Blockchain Introduction *

Jens Lechtenbörger

VM Neuland im Internet 2021

Contents

*This PDF document is an inferior version of an OER HTML page; free/libre Org mode source repository.

"Along came bitcoin, a radically different proposal for a decentralized cryptocurrency that did not need the banks, and digital cash finally succeeded. [...] This article challenges that view by showing nearly all of the technical components of bitcoin originated in the academic literature of the 1980s and 1990s (see Figure 1). This is not to diminish Nakamoto's achievement but to point out he stood on the shoulders of giants. Indeed, by tracing the origins of the ideas in bitcoin, we can zero in on Nakamoto's true leap of insight—the specific, complex way in which the underlying components are put together."

1.3 Omissions

- Focus here on blockchains à la Bitcoin, i.e., **public/permissionless** with **proof of work** consensus
- Beyond that, see survey [**DLZ+17**]
  - Ethereum, Parity, Hyperledger Fabric, private blockchains, proof of stake, PBFT, non-byzantine consensus variants, smart contracts, and more.
  - "The results show that current blockchains' performance is limited, far below what a state-of-the-art database system can offer."

2 Terminology

2.1 Crypto Recap

- "Blockchain" uses standard cryptographic primitives
- **Collision-resistant** cryptographic **hash** function
  - Map any data to "unique" hash value of fixed length, e.g., 256 bits in hexadecimal `0x420815abc...`
    - Digital fingerprint, cryptographic digest
  - Finding collisions must be computationally infeasible
- **Digital signatures** (asymmetric cryptography)
  - Participants own **key pairs**: private and public key
    - Alice **creates** signature on document with her **private key**
      - Bob uses ***her* public key to verify** signature
      - Proof that document (a) came from Alice (authentication, non-repudiation) and (b) unchanged by anybody else (integrity)

2.1.1 Key Pairs in Bitcoin

- "Account" (part of wallet) tightly coupled with key pair
  - Public key used like account number
    - Transfer of bitcoins "to" public key
  - Private key used to authenticate "transactions"

    * Alice signs transfer of bitcoins from her account to Bob's with her associated private key
    * Everybody can verify that transfer with Alice's public key
    * (Whoever has access to the private key can transfer coins)
        · (Remember Mt. Gox? 650,000 bitcoins "lost")

2.1.2 Hash Pointers

- If collision resistance is given, we can **identify** any data with its hash value
  - (If data changes, the hash value changes)
- E.g., data represents a block of transactions
  - That block is stored at some address
  - With hash pointers, we not only record the address but also the block's hash value
  - When retrieving the block, verify that it is unchanged

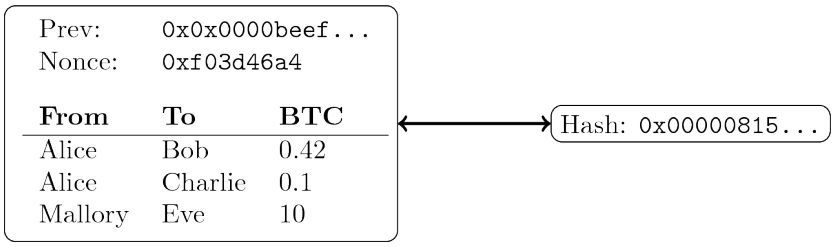

Figure 2: "Block of transactions with hash pointer" by Jens Lechtenbörger under CC BY-SA 4.0; from GitLab

2.1.3 Chaining Data

- Append-only logs can be constructed by chaining blocks with hash pointers
  - Typically, blocks organized as Merkle trees
    * Leaf nodes are transactions, internal nodes are hash pointers
    * Hash of tree's root serves as digest for block
    * Inclusion proofs (of transactions) possible with little data

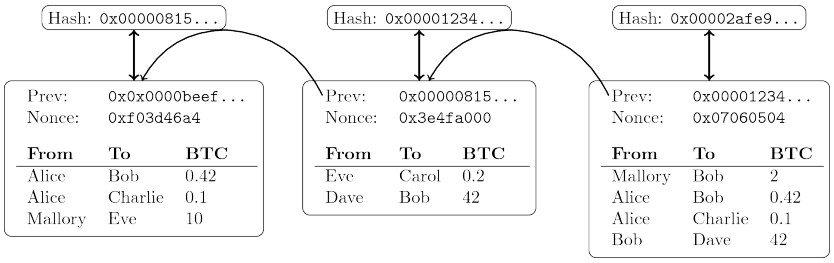

Figure 3: "Block chain of transactions" by Jens Lechtenbörger under CC BY-SA 4.0; from GitLab

2.2 Bitcoin Basics

2.2.1 Bitcoin Origin

- Announcement of bitcoin as "P2P e-cash" on Cryptography Mailing List on 2008-10-31 by Satoshi Nakamoto
  - Link to famous paper [**Nak08**]
  - [**Nak08**] does neither mention "blockchain" nor "block chain" (but "chain" and "block")
- Source code for bitcoin announced later on 2009-01-08

2.2.2 Bitcoin Goals

- **Decentralized** currency
  - No central authority; neither to issue coins nor to create accounts nor to monitor transactions
  - **Peer-to-peer** network of **miners** solves **cryptographic puzzle** to extend **blockchain** as "ledger of transactions"
    - **Proof of work**
    - Blockchain = data structure database
      - (**Database system** = DBMS + managed DBs)
      - Replicated among miners

2.2.3 Bitcoin Security

- Ownership of coins with **digital signatures**, **no double-spending**
  - Do not promise "same coin" to multiple parties
    - Miners check whether amount of proposed transaction unspent in current blockchain
    - Consensus
  - Blockchain is **probabilistically append-only** (revisited below)
    - No way to revoke/undo transactions

2.2.4 Bitcoin Transactions

- Transfer of coins from one account to another
  - Blocks in blockchain contain multiple transactions
  - When new block is mined, first transaction allows miner to transfer **block reward** to own account
    - This creates new coins
    - Amount of new coins per blocked halved about every 4 years (started at 50; 6.25 since May 2020)
  - Transactions can have multiple inputs and output (graphics here are simplified to show just one input and one output)
    - Inputs refer to (hashes of) transactions
    - Outputs subsume fees (to miner) and change (to sender)

3 The Blockchain?

3.1 The `Block Chain`

- Comments in the source code of bitcoin mention "block chain".
  - E.g., `main.h#l1013`

```
//
// The block chain is a tree shaped structure starting with the
// genesis block at the root, with each block potentially having multiple
// candidates to be the next block.  pprev and pnext link a path through the
// main/longest chain.  A blockindex may have multiple pprev pointing back
// to it, but pnext will only point forward to the longest branch, or will
// be null if the block is not part of the longest chain.
//
class CBlockIndex
```

  - Also for `CBlock`, `CWalletTx`, `CMerkleTx`

3.1.1 Trees of Blocks

- The "chain" is expected to fork into branches
- A tree structure emerges
  - The longest (actually, the most work-intensive) chain of that tree records "the truth" in Bitcoin

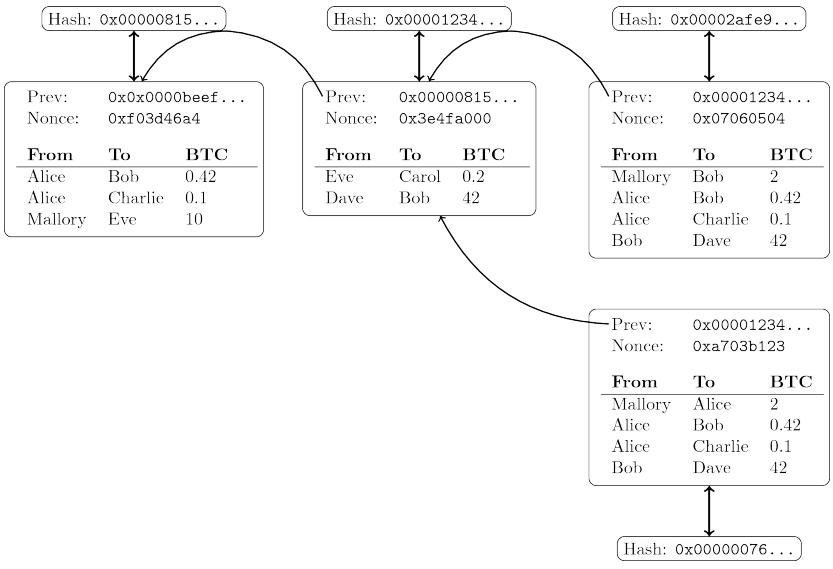

Figure 4: "Block tree of transactions" by Jens Lechtenbörger under CC BY-SA 4.0; from GitLab

3.2 The "Blockchain" in Bitcoin

- Tree-based data structure as just explained
  - No database! (Thus, no distributed database either!)
- Replicated/copied among P2P network of miners
  - Extended asynchronously
    - After solving a cryptographic puzzle: Proof of work
  - Different miners may work on different versions (forks, branches, orphans)
- Does each miner maintain "a blockchain"?
  - Comment above suggests so
- Is the entire set of chains of blocks "the blockchain" (of bitcoin)?

4 Blockchain Characteristics

4.1 Distributed Ledger?

- What exactly is a "ledger"?
  - Why invent a new term for a persistent, tamper-proof data structure?
- "Distributed" does not carry much defining value in blockchain contexts.
  - E.g., Google's infrastructure using Paxos [**Lam98**] is distributed, yet controlled centrally.

4.2 Decentralized, Public, Open, Replicated?

- No central control, no intermediaries/middlemen
- Anyone can take part
  - Obtain copy
  - Take part in maintenance/consensus

4.2.1 Reflection

- Who does take part in Bitcoin's blockchain?
  - Initially, anybody's PC was good enough
  - [**SZ18**] Then GPUs, now millions of dollars for ASICs
    - Barrier-to-entry
    - But also barrier-to-exit
      - Special-purpose hardware not much use for anything else
  - [**GBE+18**] Analysis of 10 months of data, starting July 2016
    - Weekly mining power of top Bitcoin miner about 20%

        - (Ethereum: 25%)
    - Top 4 Bitcoin miners have more than 53% of mining power
        - (Top 3 with 61% in Ethereum)
    - 90% of mining power controlled by 16 Bitcoin miners
        - (11 in Ethereum)

4.3 Immutable, Append-Only?

- Frequently, blockchains are called immutable
  - Clearly, they are append-only at best
  - Bitcoin only offers **probabilistic** guarantees
    - Forks/orphans violate append-only property

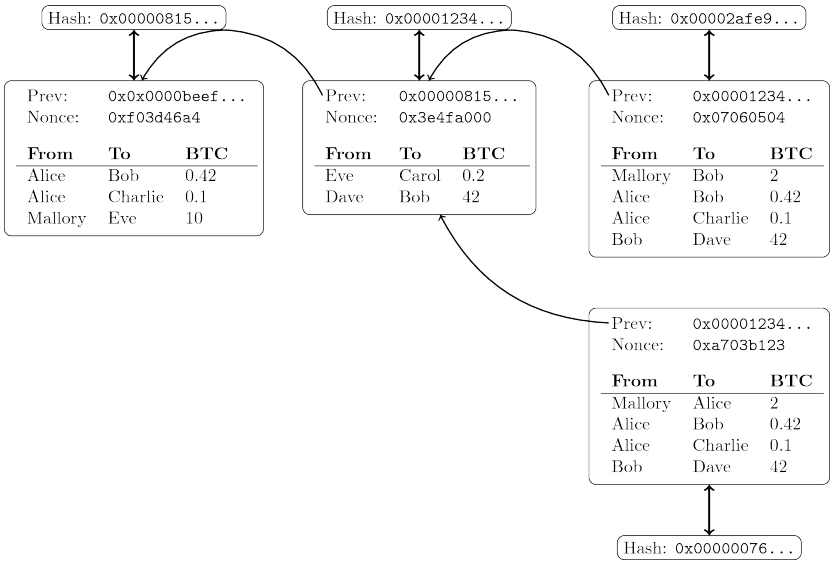

Figure 5: "Block tree of transactions" by Jens Lechtenbörger under CC BY-SA 4.0; from GitLab

4.4 Designed to Persist?

- Bitcoin **defines** rules for mining
  - E.g., mine on longest chain, publish mined blocks, include all known transactions
  - Block rewards, transaction fees as **incentives**
- Rational miners may behave differently
  - [**ES14**]: **Selfish mining**
    - Technique to collect block rewards beyond expected share corresponding to mining power

  - [**CKW+16**]: Rules change when block rewards are negligible and transaction fees dominate
    * **Selfish mining** even more profitable
    * New **undercutting attacks**
      · "At worst, consensus will break down due to block withholding or increasingly aggressive undercutting."

- Selfish mining
  - Do not publish mined bock, but extend secretly to obtain longest chain; publish only when others catch up; mining power of others is wasted on to be orphaned blocks
    * Beneficial at any percentage of mining power
    * Proposed protocol modification so that selfish mining is only effective for miners with more than 25% of mining power
      · Such miners may well exist [**ES14**; **GBE+18**]
      · Also, [**CKW+16**] show that selfish mining can be profitable for any miner
- Undercutting
  - Fork chain when block contains "lots" of transaction fees, include only some in own alternative as incentive for others to extend own alternative
  - Equilibrium where minors include only fraction of transactions, leading to indefinitely growing backlog

4.5 Justifiable?

- Proof-of-work (Bitcoin, Ethereum) is ecologically disastrous. I find its use unethical.
  - Digiconomist estimates yearly Bitcoin energy consumption of 112 TWh as of 2021-05-09 (up from 61 TWh as of 2018-04-20).

Figure 6: "Figure" under CC0 1.0; from Pixabay

    * 1135 KWh **per transaction**
      · Do you know your household's monthly energy usage?
  - With rising bitcoin prices, miners invest more in energy ...
- 2021-05-12: Tesla stopped accepting Bitcoins for fossil fuel usage

5 Conclusions

5.1 Summary

- Blockchains aim for decentralized consensus over shared data
  - No database per se, but building block
- *The* blockchain does not exist
  - Advertised characteristics need to be analyzed carefully
- As usual, choice of terms is crucial
  - Ask for meaning/definition

5.2 Constructive Thoughts

- Different (blockchain) scenarios come with different requirements
  - Specify requirements first
  - Select supporting technology afterwards
    - E.g., a digital notary service based on linked timestamping (see [**NC17**]) or a distributed database with replication may be good enough

5.2.1 Effect of Signatures

- Digital signatures (even without blockchain) provide tamper-proof, verifiable, decentralized evidence of statements/transactions.
  - (If implemented properly based on strong crypto.)
- Transactions in Bitcoin cannot be revoked.
  - Attempts of double-spending have no negative effect on attacker.
    - Double-spending "resolved" at cost of randomly chosen victim.
    - Although attacker's account is publicly visible!
    - Is that a requirement for your blockchain scenario as well?

5.2.2 Sample Application Scenarios

- Educational certificates on blockchains
  - Different security needs than e-cash
    - Must be revocable, e.g., to punish plagiarism
    - Double-spending not an issue
    - Little incentive for educational institutions to lie about previously issued certificates
- Other certifications, e.g., for real-estate may be embedded into legal regulations

  - Double spending leads attacker into jail

- Certificate Transparency
  - Notary service without blockchain
  - Append-only, open to public audits

Bibliography

License Information