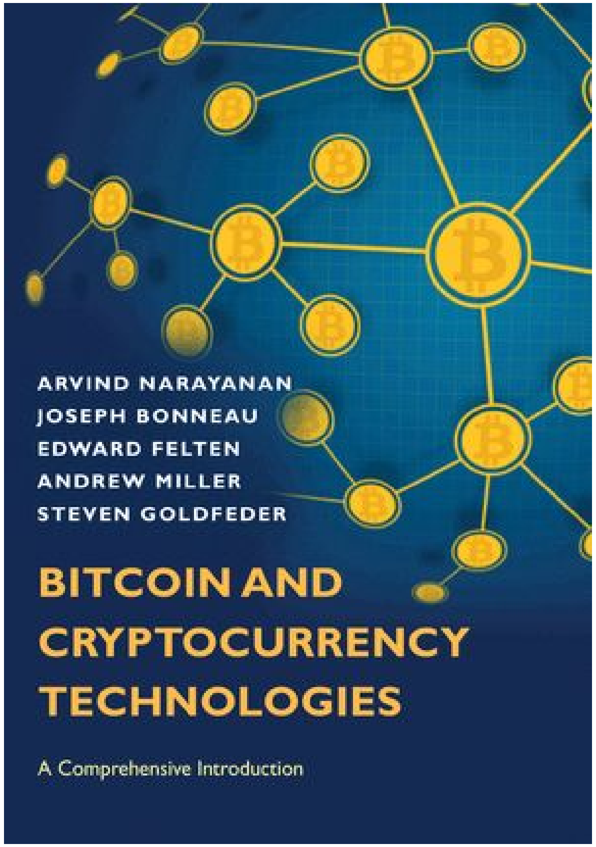
Bitcoin, Ethereum und andere Blockchains: Was steckt dahinter? Wie wenden wir sie an?

Dr. Stephan Murer
Murer Consulting GmbH
stephan.murer@ggaweb.ch

Objectives

- Understand key concepts in computer science behind cryptocurrencies
  - Hash functions and hash pointers
  - Hashed data structures (blockchain, Merkle tree)
  - Digital signature
  - Distributed consensus (byzantine agreement)
- Understand how cryptocurrencies work on the example of Bitcoin
  - Double spending problem
  - Probabilistic consensus, proof-of-work
- Generalize concept of currency transaction to more flexible agreements (smart contracts) in Ethereum
- Towards the holy grail: Scalable, distributed consensus among large numbers of non-permissioned users with immediate finality (no forks) in Algorand
- Understand the breadth of possible applications and their economic importance

Textbook



- Many concepts from there
- ISBN: 9780691171692

Agenda

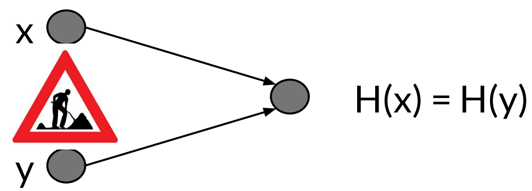
1. Cryptocurrency fundamentals
2. Bitcoin - transaction data on the blockchain
3. Ethereum - the programmable blockchain
4. Algorand - scalable distributed consensus & immediate finality
5. Applications & Market

Cryptographic Hash Functions

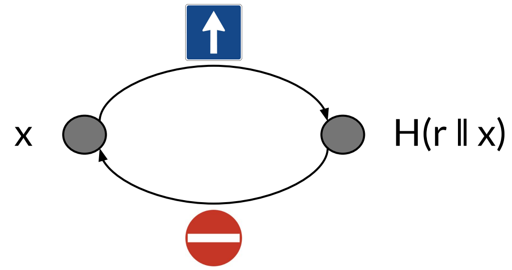
- Mathematical function
- Input: String of any size
- Output: Fixed-side bit array (256 bit in the examples)
- Efficiently computable: Computation time grows linearly with string length.
- Running time: O(n), for string length = n

h := H(s)
s: arbitrary length string
h: fixed length hash

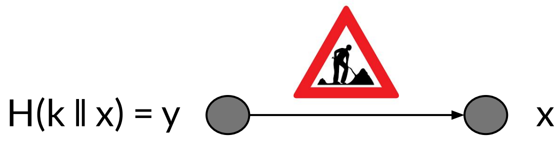
Hash Functions Properties



Collision resistant: Hard to find two values x and y, such that $x \neq y$, yet $H(x) = H(y)$.

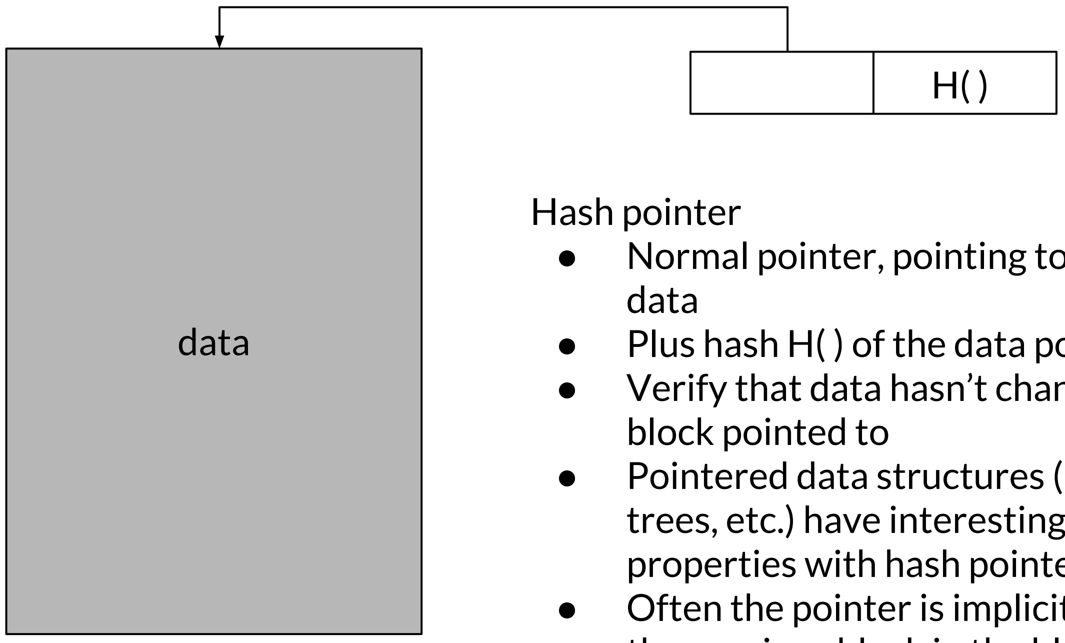


Hiding: A hash function H is said to be hiding if when a secret value r is chosen from a probability distribution that has high min-entropy, then, given $H(r \| x)$, it is infeasible to find x.



Puzzle friendliness: A hash function H is said to be puzzle friendly if for every possible n-bit output value y, if k is chosen from a distribution with high min-entropy, then it is infeasible to find x such that $H(k \| x) = y$ in time significantly less than 2^n.

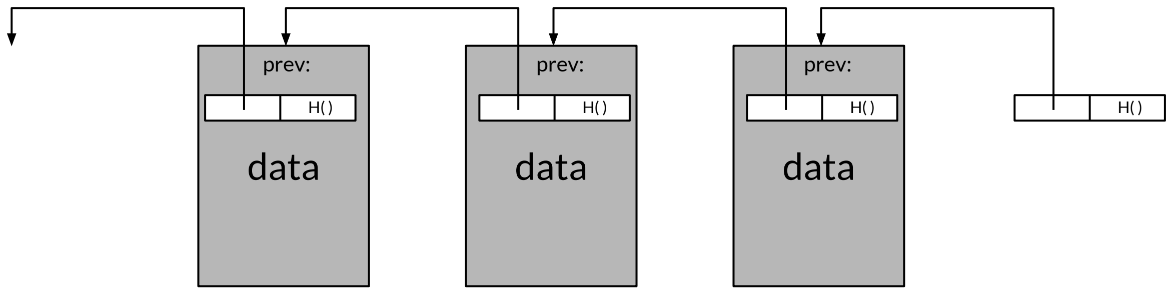
Hash Pointers



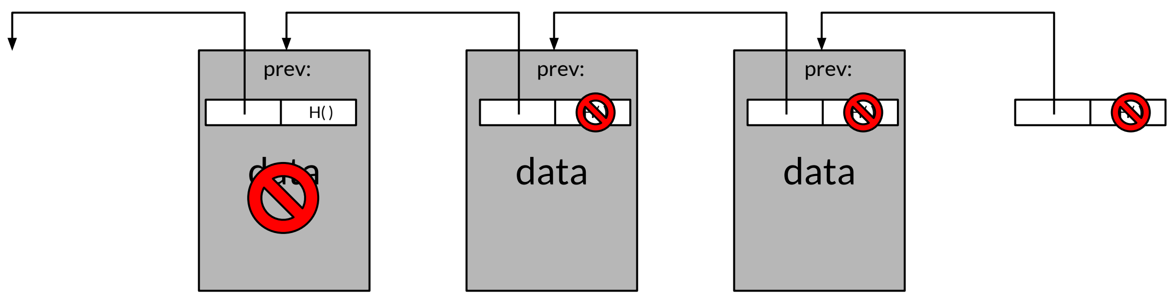
Hash pointer

- Normal pointer, pointing to block of data
- Plus hash H( ) of the data pointed to
- Verify that data hasn't changed in block pointed to
- Pointered data structures (lists, trees, etc.) have interesting properties with hash pointers
- Often the pointer is implicit (e.g. to the previous block in the blockchain)
- Only works for acyclic structures

Blockchain: Tamper-evident log

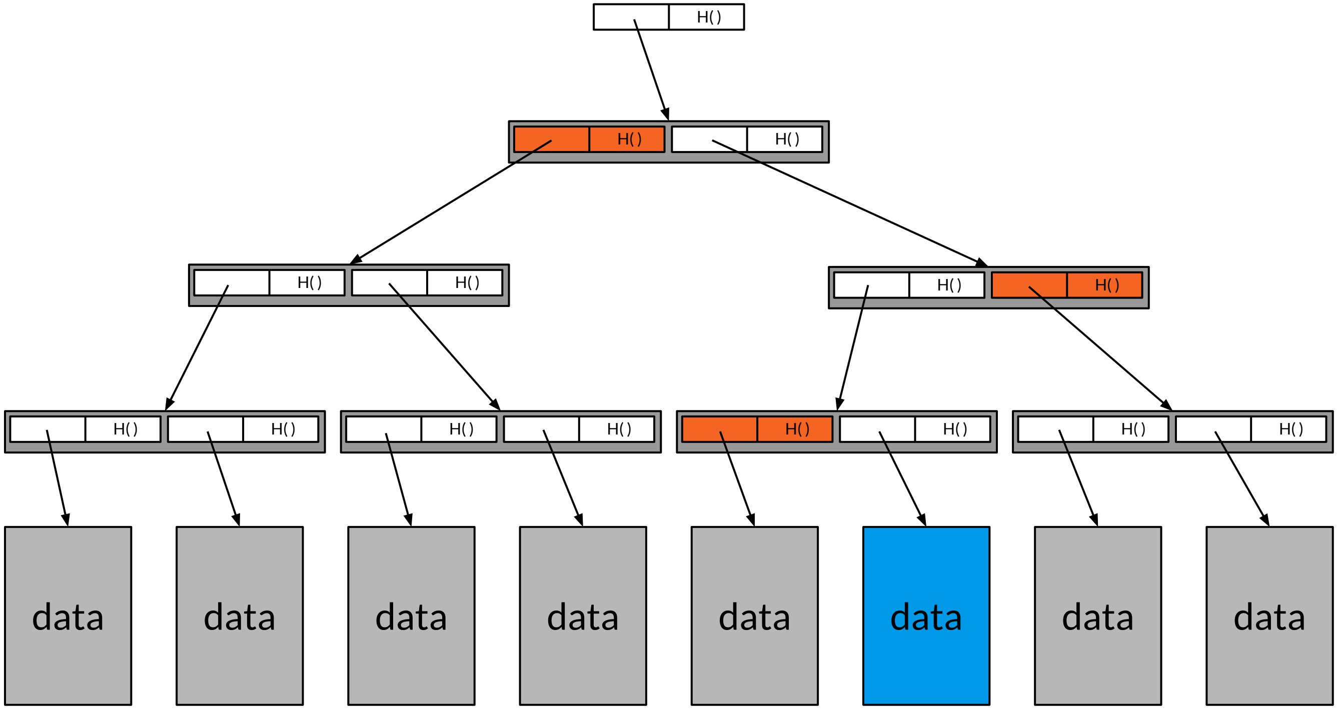


- Blockchain: Linked list with hash pointers



- Tamper-evident log (book keeping journal e.g.!)
- Alter data in block k of the blockchain
- Hash pointer pointing to that block from block k+1 is wrong
- Hash pointers pointing to previous blocks are part of block data
- Therefore: All hash pointers of succeeding blocks are wrong, including head of list
- Head of list hash pointer is all needed to check integrity of blockchain

Merkle Trees



- Merkle Tree: Binary tree with hash pointers
- Same tamper evidence property as blockchain
- log(n) proofs of (non-) membership: Provide neigbouring hashes as proof

https://en.wikipedia.org/wiki/Merkle_tree, 9.3.2017

Digital Signatures

Digital signature scheme with three algorithms:

1. (sk, pk) := generateKeys(keysize)
2. sig := sign(sk, message)
3. isValid := verify(pk, message, sig)

Two properties:

1. verify(pk, message, sign(sk, message)) == true
2. Signatures are existentially unforgeable
   Existential forgery: Given pk, a pair (message, sig) can be constructed, that looks like signed with sk, although sk is unknown, RSA, eg. is existentially forgeable, however content of message not controllable

- Public Key Cryptography offers methods with the desired properties
- Best known: RSA, Bitcoin uses Elliptic Curve Cryptography
- Better properties: shorter keys, existential unforgeability
- Very dependent on good source of randomness

https://en.wikipedia.org/wiki/Digital_signature_forgery, 7.11.2017
https://www.youtube.com/watch?v=F3zzNa42-tQ, 9.3.2017
https://blog.cloudflare.com/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/, 9.3.2017

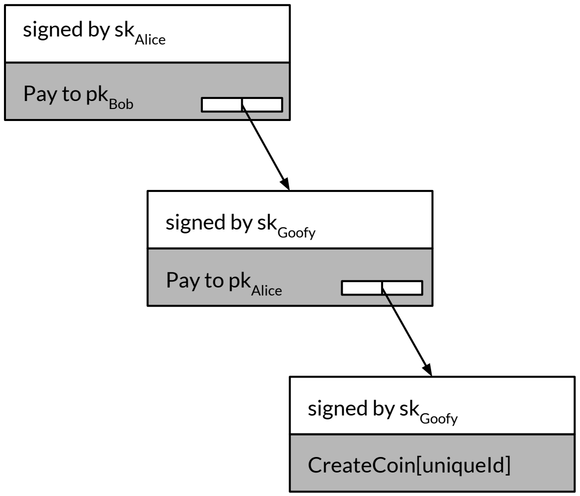
Public keys as Identities

- Public key (pk) used as identity
- Check signatures directly against identities
- New identity easily generated with generatekeys()
- In practice H(pk) used as identity, because public keys are long
- Basically random numbers, nothing that ties to your real identity (although using identity will leak real identity), identity quickly changeable
- Decentralized identity management: Everyone just generates identity
- Identities unique, as there are so many possible public keys
- Pseudonymity by using many identities

Agenda

1. Cryptocurrency fundamentals
2. Bitcoin - transaction data on the blockchain
3. Ethereum - the programmable blockchain
4. Algorand - scalable distributed consensus & immediate finality
5. Applications & Market

Simple Cryptocurrency



- Goofy creates coin by a unique coin id
- Only Goofy creates coins, his pk is universally known
- Owner of coin signs transaction record with recipient and hash pointer to either creation record or transaction record with owner as receiver
- Complete history of ownership changes, tamper evident due to hash pointer structure
- Problem: Double spending, what if Alice signs two transaction records, both spending the same coin
- Despite showing some of the key structures of cryptocurrencies, Goofycoin can't deal with double spending, not good enough!

Bitcoin Transaction Representation

```
{
  "hash":"f4184fc596403b9d638783cf57adfe4c75c605f6356fbc91338530e9831e9e16",
  "ver":1,
  "vin_sz":1,
  "vout_sz":2,
  "lock_time":0,
  "size":275,
  "in":[
    {
      "prev_out":{
      "hash":"0437cd7f8525ceed2324359c2d0ba26006d92d856a9c20fa0241106ee5a597c9",
      "n":0
      },
      "scriptSig":"..."
    }
  ],
  "out":[
    {
      "value":"10.00000000",
      "scriptPubKey":"..."
    },
    {
      "value":"40.00000000",
      "scriptPubKey":"..."
    }
  ]
}
```

Metadata

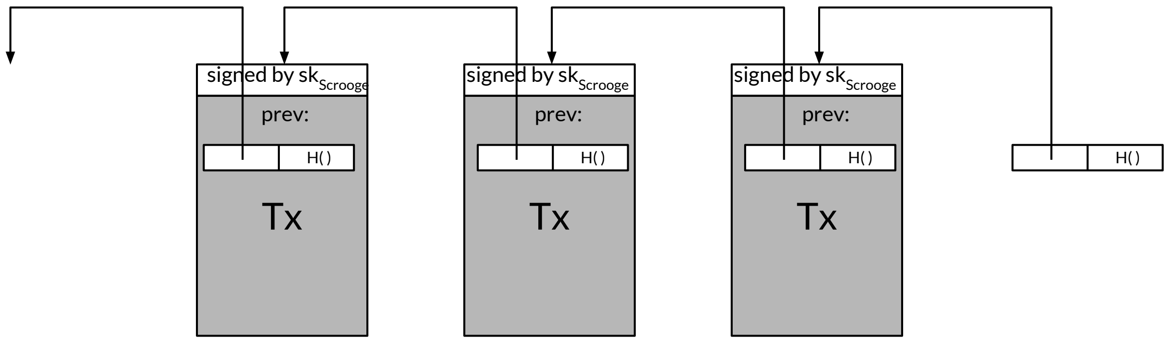
Inputs

Outputs

Bitcoins with 8 digits behind decimal point precision, smallest unit Satoshi

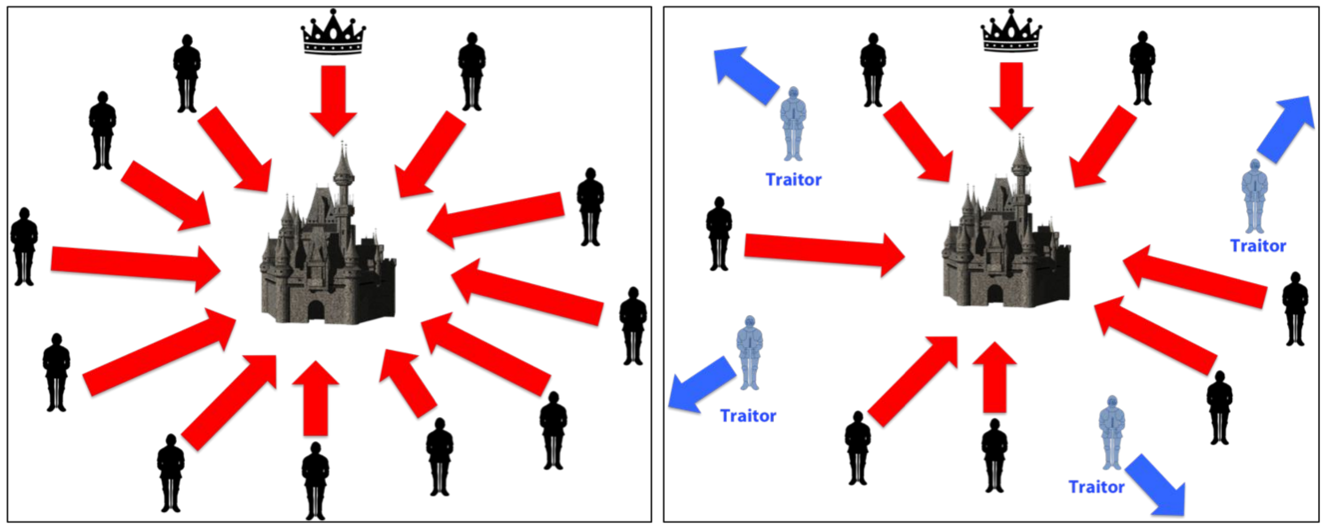
Bitcoin script: Hash of public key of signer plus operation.

Scroogecoin: Central Authority Deals with Double Spending



- Similar transactions like in Goofycoin, but all transactions stored in append-only blockchain, each block signed by Scrooge
- Blockchain assures append only property
- Easy to check integrity by looking at hash pointer
- Scrooge checks integrity of transactions (double spending), and only signs blocks with valid transactions

Byzantine Generals



Coordinated Attack Leading to Victory   Uncoordinated Attack Leading to Defeat

https://cdn-images-1.medium.com/max/800/0*-xCD-El4LZ48dji1.png, 15.3.2017

Leslie Lamport, Robert Shostak, and Marshall Pease. “The Byzantine Generals Problem” ACM TOPLAS, 4(3):382–401, 1982.

Distributed Consensus

Distributed consensus protocol: n nodes, each with input value, some faulty, some malicious. Protocol with following properties:

- Must terminate with all honest nodes in agreement on the value
- Value must have been generated by honest node

Many applications:

- Reliable distributed systems (distributed transactions, fault tolerance, etc.)
- Coordination in massively parallel systems
- Bitcoin

Consensus is possible under different assumptions

- Consensus problem cannot be solved assuming
  - at least one node failing
  - reliable, asynchronous communication
  - deterministic node behaviour (work correctly or fail)

  according to Fisher, Lynch and Patterson in 1985

- Pragmatic solutions exist 2-phase commit, PBFT, Google Chubby
- Bitcoin violates traditional assumptions:
  - Incentivize nodes to behave honestly (works well for a currency)
  - Randomness, non-deterministic behavior
  - No specific starting and ending point for consensus (eventual consensus)
- Bitcoin works better in practice than in theory, but theory would be important

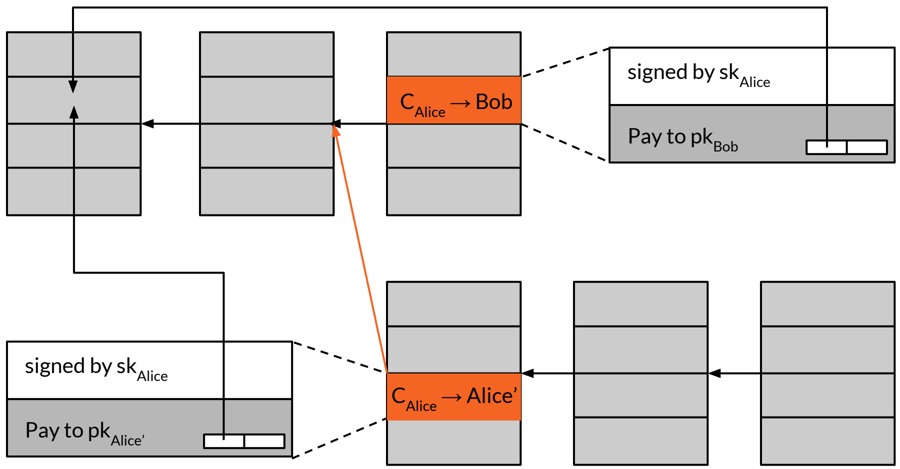
MJ Fischer, NA Lynch, MS Paterson. "Impossibility of distributed consensus with one faulty process". Journal of the ACM (JACM) 32 (2), 374-382
http://lamport.azurewebsites.net/pubs/paxos-simple.pdf, 18.3.2017
https://angus.nyc/2012/paxos-by-example/, 18.3.2017
http://pmg.csail.mit.edu/papers/osdi99.pdf, 19.4.2017

Implicit Consensus in Bitcoin

Bitcoin consensus algorithm (assuming random node selection ability, not vulnerable to sybil attack):

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

Double Spend Attack



Alice pays Bob

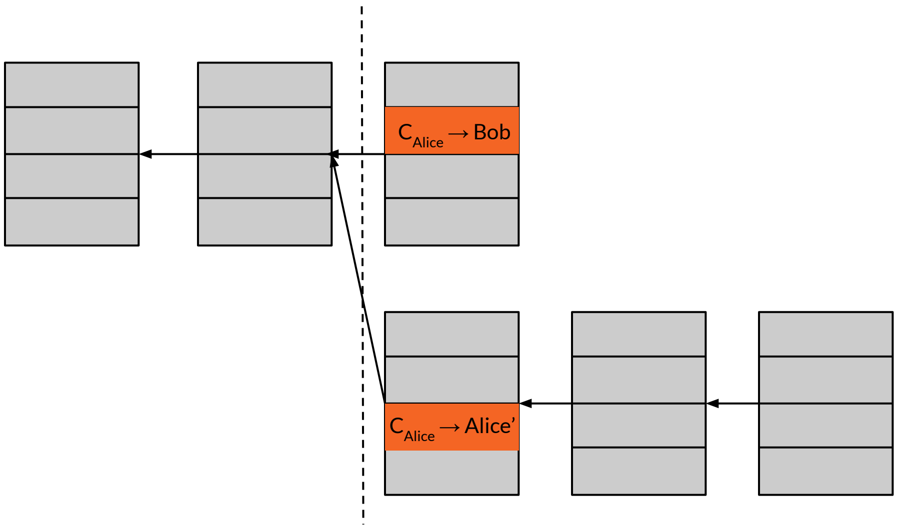
- Broadcasts transaction with Bob as receiver pointing to coin owned by Alice
- Transaction gets included into latest block by honest node
- Bob sees transaction confirmation in blockchain

Double spend

- Next random node to propose block controlled by Alice
- Ignores latest block with payment to Bob
- Creates new block pointing to previous block with transaction benefitting Alice'

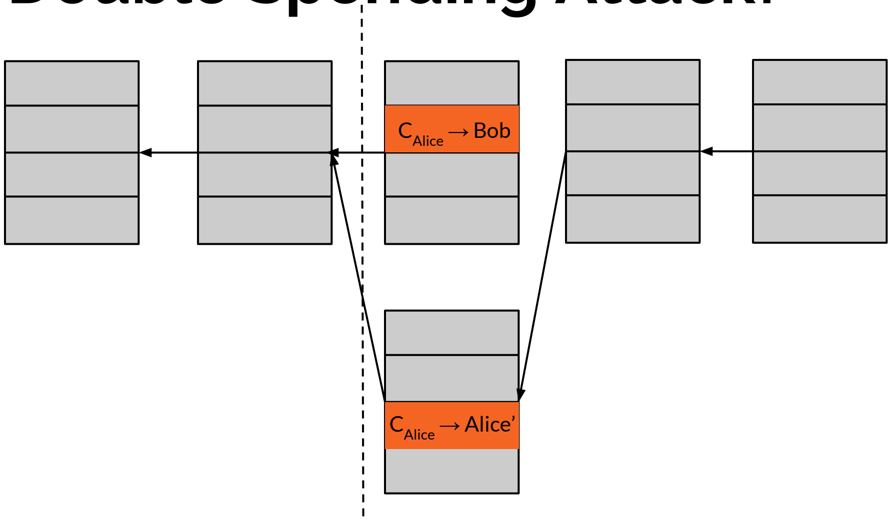
- Can this attack be successful?
- Depends on whether long-term consensus agrees on the fraudulent block with the transaction $C_{Alice} \rightarrow$ Alice' or the honest one with $C_{Alice} \rightarrow$ Bob gets included into the blockchain.

Which Block gets Inserted?



- Temporarily, there may be branches in the blockchain
- Honest nodes extend the longest branch of the blockchain
- Unclear directly after attack, both branches same length, both blocks are valid at this point in time
- Generally first block detected on network gets accepted, but can be either due to network latency
- So, potentially fraudulent branch will win and honest block gets orphaned
- May be helped by Alice bribing other nodes to build on her branch

How can Bob Protect himself against a Double Spending Attack?



When does Bob accept transaction as confirmed?

- Sees signed transaction on network → zero confirmation transaction, can easily be faked by malicious node
- Transaction in one block → single confirmation
- Other blocks point to block with transaction → multiple confirmations

- More confirmations → higher probability that transaction is on consensus chain, ie. valid, careful payment receiver waits for a number of confirmations
- Probability for block not being on the consensus chain reduces exponentially with number of confirmations, 6 confirmations are common practice in Bitcoin

Can we Incentivize Nodes to Behave Honestly?

- Block reward
  - Node creating a block can include a special transaction creating coins
  - Block reward only valuable, if included in consensus branch (like other transactions)
  - Network follows longest branch rule, block reward incentivizes nodes to extend longest (consensus) branch
- Transaction fee
  - Payers can choose to leave a small difference between coins spent and paid
  - Nodes including transaction into block can pay this difference to themselves
  - Will users include transaction fees to get good service? Unclear yet!
  - Transaction fee will have to take over as incentives, when block rewards run out in 2034

Proof of Work

- Why wouldn't everybody create as many nodes as possible, get the incentives, or worse monopolize the consensus mechanism?
- We still don't know how to select a random node from an unknown set of nodes, no node ids, permissionless distributed ledger
- Approximate random selection by selecting nodes in proportion to an overall resource nobody can monopolize (we hope!)
  - Compute power: Proof of work
  - Ownership of currency: Proof of stake
- As opposed to permissioned distributed ledger, with identified node, and different consensus protocols (eg. PBFT for Hyperledger)

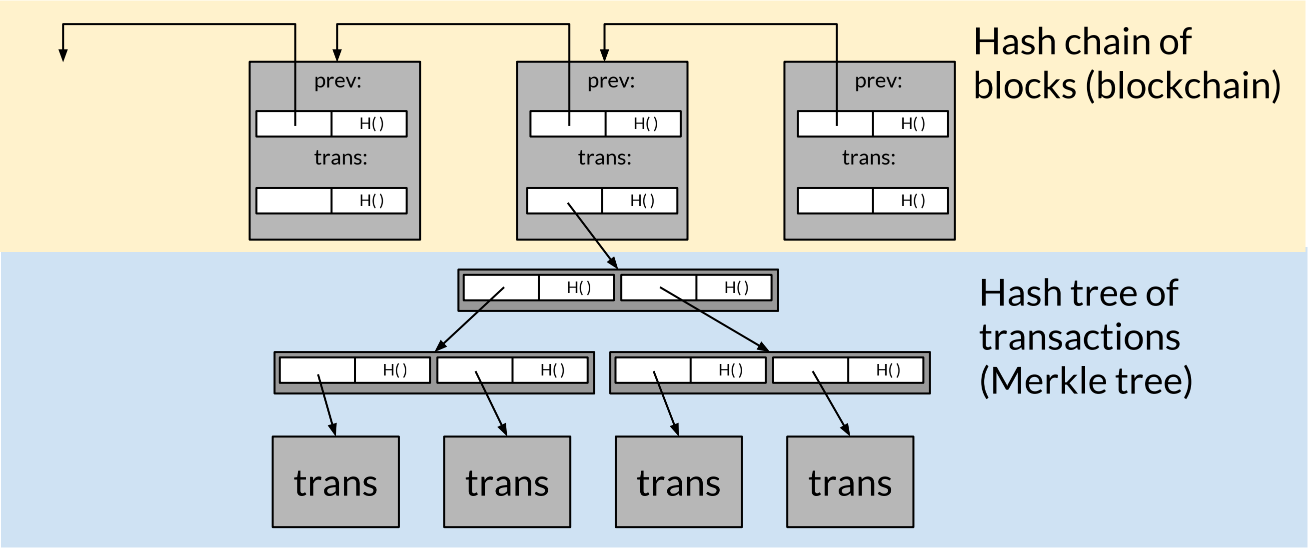
Proof of Work in Bitcoin: Hash Puzzles

Find nonce such that:

H(nonce ‖ prev_hash ‖ tx ‖ tx ‖ … ‖ tx) < target

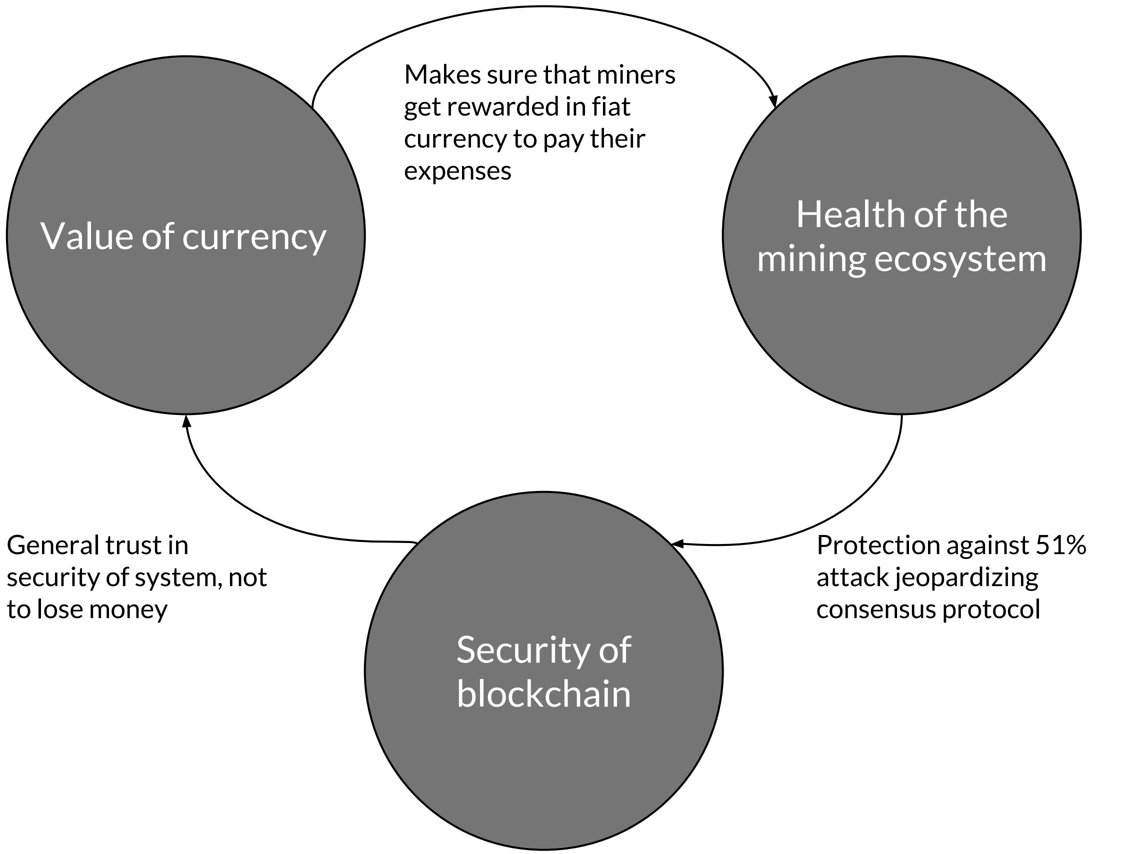
- Puzzle-friendlyness → try different nonces one-by-one until condition satisfied
- "target" defines how hard the puzzle is
- No need to centrally select random node, nodes independently compete in finding nonce satisfying condition
- Nodes will statistically "win" proportionally to their power computing hashes

Bitcoin Blocks



- Block consists of block header, hash pointer to previous block, hash pointer to transaction tree
- Block header contains nonce, difficulty, timestamp, etc.

Bringing it Together ...

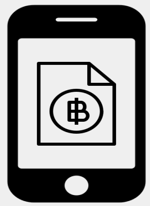


- Bootstrap problem: At the beginning Nakamoto was the only miner, bitcoin had little value, chain in secure
- Unclear, why bitcoin took off, probably good story, publicity

Bitcoin Summary

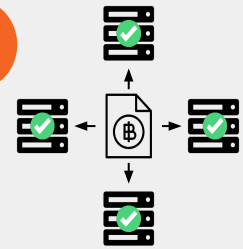
at the client

1



create transaction

2



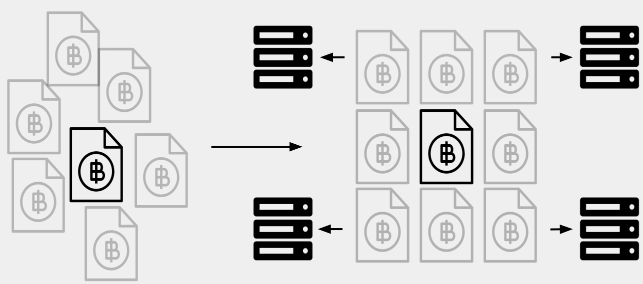
distribute

5

Check for validity of transaction in the blockchain

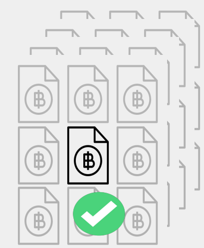
at the mining node

3



Every 10 minutes collect valid transactions, form block, and publish block when nonce is found.

4



Accept new block as head of the blockchain

Agenda

1. Cryptocurrency fundamentals
2. Bitcoin - transaction data on the blockchain
3. Ethereum - the programmable blockchain
4. Algorand - scalable distributed consensus & immediate finality
5. Applications & Market

Bitcoin & Ethereum

Similar:

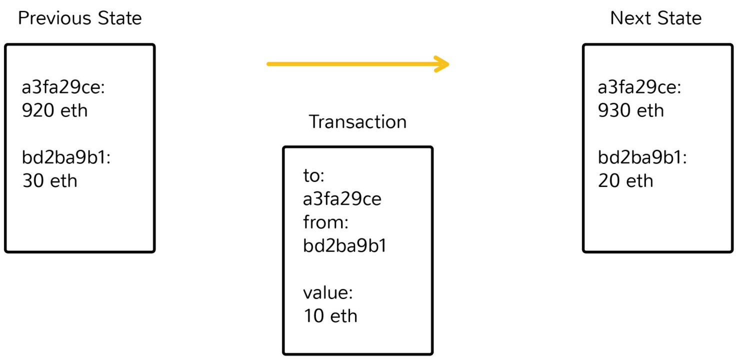
- Blockchain
- Built-in cryptocurrency
- Proof-of-work-mining
- Public, permissionless

Differencies:

- Smaller blocks (~70 vs. ~2000 tx), lower block time (~14 s vs. ~10 m)
- Memory-bound (ETHHASH) vs. compute-bound proof of work (SHA-256)
- Fully programmable smart contracts vs. simple scripts / fixed transactions
- Unlimited ETH supply (some pre-mined) vs. limited BTC supply, no pre-mining
- Gas & gas price
- Accounts & transactions vs. transactions & UTXOs

https://bitsonblocks.net/2016/10/02/a-gentle-introduction-to-ethereum/, 8.11.2107

Ethereum accounts and transactions



- Ethereum accounts
  - External accounts, controlled through private key, balance in ether (ETH), can send transactions (transfer ETH or trigger contract code), state: balance
  - Contract accounts, has code (smart contract), code execution triggered by transactions or messages from other contract accounts, state: balance, can send messages to other accounts
- Transaction
  - Value, data, gas & gas price

https://www.coindesk.com/information/how-ethereum-works/, 8.11.2017
https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369, 8.11.2017

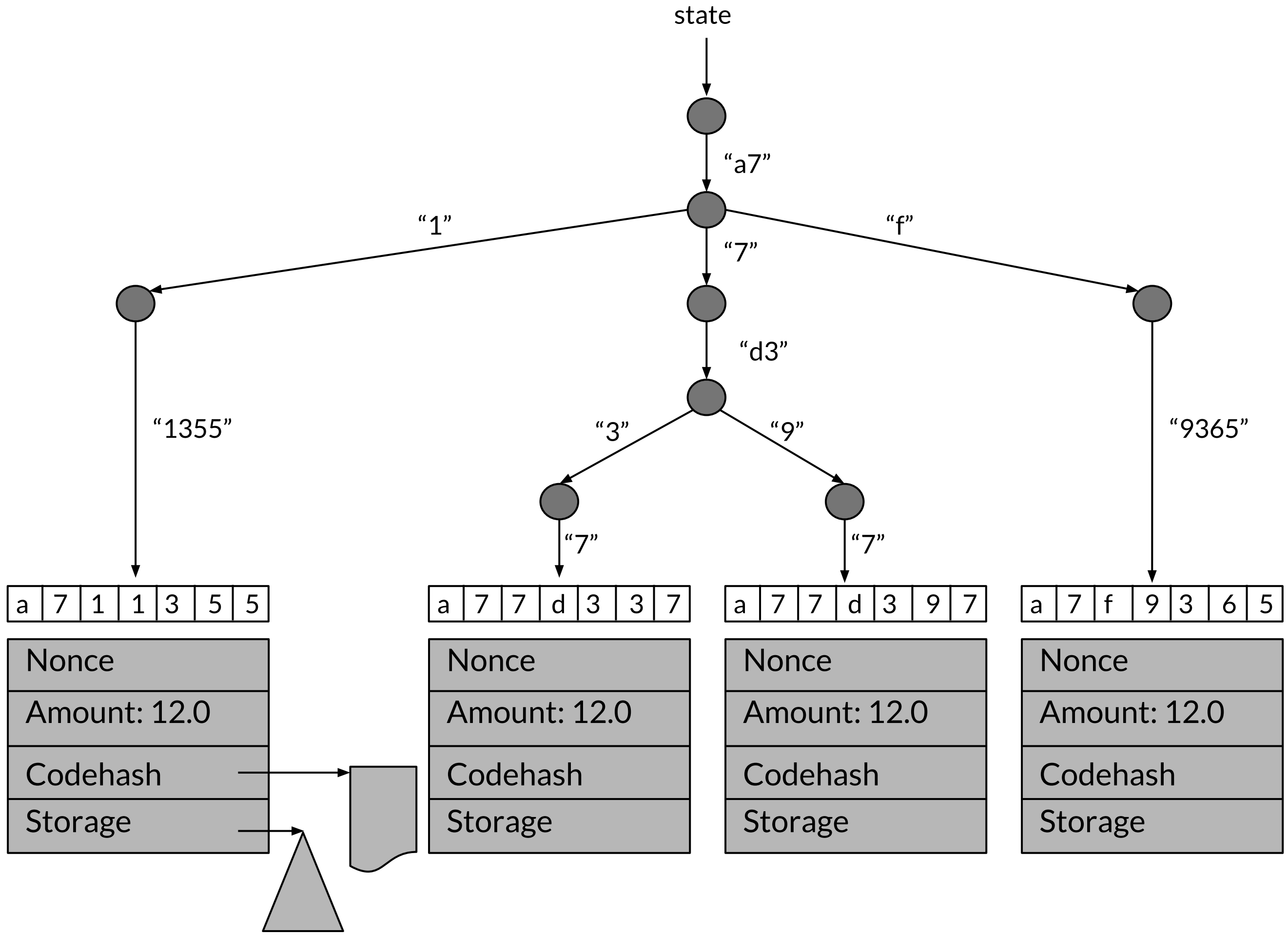
Smart Contracts

```
contract ProofOfExistence {

    // key/value store mapping doc hashes to booleans
    mapping (bytes32 => bool) private proofs;

    // calculate document hash and store the proof for a document
    // state changing function
    function notarize(string document) {
        proofs[sha256(document)] = true;
    }

     // check if a document has been notarized
     // non-state changing function
    function checkDocument(string document) constant returns (bool) {
        return proofs[sha256(document)]; //unmapped value is always 0 (=false)
    }
}
```

- Contract programming Language: Solidity
- Contracts similar to classes: combining state and functionality
- Ethereum virtual machine (EVM)
- Uses gas for execution, gas price x amount of gas pays miner
- Limited gas amount ensures termination despite turing completeness of language

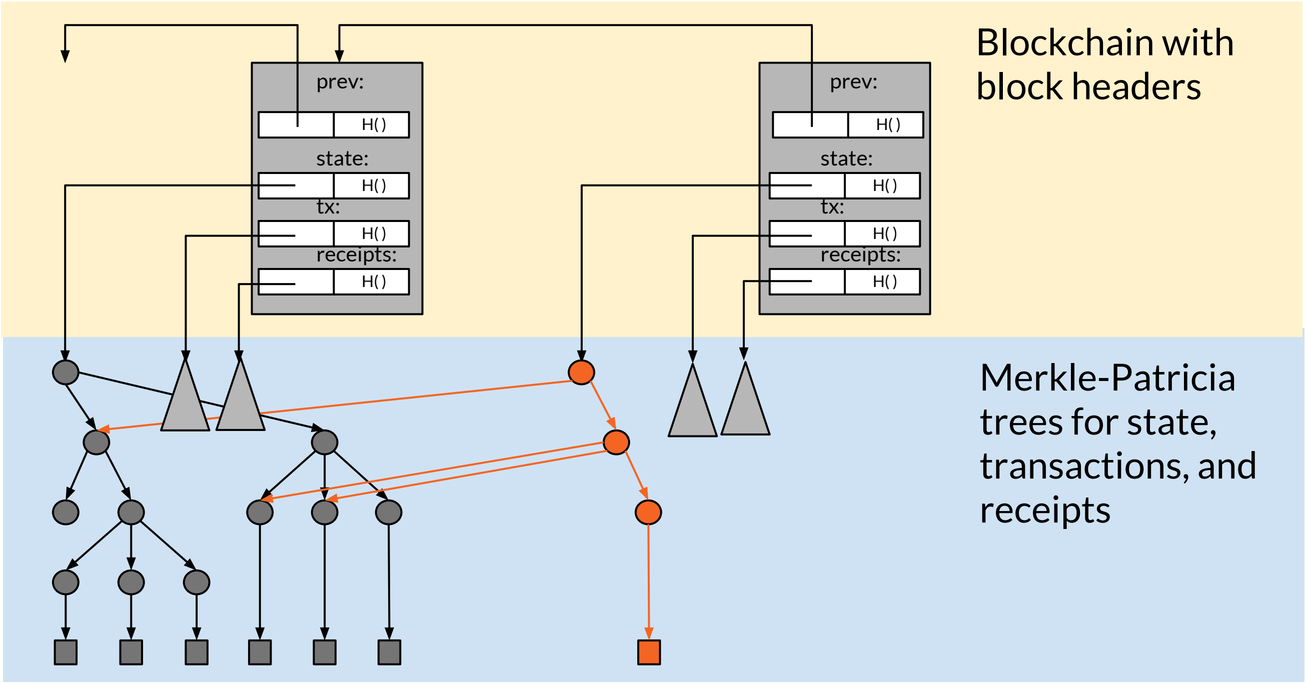
https://blog.zeppelin.solutions/the-hitchhikers-guide-to-smart-contracts-in-ethereum-848f08001f05, 8.11.2017

Merkle Patricia Tree for State Key Value Store



https://easythereentropy.wordpress.com/2014/06/04/understanding-the-ethereum-trie/, 7.11.2017
https://en.wikipedia.org/wiki/Radix_tree, 7.11.2017
https://ethereum.stackexchange.com/questions/6415/eli5-how-does-a-merkle-patricia-trie-tree-work, 7.11.2017

Ethereum Blocks



- Trees linked between blocks on a node, efficient changes just for changed state

Agenda

1. Cryptocurrency fundamentals
2. Bitcoin - transaction data on the blockchain
3. Ethereum - the programmable blockchain
4. Algorand - scalable distributed consensus & immediate finality
5. Applications & Market

Limitations of Bitcoin & Co.

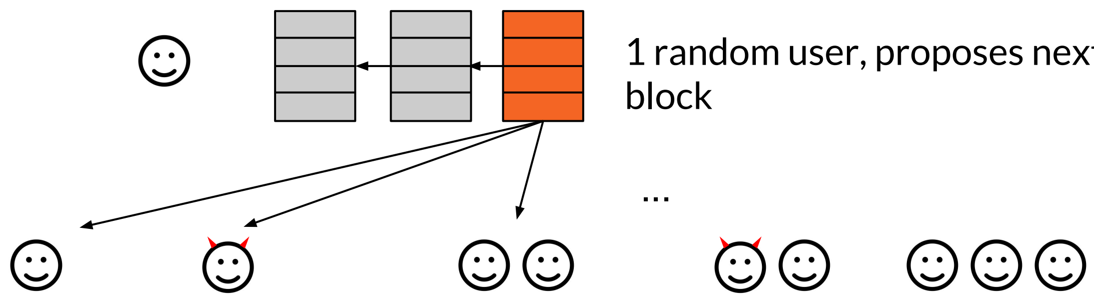
- Enormous waste of compute resources / energy
- Concentration of power with large miners / mining pools
- Scalability (Bitcoin 7 Tx/s), others faster but limited
- Forks, eventual consistency
- Anonymity

Various approaches to improve:

- Permissioned ledgers
- Alternative mining puzzles
- Different block parameters
- Sidechains
- Fundamental: Algorand

https://www.coindesk.com/information/what-is-the-difference-between-open-and-permissioned-blockchains/
http://faculty.cs.tamu.edu/bettati/Courses/489CryptoCurrencies/2017A/Slides/AlternativeMiningPuzzles.pdf
https://blockstream.com/sidechains.pdf

Algorand: High-level structure



Randomly selected committee runs distributed consensus, signs consensus block, propagates signatures

https://people.csail.mit.edu/nickolai/papers/gilad-algorand-eprint.pdf

Self selecting committees

- Committee selects itself through cryptographic lottery (fair, winner can proof he won)
- User i part of committee r if and only if:
  $. H(\mathrm{sign}(sk_i, Q_r)) < p$

probability to be on committee (in practice weighted with owned assets)

quantity shared by all users based on last block

digital signature of Q_i with secret key sk_i (users are identified by their public keys)

hash of signature to get good random distribution

binary decimal point (result is between 0 and 1)

- No communication needed: scalable
- Only user i knows, he is on committee
- Can proof by propagating $\mathrm{sign}(sk_i, Q_r)$

Scalable distributed consensus

- 9 communication steps needed to reach consensus on block, or return 0 block
- Committee can be different in each round (cannot be bribed)
- ⅔ of money needs to be honest
- Forks with negligible probability
- Fulfills all properties of distributed consensus
- Speed depends on network latency, but 100x bitcoin tx throughput seems feasible
- x000 committee size, round requires each member to send/revieve short message
- practical tx conformation below 60s

Scalable, permission-less, but consensus-driven, no forks, finality, low-latency

https://www.youtube.com/watch?v=mqqEjzmQva8 (explained in a video by Silvio Micali)

Agenda

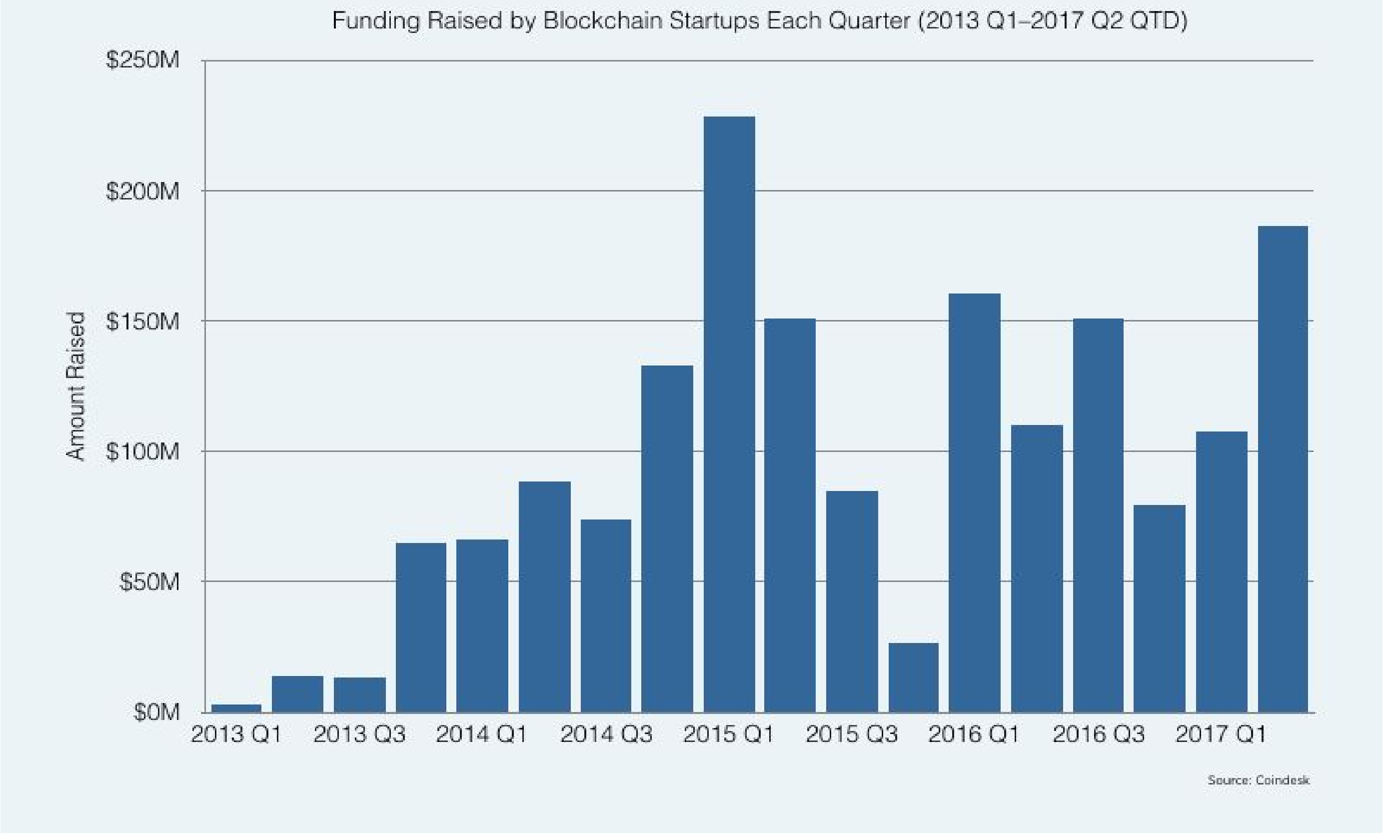
1. Cryptocurrency fundamentals
2. Bitcoin - transaction data on the blockchain
3. Ethereum - the programmable blockchain
4. Algorand - scalable distributed consensus & immediate finality
5. Applications & Market

BTC/ETH market

- Total market cap all currencies: ~$200 \cdot 10^9$ $
- Total volume 24h: ~$10 \cdot 10^9$ $
- Money supply M1 Germany: $2 \cdot 10^{12}$ $
- TARGET2 (ECB) volume 24h: $1.8 \cdot 10^{12}$ €

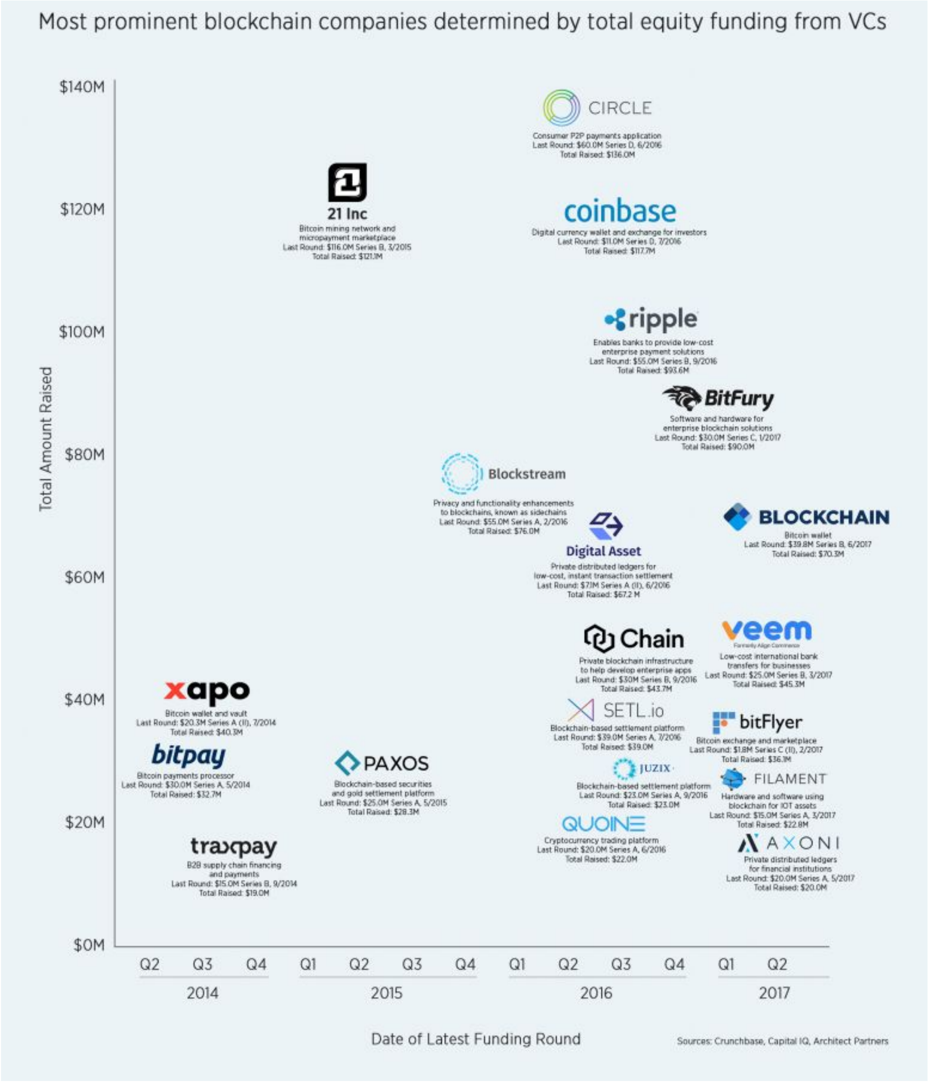
https://coinmarketcap.com/, 9.11.2017

Blockchain venture capital



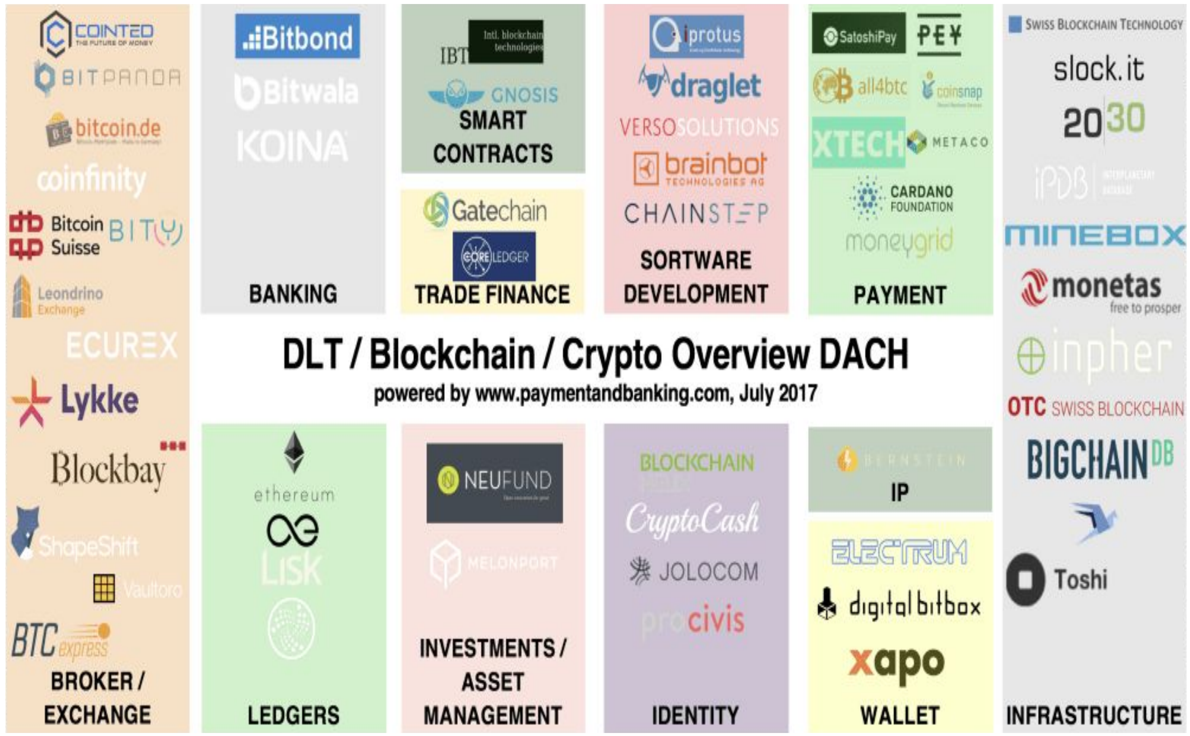
http://architectpartners.com/ecosystem_thoughts/top-20-vc-backed-blockchain-companies/, 8.11.2017

Largest blockchain startups



http://architectpartners.com/ecosystem_thoughts/top-20-vc-backed-blockchain-companies/, 8.11.2017

DACH startups



https://paymentandbanking.com/infografik-dlt-blockchain-crypto-overview-dach/

Zürich startups

- E-Government on blockchain
- Digital identity
- E-Voting
- Collaboration with university of Zürich

- Trade finance on smart contracts
- Project batavia
- UBS & IBM founding partner
- Today with Caixa, Erste Bank, Bank of Montreal, Commerzbank

- Quality-ensured pharmaceutics delivery chain
- Blockchain and IoT
- Spin-off from university of Zürich

- Clearing and settlement for OTC financial instruments in Switzerland
- Partnering with Hochschule Luzern

https://procivis.ch/ https://modum.io/
https://www.finews.ch/news/banken/29100-ubs-blockchain-handelsfinanzierung-batavia
https://otc-blockchain.ch/

Initial Coin Offering

- Initial Coin Offering (like Initial Public Offering / IPO)
- Basically Crowdfunding / collecting donations in digital currency
- Founders of new currency sell digital tokens of a future currency or promises for future services agains established digital currencies (BTC, ETH) or fiat currencies
- Buyers speculate on fast value increase of the new coins
- In contrast to classical capital market transactions, this is unregulated
- Example TEZOS:
  - Breitman couple founds company to build new, better blockchain
  - ICO results in 232M$ for Zug based foundation (today > 400M)
  - Breitmans get substantial amount for preliminary work
  - Conflict between founders and foundation jeopardizes project, money potentially lost

https://tokenmarket.net/
https://www.tezos.com/
https://www.finews.com/news/english-news/29275-tezos-swiss-foundation-zug-battle-arthur-breitman-kathleen-breitman-johann-gevers-crypto-ico

Conclusion

- Overview of a fascinating field in computer science, that has developed only in the last 10 years
- Core technology for Fintech
- Huge economic potential
- Towards the transactional Internet
- Elegant link between very theoretical results and very meaningful applications in key fields of computer science
- Something every young computer scientist should understand as it will be important for future jobs in IT, particularly:
    - Finance / Fintech
    - IT Security

Danke!

рахмат
danke
謝謝
ngiyabonga
spas
teşekkür ederim
Баярлалаа
mersi
kia ora
barka
welalin
tack
mahalo
tapadh leat
спасибо
faafetai lava
vinaka
спасибі
blagodaram
dank je
misaotra
matondo
paldies
grazzi
kiitos
dankie
thank you
gracias
хвала
nanni
nandri
dhanyavad
maururu
asante
manana
bedankt
bayarlalaa
gràçie
hvala
köszönöm
dankon
aciū
akun
obrigada
murakoze
tenki
enkosi
djiere dieuf
tau
mochchakkeram
дякую
mamnun
chokrane
dziękuję
chnorakaloutioun
gratias ago
gràcies
sulpáy
taiku
go raibh maith agat
sobodi
děkuji
sagolun
sukriya
kop khun krap
obrigado
mèsi
didi madloba
ありがとう
tanemirt
grazie
arigatô
takk
dakujem
trugarez
najis tuke
kam sah hamnida
rahmat
terima kasih
rahmet
ευχαριστώ
diolch
dhanyavadagalu
shukriya
mercé
мерси
তোমাকে ধন্যবাদ
감사합니다
xièxie
merci