Goûter des Doctorants : Cryptocurrencies

Adrien Koutsos

January 29, 2018

1. Introduction
2. Quick Crypto
3. Lets Try!
4. Blockchain
   - Blockchain and Merkle Tree
   - Consensus Problem
   - Block Mining
5. Variants, Futur Changes
6. Conclusion

1 Introduction

2 Quick Crypto

3 Lets Try!

4 Blockchain
- Blockchain and Merkle Tree
- Consensus Problem
- Block Mining

5 Variants, Futur Changes

6 Conclusion

Introduction

Cryptocurrency

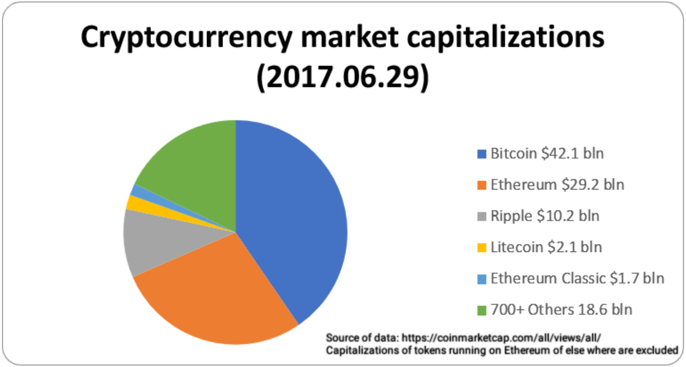
- Money based on thin air: not backed by anything from the "real world".
- Decentralized ledger: no authority says who has what.
- New currency tokens issued automatically, at a fixed rate.

Standard currency

- Money based on thin air: not backed by anything from the "real world".
- Centralized system: banks and/or governements maintain the ledgers.
- New currency tokens issued by governements, depending on policies.

Introduction

- Bitcoin probably the first and most famous cryptocurrency.
- Other famous cryptocurrencies: Ethereum, Litecoin ...
- Lots of speculations, not so much applications:



1 Introduction

2 Quick Crypto

3 Lets Try!

4 Blockchain
- Blockchain and Merkle Tree
- Consensus Problem
- Block Mining

5 Variants, Futur Changes

6 Conclusion

Cryptographic Hash Function

Definition

$H : \mathcal{M} \mapsto \{0;1\}^\eta$ such that:

- **Hiding:** Given $H(x)$, computationally infeasible to find x.
- **Collision-Resistance:** Given x and $H(x)$, computationally infeasible to find $y \neq x$ such that $H(x) = H(y)$.

Formal Property

H is Collision-Resistant against Hidden-Key Attacks if for all PPTM $\mathcal{A}$ with oracle access we have:

$$\mathbf{Pr}\left[\mathsf{k} : \mathcal{A}^{H(\cdot,\mathsf{k})}(1^\eta) = (m_1, m_2) \wedge m_1 \neq m_2 \wedge H(m_1,\mathsf{k}) = H(m_2,\mathsf{k})\right]$$

is negligible in η (k is drawn uniformly at random in $\{0,1\}^\eta$).

Signature Scheme

Definition

$$\text{sign} : \mathcal{M} \times \mathcal{SK}_\eta \mapsto \{0;1\}^\kappa$$
$$\text{verify} : \mathcal{M} \times \{0;1\}^\kappa \times \mathcal{PK}_\eta \mapsto \{0;1\}$$

such that:

- η is the key length, κ the signature length.
- **Correction:** $\text{verify}(m, \text{sign}(x, \text{sk}), \text{pk}) = 1$.
- **Unforgeability:** Given m and $\text{sign}(m, \text{sk})$, computationally infeasible to find $s \neq \text{sign}(m, \text{sk})$ such that $\text{verify}(m, s, \text{pk}) = 1$.

1 Introduction

2 Quick Crypto

3 Lets Try!

4 Blockchain
- Blockchain and Merkle Tree
- Consensus Problem
- Block Mining

5 Variants, Futur Changes

6 Conclusion

How to build a cryptocurrency

Naïve first approach

- **Identities:** Public signature keys.
- **Money transfer:** $\mathsf{sign}(\text{"}\mathit{IOU} : \mathit{Bob\text{-}to\text{-}Alice} : 100\text{"}, \mathsf{sk}_{Bob})$

How can Alice use this money?

Naïve second approach

- **Identities:** Public signature keys.
- **Money:** *IOU* messages.
- **Money transfer:**
  $\mathsf{sign}(\text{"}\mathit{IOU} : \mathit{Bob\text{-}to\text{-}Alice\text{-}from\text{-}Charlie} : \text{"} \cdot m \cdot \text{"}100\text{"}, \mathsf{sk}_{Bob})$
  where m is a *IOU* message from *Charlie* to *Bob* from _.

Double spending!

Obstacles to Cryptocurrencies

Obstacles

- Check identities of people: cryptographic signatures.
- Creating initial coins: actually pretty easy, and even helps.
- No double spending: consensus problem.

Consensus Problem

The problem

A finite number of agents $A_1, \ldots, A_n$ need to have a *comon view* on some set of data, but:

- They communicate through an adversarial network (block messages, forge messages . . .).
- Some agents may be compromised/corrupted.

Requirements

- **Asynchronous:** people come and leave all the time.
- **Validity:** if enough honest agent, consensus decision is the same for all honest agents.
- **Progress:** cannot DoS the cryptocurrency, and transactions eventualy take place.

Consensus Problem

Theorem: Byzantin General Problem

If more than *one third* of the agent are corrupted, cannot guarrantee all three properties.

Remark

Paxos well-known algorithm for consensus in non-adversarial network (just node failures considered).
Very complicated, no full formal analysis (I think), but works (variants used by Google, Microsoft, . . .).

1 Introduction

2 Quick Crypto

3 Lets Try!

4 Blockchain
- Blockchain and Merkle Tree
- Consensus Problem
- Block Mining

5 Variants, Futur Changes

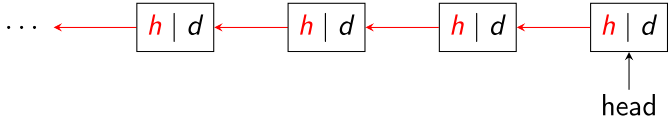
6 Conclusion

Blockchain

Definition

List of back-chained block, where each block contain data and the hash of the previous block.

- **Tamper-Resistance:** Given the head of a blockchain, you cannot tamper with any block of the chain.

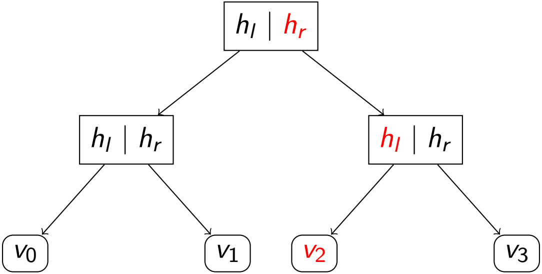


Merkle Tree

Definition

A binary where internal nodes contain the hashes of its left and right child, and leaves contain some data.

- **Proof of membership:** in $\sim \log(n)$ space/time.
- **Proof of non-membership:** in $\sim \log(n)$ space/time, if *sorted*.



How Bitcoin works

Functioning

Network of nodes, each having a replica of the full blockchain (almost).

- Transactions are broadcasted through the network.
- Nodes collect the unpublished transactions into a *block*.
- Try to publish the block to extend the chain (details later).
- If receive a new valid block before publishing, go back to the beginning.

Details

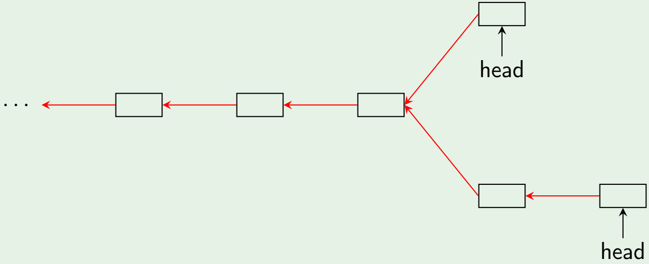
- Block are represented using a Merkle Tree.
- Broadcast algorithm is the simplest imaginable (I think).

How to have consensus

Bitcoin reaches consensus through the following rule:

Always extend the longest chain.

View of a node



Remark

Transactions in the head block can disappear if a longer branch appears.
Rule of thumb: a transaction is fully commited after 6 blocks.

How Bitcoin works

Functioning

Network of nodes, each having a replica of the full blockchain (almost).

- Transactions are broadcasted through the network.
- Nodes collect the unpublished transactions into a *block*.
- **Try to publish the block to extend the chain (details now).**
- If receive a new valid block before publishing, go back to the beginning.

Block Publishing

Constraints

- Everybody can publish at ant time, attack:
  - Send money to A.
  - Wait for 6 blocks, A transfer you what you bought.
  - Extend a previous block where you own the bitcoins.

  Being able to publish is rare and random.
- Block published too fast: forks all the time.
  Being able to publish is rare and random.
- Need incentives for people to host nodes.
  Nodes publishing are paid.
- Need incentives for nodes to be honest.
  Nodes publishing are paid *in the current branch*.

Block Publishing

Block Mining: Proof of Work

Given a Merkle Tree representation of a set of transactions m, a previous block hash p, look for n such that $H(n \cdot p \cdot m)$ is in some small set.

- $H(n \cdot p \cdot m)$ starts with more than d zeros ($d \approx 60$).
- Difficuty recomputed every 2048 blocks ($\approx$ 2 weeks) to be on average every 10 minutes.
  10 minutes deemed large enough to avoid too much forking, and to have time to properly broadcast the block.

Block Publishing

Block Mining

- Miner who find a block add to the transactions a reward for themselves.
- 50 Bitcoins initially, divided by 2 every 4 years (25 today).
- Therefore controlled inflation and coins creation (at most 21 millions Bitcoin, in 2140).
- Transactions can include a fee for the miner, if the block reward is not enough.

Mining In Practice

Block Mining

Initially, meant to be CPU mining: *one CPU, one vote* (I think).

Mining In Practice

Computing hashes is very paralellizable: GPU mining.

Mining In Practice

When the value of Bitcoin started to go up, ASIC (*Application-specific integrated circuit*) mining.

Mining In Practice

Block Mining Today

- Rentable only if using ASIC and cheap electricity (e.g. China).
- People group into mining pools to reduce variance.
- Very energy consuming: 82 810 MWh per day ($\approx$ Marocco, or 2.8 millions US households).
- Number of Hashes per seconds: 12,132 Peta Hashes/second.

Source: digiconomist.net

1 Introduction

2 Quick Crypto

3 Lets Try!

4 Blockchain
- Blockchain and Merkle Tree
- Consensus Problem
- Block Mining

5 Variants, Futur Changes

6 Conclusion

Mining In The Future

Reduce Energy Consumption

Mining power not proportional to computational power, but to:

- **Proof of Stack:** money you own (e.g. Ethereum soon?).
- **Proof of Space:** space you allocated (e.g. SpaceMint).
- **Proof of Useful Work:** miner solves PDE, protein folding ... (do not exists yet).

Concurrent Cryptocurrencies

Try to improve on Bitcoin

- **Litecoin:** Supposed to have ASIC resistant hash function (failed).
- **Ethereum:** Allows for a Turing-complete language for transactions. Lots of funny attacks (DAO: 50 millions $ stolen, Parity: 300 millions $ blocked).

A Word on Verification of Cryptocurrencies

Two approaches to formal proofs of cryptocurrencies

- **Byzantine style proofs:** assume more than x percents of honest nodes.
- **Game theoretic proofs:** show that we have a Nash Equilibrium. *(Sometimes false, e.g. mining pools)*

1 Introduction

2 Quick Crypto

3 Lets Try!

4 Blockchain
- Blockchain and Merkle Tree
- Consensus Problem
- Block Mining

5 Variants, Futur Changes

6 Conclusion

Conclusion

What I talked about

- Introduced some cryptographic tools used in cryptocurrencies: hashes, signatures, blockchains, Merkle trees.
- Discussed the difficulties encountered when building a cryptocurrency: double spending, consensus problem.
- Explained how bitcoin works (all cryptocurrencies work in a similar way).

My personal opinion

- Proof of work is a nice idea (with horrible consequences in practice).
- Conceiving a system such that the incentives of the agents are to play by the rule is *fun* (Cf proof of stacks/space).
- Ethereum has at first sight lots of potentiel fun applications. Although to my knowledge, only gambling and financial products (e.g. ICO).

Thanks for your attention