SpaceMint
Overcoming Bitcoin’s waste of energy

Georg Fuchsbauer

joint work with
S. Park, A. Kwon, K. Pietrzak, J. Alwen and P. Gaži

Journées nationales pré-GDR SI 31/05/17

Overview

1 Bitcoin
- Transactions
- Blockchain
- Proof of work
- Problems with PoW

2 SpaceMint
- Proofs of space
- Issues with PoSp
- New blockchain format

Ledger

Public ledger

<table>
<thead>
<tr>
<th></th>
<th>Alice</th>
<th>Bob</th>
<th>Charlie</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Alice: transfer 1 → Bob
Public ledger (records all transactions)

Charlie → Alice

Alice: transfer 1 → Bob

how to identify?

Digital signatures

- Alice can create a **key pair**
  - **private key** used to sign messages
  - **public key** lets anyone verify signatures
- **Unforgeability**: no one can forge signature w/o knowing secret key

- Public key ↔ coin
- Private key: enables spending of coin

Transactions

- Alice owns \( pk_A \) i.e. it’s in the ledger
- Bob creates \( pk_B \)
- Alice signs \( pk_A \rightarrow pk_B \) and adds to ledger

Double-spending

- Alice signs \( pk_A \rightarrow pk_B \)
- Alice signs \( pk_A \rightarrow pk_C \)

Ledger only accepts if

- exists transaction \( * \rightarrow pk_A \)
- no transaction \( pk_A \rightarrow * \)

Transactions

- Alice owns \( pk_A \)
- Bob creates \( pk_B \)
- Alice signs \( pk_A \rightarrow pk_B \) and adds to ledger

Double-spending

- Alice signs \( pk_A \rightarrow pk_B \)
- Alice signs \( pk_A \rightarrow pk_C \)

Ledger only accepts if

- exists transaction \( * \rightarrow pk_A \)
- no transaction \( pk_A \rightarrow * \)
Decentralization

How to eliminate authority that

- checks validity of tx’s
- publishes new tx’s in ledger

The Blockchain

- blocks linked by including hash of previous block
  \( \Rightarrow \) cannot modify block w/o changing everything after
  acts as fingerprint for whole chain

Cryptographic hash functions

- outputs look random
  \( \Rightarrow \) small mods result in huge changes
  \( \Rightarrow \) hard to find preimage
  \( \Rightarrow \) best way to find input with hash from some subset is randomly trying
Proof of work

- prove that you’ve performed work
- e.g. prevent spam: Hashcash

\[
X\text{-Hashcash: } 3105171100:gfuchsba@inria.fr:0101
\]

- try out \( \approx 2^{20} \) values (\( \sim 1s \))
- easy to verify (\( \sim 1\mu s \))

Mining

- collect transactions
- find value \( $ \) yielding small hash
- broadcast block

\[
\text{tx}_1 \quad \text{tx}_2 \quad \text{tx}_3
\]

\[
h \quad $ \quad \text{H} \quad 0...000730
\]

\( 69 \times \)

\[
\text{if } \quad \text{tx’s are valid}
\]
\[
\quad \text{hash is small enough}
\]
\[
\Rightarrow \text{add block to local copy of blockchain}
\]

Incentive?
\( \Rightarrow \) reward bitcoins!
(all bitcoins created this way)

Forks

- Double-spending!
Forks

"Always mine on the longest chain"

Secure if majority of miners is honest
⇒ wait for 6 blocks before accepting payment

The "51%-attack"

Proof of stake

• prob. of mining ∼ number of coins owned

• Problems:
  – Nothing-at-stake problems
  – Participation: miners = holders

Why does it work?

• Miners incentivized by rewards
• Probability of mining block ∼ computing power
  ⇒ no Sybil attacks!
• Rational to mine on longest chain
  ⇒ quick consensus

↑ Can proof of work be replaced by something else?

↑ Problems
  • specialized hardware + cheap electricity
    ⇒ mining oligarchy
  • Bitcoin consumes electricity like a town of 100k population
    ⇒ polluting

↑ Problems
  • Nothing-at-stake problems
  • Participation: miners = holders
Proof of space

- prove that you've allocated disk space

Trivial solution

Initialization:
- store lookup table
  (1, f(1))
  ...
  (N, f(N))
sorted by output

Verifier

Prove:
- compare with $F$
  $F[i_1],...,F[i_n]$ 

inefficient for verifier

A better solution

Initialization:
- fill nodes of graph $G$
  dep. on $id$
- hash content
  $\gamma$(hash using Merkle tree)
  (use hard-to-pebble graph)

Verifier

Prove:
- $y_1,\ldots,y_n$
  $\pi = (f^{-1}(y_1),\ldots,f^{-1}(y_n))$

⇒ Proof of work

Time/memory trade-offs:
- Store $N^{2/3}$, invert in time $N^{2/3}$

[DFKP'15]
Proof of space

- prove that you’ve allocated disk space
  
  [DFKP’15]

  **Initialization:**

  Verifier → Prover
  
  - fill nodes of graph $G$ dep. on $id$
  - hash content

  **Prove:**

  nodes $i_1, \ldots, i_n$
  
  - check consistency with $\gamma$

  $\pi = (G[i_1], \ldots, G[i_n])$


**SpaceMint**

- replace proof of work by proof of space

- **Advantages:**
  
  - *green:* low electricity; reusable hardware
  - decentralized

- **Challenges:**
  
  - PoS is *interactive*
  - Nothing-at-stake problems
    
    * Mining multiple chains
    * Grinding blocks

- Miner initializes space with $id = pk$

- broadcasts $\gamma$

- $\gamma$ gets added to chain
Who gets to add the block?

- **Quality** of proof?
  - define fct. of proof $\pi$: quality $\sim$ space allocated
  - block with best proof gets added to chain
- Blocks define **quality of chain**
  - always mine on best chain

Does this work?

$\neq$ Bitcoin: 1
- easy to generate proofs!
  - miners try to extend every chain
  - no consensus!
- Forbid extending 2 chains

$\neq$ Bitcoin: 2
- easy to check if good solution!
  - miners might not extend best chain
  - no consensus!
- Take challenge from past

same challenge!
SpaceMint

≠ Bitcoin: ③
⇒ miners might grind blocks leading to good challenge in future
⇒ proof of work
Make challenge hash of π only

Transactions not hashed
⇒ not consolidated in chain!
Blocks not linked to previous block
⇒ consensus??

More ecological?
• no ongoing cost
• resources recyclable
• unused disk space ⇒ decentralized

Use signatures (tied to proof) to link blocks