Writing Smart Contracts
01 Introduction

Peter H. Gruber

Supported by the Algorand Foundation

Blockchain = a digital ledger

	Alice	Bob	Charlie
In the beginning, Alice has 100 coins and Bob 50			
Alice pays 10 coins to Bob for bread			
Charlie repairs the house of Alice for 30 coins			
Alice goes to Bobs and buys coffee for 1 coin			

How many coins do Alice, Bob and Charlie have in the end?

Blockchain = a digital ledger

	Alice	Bob	Charlie
In the beginning, Alice has 100 coins and Bob 50	100	50	0
Alice pays 10 coins to Bob for bread	90	60	0
Charlie repairs the house of Alice for 30 coins	60	60	30
Alice goes to Bobs and buys coffee for 1 coin	59	61	30

So the blockchain is only about money?

No. It is a way of ...

- Recording information
- In sequential order
- In a tamper-proof manner

Examples

- Who invented what first? (Haber and Stornetta 1991)
- Who did what? (reputation)
- Who voted for what?
- Who traded with whom?
- Where does an object (and its components) come from?
- Who owns what? ← *apparently what interests most people*

The three cryptographic problems

Starting point:

Alice $\xrightarrow{\text{message}}$ Bob

- Adversaries could intercept or falsify the message
- Goal: secure communication over insecure channels

The three basic cryptographic tasks

- Encrypting
- Signing
- Hashing

The three cryptographic problems (2)

Encrypting – keep a secret

- How can Alice and Bob keep their secret?
  - Alice encrypts using the **public** key of Bob.
  - Bob decrypts using his **private** key.
- Usually **not** used on the Blockchain
- Example: PGP algorithm

Signing – ensuring authenticity/identity

- How can Bob be sure the message is from Alice?
  - Alice signs a message using her **private** key.
  - Bob can verify the authenticity using the **public** key of Alice.
- Verify authorisation of blockchain transactions
- Example: Ed25519 signature algorithm

The three cryptographic problems (3)

Hashing – ensure immutability

- How can Bob know that Alice's message has not been changed?
  - Alice calculates the hash of **her original message**
    She makes it public
  - Bob calculates the hash of the **received message**
    He compares it to Alice's hash
- Link blocks on the blockchain
- Example: SHA-256 algorithm

Two blockchain ideas

Goal

- Agree on **one** record of legitimate transactions (*ledger*)

The process

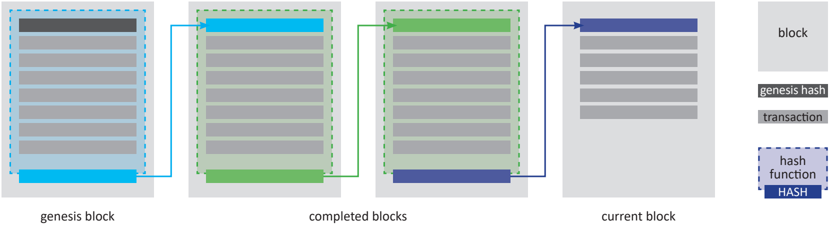
1. Everybody can submit candidate transactions
2. Validators (*miners*) validity
   - Check signature, sufficient funds, ...
3. Valid transactions are added to the ledger

Two problems (and two ways to steal money)

- Agree on which transactions are legit → consensus
- Ensure the immutability of the ledger → chaining & hashing

Blockchain idea #2: Chaining & Hashing

- Start from *genesis block*
- Concatenate *all(!)* subsequent blocks via hashing algorithm
  - Block k contains hash of block $k-1$
  - Block $k-1$ contains hash of block $k-2$ (and so on)
- Change historic entry $\leftrightarrow$ rewrite subsequent chain
  - *How to avoid this?*
  - Make it costly to "rewrite history" (*next slide*)



Costly Chaining & Hashing

How to render proposing a new block costly?
→ link to limited (i.e. valuable) resource

Physical proofs

- Proof of ...
  - work (BTC, ETH 1.0)
  - elapsed time (Intel's trused execution environment)
  - (disk) space (or retrievability, data possession; Chia)
  - capacity (Signum)

Economic proofs

- Proof of ...
  - stake (ETH 2.0, ...)
  - authority (Lugano's triple-A blockchain)
  - burn (Slimcoin)
  - Pure proof of stake (Algorand)

Blockchain idea #1: Consensus

- Goals
  - Efficiency – no unnecessary calculations
  - Security – high threshold for attacks

Byzantine General's Problem

- How to establish trust in the presence of adversaries?
  - Whom can we trust? Who has been bribed?
  - If 51% of nodes were compromised,
    a majority vote would confirm fake transactions.

Solution

- Alternative to pure majority vote
  - Randomly choose nodes that can vote
  - Attacker does not know whom to bribe
    → Higher threshold for attacks

*Consensus algorithms – discussion

Decentralized Byzantine Agreement

- Minimize waste
- Robustness against attacks
- Competition-friendly to encourage wide participation

Tragedy of the Commons

- Need cost efficient incentivisation to provide common goods
  - Provision of infrastructure and participation in consensus
  - Participation in governance

Network effects

- Value of network grows with size. How to start?

Glossary

- **Blockchain:** a digital ledger or growing list of records/blocks, that are linked together using cryptography. Properties: decentralized, distributed, often public.
- **Layer 1:** Base or infrastructure layer of a blockchain, e.g. Bitcoin, Ether, Algorand.
- **Key pair:** Two keys (public–private) in public key cryptography.
- **Private key:** Long number, used for decrypting and signing messages. Keep secret.
- **Address:** or public key. Used to verify signatures or encrypt messages.
- **Mnemonic:** Set of English words representing the private key.
- **Wallet:** Collection of several (public/private) keys.
- **Hash function:** Injective cryptographic function that is difficult to invert.
- **Digital Signature function:** Injective, verifiable cryptographic signature. Sign with private key, verify with public key.
- **Token:** Class of entries in a blockchain representing an absolute claim, often ownership.
- **Fungible Token:** Type of token for which all instances are identical. Cryptocurrency.
- **Non Fungible Token (NFT):** Type of token which is unique. Art, tickets, messages.
- **Native token:** Cryptocurrency associated with blockchain protocol. Used to pay transaction fees.
- **Central Bank Digital Currency (CDBC):** Cryptocurrency issued by a central bank, representing one unit of fiat money.
- **Smart Signature:** Logic that can approve (or not) a proposed transaction. Stateless.
- **Smart Contract:** Logic that can interact (read/write) with the blockchain. Stateful.

Official terms and definitions: `https://www.federalregister.gov/d/2022-05471/p-58`

A brief history of Algorand

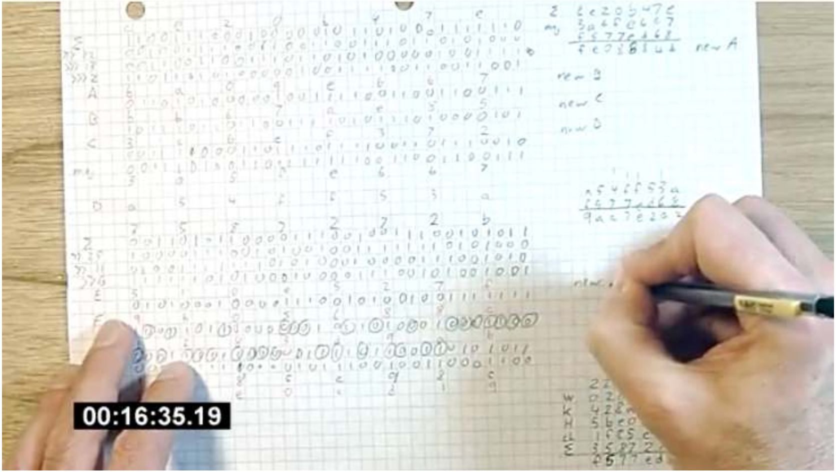
- 2017 work started; goal to improve over Bitcoin's inefficiencies
- 2017 company founded by Silvio Micali (MIT, Turing price)
- 2018-Feb 4M USD seed funding from Pillar, Union Sq. Ventures
- 2018-July Launch of testnet
- 2018-Oct 62M USD venture capital funding
- 2019-May launch of Algorand University program
- 2019-June auction of ALGO token
- 2019-July launch of mainnet
- 2020-Feb first version of PyTEAL
- 2021 First carbon negative blockchain

See also
`https://arxiv.org/abs/1607.01341`
`https://dl.acm.org/doi/10.1145/3132747.3132757` (first video)
`https://www.algorand.com/about/our-history`
`https://www.algorand.com/about/sustainability`
`https://github.com/algorand/pyteal/releases`

Algorand and USI

- USI joined the Algorand Global University Program in May 2019
- Algorand supports the USI foundation
- USI runs a relay node
- USI collaborates on research and teaching

SHA-256 Hashing Algorithm



How to calculate one SHA-256 hash by hand in 16 minutes
`https://www.youtube.com/watch?v=y3dqhixzGVo`