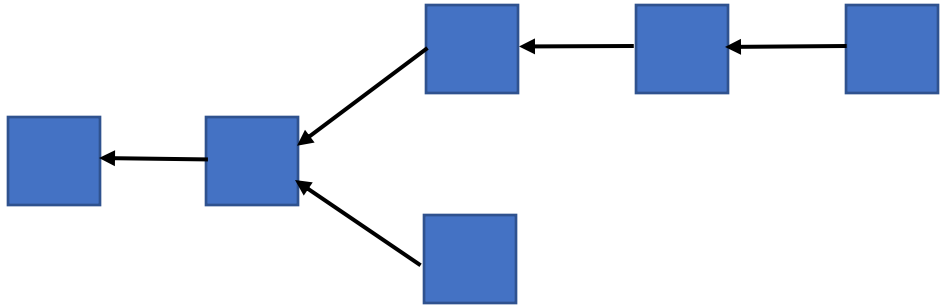
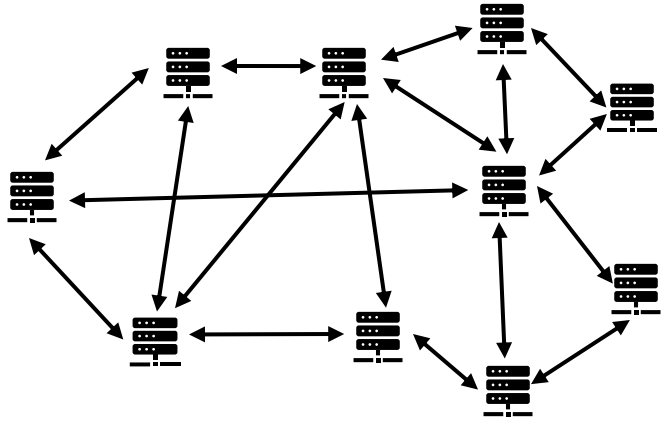
Ethereum and Smart Contracts

CS839 – Kai Mast

Last Time: Nakamoto Consensus





First probabilistic consensus protocol

- Leadership election using Proof-of-Work (or Proof-of-Stake)
- Eventually converges to a single totally-ordered chain of blocks
- Each blocks contains a set of transactions

Operates on a public network

- No global membership registry
- Anyone can join or leave
- Highly resilient against node failures

Limitations with Bitcoin

- Only one "built-in" application: the Bitcoin currency
- Bitcoin script allows some flexibility, but it is not Turing-complete

Can we build an "Internet Computer" that supports arbitrary applications?

(Sometimes called Web 3.0 or Web3)

Ethereum

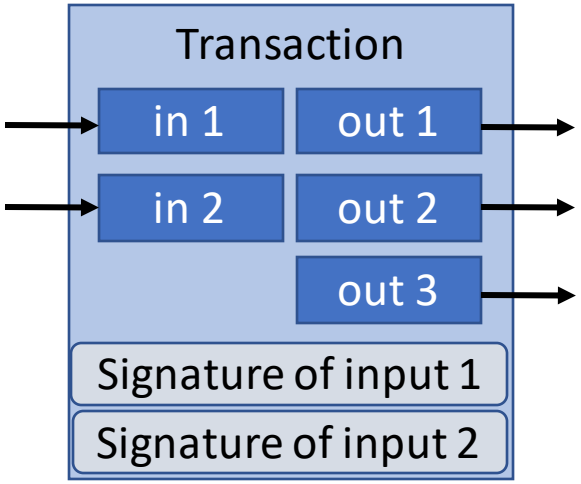
- Proposed in 2013 by Vitalik Buterin, Gavin Wood, Joseph Lubin, and others
- Relies on Nakamoto Consensus with some modifications
  - Ethereum 2.0 is moving to a different protocol
- Introduces the notion of Smart Contracts
- Today, Ether (also ETH or Ξ) is the second most popular/valued cryptocurrency

Vitalik Buterin
(Source: Bloomberg.com)

Gavin Wood
(Source: Web3 Foundation)

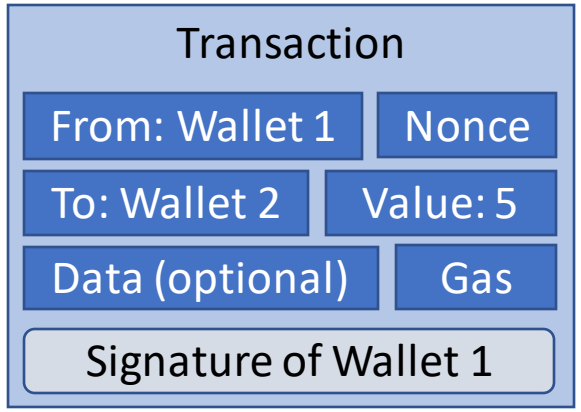
Joseph Lubin
(Source: The Telegraph)

Data Models



UTXO (Bitcoin)

- Generated by a transaction
- Can be used at most once
- Entire value is consumed, if used



Accounts (Ethereum)

- Is controlled by a public key or a contract
- A single account may be involved in many transactions
- Can store arbitrary state in addition to Ether

The Ethereum Virtual Machine

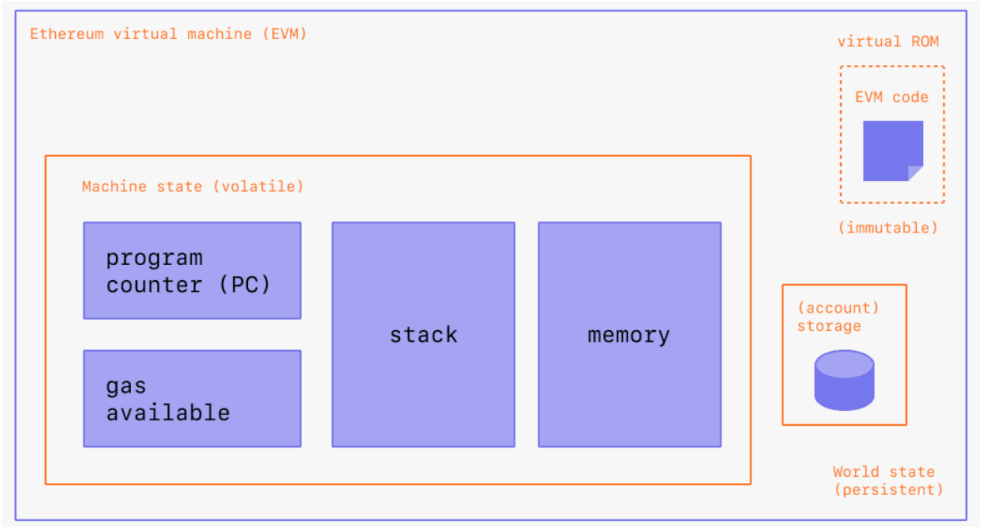
- Interprets Bytecode (similar to Assembly)
- EVM bytecode has low-level instructions, such as CALL, ADD, or RETURN

Pros:

- Less complexity than an interpreted language
  - Smaller attack surface
  - Potentially higher performance
- Allows for different high-level languages to compile to EVM bytecode

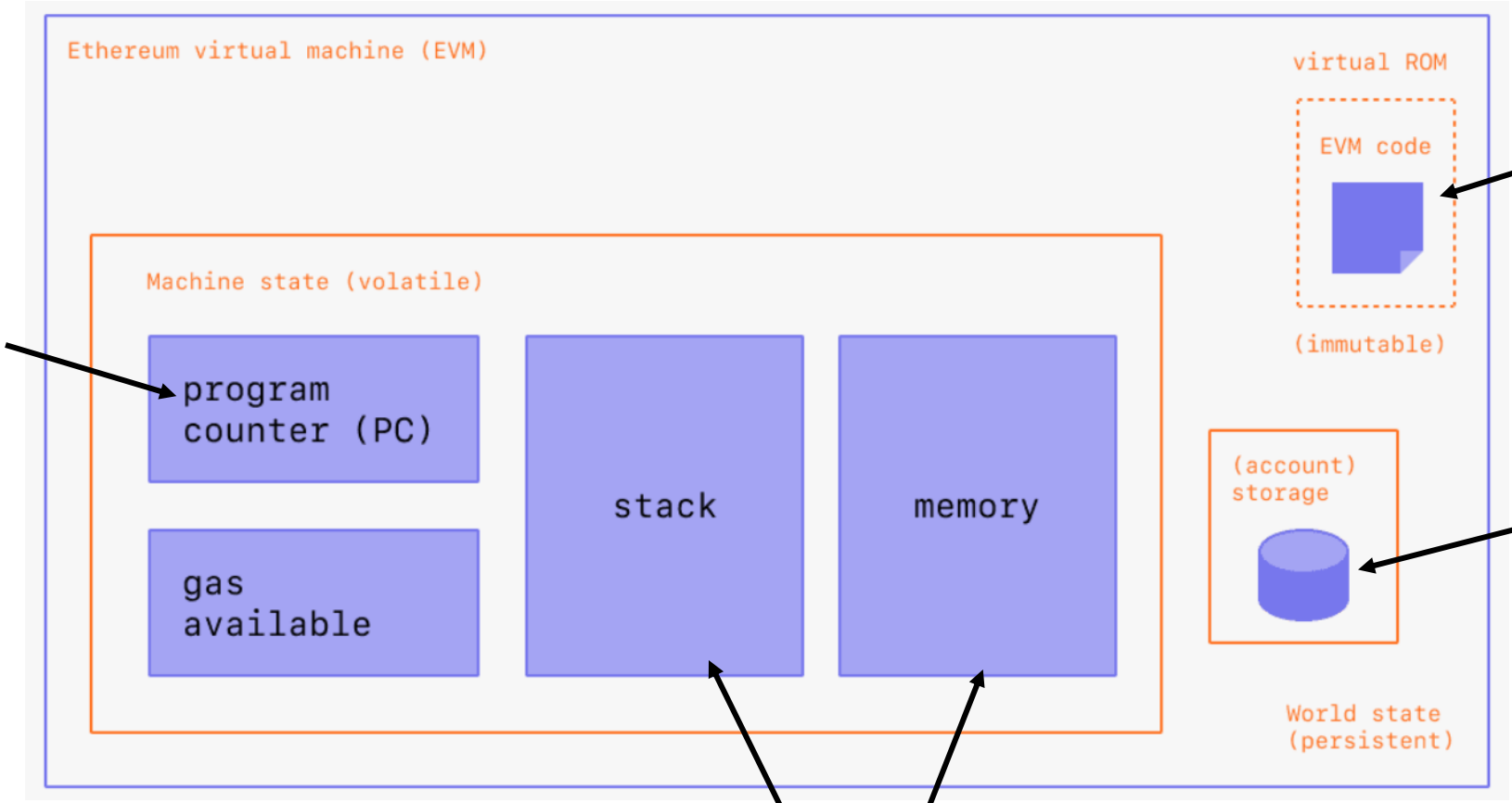
Cons:

- Bytecode is harder to reason about than uncompiled code



Source: Ethereum.org / EVM Illustrated

The Ethereum Virtual Machine



The actual bytecode

Keeps track of where in the program we are

Durable state: Stays around across function invocations

Allows storing temporary state

Gas

- Used to pay for computation
- Issuer of transaction sets gas price and limit
- Different instructions of the EVM have different gas cost
- Transactions that exceed the gas limit abort
- Any unused gas is refunded to the issuer

Logo from ethgasstation.info

Why?

- Takes the role of transaction fees in Bitcoin
- Prevents attackers from issuing infinitely executing code

Smart Contracts

- Allows encoding arbitrary application logic into an account
- Arbitrary state can be stored in the contract
- Functionality is exposed through public functions
- Functions can be invoked by clients or other smart contracts

```
pragma solidity >=0.4.16 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Example from soliditylang.org

Smart Contracts: Function Invocation

How does one call a contract's function?

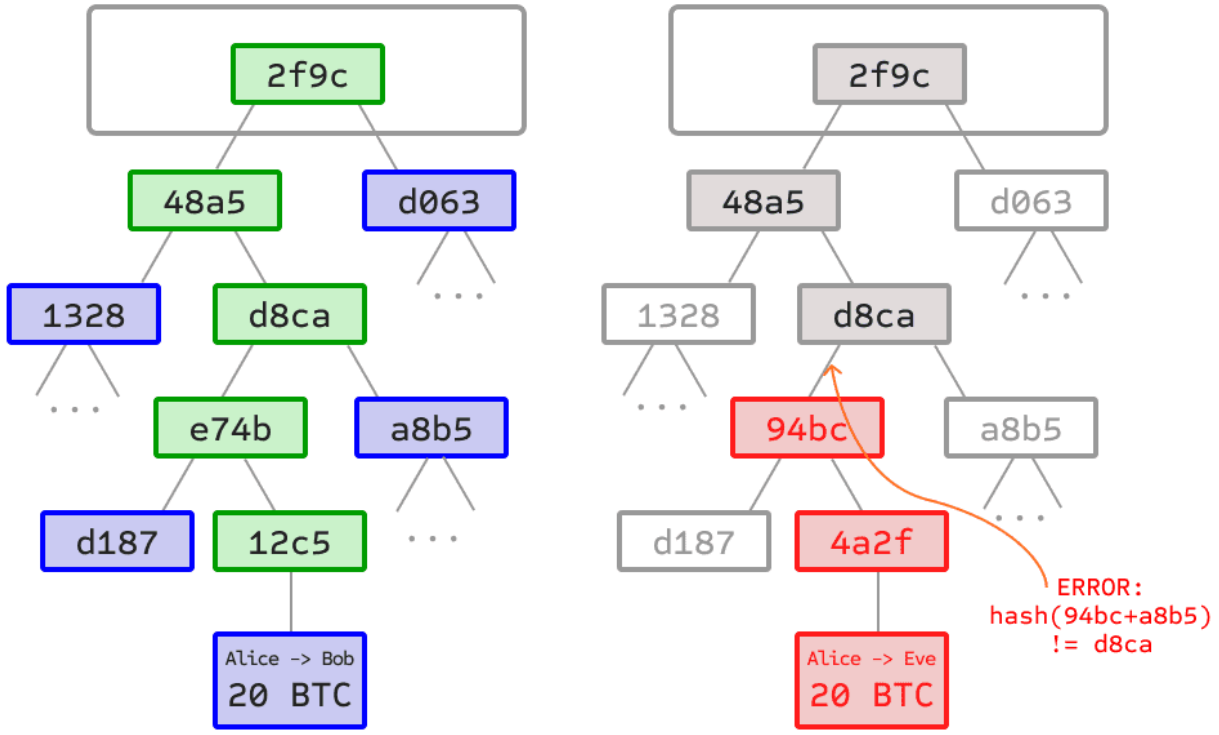
- Issue a transaction with the contract as recipient
- Data contains function call and arguments

How does a contract call another contract?

- The calling smart contract send a message to the other contract
- Messages are not stored on the blockchain explicitly
  - They can be (re-)generated by executing the calling transaction

Recap: Merkle Trees

- Recursively "fold" a list of data items into hashes
- Only root of the tree needs to be stored on the blockchain
- Merkle trees allow generating proofs that can be verified against the tree's root



Source: Ethereum Whitepaper

Recap: Light Nodes

- We may not be able to run a full blockchain node
  - e.g., on embedded devices or phones
- Light nodes only track headers of the blockchain
- State can be verified using Merkle proofs

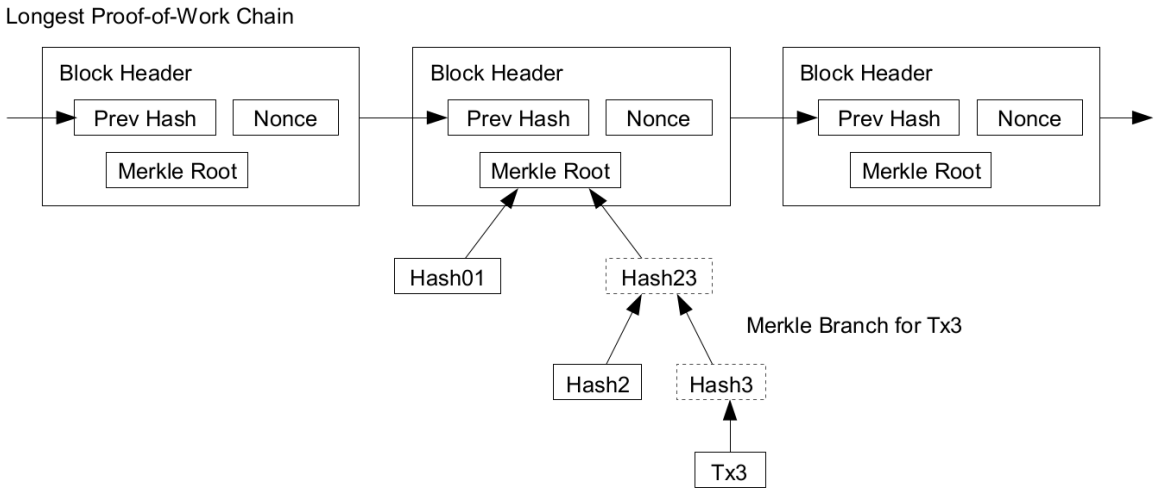
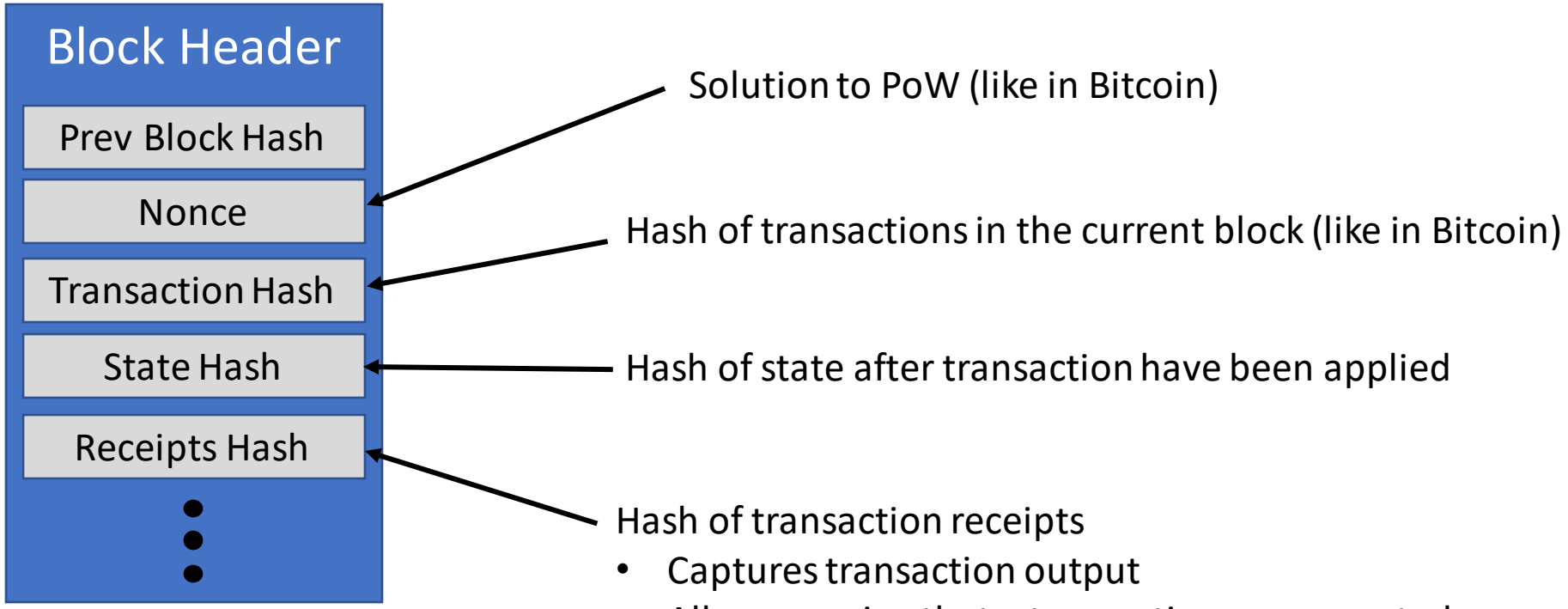


Figure from the Bitcoin Whitepaper

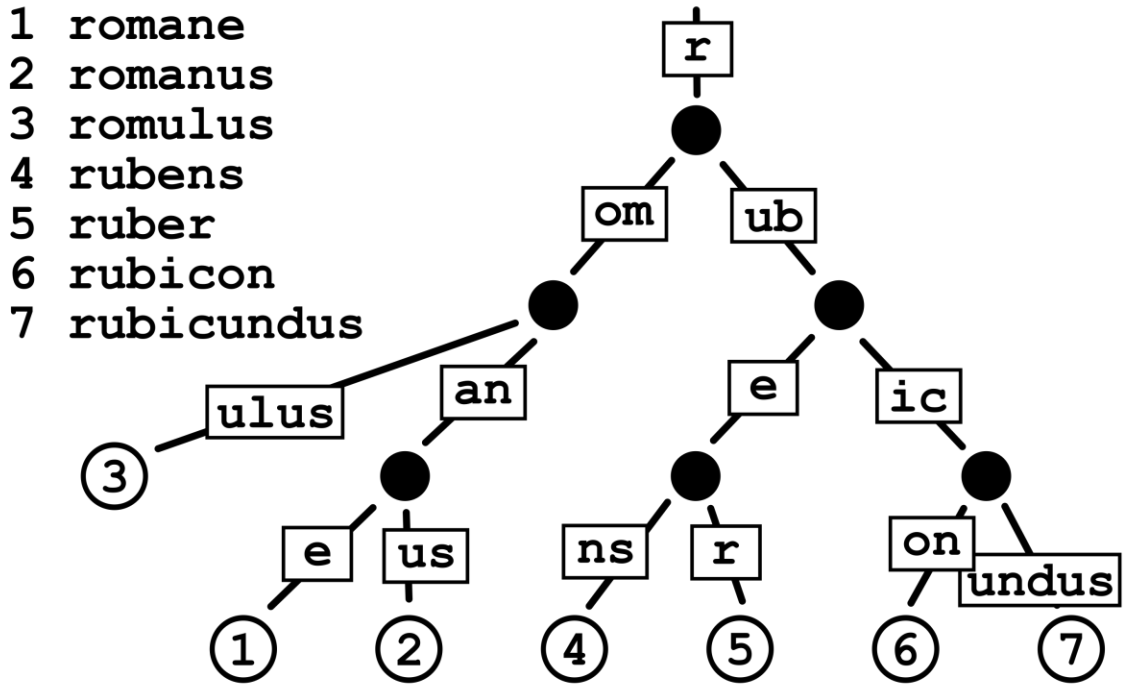
Ethereum State



Patricia Merkle Trees

Why not Merkle trees?

- We need not only proof the existence but also the absence of a data item
- State of the same contract accounts should be stored close together



Patricia Trees are a variant of Radix Trees
(Figure by Claudio Rocchini)

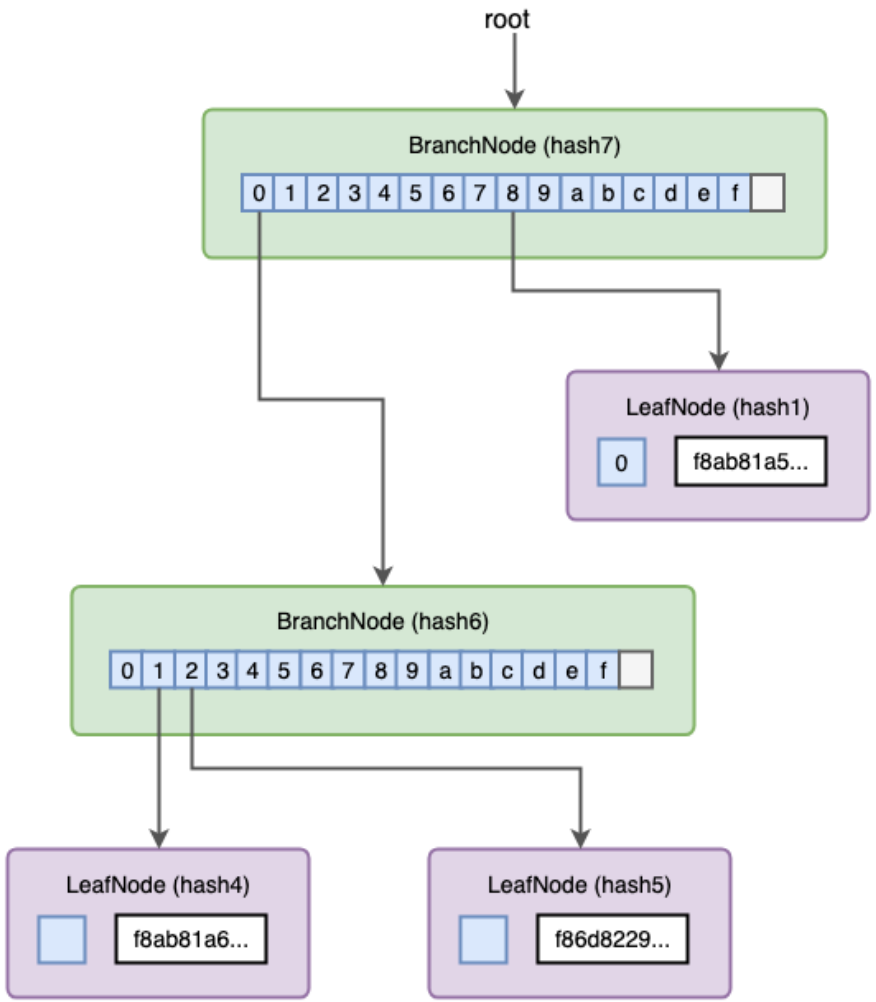
Patricia Merkle Trees

Components of a Patricia Merkle Tree in Ethereum

Branch: Up to 16 children (4 bits) and a value

Path: A single child and a prefix

Leaf: A suffix and value



(Source: Leo Zhang)

Ethereum Block Generation

Goal: Higher throughput and faster confirmation times compared to Bitcoin

Throughput:

- A function of block size, block frequency, and forks
- In Ethereum, blocks are created every 10-12 seconds, not every 10 minutes

Confirmation and Safety:

- Depth (in blocks or mining power) of the transaction indicates confirmation likelihood
- Total mining power behind the longest chain determines safety (in PoW)

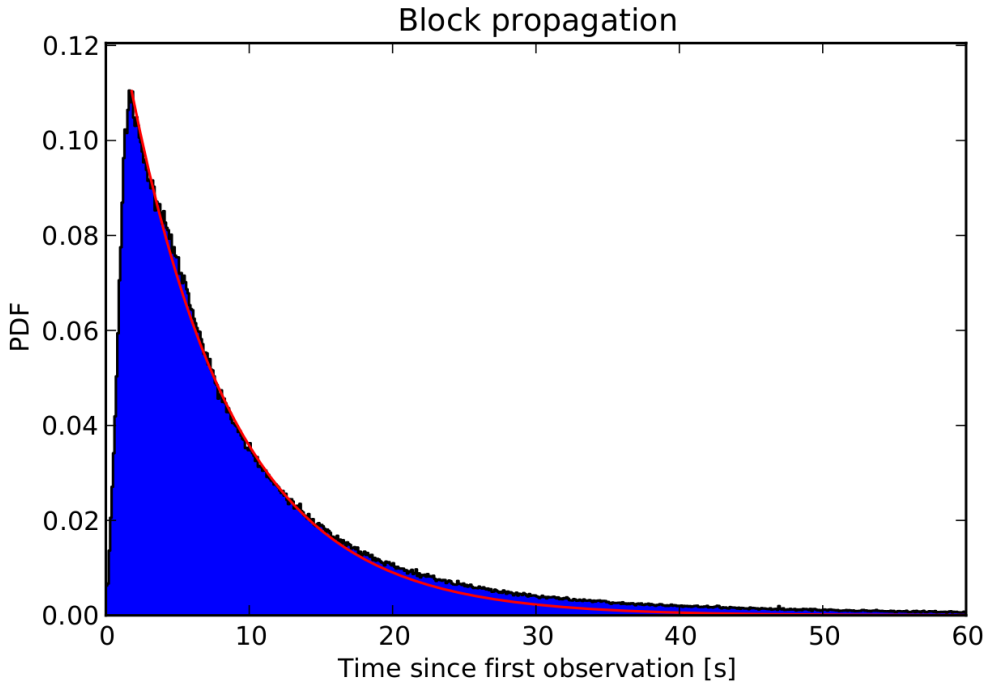
Ethereum Block Generation

"Information propagation in the Bitcoin network" (Decker and Wattenhofer, 2013)

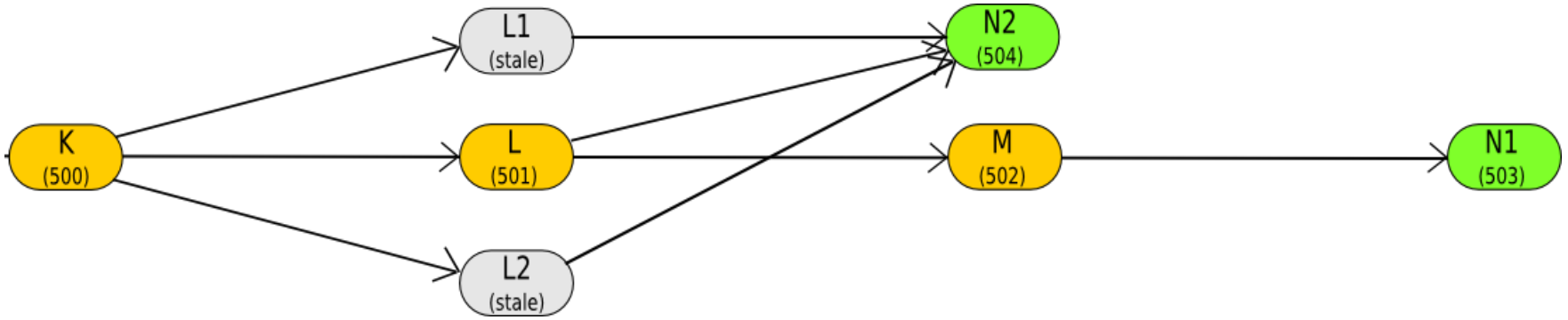
- 95% of nodes can be reached in <13seconds
- 50% of nodes are reached within 6 seconds
- Numbers might be slightly different today

Why does block propagation take so long?

- Nodes verify/execute blocks before forwarding
- Gossip network introduces additional network hops



Greedy Heaviest Observed Subtree (GHOST)



(Source: Vitalik Buterin / blog.ethereum.org)

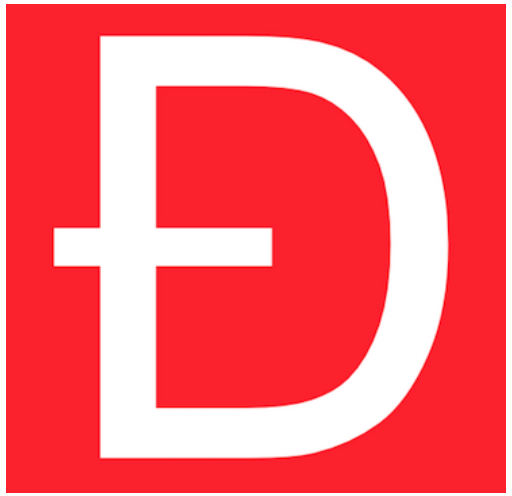
Idea: Leverage orphan (or uncle) blocks to get faster confirmation times

- Blocks do not only reference their direct parent, but also uncle blocks
- Orphan blocks' transactions are not applied to the blockchain state, but their mining power still counts towards the "heaviest" chain

Decentralized Autonomous Organizations

Goal: Establish an organization, e.g., an investment firm, entirely on the blockchain

- One of the first envisioned applications for Ethereum
  - Mentioned in the Whitepaper
- Users pool money in a smart contract
- Majority vote (usually 2/3rds) can spend the money, e.g., to invest



Logo for "The DAO"

Ethereum Governance

Approval Process:

- Protocol changes introduced as "Ethereum Improvement Proposal" (EIP)
- Core developers decide on whether to accept the proposal
- Network as a whole needs to accept proposal to take effect
  - EIP are first adopted on test networks

Applying changes through forks:

- "Soft forks" (backwards compatible)
  - Application-level, API, or ABI changes
- "Hard forks" (backwards incompatible)
  - Changes to the consensus protocol or the blockchain structure
  - Often results in network splits, e.g., Ethereum (ETH) and Ethereum Classic (ETC)

Ethereum Request for Comments (ERCs)

- A kind of EIP that does not require changes to the client or node code
- Like "interfaces" in Object-Oriented Programming
  - Eases interoperability between different contracts
  - Avoids re-inventing abstractions
- Two most important standards
  - ERC20: Fungible Tokens
  - ERC721: Non-fungible Tokens
  - But, many others exist

ERC20 Tokens

```
function name() public view returns (string)
function symbol() public view returns (string)
function decimals() public view returns (uint8)
function totalSupply() public view returns (uint256)
function balanceOf(address _owner) public view returns (uint256
balance)
function transfer(address _to, uint256 _value) public returns
(bool success)
function transferFrom(address _from, address _to, uint256 _value)
public returns (bool success)
function approve(address _spender, uint256 _value) public returns
(bool success)
function allowance(address _owner, address _spender) public view
returns (uint256 remaining)
```

Public Methods of an ERC20 Token

How much can tokens be split up?

Allows giving another party access to your token(s)

When a party gives you access to their tokens, this show how many you can spend

- Specification for smart contracts to generate, sell, and transfer tokens
- Used for ICOs and other custom tokens
- Standard allows for interoperability between tokens

ERC721 Tokens

- Used for NFTs
- Not every token is the same!

CryptoKitties is one of the most popular NFTs

Like ERC20 Tokens, but:

- Every token has an ID
- Tokens cannot be divided

Transaction volume exploded in 2018 due to Cryptokitty trading

That's all for today

- I will start sending out paper questions every Sunday
- Please check in with me about (potential) projects, if you haven't yet