CENTRALIZATION VS. DECENTRALIZATION

Centralization vs. decentralization

Competing paradigms that underlie many digital technologies

Centralized: Online Social Networking Services (Facebook, Google)

Decentralized: Internet, Email service and the SMTP protocol

Decentralization is not all-or-nothing: For example, E-mail. Email has a decentralized protocol (e.g., SMTP), but dominated by centralized webmail services

Aspects of decentralization in Bitcoin

1. Who maintains the ledger?
2. Who has authority over which transactions are valid?
3. Who creates new bitcoins?
4. Who determines how the rules of the system change?
5. How do bitcoins acquire exchange value?

Beyond the protocol:

Exchanges, Wallet software, Service providers...

Aspects of decentralization in Bitcoin

1. **Peer-to-peer network**:

   Open to anyone, Low barrier to entry

2. **Mining**:

   Open to anyone, but inevitable concentration of power
   often seen as undesirable

3. **Updates to software**:

   Core developers trusted by community, have great power

DISTRIBUTED CONSENSUS

Bitcoin’s key challenge

- Key technical challenge of decentralized e-cash: distributed consensus
  - or: how to decentralize ScroogeCoin

- Traditional motivation: reliability in distributed systems
  - Distributed key-value store enables various applications: DNS, public key directory, stock trades …

Defining distributed consensus

There are "*n*" nodes, each have an input value. Some nodes are faulty or malicious. A distributed consensus protocol has the following two properties:

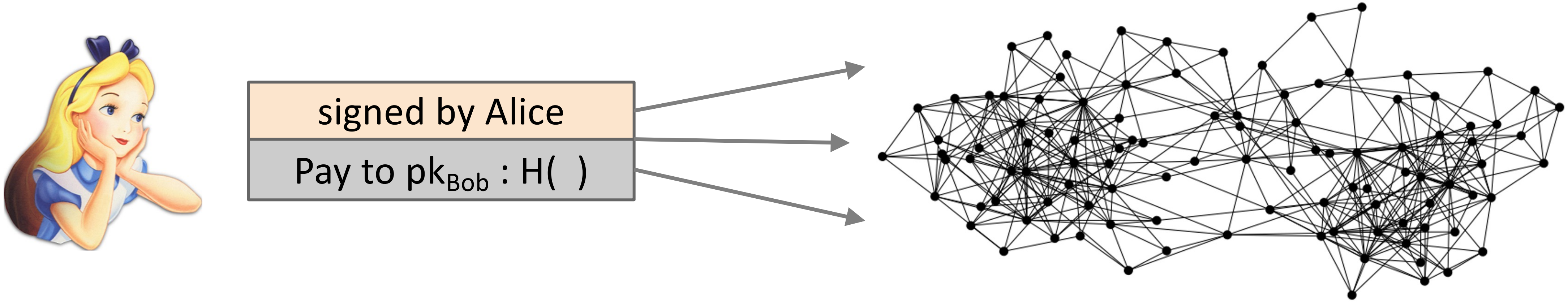
1. The protocol terminates and all honest nodes are in agreement on the same value

2. This value must have been proposed by some honest node

What does this mean in the context of Bitcoins?

Bitcoin is a peer-to-peer system

When Alice wants to pay Bob:
she broadcasts the transaction to all Bitcoin nodes



signed by Alice

Pay to pk_{Bob} : H( )

Note: Bob’s computer may not be in the picture or online!
In fact, running a Bitcoin node is not important for Bob to receive the funds. The Bitcoins will be his regardless

What nodes need to reach a consensus on?

- Which transactions were broadcast on the network
- Order in which these transactions occurred

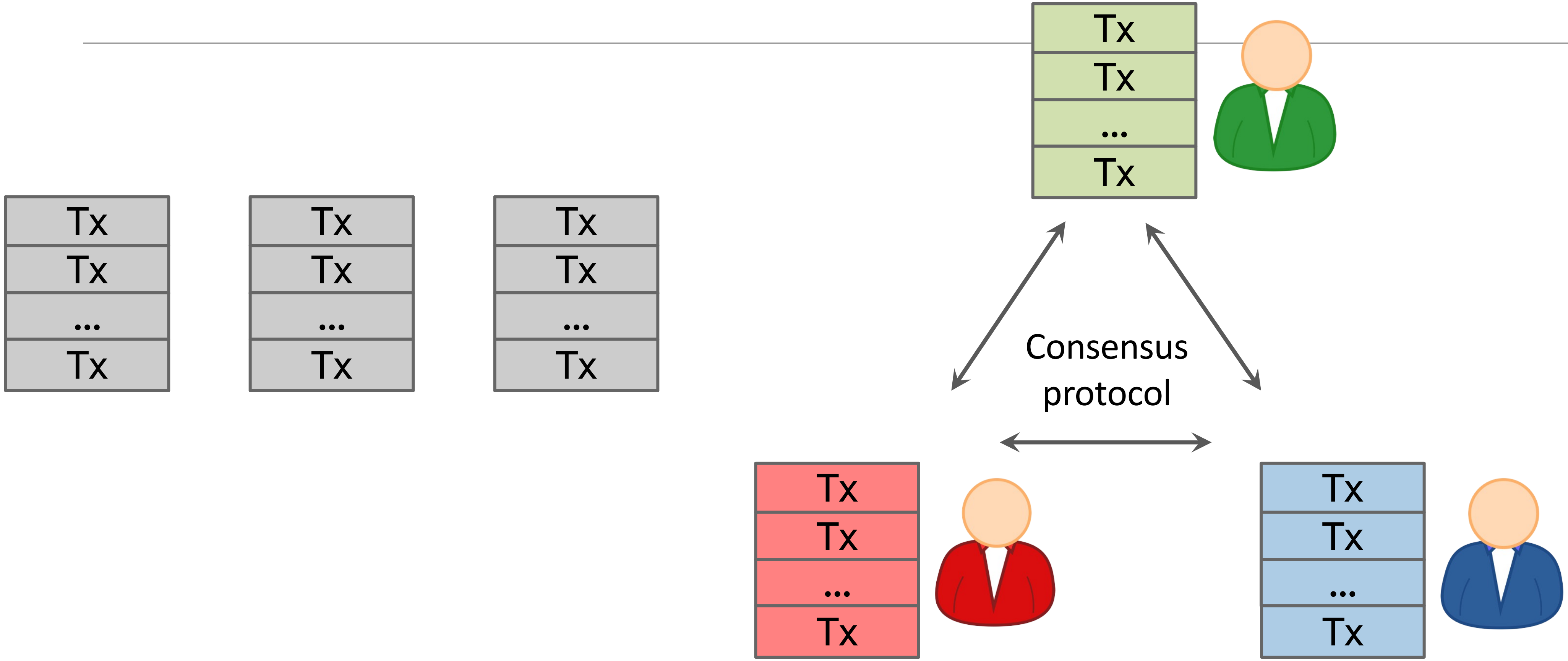
→ Result of the consensus protocol: Single, global transaction ledger for the system

How consensus could work in Bitcoin

At any given time (in the bitcoin peer-to-peer network):

- All nodes have a sequence of blocks of transactions (called, ledger or block chain) they’ve reached consensus on
- Each node has a set of outstanding transactions it’s heard about (but not yet included in the block chain)
  - For these transactions consensus has not yet happened
  - Each node may have a slightly different outstanding transaction pool

How consensus could work in Bitcoin



Consensus protocol

OK to select any valid block, even if proposed by only one node

Why consensus is hard

Nodes may crash

Nodes may be malicious

Peer-to-peer network is imperfect

- Not all pairs of nodes connected (and may participate)
- Faults in network
- Latency

No notion of global time → constraints the set of consensus algorithms that can be used

Many impossibility results

- **Byzantine generals problem**: Consensus impossible to achieve if 1/3 or more generals are traitors

- **Fischer-Lynch-Paterson (deterministic nodes)**: consensus impossible with a single faulty node (under certain conditions)

Understanding impossibility results

The earlier results proven for specific models

- Specifically, distributed databases
- A distributed database model (and assumptions under it) doesn't carry over to Bitcoins!

These results say more about the model than about the problem

What does it mean?

- It may be possible to develop consensus protocols that work for Bitcoin networks

Bitcoin consensus: theory & practice

Bitcoin consensus works better in practice than in theory

Theory is still catching up

BUT theory is important, can help predict unforeseen attacks

So why is the problem of consensus different in Bitcoins?

Introduces incentives

- Possible only because it’s a currency!
- So in Bitcoins we do not have to solve the consensus problem in general, but only the one for a currency system

Embraces randomness

- Does away with the notion of a specific starting and ending point for consensus
- Consensus happens over long time scales — about 1 hour
- In summary, consensus in Bitcoins is not deterministic – Even at the end of 1 hour nodes may not be 100% sure that their view of the block chain is the consensus view
  - Although the probability of that not being the case is very low

BITCOIN’S CONSENSUS ALGORITHM

Why having identity is useful for consensus?

Answer: It makes the consensus protocol easy to design! But how?

1. **<u>Pragmatic</u>**: some protocols need node IDs
   - Protocols could have instructions of the form “Now node with lowest ID, do something..”
   - Without identities, instructions are constrained
2. **<u>Security</u>**: assume less than 50% malicious
   - If nodes have identities, and difficult to create new node identities then some assumptions about the number of malicious nodes can be made
   - This can be used to prove certain security properties

How to overcome lack of identity in Bitcoins?

Weaker assumption: select random node in the bitcoin network

Analogy: lottery or raffle

- When tracking & verifying identities is hard, we give people tokens, tickets, etc.

Key assumptions:

- Now we can pick a random ID & select that node
- Multiple sybil nodes by the adversary are able to get only a single token (random ID)

Key idea: implicit consensus

1. In each round (corresponds to a different block in the block chain), random node is picked

2. This node proposes the next block in the chain
   - No consensus or voting done by this node!

3. Other nodes implicitly accept/reject this block
   - by either extending it
   - or ignoring it and extending chain from earlier block

4. <u>Every block contains hash of the block it extends</u>

Consensus algorithm (simplified)

1. New transactions are broadcast to all nodes
2. Each node collects new transactions into a block
3. In each round a random node gets to broadcast its block
4. Other nodes accept the block only if all transactions in it are valid (unspent, valid signatures)
5. Nodes express their acceptance of the block by including its hash in the next block they create

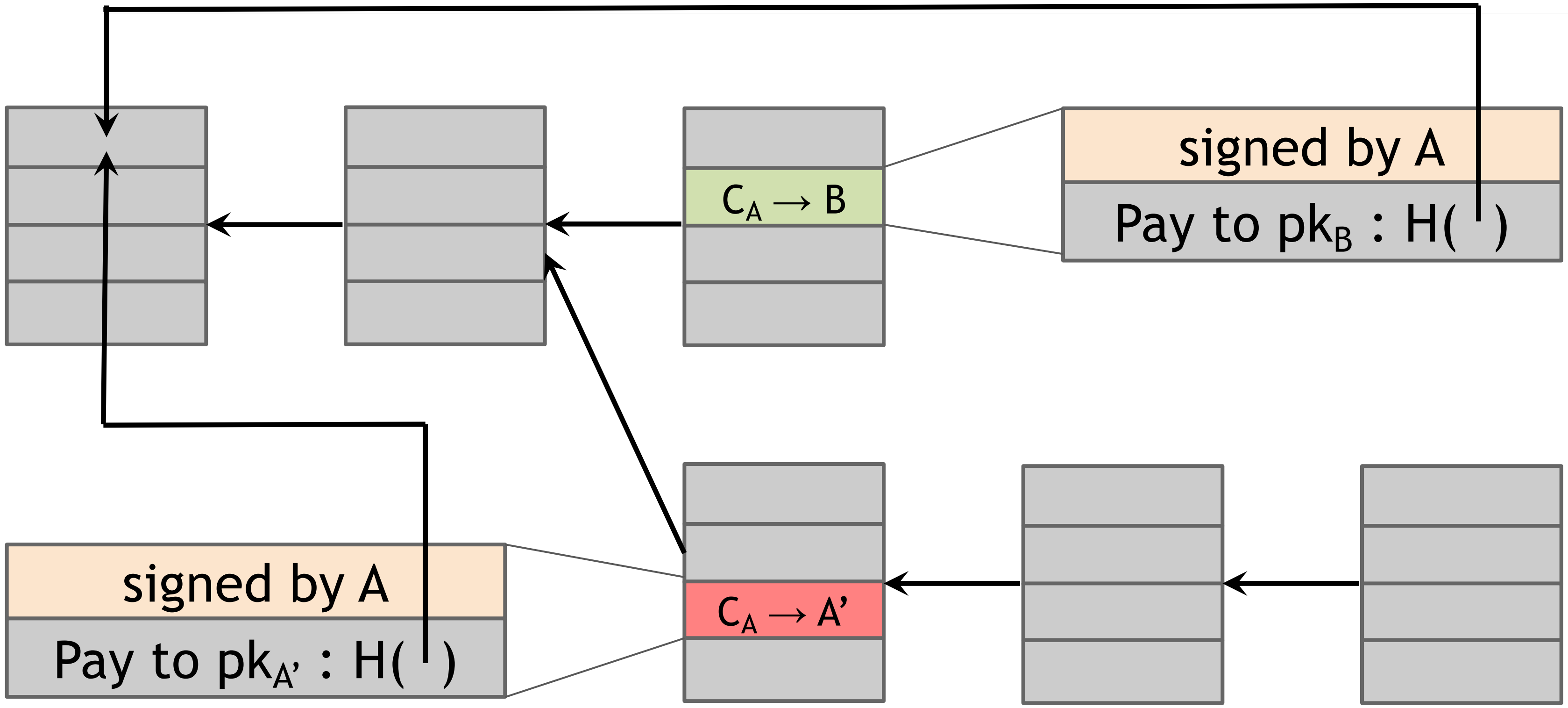
Now let’s analyze if this works!

Assume a malicious adversary.

Can this adversary subvert the implicit consensus process by:

1. **Stealing Bitcoins?**
2. **Denial of service?**
3. **Double spend?**

What can a malicious node do?

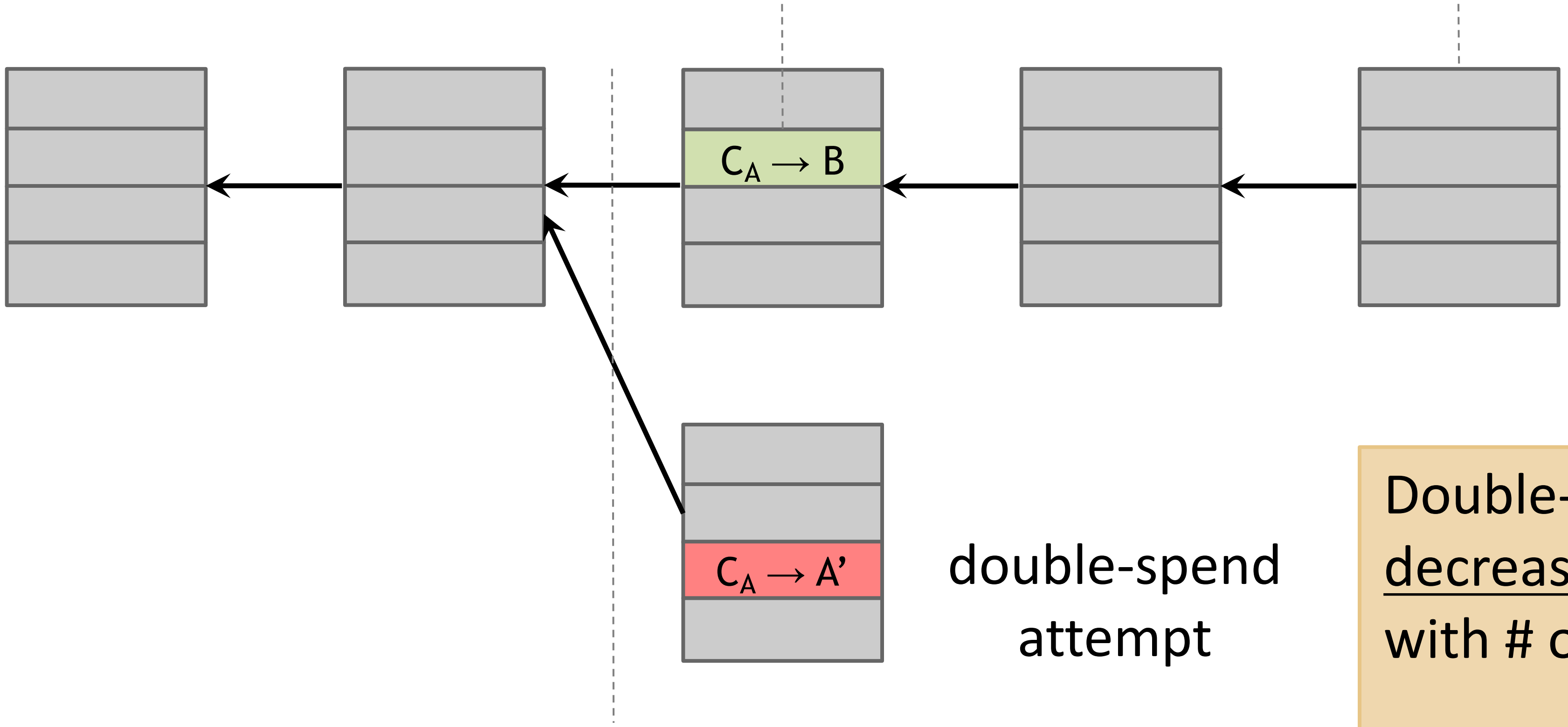


Double-spending attack

Honest nodes will extend the <u>longest valid branch</u>

In practice nodes extend the block that they first detect on the peer-to-peer network (not a solid rule)

From Bob the merchant’s point of view

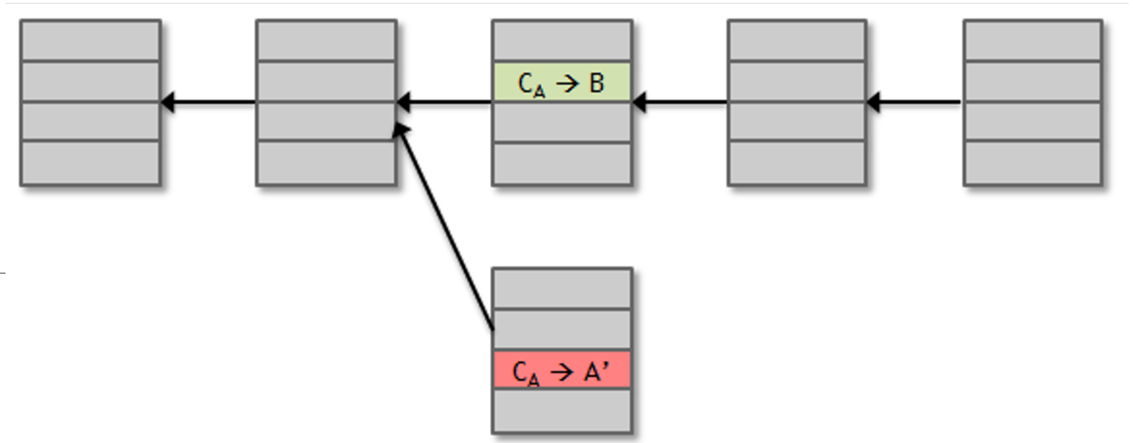


Hear about $C_A \to$ B transaction
0 confirmations

Double-spend probability <u>decreases exponentially</u> with # of confirmations

Most common heuristic:
6 confirmations

Recap



Protection against invalid transactions is cryptographic, but enforced by consensus

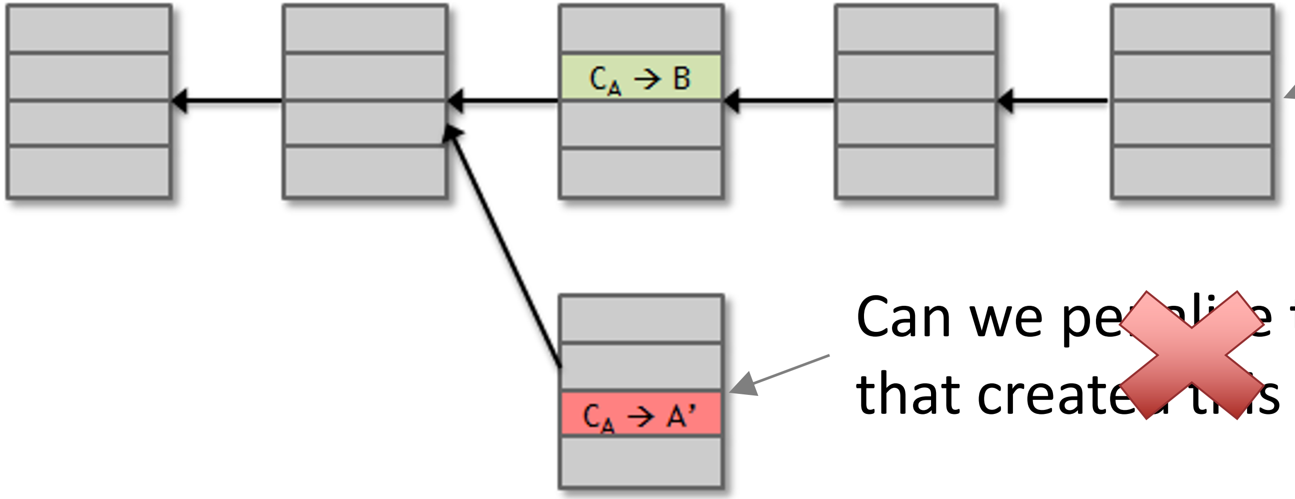
Protection against double-spending is purely by consensus

You’re never 100% sure a transaction is in consensus branch. Guarantee is probabilistic

INCENTIVES AND PROOF OF WORK

Assumption of honesty is problematic

Can we give nodes incentives for behaving honestly?



Can we reward nodes
that created these blocks?

Can we penalize the node
that created this block?

Everything so far is just a distributed consensus protocol
But now we utilize the fact that the currency has value

Incentive 1: Block Reward

Creator of block gets to

- include special coin-creation transaction in the block
- choose recipient address of this transaction

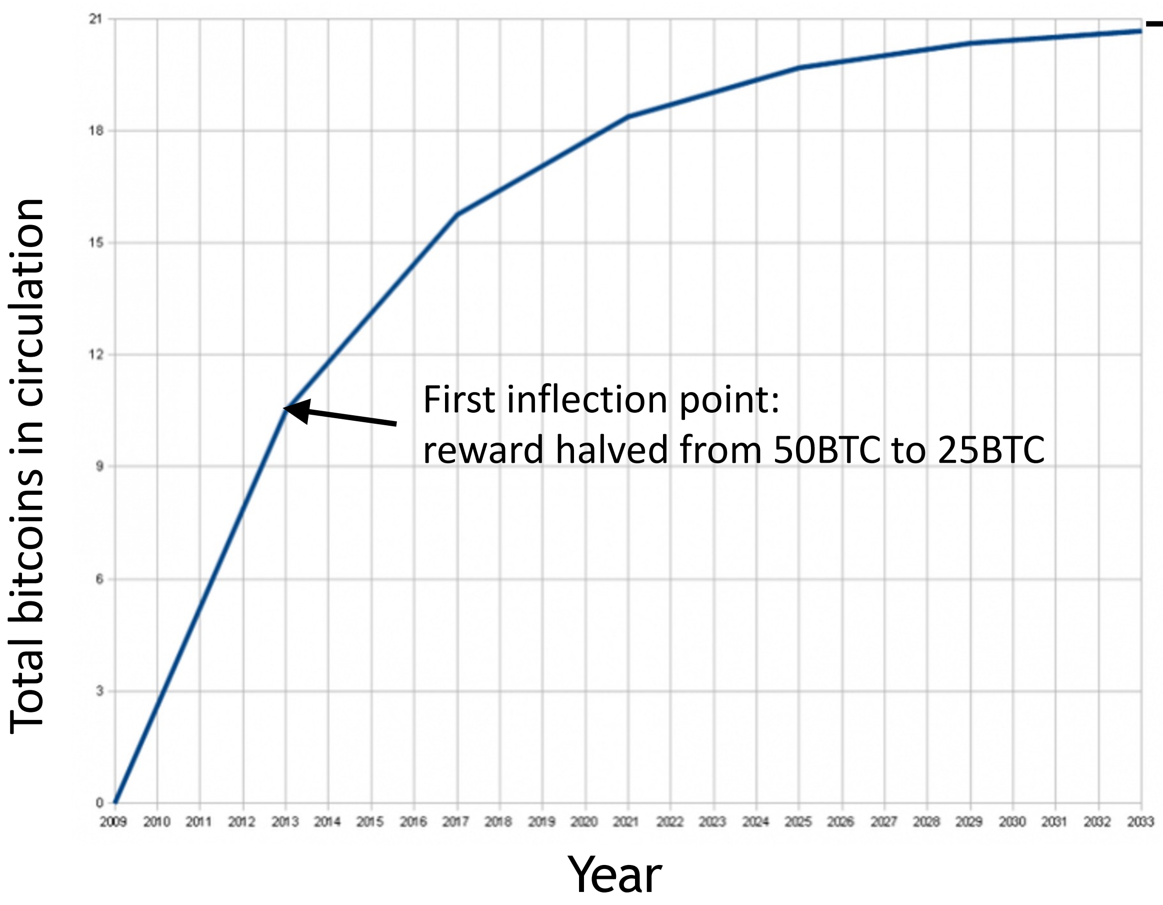
Value is fixed: currently 12.5 BTC, halves every 210,000 blocks created (or every 4 years at the current rate of block creation)

- We are now in the third period – first period block reward was 50 BTC

Block creator gets to "collect" the reward only if the block ends up on long-term consensus branch!

- Subtle but powerful trick: Incentivizes nodes to behave in way that will get other nodes to extend their block

There’s a finite supply of bitcoins



Total supply: 21 million

Block reward is how new bitcoins are created

Runs out in 2040. No new bitcoins unless rules change

Does that mean that after 2040, nodes will no longer have incentive to behave honestly?

Not really!

Incentive 2: Transaction Fees

Creator of transaction can choose to make output value less than input value

Remainder is a transaction fee and goes to block creator (that first puts that transaction into that block)

Purely voluntary, like a tip

- But system will evolve, and will become mandatory, as Block rewards run out

Remaining problems

1. How to pick a random node?

2. How to avoid a free-for-all due to rewards?
   - Everybody may want to run a bitcoin node in order to get this free reward (lock reward and Transaction fee)

3. How to prevent Sybil attacks?
   - An adversary may create a large number of Sybil nodes to subvert the consensus process

Proof of work

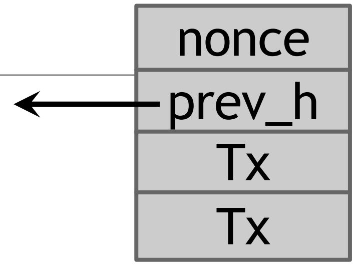
To approximate selecting a random node: *select nodes in proportion to a resource that no one can monopolize (we hope)*

- In proportion to computing power: **proof-of-work** *(Used in Bitcoins)*
- In proportion to ownership of the currency: **proof-of-stake** (*Not used in Bitcoins – but a legitimate model used in other cryptocurrencies*)

Equivalent views of proof of work

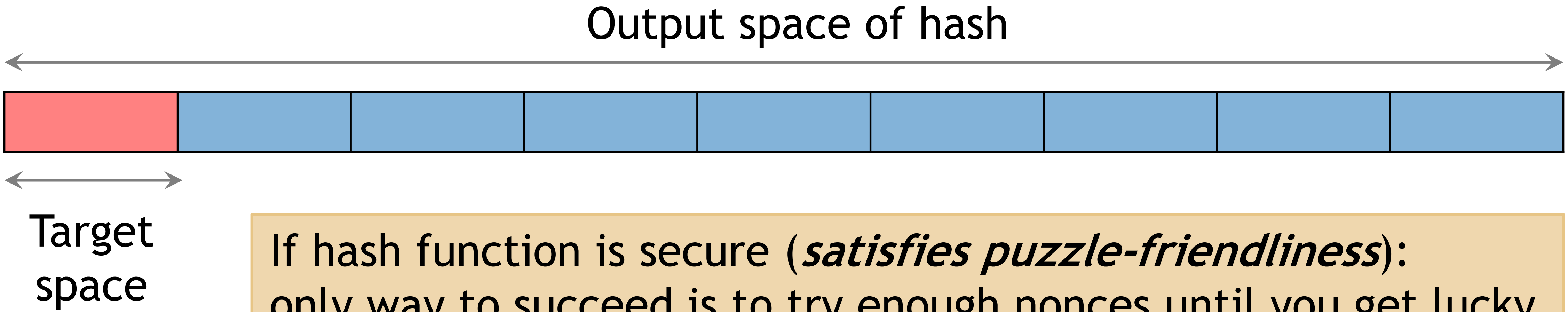
1. Select nodes in proportion to computing power
2. Let nodes compete for right to create block
3. Make it moderately hard to create new identities

Hash puzzles



To create block, find nonce s.t.
H(nonce ‖ prev_hash ‖ tx ‖ … ‖ tx) is very small

In other words, *H(nonce ‖ prev_hash ‖ tx ‖ … ‖ tx)* < *target*



If hash function is secure (***satisfies puzzle-friendliness***):
only way to succeed is to try enough nonces until you get lucky

Advantage of such a PoW system?

It completely does away with the problem of magically picking a random node (to propose a block)

Nodes independently compete by attempting to solve hash puzzles

- Once in a while, one will succeed and propose the next block

Result: Such a system is completely decentralized → No one gets to decide which node proposes the next block

PoW property 1: difficult to compute

Difficulty varies with time

As of 2015: difficulty level is over 10^{20} hashes/block

- i.e., size of target space <= $1/10^{20}$ size of hash’s output space
- Such a computation not possible with commodity laptops

Only some nodes bother to compete — **miners**

- This process of repeatedly solving hash puzzles is called ***bitcoin mining***

Technically anyone can mine → however mining power is concentrated in a mining ecosystem

PoW property 2: parameterizable cost

Nodes automatically **re-calculate the target** (size of target space as a fraction of the output space) every two weeks

Goal: average time between blocks = 10 minutes

In other words, recalculation takes place after 2,016 blocks!

Prob (Alice wins next block) =
fraction of global hash power she controls

Why is such a re-adjustment needed?

It is inefficient if blocks are proposed too close to each other

Would not be able to put multiple transactions in a single block!

Why 10 minutes?

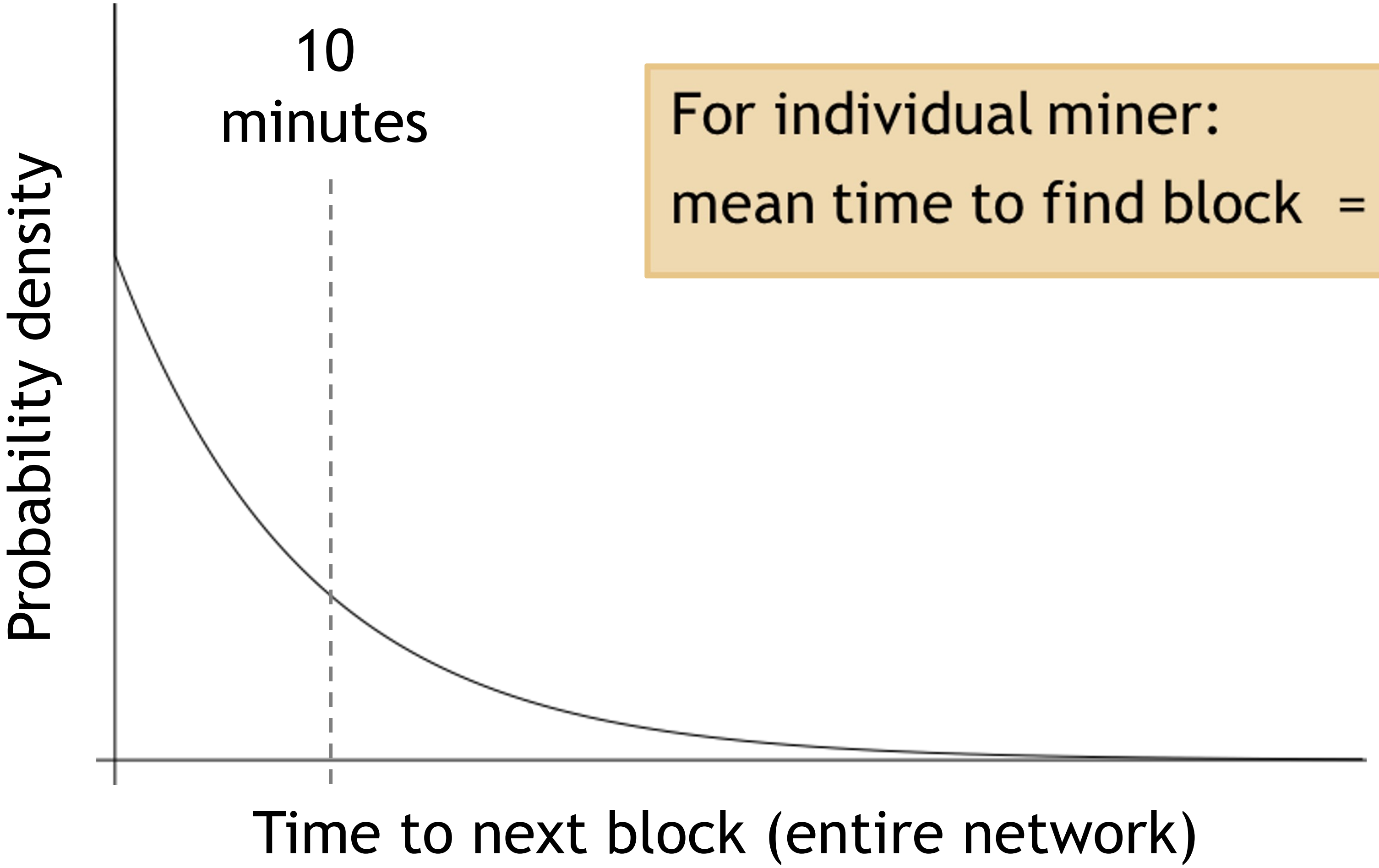
- Not significant!
- Can change it to 5 minutes, and system would still work

Key security assumption

Bitcoin attacks infeasible if **majority of miners** weighted by hash power follow the protocol (or are honest)

This will ensure a more than 50% chance that the next block is proposed by a honest node!

Solving hash puzzles is probabilistic



For individual miner:

$$\text{mean time to find block} = \frac{\text{10 minutes}}{\text{fraction of hash power}}$$

PoW property 3: trivial to verify

Nonce must be published as part of block

Other miners simply verify that
$H(\text{nonce} \parallel \text{prev\_hash} \parallel \text{tx} \parallel \ldots \parallel \text{tx}) < \text{target}$

Advantage?

No centralized verifier needed! Any node or miner can verify that the block was correctly mined

Mining economics

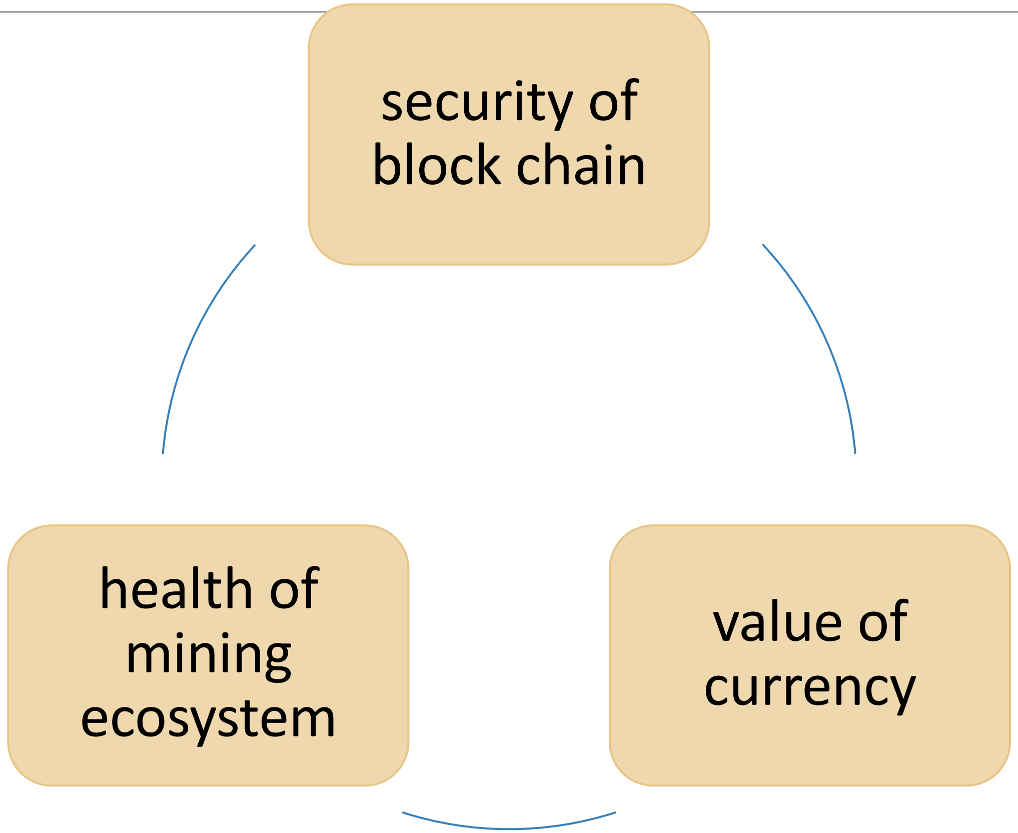
If mining reward (block reward + Tx fees)	>	mining cost (hardware + electricity cost)	→	Profit

Complications:

- Fixed (hardware) vs. variable (electricity) costs
- Reward depends on rate at which miners propose blocks (ratio of their hash rate to the global hash rate)
- Cost in dollars, but reward in BTC → profit depends on exchange rate

Solving more than 10^{20} hashes to obtain 12.5 BTC at current exchange rate is profitable!

Bitcoin is bootstrapped



security of block chain

value of currency

health of mining ecosystem

What can a “51% attacker” do?

Steal coins from existing address?	✗
Suppress some transactions?	
• From the block chain	✓
• From the P2P network	✗
Change the block reward?	✗
Destroy confidence in Bitcoin?	✓✓

References

CS 4593/6463 – Bitcoins and Cryptocurrencies, Prof. Murtuza Jadliwala, University of Texas, San Antonio

Note: most of the slides used in this course are derived from those available for the book "Bitcoins and Cryptocurrencies Technologies – A Comprehensive Introduction", Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller & Steven Goldfeder, 2016, Princeton University Press.