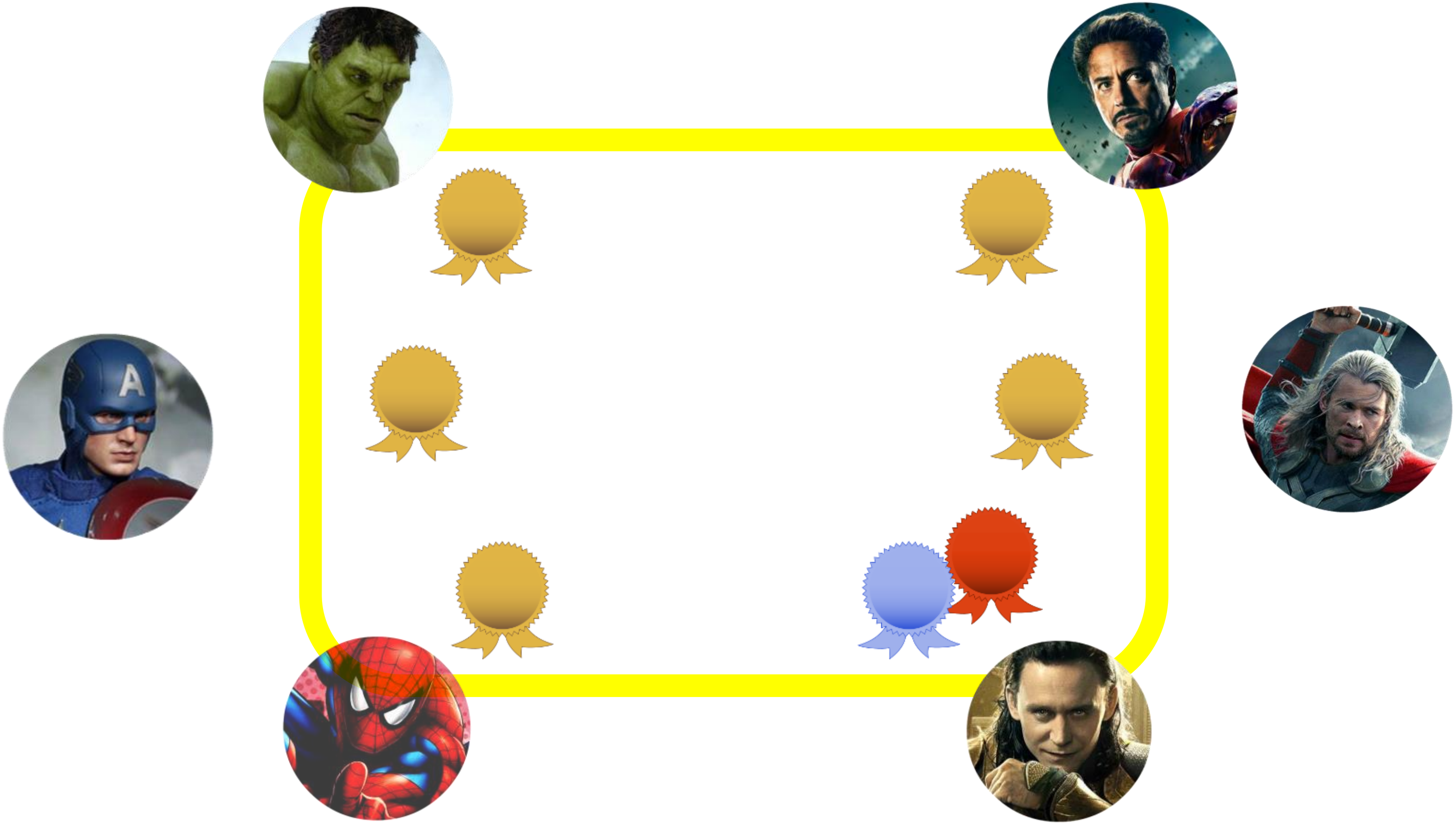
Everyone votes



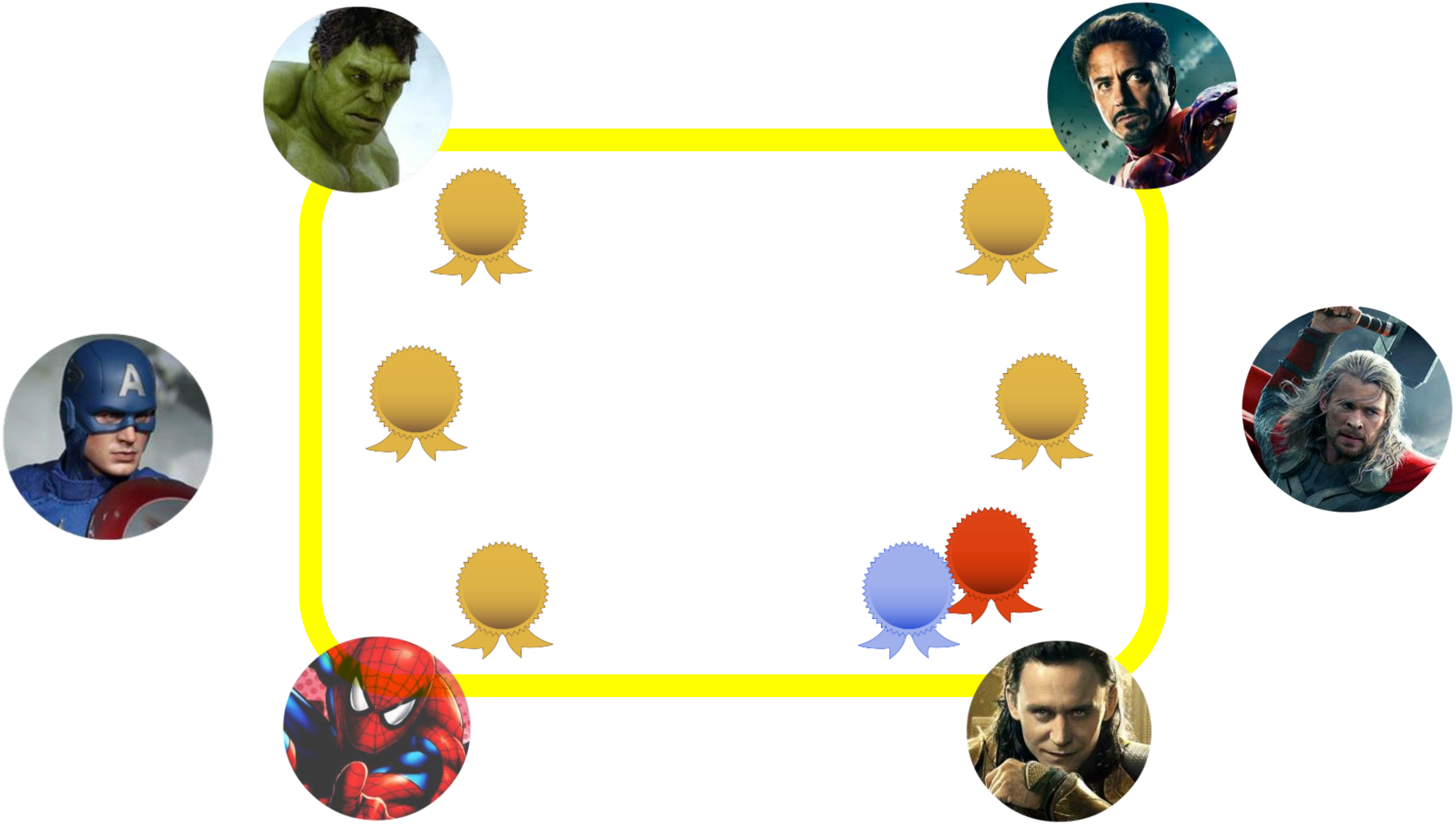
3 Confirm upon enough votes



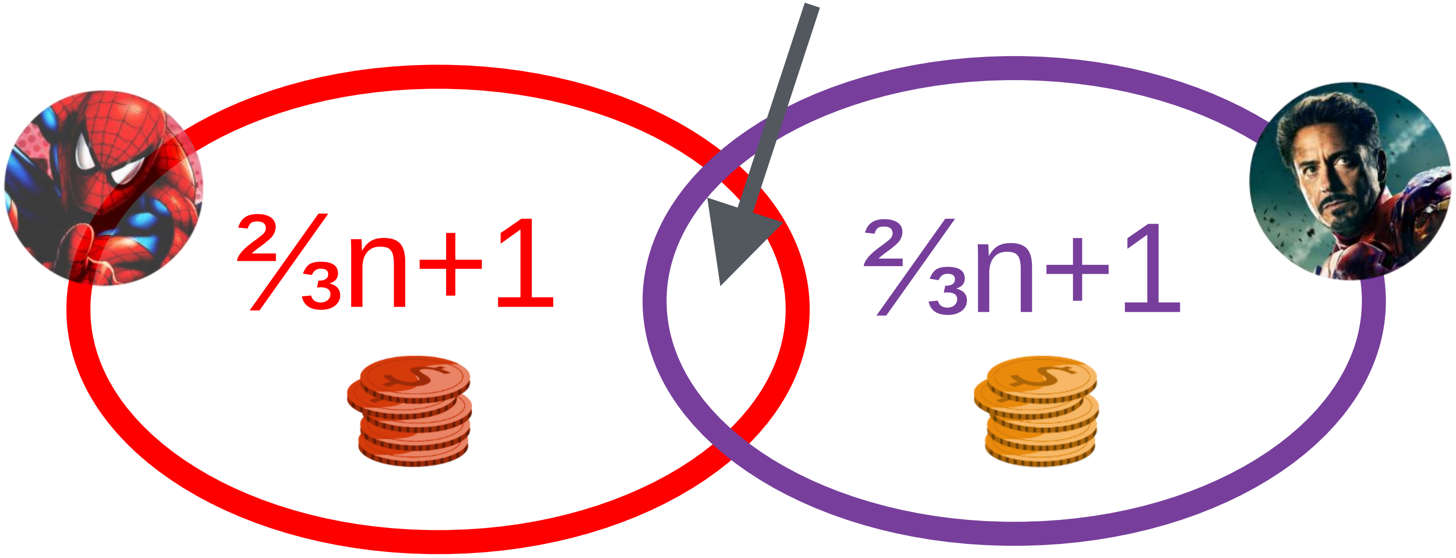
Honest nodes vote uniquely.



Wait for ⅔n+1 votes

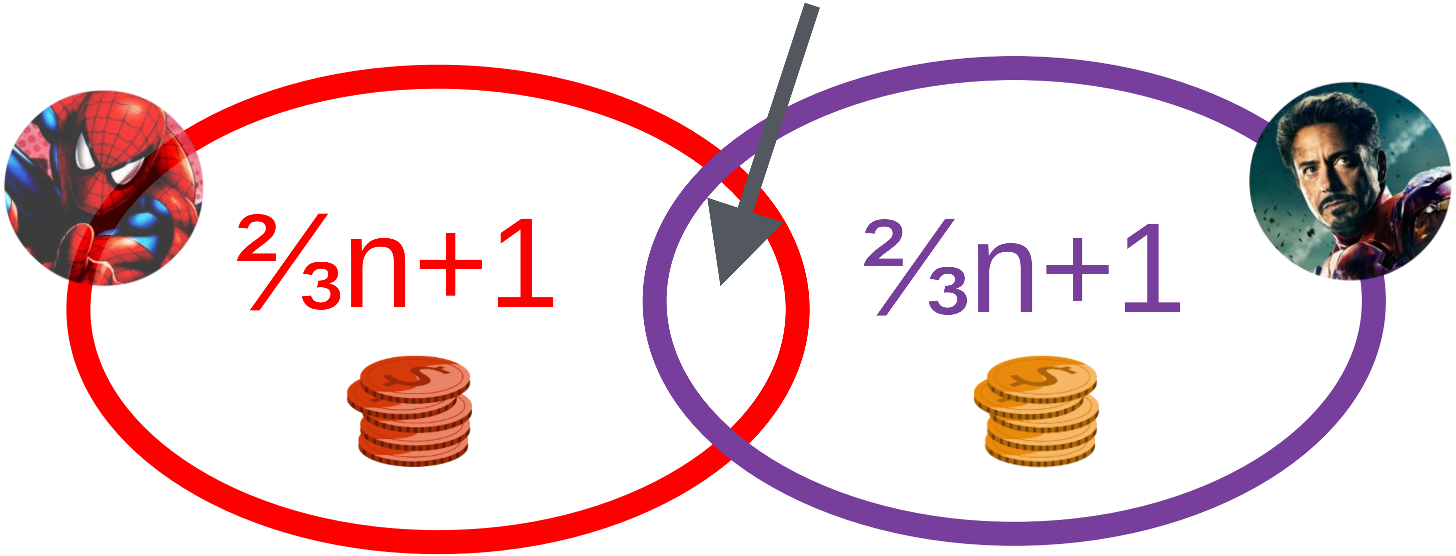


Must intersect at an honest node



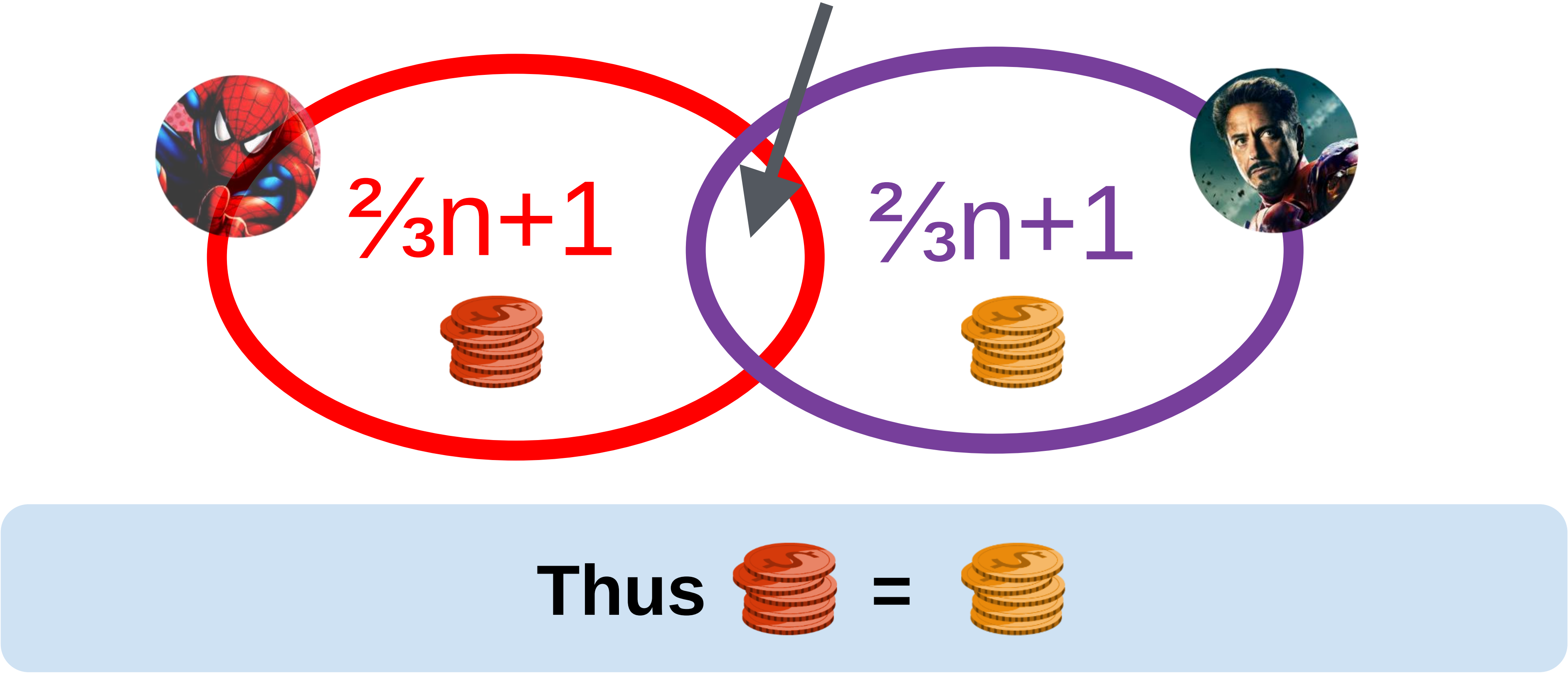
Assume $\frac{2}{3}n+1$ honest

Must intersect at an honest node



Assume < 1/3n malicious

Must intersect at an honest node



Assume ⅔ honest and online

Assume ⅔ honest and online

✓ **Consistency**

✓ **Liveness**

Assume $\frac{2}{3}$ honest and online

✓ **Consistency**

✓ **Liveness**

✓ **Consistency**

✗ **No liveness**

Dealing with faulty proposers:

- "Time-out" and move on to the next leader
- Approach 1: Require "many" rounds of confirmation before moving to the next iteration [PBFT,Algorand]
- Approach 2: Or can pipe-line [Casper,HotStuff,..]: Can move on directly, but don't finalize the whole chain (c.f. Naka).

Summing Up

The "Permissionless" Model **w/o set-up**

.>

- Nodes don't know the exact # of nodes => **synchronous**
- Nodes come and go: "late joining" => **½ honesty**
- No authentication => **need POW**

Under all those assumptions, Naka works!

Permissionless with PKI (Proof of Stake)

Two approaches:

1. Variants of Nakamoto consensus that remove proof of work [PS'17,GKL'17]

Pro: handle **dynamic participation:**
we don't know how may people show up; security holds
(assuming that ½ of **online nodes** are intact).

Con: roughly as **slow confirmation** as Nakamoto consensus

2. **Sortition to Elect a Committee** and next use **Byzantine Fault Tolerance (BFT)**
[Micali'17,Chen-Micali'17,TenderMint'16]

Pro: has been researched since 1970s;
fast confirmation, partial synchrony

Con: requires all **honest/intact nodes to be online**
(security relies on > 2/3 of all players being online and intact)

Incentives (for POW blockchains)

Why do miners “mine”?

Block rewards: each miner who find a new block gets a reward

Transaction fees, but let’s ignore for now

Two issues

1. **Fairness**: honest players get less than their "fair" rewards:
   - Not "**incentive-compatible**"!
2. **High-variance** of Rewards
   - [PSS'16]: needed to ensure consistency
   - Join a **mining pool**

Ideal Fairness

In **any length k** segment of the chain,
fraction of blocks mined by an **X-fraction**
“coalition” of honest users is **X**

ε -approx Fairness

In **any length k** segment of the chain, fraction of blocks mined by an **X-fraction** "coalition" of honest users is **$(1- \varepsilon)$ X**

Distribute rewards + fees over k-length sliding window:
Implies **Coalition-safe 3ε-NE**

If each block in the chain were selected like a random lottery,

⇒ **ε -approx fairness** for any ε >0

(by Chernoff bound)

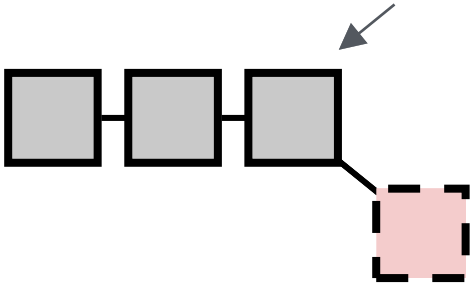
Nakamoto’s Blockchain

Completely UNFAIR

- An attacker controlling close to **1/2**, may get almost **ALL** the blocks
- An attacker controlling close **1/3** may get **1/2** the rewards,

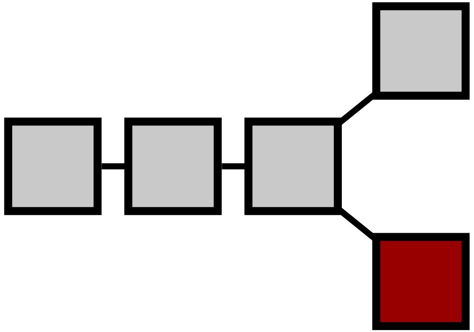
Selfish Mining [bitcoinforum'10, Eyal-Sirer'13]

Honest nodes see this chain



Adversary withholds a private fork

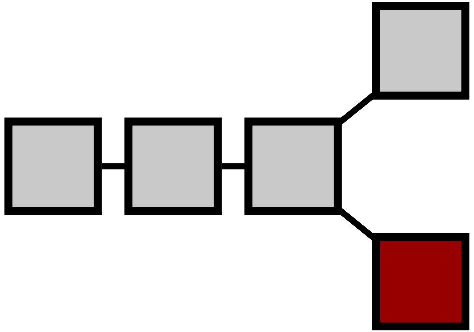
Selfish Mining



An honest node
mines next block

Adversary immediately releases block
Combine with a network rushing attack

Adversary can erase honest nodes' work



An honest node
mines next block

Adversary immediately releases block
Combine with a network rushing attack

$\rho=1/3$

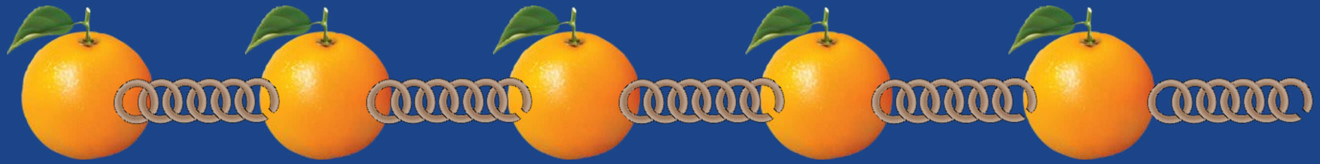
$$(2/3t - 1/3t)/(2/3\ t) = 1/2$$

By deviating get more than "fair" share of rewards

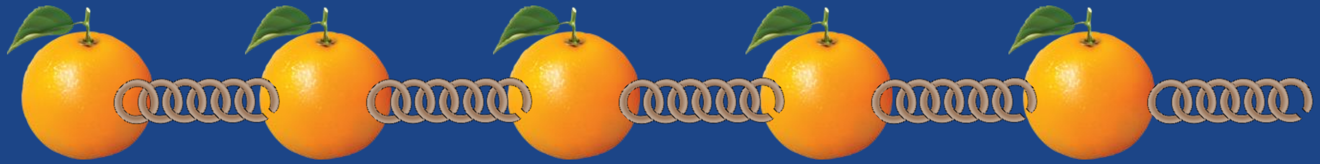
Thm [PS’17]: for any ε >0, there exists a secure blockchain that satisfies

ε -approx fairness

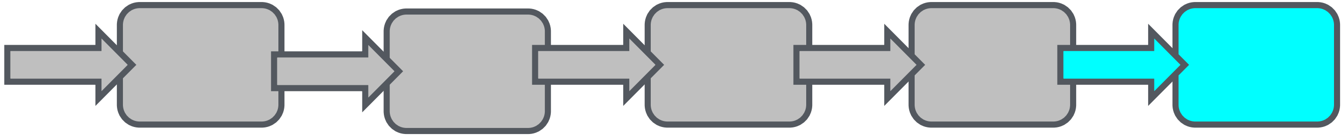
Fruitchain



ORANGE is the new BLOCK

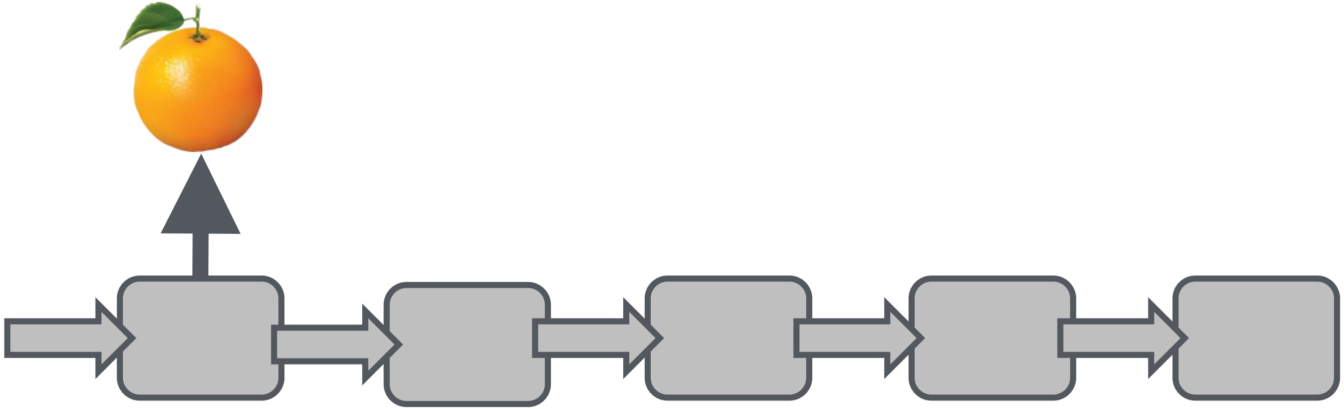


Each step: An honest node has a chance of **mining a block**

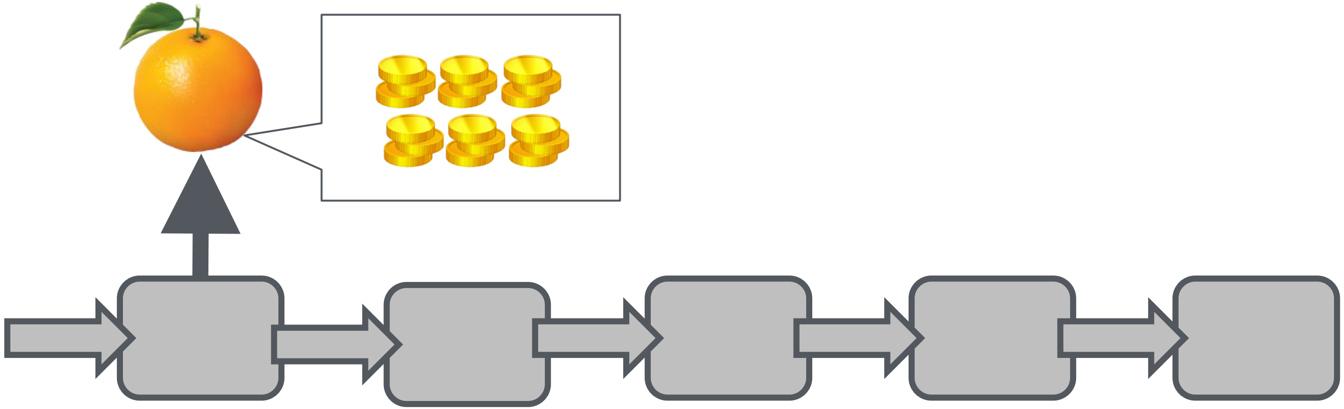


Each step: An honest node has a chance of

mining a block **and mining a fruit**

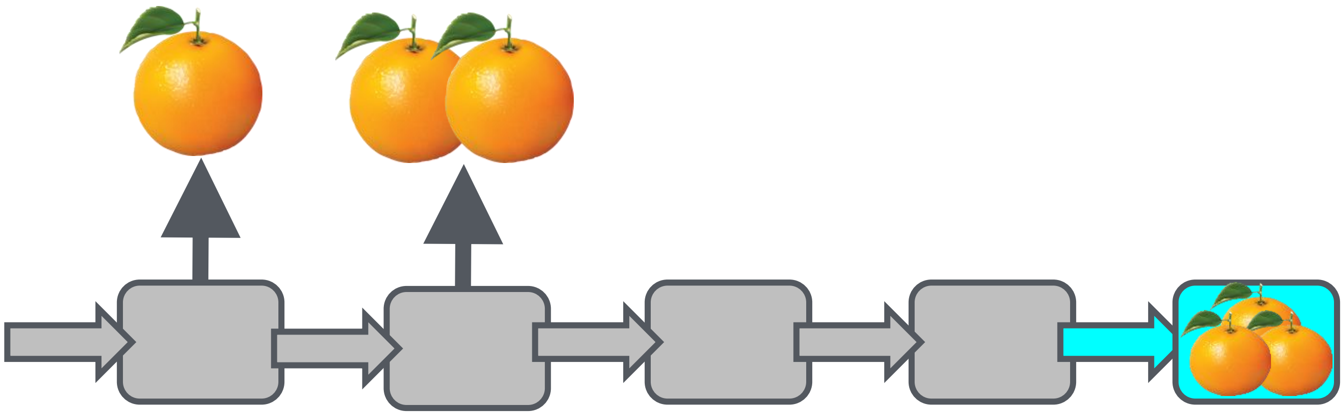


A fruit contains transactions, blocks don't

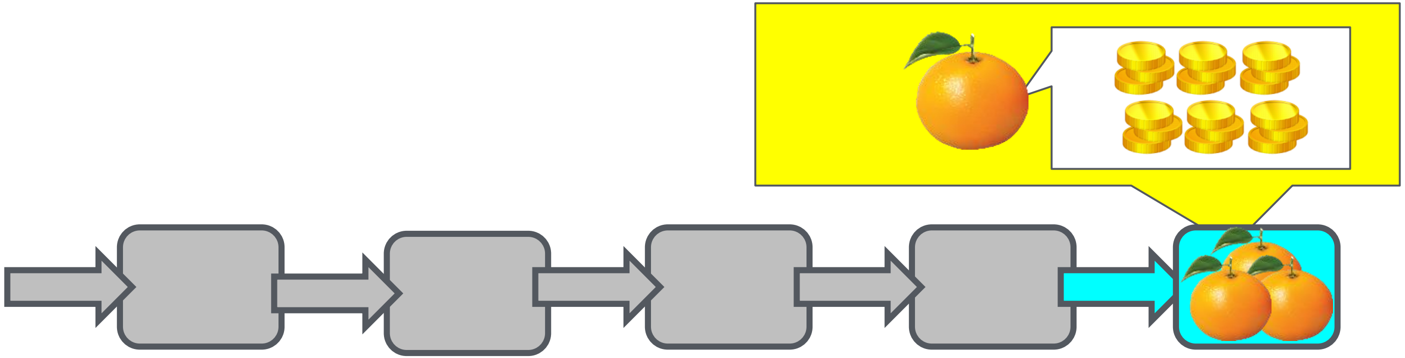


An honest node includes

“recent” fruits in a newly mined block



Fruits contain transactions, blocks contain fruits



Honest fruit will not get erased (by liveness, eventually some good guy will pick them up)

Adversary can amass fruits and release them all together?

Old fruits perish

(only “recent” fruits count)

Thm: for any ε >0, there exists a secure blockchain that satisfies **ε -approx fairness**

=> **ε-Incentive-Compatible** blockchain for ε = 1/poly(k)

Open to get ε = neg.

Fruit chain method also extremely useful to improve bandwith!
Similar ideas are currently used in Ethereum's proof of stake protocol.