Principles of Distributed Ledgers

Lecture 1: Introduction to blockchains

Lewis Gudgeon, **Daniel Perez**, Paul Pritz, **Sam Werner**

January 20, 2023

Imperial College London

Course logistics

Lecturers

Lewis Gudgeon

Daniel Perez

Paul Pritz

Sam Werner

Outline

Weekly sessions:

- Fridays: 2 hour lecture
- Tuesdays: 2 hour tutorial/lab

Reading material:

- Required and recommended reading material will be posted online for each week

Guest lecturers:

- Week 5: Chainlink
- Week 8: Patrick McCorry

Note: In week 4 there will be no tutorial. Please bring a laptop to the lecture in week 4.

Coursework, labs, exam

Labs:

- Based on the lecture material of the same week
- Programming-focused

Coursework:

- Week 6
- Individual

Exam:

- 2 hour exam
- Open book (1 A4 sheet)

External Students – Registration for DoC Courses

1. Apply at: `https://infosys.doc.ic.ac.uk/externalreg/`
   - Course not on this list? It is not available (**do not** approach individual lecturers)
2. Then, your department's endorser will approve/reject your application
3. If approved,
   - DoC's External Student Liaison will approve/reject your application
4. If approved (again!),
   - Students will get access to DoC resources (DoC account, Scientia, ...) + added to MS Team of module (only if necessary)
   - No access after a few days? Check status of approval and contact relevant person(s)

Key Dates

- Exams for DoC $3^{rd}/4^{th}$ yr. courses take place at the end of the Term in which the course is taught (week 11)
- Registration for Spring term exams opens end of January

History of blockchain

Timeline

Bitcoin whitepaper

Various blockchains launch

DeFi Summer

2008

2009

Initial Bitcoin release

2012

2017

ICO boom

2020

2022

Toughest year for DeFi

Three Arrows Capital

2008: Bitcoin whitepaper

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

1. Introduction

Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot

Creation of Bitcoin

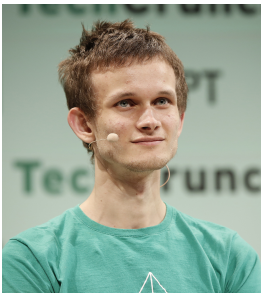
> *"A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution."*

Why was it novel?

- Trustless
- Permissionless
- Censorship resistant
- No double spending!

2013 - 2015: Creation of Ethereum

- **2013**: Vitalik Buterin introduces “Ethereum”
- **2015**: Initial Ethereum release
- Vitalik wanted a more powerful scripting language for application development
- Develop **smart contracts**
- Ethereum Vitual Machine (EVM) compiles smart contract code into EVM bytecode



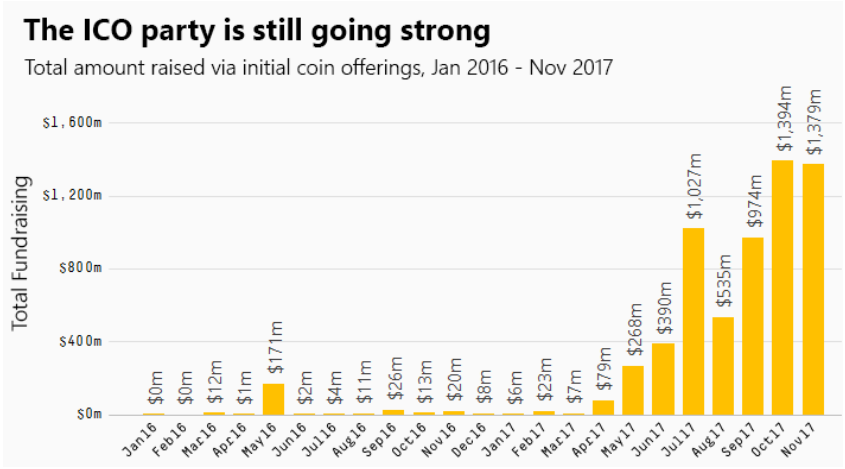
Smart contracts

- Nick Szbabo (1994): "a computerised transaction protocol that executes the terms of a contract."
- Objects that live on the blockchain
- Building blocks of Ethereum applications
- Automatically execute code when "contract terms" are met
- Transactions can update the state of a smart contract

Building Tokens

- Smart contracts started to be used to create tokens
- "Tokens" are smart contracts that contain the logic for storing and updating balances of token holders
- Digital assets that run on top of an existing blockchain
- Anyone can trade tokens

2017: The ICO boom



Source: https://www.visualcapitalist.com/video-ico-explosion-one-animated-timeline/

2020: Emergence of Decentralised Finance (DeFi)

"Decentralized Finance (DeFi) is a peer-to-peer powered financial system."

Properties of DeFi

Properties of an idealised DeFi ecosystem:

- Non-custodial
- Permissionless
- Openly auditable
- Composable

Applications?

- Non-custodial stablecoins
- Protocols for loanable funds
- Derivatives and other financial products
- ...

The promise of DeFi

DeFi Total Value Locked Hits All-Time High of $236 Billion

Explained: How DeFi could one day liberate finance

Silicon Valley bets on crypto projects to disrupt finance

DeFi – The Future of Finance

How NFTs could be the future standard for trading and investing

DeFi Optimist

How decentralized finance will transform business financial services – especially for SMEs

The Simplification of DeFi Products Will Cement It as the Future of Finance

Coinbase is launching a marketplace for NFTs

Why NFTs are the future of creative expression

UniSwap V3 the Top Defi Exchange Facilitating 4000X Capital Efficiency

The struggles of DeFi

Global regulators target blockchain-based 'decentralised finance'

Anyone Seen Tether's Billions?

Regulatory risks grow for DeFi as a 'money laundering haven'

DeFi Protocol Compound Mistakenly Gives $162 Million To Users, CEO Begs Them To Give It Back

CREAM Finance Exploited for $130M in DeFi's Third-Largest Hack

Legislation on stablecoins needed 'urgently', say top US regulators

Defi Protocol Harvest Finance Hacked for $24 Million, Attacker Returns $2.5 Million

WANTED! $1m bounty on offer for information on cryptocurrency firm tether's so-called 'stablecoin' backing

$600 million gone: The biggest crypto theft in history

China 'Banned' Crypto. Can The SEC Try Doing The Same?

DeFi Protocol Pickle Finance Hacked For $20 Million

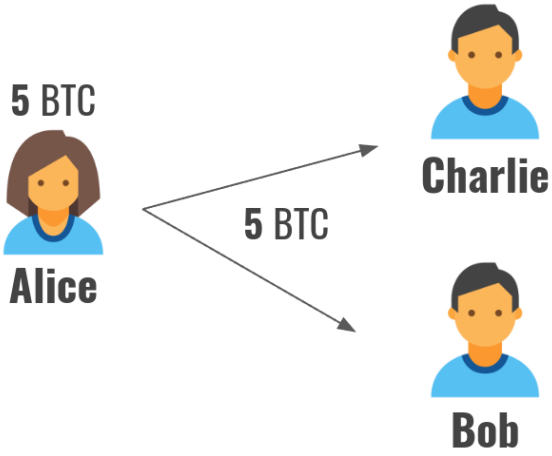
DeFi Pessimist

Binance Chain DeFi Exchange Uranium Finance Loses $50M in Exploit

90% of NFTs Will Be Worthless in 3 to 5 Years, Coinbase Cofounder Warns

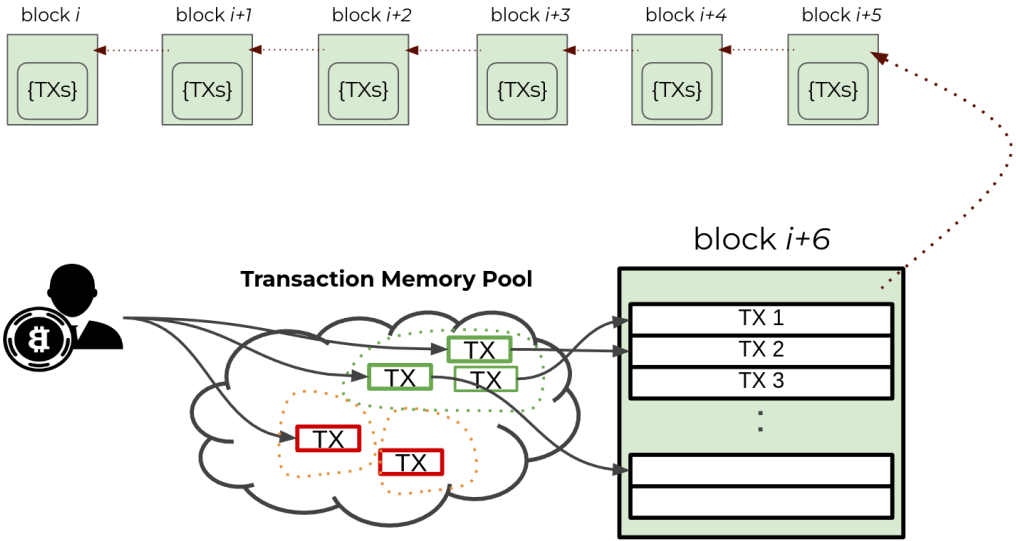
How Bitcoin works

Double-spending problem



What is a blockchain?

- A data structure that stores information, such as transaction data
- Peer-to-peer network
- Data is recorded in multiple identical data stores (ledgers) that are collectively maintained by a distributed network of computers (nodes)
- Consensus algorithm (all nodes see the same data)


block i
block i+1
block i+2
block i+3
block i+4
block i+5
{TXs}
{TXs}
{TXs}
{TXs}
{TXs}
{TXs}
block i+6
Transaction Memory Pool
TX
TX
TX
TX
TX
TX 1
TX 2
TX 3

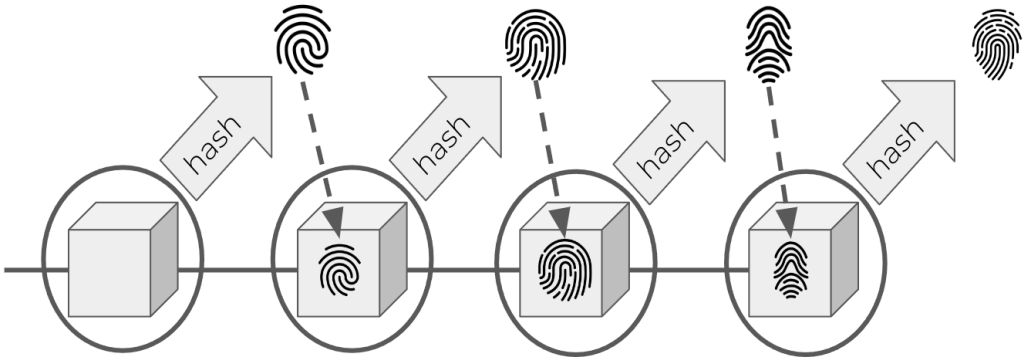
What's a block?

A data structure that stores information, such as transaction data.

A block consists of:

1. `block header`:
   - Identifies a particular block of transaction
   - Serialized in 80 byte format
   - Format:
     `<version><previous_block_header_hash><merkle_root_hash><time><...><nonce>`
2. `txn_count`: total number of transactions
3. `txns`: every transaction in the block

A block is not valid unless serialised size is $<=$ 1MB.


hash
hash
hash
hash

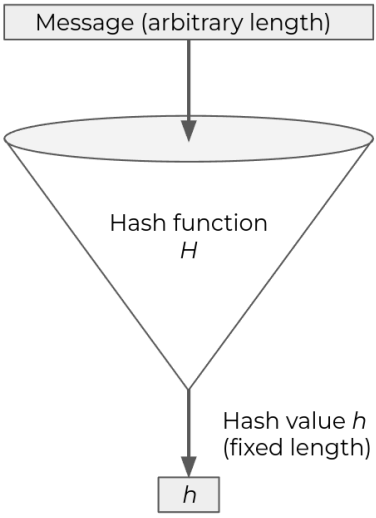
Hash functions

A one-way deterministic function for mapping input data of arbitrary size to a fixed-size bit string.

Characteristics:

- **Pre-image resistance:** For a given hash value h in the output space of hash function H, it is hard to find any input x such that $H(x) = h$.
- **Second pre-image resistance:** For an input value x and a hash function H, it should be hard to find a different input y, such that $H(y) = H(x)$.
- **Collision resistance:** For a hash function H, it is hard to find two different inputs x and y, such that $H(x) = H(y)$.

Hash functions



Hash functions

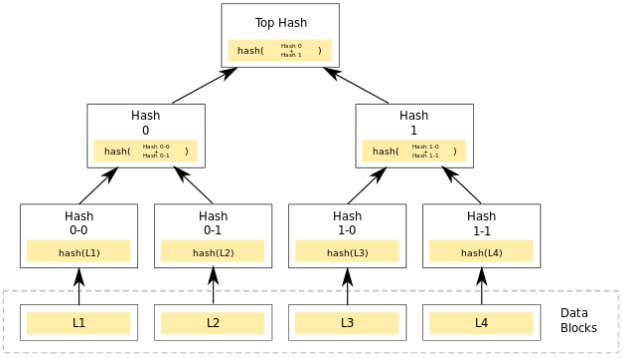
Examples:

- SHA-256(*"Short input"*) $\rightarrow$
  `0x762c13ac8b2afa3f449e06c3413f703d8e545b095471764c224380e25b14b0e0`
- SHA-256(*"A very long input with all sort of characters: ¨¥f©ø¶§f˙f"*) $\rightarrow$
  `0xd0caa87584c3e073fbb8b0986e3593397980568fdeb921d22ad38c8e1039dec1`

Merkle tree

Merkle trees are used to encode a set of values into a single hash: the Merkle root

- Created by hashing nodes together starting from leaves
- Allow for logarithmic time inclusion proof generation
- Allow for logarithmic time/storage inclusion proof verification



Example merkle tree

Proof of Work (PoW)

- **PoW**: a leader election process in which participating nodes (*miners*) invest computational power in solving cryptographically hard, memoryless puzzles
- The node that generates a valid PoW solution determines the new set of valid transactions that should be added to the next block
- Partial pre-image attack on SHA256 in Bitcoin (generating a valid solution is hard, verifying a hash against a pre-image is easy)

Proof of Work

Proof-of-Work $\Rightarrow$ `SHA-256(SHA-256(<block_header>)) <= Target (a number from 0 to 2^256-1)`

<version><previous_hash><time><…>**<nonce>**

- Probability of a hash being a solution: $p = \frac{Target}{2^{256}}$ (binomial process)
- Expected number of hashes: $E[Numberofhashesperblock] = \frac{1}{p} = \frac{2^{256}}{Target}$
- Block solve-times follow a Poisson distribution (mean = 10 minutes)
- Important to maintain a stable block solve-time for TX throughput
- *Difficulty* represents how difficult the current target makes it to find a block, relative to how difficult it would be at the highest possible target
- Difficulty algorithms adjust the target (easiness) of the PoW puzzle

Mining

Miners are nodes that try to solve the PoW puzzle.

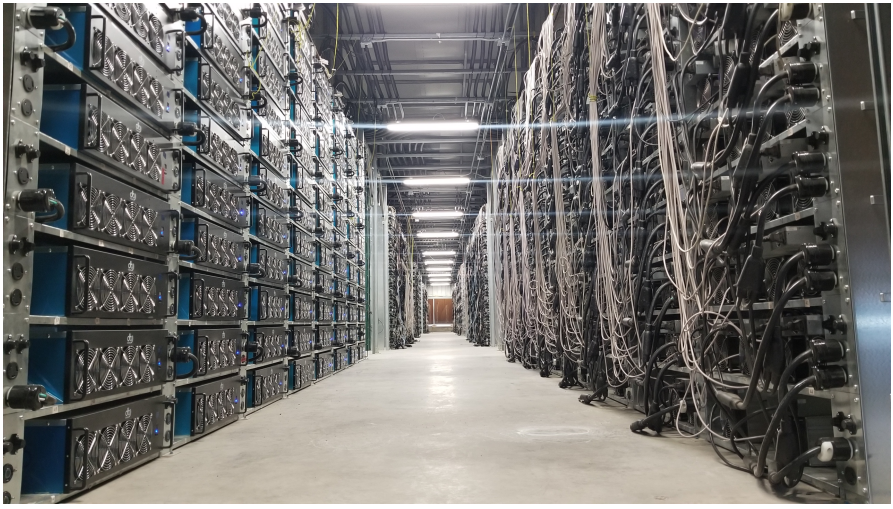
Nodes are incentivised to select transactions because of:

- Transaction fees (difference between TX outputs and TX inputs)
- Block reward (newly minted coins per new block)

In Bitcoin:

- Finite supply of 21 million BTC
- Reward halves approx. every 4 years
- Current reward is 6.25 BTC per block

Mining in practice



Source: https://www.americanbanker.com/payments/news/a-greener-approach-to-crypto-mining

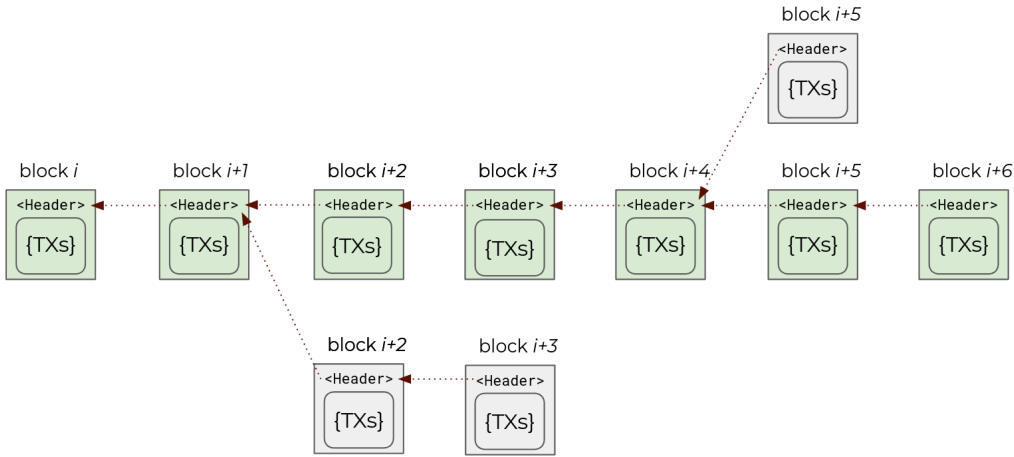
“Longest chain” rule

Longest chain rule == accept the “correct” version of the transaction history (blocks)

- Allows nodes to reach agreement on which history is the right one
- How many hashes one would have needed to perform to mine each block

Block tree: all valid blocks whose previous linked blocks are known (up to genesis block).

Active chain: a single path from the genesis block to a leaf node of the block tree (every path is a valid path), however, nodes pick the one path with the most “work” performed (sum of the difficulties).


block i+5
<Header>
{TXs}
block i
<Header>
{TXs}
block i+1
<Header>
{TXs}
block i+2
<Header>
{TXs}
block i+3
<Header>
{TXs}
block i+4
<Header>
{TXs}
block i+5
<Header>
{TXs}
block i+6
<Header>
{TXs}
block i+2
<Header>
{TXs}
block i+3
<Header>
{TXs}

Forks

Soft fork:

- Protocol changes that remain backward compatible with older protocol versions (clients adhering to the previous protocol rules)
- Example: a stricter limit on the block size (e.g. 0.5 MB instead of 4MB)
- Generally just requires the majority of miners to upgrade

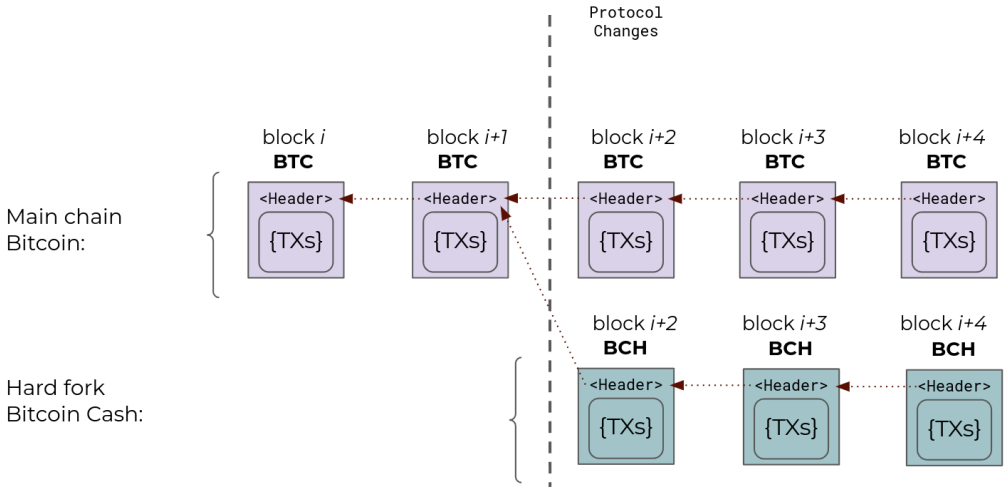
Hard fork:

- Protocol changes which can incur a permanent split of the blockchain (not backward compatible) – blocks considered invalid under previous protocol rules
- A subset of network participants will reject branches that build on blocks that are invalid to them
- Examples: increased block size

Case study: Bitcoin Cash (BCH)

- On 1 August 2017 (block number 478559), Bitcoin hard forked into Bitcoin (BTC) and Bitcoin Cash (BCH)
- Bitcoin Cash implemented a larger block size (from 1 MB to 32 MB), new difficulty algorithm, etc.
- Holders of BTC received an equal amount of BCH on the "new" chain

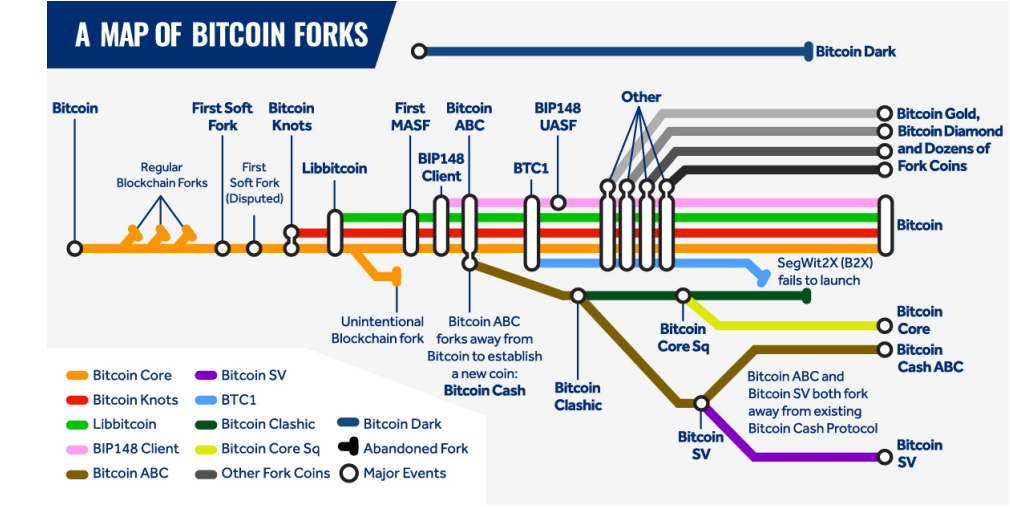
BTC and BCH fork



Post-fork price: BTC

Post-fork price: BCH

Bitcoin forks



Source: https://www.visualcapitalist.com/major-bitcoin-forks-subway-map/

Transactions

- Created $\rightarrow$ propagated $\rightarrow$ validated $\rightarrow$ added to the ledger of transactions
- Publically broadcasted (no confidential information)
- Data structures that encodes the transfer of funds from an input source to an output destination
- 300 - 400 bytes of data

Structure:

`<version><input_counter><inputs><output_counter><outputs><lock_time>`

Unspent Transaction Output (UTXO)

- Fundamental building block of a transaction
- Chunks of BTC that are locked to some address
- Whenever someone receives BTC, this amount is recorded as a UTXO
- Transactions consume one or more UTXOs
- UTXOs must be consumed entirely
- Wallet applications typically combine many small UTXOs into one that is equal or larger than the desired transfer amount
- Sending BTC == creating a UTXO that belongs to the recipient

Unspent Transaction Output (UTXO)

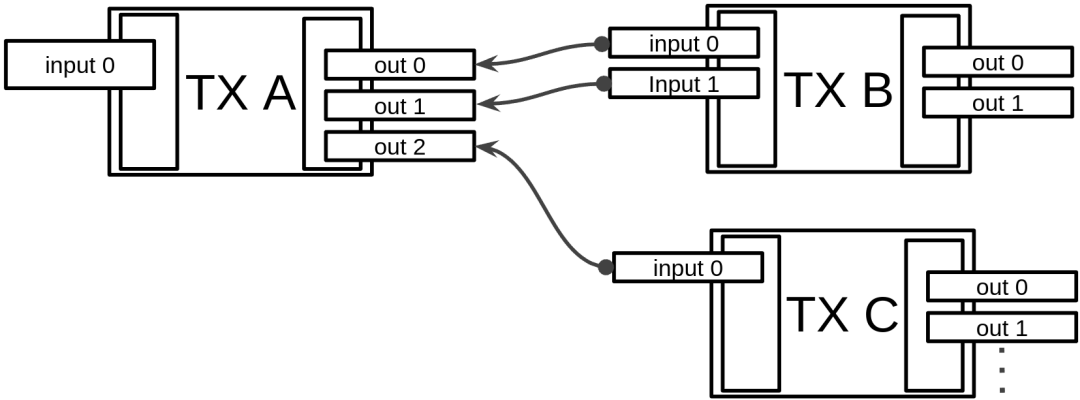
Transaction **inputs**:

- UTXOs consumed by a transaction (reference to tx hash + sequence number)
- Unlocking scripts that meet spending conditions (e.g. signature proving ownership)

Transaction **outputs**:

- UTXOs created by a transaction (amount + locking script)

Transaction fees: inputs - outputs


input 0
TX A
out 0
out 1
out 2
input 0
Input 1
TX B
out 0
out 1
input 0
TX C
out 0
out 1

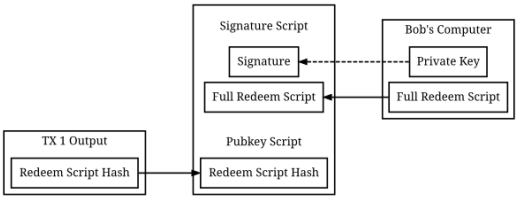
Bitcoin scripting

- Bitcoin uses a scripting language
- Stack-based
- Has conditional jump
- Not Turing-complete (no loop)
- Mostly used to define how to spend Bitcoins
- Transaction is invalid if script fails

```
OP_2
41 ; push 65 bytes
PUBKEY1
41
PUBKEY2
41
PUBKEY3
OP_3
OP_CHECKMULTISIG
```

Bitcoin Scripting and UTXO

- Bitcoin follows an UTXO model
- Bitcoin uses "redeem scripts" to enable scripting with UTXO
- Only a hash of the script is published beforehand
- Full scripts are published when spending



Using a Pay to Script Hash (P2SH)

Flow of a Bitcoin redeem script

1. Write script
2. Hash script to create address
3. Receive Bitcoins
4. Publish script and required data (usually signature) using a transaction

From Bitcoin Scripting to Smart Contracts

We need more features to write general programs

- Persistent state
  - → account-based
  - → storage primitives
- Turing-completeness (loops)
  - → jump primitive
- More transparency?
  - → code deployed before usage

How Ethereum works

Similarities with Bitcoin

Bitcoin has many similarities with Ethereum:

Data structure Also uses a chain of blocks pointing to a parent using its hash

Forks Forks are formed in the same way in Ethereum and Bitcoin

Proof-of-work Until recently, Ethereum used proof-of-work similarly to Bitcoin

The case of Ethereum classic

TheDAO hack (2016)

- TheDAO raised ~$150M in ICO
- Soon after, it got hacked ~$50M
- Price of Ether halved
- Ethereum community decided to hard-fork

Ethereum Classic

- Ethereum Classic is a version of Ethereum that does not revert TheDAO hack
- Ethereum Classic still runs its network in parallel to Ethereum

Main differences with Bitcoin

- Proof-of-Stake
- Block rewards
- Accounts
- Virtual machine

vs

Bitcoin
(BTC)

Ethereum
(ETH)

Proof-of-Stake

- Proof of stake is an alternative consensus mechanism to proof-of-work
- Validators “stake” some ETH to participate in the network
- Each “epoch”, a validator is randomly selected to create a new block
- Other validators verify and attest new blocks
- If a validator produces an invalid block, his ETH stake is “slashed”

Pros and cons of proof-of-stake

Pros

- Energy efficient (no mining)
- Low barrier to entry (only 32 ETH)
- Typically more decentralized
- Requires less participation incentive

Cons

- Not as battle tested as proof-of-work
- Generally more complex
- Larger attack surface

Validator rewards

- Validators receive rewards for staking their ETH and participating in the network
- Rewards are proportional to the percentage of total staked ETH the validator has staked

Reward types

- source vote: voted for the correct source checkpoint
- target vote: voted for the correct target checkpoint
- head vote: voted for the correct head block
- sync committee reward: participated in a sync committee
- proposer reward: proposed a block in the correct slot
- block fees: fees associated with a particular block

Validators penalties and slashing

Penalties

Validators are penalized when they do not vote in a timely manner

- Validators are penalized the amount they should have received if they are late for a source or target vote
- Validator are not penalized if they miss a head vote or are late to propose a block

Slashing

Validators are slashed and forcefully removed if they, that is if they:

- Propose and sign two different blocks for the same slot
- Attest to source/target blocks that "surrounds" another one
- Double vote by attesting to two candidates for the same block

Proof-of-stake and finality

"Finality" means that a block cannot be changed (without burning significant funds) anymore

- In proof-of-work, probabilistic finality is used: final after n blocks
- In proof-of-stake, the first block of each epoch is a "checkpoint block"
- Two checkpoint blocks have a "supermajority link" if they are both attested by at least 66% of the total ETH staked
- A transaction is finalized if it is included between two checkpoint blocks that have a supermajority link

Ethereum account types

There are two types of accounts in Ethereum:

Externally owned accounts

- Account is created by generating private/public key pair
- Address is derived from the public key
- Can initiate transactions

Contract accounts

- Deployed as smart contracts and controlled by their code
- Do not have an associated private key
- Cannot initiate transactions

Ethereum accounts

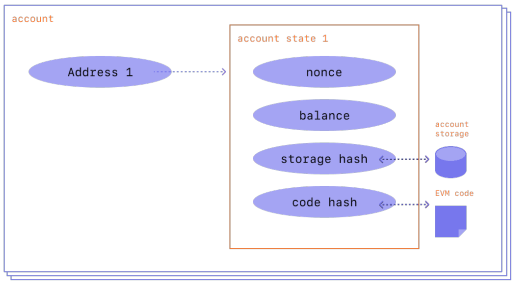
An Ethereum account has the following information associated with it:

Nonce Number of transactions sent from account

Balance Number of wei (1 ETH=1e18 wei) owned by this address

codeHash The hash of the code for this account (empty for EOA)

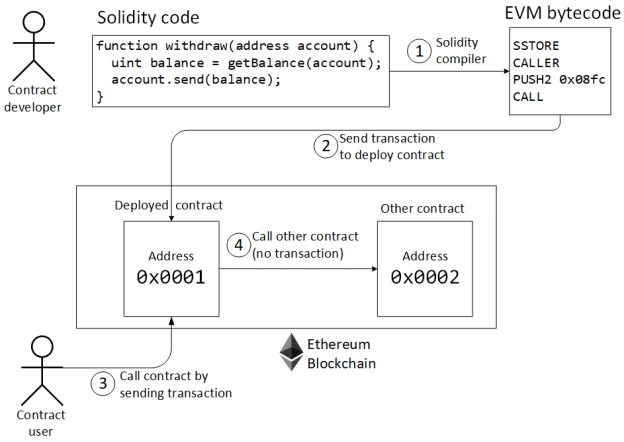
storageRoot Hash of the storage (represented as a Merkle tree) of this account



Ethereum account overview (from Ethereum docs)

Smart contracts

- Programs deployed on a blockchain
- Usually written in a high-level language and compiled into bytecode
- Interacted with using transactions
- Uses some metering mechanism



Smart contract example

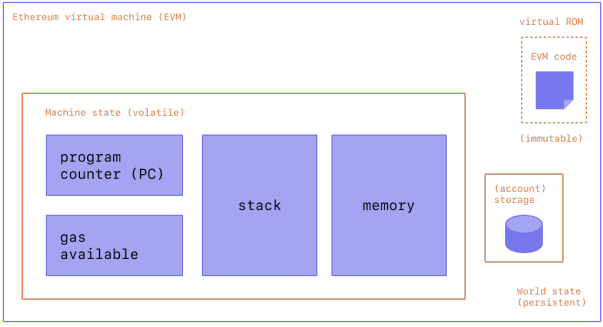
```
contract Coin {
  address public minter;
  mapping (address => uint) public balances;
  constructor() { minter = msg.sender; }

  function mint(address receiver, uint amount) public {
    require(msg.sender == minter);
    require(amount < 1e60);
    balances[receiver] += amount;
  }

  function send(address receiver, uint amount) public {
    require(amount <= balances[msg.sender]);
    balances[msg.sender] -= amount;
    balances[receiver] += amount;
  }
}
```

Ethereum Virtual machine

- Stack-based virtual machine
- Very low-level: no functions, only jumps, no types
- Has regular VM instructions: `ADD`, `SUB`, `PUSH`, `POP`
- Has instructions to interact with environment: `SENDER`, `CALL`, `BALANCE`
- Has ephemeral and permanent storage
- Uses 256-bits words



EVM overview (from Ethereum docs)

Alternative chains

Scalability

The throughput of a blockchain can be informally defined as

$$\text{transactions/second} = \frac{\text{number of transactions per block}}{\text{block time}}$$

where the block time is the number of seconds between two blocks.

- Bitcoin: ~7 tps
- Ethereum: ~25 tps

The main trade-off is that the higher the throughput, the harder it is for validators to participate, and the more centralized the system becomes.

Alternative blockchains

Many other blockchains exist, some of which are trying to improve on the scalability/throughput issues seen by Bitcoin or Ethereum.

Solana

- Launched in 2020
- Used mainly for DeFi and NFTs
- Focus on speed: block time of 400ms, ~5,000 TPS
- Uses Proof-of-Stake as its main consensus algorithm
- Uses Proof-of-History, which provides a safe way to timestamp messages
- Relatively centralized, 35% of network controlled by small set of validators
- Relatively frequent outages (12 so far)

Cardano

- Launched in 2017
- Very research-heavy, many top-tier papers published in the process
- One of the first proof-of-stake blockchain
- Somewhat slow development, smart contracts only recently added
- Not a lot of usage at this point

TRON

- Launched in 2018
- Uses delegated proof-of-stake
- High-throughput (~2k tps), low fees
- Used a lot for games
- Fairly centralized: top nodes have massive amount of control

Reading Material

Reading Material

Required reading:

- Bitcoin whitepaper
- Bitcoin SoK
- Ethereum whitepaper

Outlook

Tutorial: Implementing a simple blockchain node

- Load blockchain state from file
- Produce new blocks (load transactions from file) using proof-of-work
- Write new state to file

Can use any language but we only help with Python and Golang

Next week's lecture

Next week's lecture will be about the EVM, Ethereum smart contracts, and Solidity programming.