Blockchain and Consensus

Module 2

Clemens H. **Cap**
clemens.cap@uni-rostock.de

Department of Computer Science
University of Rostock
Rostock, Germany

Blockchain & Smart Contracts
DAAD-sponsored Summer School Blossom
Sept. 2019 Tallinn, Estonia
Version 1.4 typeset September 12, 2019

Citing these slides

Clemens H. **Cap**, Blockchain and Consensus, Blossom Summer School Module 2, 2019, Tallinn, Estonia. Github Repository: clecap/blockchain-masterclass

Cut – Copy – Paste, Steal-and-Reuse is wide-spread and nevertheless **illegal**.

You may use the contents of these slides provided you give proper credit by citing them as outlined above.

The contents of these slides is of informational nature only. The slides do not constitute legal, financial or investment advice. The author assumes no legal responsibility for completeness or correctness. An increasingly wild-west behavior in web & world makes these remarks necessary. :-)

Table of Contents

Why Distributed Systems?

- Higher **performance** — concurrent compute
- Higher **data rate** — concurrent read, write
- Smaller **latency** — node 'round the corner
- Higher **availability** — backup nodes ⇒ less downtime
- Higher **configurability** — take offline for reconfig
- Higher **reliability** — multiple computations & crosscheck
- Higher **stability** — no single point of failure
- ...

Why No Distributed Systems?

Too high overall **system complexity**

- **Heterogeneity** — HW, SW, versions, admin discipline
- Larger **attack surface** — $\Leftarrow$ more nodes
- More people involved — $\Rightarrow$ less consensus, more misunderstandings
- Smaller **reliability** — $\Leftarrow$ more & remote failure modes
- Smaller **stability** — System effects
- ...

"Definition" of a Distributed System by Leslie Lamport

A distributed system is one that **prevents you from working** because of the failure of a machine that you had never heard of.

Main task in a distributed system:

Contain the inherent complexity.

Main task in a distributed system:

Use the advantages while avoiding their price

Informal Problem Statement: What consensus is about.

Achieve reliable system operation in a distributed system

- distributed system — fully distributed? some trusted nodes? PKI needed?
- failure model — how? when? which? detectable?
- communication model — synchronous, asynchronous, bounded
- termination model — is it terminated or has it failed?

Limitations: Why consensus is difficult.

- **Termination** cannot be proved
- **Correctness** cannot be proved
- **Location** of failure impossible — remote or router or intermed?
- **Detection** of failure impossible — crashed or slow or looping?

Question: What actually **is** "reliable system operation" as a notion?

Answer: Need to **simplify** definitions and employ **models**!

A short review of TCS...

How would I see that termination cannot be proved / decided algorithmically?

How would I see that correctness cannot be proved / decided algorithmically?

Function 1 of 3: Measure of Value

Problem: 90 minutes lecture Cap = **???** minutes dentist

Questions:

- Is there an **objective** measure of value? **No!**
- **Where** do value measures **originate**? It's about demand and supply!
- **Who defines** it? Collective behavior
- How to implement **stable** trust? Assume greed & rationality ⇒ Game theory

Function 2 of 3: Medium of Value Exchange

Problem:

- Clemens Cap offers lecture wants fish'n chips
- Summer school student has a cow
- **How to transform** a cow into fish'n chips?

Questions:

- How do we **split values** into smaller denominations?
- How do we implement exchange if **offered and wanted medium are different**?
- Can we have a **common value standard**?
- How do we implement **chains of value exchange**?

Function 3 of 3: Deferring Value Exchange (aka Store)

Problem:

- Clemens Cap offers summer school lectures in **2005-2035**
- Clemens Cap wants a steak with fries and salad in **2045**

Questions:

- **How** do we store value (or: defer value exchange)?
- Does the value **change** while stored?
  - Shall it increase? (eg: savings, investment)
  - Shall it decrease? (eg: inflation, discounting, stimulating exchange)
  - May it vanish? (eg: crash, theft, for promoting thorough risk assessment into storage method)
- Is there a **backing** of the value? (eg. in paper, shares, gold, time, energy etc.)

Money as Unlimited Optionality

> At no point does anyone in the chain know what to do with money in the real economy. But in an indefinite world, people actually *prefer* unlimited optionality; money is more valuable than anything you could possibly do with it. Only in a definite future is money a means to an end, not the end itself.

Figure 1: Peter Thiel: From Zero to One: Notes on Startups, or How to Build the Future. Currency Publisher, 2014.

Money may be considered as

- a **right**
- to execute a **specific transferal transaction**
- which can be executed by the **owner** of the right
- **exactly once**
- and which is **transferred** to another person
- **only** by executing the transferal transaction

Consider

- What are these elements in traditional money?
- What are these elements in various forms of digital moneys?
- Which aspects are easy to implement in a digital manner – and how?
- Which aspects are easy to implement in centralized architectures? In P2P?

Mechanism:

- **Central bank** creates monetary units.

  Different mechanisms (printed money, book money, fractional reserve banking, mortgage loans etc.)

- **Parliament** creates backing by law.

  Where entitled to settle debt in £?

Structural Analysis:

- Single point of **failure**.

  One bank(er) running amok may produce inflation.

- Single point of **responsibility**.

  More immune against external influences. Not so much under populistic political control.

- Democratic, but multiple **limitations**.
  - Very slow reaction.
  - Via long chains of intermediaries.
  - Good or bad?

tweedback

Session: blossom19

Quiz: How many different crypto currencies did you ever own?

(a) None

(b) 1

(c) 2

(d) 3 or more

tweedback

Session: blossom19
Quiz: How many different crypto currencies did you mine?

(a) None
(b) 1
(c) 2
(d) 3 or more

Discussion:

Which motives did **you** have for owning / studying Bitcoin / blockchain?

Bitcoin
Some motives

1. Protect against inflation

Prevent political decisions in a monetary system which eventually could lead to inflation.

Figure 2: German Papiermark

2. Protect against next Lehman crisis

So Bitcoin was born in an age when the financial sector divulged a dark secret, and showed that trust in banks could falter. Trustless money was the answer of Satoshi Nakamoto.
(Christine Masters: The Lehman Brothers Bankrupty: How it Triggered the Rise of Bitcoin)

Figure 3: Berlin bank run, July 1931

3. Escape negative interest rates

Negative interest rates are ... well ... *negative* for us:

- Urge consumers to spend.
- Undermine financial decision autonomy of citizen.
- Ruin financial provisions for old age.
- Are politically doubtful (is "happyness = permanent growth" ?)
- Could be escaped with value store under control of citizens.

4. Denial of service based on policy or identity

BBC News: PayPal has said that its decision to stop people from using its service to make dontations to Wikileaks was made after a letter from the US government. ... Datacell claimed in its statement that Visa had come under political pressure and had "put priority on political influence over the law".

See also: Süddeutsche Zeitung

Figure 4: Credit card companies cooperating with KKK

Question

Can we build a monetary system which is immune against attacks on civil liberty?

Necessary features

- **Fully decentralized P2P** with no single point of action
- **Open** to **anonymous & private** participation of everybody
- Governed by a **majority consensus** of participating entities
- **Highly replicated** and thus robust against attacks (especially: (d)DoS & Sybil)
- Secured by **cryptography** ,not by human trust or social power
- Majority of nodes adhering **consistently** to governance decided upon by majority

Phase 1: Email

- Are data important?

Phase 1: Email

- Are data important?
- Data are important!

Phase 1: Email

- Are data important?
- Data are important!
- Data are hype

Phase 1: Email

- Are data important?
- Data are important!
- Data are hype
- Everybody does *something* with data

Digital Disruption

Phase 1: Email

- Are data important?
- Data are important!
- Data are hype
- Everybody does *something* with data

Problem: Nobody adjusts the processes

- Using email to broadcast holiday pictures to friends
- Implementing "digital teaching" by distributing PDF

Phase 2: Intermediaries

- Recognize special needs

Phase 2: Intermediaries

- Recognize special needs
- Adjust processes to application scenarios

Phase 2: Intermediaries

- Recognize special needs
- Adjust processes to application scenarios
- Uber, Tinder, AirBnB, Facebook, Google & Co. introduce specialized solutions

Phase 2: Intermediaries

- Recognize special needs
- Adjust processes to application scenarios
- Uber, Tinder, AirBnB, Facebook, Google & Co. introduce specialized solutions
- Everybody provide their preferences and private data

Phase 2: Intermediaries

- Recognize special needs
- Adjust processes to application scenarios
- Uber, Tinder, AirBnB, Facebook, Google & Co. introduce specialized solutions
- Everybody provide their preferences and private data
- Everything becomes available for free

Digital Disruption

Phase 2: Intermediaries

- Recognize special needs
- Adjust processes to application scenarios
- Uber, Tinder, AirBnB, Facebook, Google & Co. introduce specialized solutions
- Everybody provide their preferences and private data
- Everything becomes available for free

Problem: User lock-in in TOS

- What *exactly* are they doing to my data?
- Why can't I have it my way? (no ads, spam filtering, UI adaption, platform migration, data sovereignty, ...)

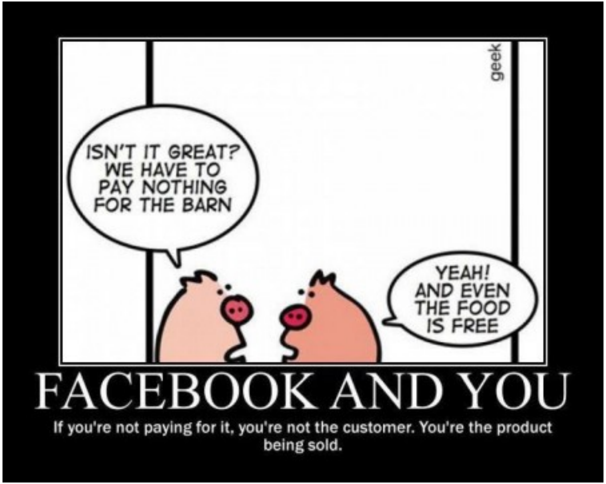

Figure 5: If you are not paying for it, you are the product being sold.

tweedback

Session: blossom19

Quiz: How many of the following systems have you used?

(a) 0
(b) 1
(c) 2
(d) 3 or more

Selection

Friendica	Diaspora	Identica	Libertree
Mastodon	Movim	Twister	Galaxy2

Digital Disruption

Problem 1: Value generation

Without value generation of intermediaries there are **no incentives** for

- dissemination & marketing & branding
- un-nerding & mainstreaming
- user studies on UI quality
- bug removal & feature proliferation & lanuage localization

Problem 2: Adherence to community standards

How do we **enforce** community rules by open & democratic standards

- Consensus — With n nodes complexity n^2
- Benevolent dictator

Linus Torvalds	✓
Mark Zuckerberg	?
Christine Lagarde	?
Mario Draghi	?

Platonic problem: Quis custodiet ipsos custodes? (Who guards the guardians?)

Digital Disruption

Value generation: Bitcoin blockchain comes with

~~batteries included~~

~~$ included~~

฿included.

Problem solved ✓

Adherence to community standards:
Bitcoin started with this goal for the monetary system and there solves it successfully.

Bitcoin enforces community standard:

$$\sum \text{deposits} - \sum \text{withdrawals} = \text{balance}$$

$$\text{balance} \geq 0$$

Ethereum enforces complex community standards (aka *smart contracts*)

Problem solved ✓

```
pragma solidity >=0.4.22 <0.6.0;

/// @title Voting with delegation.
contract Ballot {
    // This declares a new complex type which will
    // be used for variables later.
    // It will represent a single voter.
    struct Voter {
        uint weight; // weight is accumulated by delegation
        bool voted;  // if true, that person already voted
        address delegate; // person delegated to
        uint vote;   // index of the voted proposal
    }

    // This is a type for a single proposal.
    struct Proposal {
        bytes32 name;   // short name (up to 32 bytes)
        uint voteCount; // number of accumulated votes
    }

    address public chairperson;

    // This declares a state variable that
    // stores a `Voter` struct for each possible address.
    mapping(address => Voter) public voters;

    // A dynamically-sized array of `Proposal` structs.
    Proposal[] public proposals;

    /// Create a new ballot to choose one of `proposalNames`.
    constructor(bytes32[] memory proposalNames) public {
        chairperson = msg.sender;
        voters[chairperson].weight = 1;
```

Figure 6: Delegated voting smart contract specification. https://solidity.readthedocs.io/en/v0.5.3/solidity-by-example.html

How does the blockchain solve the phase 3 problems?

Everybody generates their identity themselves

- **Thus:** Nobody unfairly cut out
- **How:** Randomly generate a public-private key pair (e, d)
- **Use:** Know private key d (proof: signature) $\Rightarrow$ own account of public e
- **Issue:** Collision of random key pairs?
- **Solve:** Very small chance of 2^{-256}

Everybody can/may operate a bitcoin node

- **Thus:** There is always a bitcoin bank open for you :-)
- **Issue:** Why would anybody want to do that?
- **Solve:**
  - Fun
  - Mining bounties
  - Mining fees

Everybody broadcasts & stores all transactions & replies to account status queries

- **Thus:** Robust, available storage in face of node failures and network partitions
- **Issue:** Scalability? Side & state channels, light weight nodes ... not our topic ✓
- **Issue:** Consistency? **Big problem & our topic**

Where and how are the following aspects required in bitcoin?

- Game Theory
- Mechanism Design
- Real Time
- Feedback
- Proof of Work
- (Crypto) Hash Chains
- Signatures
- Randomization
- Consensus

Consensus

Main sources of blockchain consensus problems

1. Network & processing latencies – unavoidable side effect

- Alice generates, signs & broadcasts a transaction.
  Bob has heard from it, Carol not yet
- Donald has formed a new block, Eric has not yet heard from it
- Fred has just formed a new block, Greg has also minted one at the same time

2. Double spending – active attack

Mallory maliciously sends out conflicting transactions to different nodes.

3. Malicious nodes – active attack

Mallory maliciously gives inconsistent answers to requests

4. Sybil nodes – active attack

Mallory acts as Mallory-1, Mallory-2, Mallory-3 to influence "majority" consensus

Overall Problem

Achieve reliable system operation in a system

- distributed system — several nodes
- failure model — how, which, detectable
- attack model — active, passive; capabilities
- communication model

Consistency versus moral correctness

Example 1: Mallory was doublespending – first to Bob, only much later to Carol. Bob should get the money – but Carol gets it

Example 2: Mallory doublespends to Bob and to herself and – by chance – manages to mint a block with the spending to herself.
Carol mints a block with the spending going to Bob.
Bob confirms the payment with Carol, does not wait long enough, ships the merchandise and loses the payment.

One ~~man~~ hash – one vote: Hash beats node numbers!

Example: Different versions $v_1 \neq v_2$ of the algorithm are used in the network.
Minority supports v_1 but wins due to more hash performance.

Random & dynamic elements

Example: Different versions $v_1 \neq v_2$ of the algorithm are used in the network.
Hash minority supports v_1 but by mere chance produces 5 blocks in a row.
Ultimate fate depends on future behavior of hash majority (eg. late switching over to v_1).

Let us look at a more simple model in form of an anecdote!

There are $n = 4$ armies around Byzanz.

Every army is commanded by a general.
One of the generals is the commander.
It is possible that one of the generals is a traitor.
The goal of the traitor is to confuse the armies.
As a result, a too small number of armies attack and the battle is lost.

tweedback

Session: blossom19

Quiz: The following is true for Byzantine generals:

1. What's that?
2. Heard of it
3. Know algorithm
4. Know proof or have programmed it

Military situation:

- If $n - 1 = 3$ or more armies attack they will win the battle.
- If $n - 2 = 2$ or less armies attack they will lose the battle.

Communication:

- The generals communicate via army-to-army messengers.
- Every sent message is delivered correctly.
- The receiver of a message knows who sent it.
- The absence of a message can be detected.

Military order:

- A loyal commander gives the same commands to his generals.
- A loyal general obeys the commander.
- The commander may be a traitor.
- To protect themselves against a traitor in command,
  the generals may disobey the commander
  provided there is consensus to do so.

Is there a protocol to win the battle?

- **First** look at a situation where this does not work out:
  3 generals of which 1 traitor
- **Then** look at a situation where this works out:
  4 generals of which 1 traitor
- **Generalize** the situation without full proof of scheme.
- **Finally** collect the lose ends.

Case 1: Commander is a traitor.

- Commander to General1: Attack!
- Commander to General2: Retreat!
- General1 to General2: He ordered attack.
- General2 to General1: He ordered retreat.
- Loyal General1 receives two contradicting statements: "Attack!" and "He ordered retreat"
- Loyal General1 cannot make local majority decision.
- Loyal General1 cannot distinguish 2 cases:
    1. Commander is a traitor
    2. General2 is a traitor.

Case 2: Commander is loyal

Without loss of generality: Assume General2 is a traitor.

- Commander to General1: Attack!
- Commander to General2: Attack!
- General1 to General2: He ordered attack.
- General2 to General1: He ordered retreat.
- Loyal General1 receives two contradicting statements: "Attack!" and "He ordered retreat"
- Loyal General1 cannot make local majority decision.
- Loyal General1 cannot distinguish 2 cases:
  1. Commander is a traitor
  2. General2 is a traitor.

Case 1: Commander is a traitor.

- Commander to General1: Attack!
- Commander to General2: Retreat!
- Commander to General3: Attack!
- Loyal Generals exchange received messages.
- Loyal General1 receives: "Attack", "He ordered retreat", "He ordered attack"
  Loyal General1 takes local majority decision "Attack"
- Loyal General2 receives: "Retreat", "He ordered attack", "He ordered attack"
  Loyal General2 takes local majority decision "Attack"
- Loyal General3 receives: "Attack", "He ordered retreat", "He ordered attack"
  Loyal General3 takes local majority decision "Attack"
- Loyal Generals1,2,3 attack, thereby taking the same consensus decision.
  Irrelevant that General2 disobeys the commander, since commander is a traitor.

Case 2: Commander is loyal

Assume General3 is a traitor.

- Commander to General1: "Attack"
- Commander to General2: "Attack"
- Commander to General3: "Attack"
- Generals exchange received messages.
- Loyal Generals1,2 receive "Attack", at least one "He said attack" and one more.
- Loyal Generals1,2 take local majority decision to attack
  thereby taking the same consensus decision
  which also is the same as the decision of the commander

$n = 3$

No consensus is possible.

$n = 4$

A consensus is possible.
Everybody talks to everbody else what everybody else had said.

General n

A consensus is possible if **strictly less than** $\frac{n}{3}$ nodes are traitors.
The general protocol uses multiple hierarchical
"X said that Y said that Z said that U said..."
type of messages.

Protocol thus far not truly distributed!

The role of Commander is a single point of decision.
No complete homogeneity of nodes!
But: This can be solved as well!

The commander protocol established:

- All loyal partners end up with the same opinion
- If Commander is loyal: This is what the commander ordered.
- If Commander is traitor: Loyal partners still share a consistent view
  Never mind that this is not what Commander ordered.
  He is a traitor and gave conflicting orders.
  Thus he did not really give an order.

Final solution:

- Use 4 rounds: Every general may play commander once.
- Result: All loyal generals have the same opinion on what the other generals (including the traitor) believe.
- Important is only the **consensus among the loyals**
- The loyals now run the same deterministic decision algorithm on identical input.
- All loyals end up with the same overall decision.
- More efficient: Combine rounds by sending vectors.

n nodes

t traitors

Criterion: $n > 3t$

Communication complexity: $O(n^t)$

Assume $n = 100.000$ bitcoin participants

Assume $t = 10.000$ traitors

Communication complexity becomes ... **oops**

Cryptographic Approach

- Assume a PKI
- Every general signs his messages
- Traitor can no longer communicate in a contradicting manner!

What would be the **disadvantages** of the cryptographic approach to BFT if it *were* used in bitcoin?

Federated Approach

- Cut down on growth of complexity
- Use small local clusters with $n \sim 15$ nodes
- Delegate one node in the cluster as representative to next level
- Use small regional clusters with $n \sim 15$ representatives
- Continue with this concept

What would be the **dangers** of the federated approach if it *were* used in bitcoin?

2016 12th European Dependable Computing Conference

A Performance Comparison of Algorithms for Byzantine Agreement in Distributed Systems

Shreya Agrawal
Cheriton School of Computer Science
University of Waterloo
shreya.agrawal@uwaterloo.ca

Khuzaima Daudjee
Cheriton School of Computer Science
University of Waterloo
kdaudjee@uwaterloo.ca

Figure 7: Research on variants of Byzantine agreement.

TABLE I
A SUMMARY OF FEATURES OF THE ALGORITHMS UNDER EVALUATION

Algorithm	**Type**	**n**	**Rounds**	**Bit Complexity**	**Decision value**	**Communicating nodes**	**Remarks**
Ben-Or, Pavlov, Vaikuntanathan [7] (*Quorum*)	Randomized	$4k+1$	$O(\log n)$	$n^{O(\log n)}$	String of $O(\log n)$ bits	All-to-all communication and within quorums of size $O(\log n)$	Everywhere byzantine agreement
Braud-Santoni et al. [9] (*Pull-Push*)	Randomized	$3k+1$	$O(\frac{\log n}{\log \log n})$	$\tilde{o}(n)$	String of $O(\log n)$ bits	With samplers of size $O(\log n)$	Almost-everywhere to everywhere
Kowalski and Mostefaoui [30] (*EIG*)	Deterministic	$3k+1$	$k+1$	$O(n^3 \log n)$	Single bit	All-to-all communication	Uses EIG data structure

Figure 8: Research on variants of Byzantine agreement: For the deterministic case is stays pretty bad!

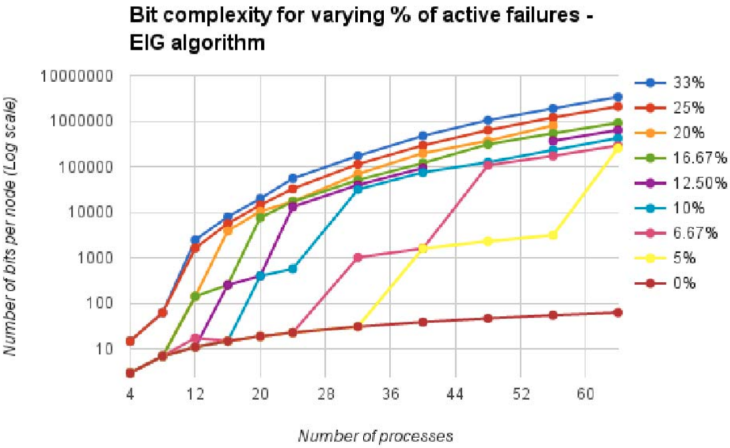

Figure 9: Research on variants of Byzantine agreement: Still pretty bad performance scalability for small number of nodes.

When storing data in a distributed system, we are interested in 3 properties **C – A – P**.

C: Consistency

Every read receives the most recently succesfully written value – or an error.

Note: "Most recently written" is a topic for a summer school on it's own.

- Communication latency may change the order.
- Fault tolerance mechanisms may change the order.
- According to Einstein, physics itself knows no consistent order on all events.
- Needs Lamport & vector clocks, virtual synchrony, atomic broadcast & co.

CAP Theorem
Definition of properties (2)

A: Availability

Every request receives a non-error response.
There are no guarantees on consistency of the result.

P: Partition Tolerance

Gilbert & Lynch: No set of failures less than total network failure is allowed to cause the system to respond incorrectly.

CAP theorem as conjectured by Brewer in 2000

Out of $\{C, A, P\}$ an implementation can do at most two.

CAP theorem as formulated by Gilbert & Lynch

In a network subject to communication failures, it is impossible for any web service to implement an atomic read/write shared memory that guarantees a response to every request.

See:

- Brewer: Towards Robust Distributed Systems
- Gilbert & Lynch: Perspectives on the CAP Theorem
- Gilbert & Lynch: Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services.
- Abadi: Consistency Tradeoffs in Modern Distributed Database System Design (For extensions of the CAP-theorem to the **PACELC-theorem** describing **further trade-offs between consistency and latency**.)

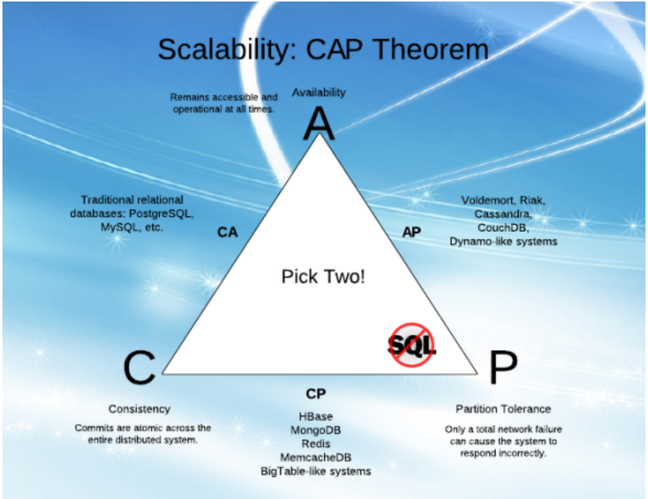

Figure 10: The tricky choice of the system architect. Image by image source.

The crucial partition decision

Suppose an operation times out. You now can

- **cancel** the operation and decrease availability.

  – XOR –

- **proceed** with the operation and risk inconsistency.

CA systems drop partition tolerance

Put everything related to a specific transaction on one node or an atomically failing cluster.

Analysis:

- Does not scale well.
- Is not robust against losses of sites and/or connectivity.
- **Traits:** Commit & multi-phase protocols involving all nodes, closely coupled (single-rack) cluster architectures

AP systems drop consistency

Eventually consistent systems accept outdated responses once in a while.
The system status *finally will* converge to the most recently written value.

Analysis: Participants get wrong answers once in a while.

- Participants **must not rely-and-react immediately** on the answer of the system.
- **Solution 1:** Design systems with **room for error and non-finality**
  Example 1.1: Allow for compensatory transactions.
  Example 1.2: Allow for manual exception handling.
- **Solution 2:** Design systems with **slack time until finality**
  Example: **Bitcoin:** Wait 6-8 blocks from "*transaction has cleared system*" to "*transaction may be considered paid and goods may be shipped*".
- **Traits:** Mechanisms for expiration & lease (TTL), conflict detection & resolution.

CP systems drop availability

On suspecting a partition event, wait until data is consistent and r
remain unavailable until that moment.

Analysis:

- Network partitioning and healing difficult to detect.
- Logic for getting failed or disconnected nodes consistent & online may be complex.
- **Traits:** Pessimistic locking & majority counting protocols, unavailable partition minorities

ACID

- **A**tomicity: Each TX is an undivisible unit – failing or succeeding completely.
- **C**onsistency[a]: TXs transform DB from valid state to valid state.
- **I**solation: Effects of an incomplete TX are not visible to other TXs.
- **D**urability: A committed TX has its effects recorded in persistent DB state.

[a] **Cave**: This is a different notion of consistency than in CAP! More on this here.

BASE

- **B**asically **A**vailable but not necessarily guaranteed availability
  Reads & writes may go missing but will not compromise (later) (eventual) consistency
- **S**oft state: No hard guarantees on a state
  which has (not yet) converged but will do so later
- **E**ventually consistent: State will sooner or later converge.

CAP Theorem
Why would we want to settle for BASE?

BASE offers

- Simpler system design
- Faster transactions
- Better scalability
- Higher availability
- Smaller downtime

Price to pay: Only weak consistency, which means...

- **Delayed data** may occur: Data was like that some time ago.
- **Stale data** may occur: State is shown, but no longer exists.
- **Mechanisms** are necessary which detect and fix this

Assume transaction $\lambda x.x + 100$

If state is consistent: Apply transaction to last (=correct, most recent) state.

If state is not consistent: No guarantee on correctness of base state.

Repeated reads of state provides to client: $88, 200, 94, 451, \ldots$

Solution 1: **Commuting** transactions

If all transactions **commute** $\Rightarrow$ do whenever you want.

Add and subtract transactions commute, but

- Human spending decisions do not commute
- Balance sheet transactions do not commute
  Changing TX sequence may lead to temporarily overdrawn account.

Solution 2: **Sequence numbers** on states.

Solution 3: **Chain** of states.

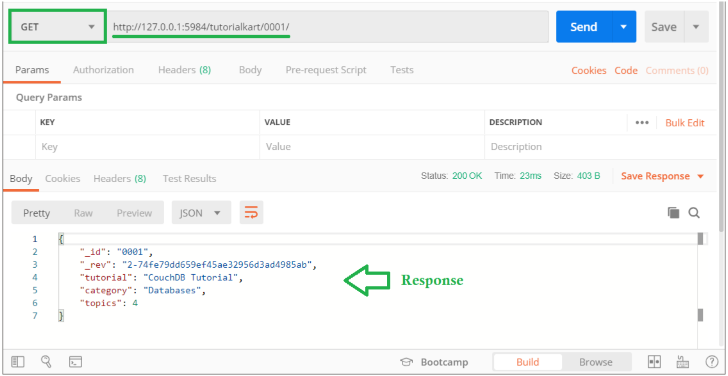

Figure 11: NoSQL Database CouchDB using revision stamps to make sure that transaction is operating on the correct DB state.

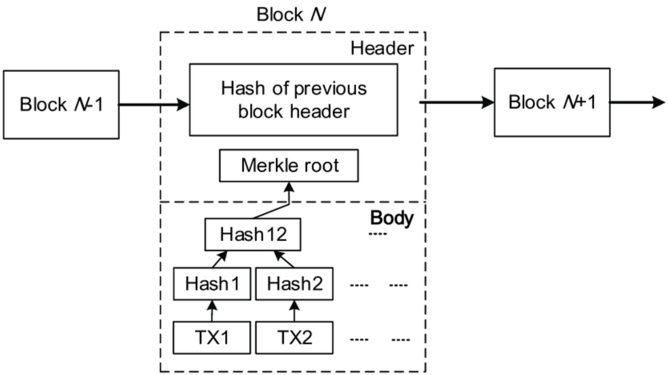

Figure 12: Blockchain presents a sequence of states in time.

Chain provides a sequence of states

- But: There may be several TX involving the same account arriving at different nodes in different order
- Resolution by *real-time clocks*: Unreliable (clock-drift)
- Resolution by *time-stamp algorithm*: Too complex algorithm.
- Resolution in bitcoin:
  Locally: By random winner of PoW
  Globally: Selfish nodes prefer as chain the longest branch

Additional roles of chain

- Conflict resolution by "**rule of longest branch**"
- Cannot change past without redoing entire chain **linked (crypto)hash pointers**
- Redoing entire chain is very costly **proof of work**

The classical three:

1. **Proof of Work (PoW)**
2. **Proof of Stake (PoS)**
3. **Proof of Authority (PoA)**

And four more:

4. Proof of Weight (PoW)
5. Byzantine Fault Tolerance (BFT)
6. Directed Acyclic Graphs (DAG)
7. Consensus by Delegation (CD)

Idea: A limited resource is restricting the number of votes
Examples: Bitcoin, Ethereum, Litecoin, Dogecoin
Pro: The classical, *orthodox* **blockchain scheme**

- Stable and secure
- Established track record of success
- All nodes anonymous

Con: Resource consumption

- Slow
- Power consumption
- Power consumed is wasted — no useful job done (or: rainbow table precomputation)
- Incentive for mining pool cooperation is recentralization

See also: Satoshi Whitepaper

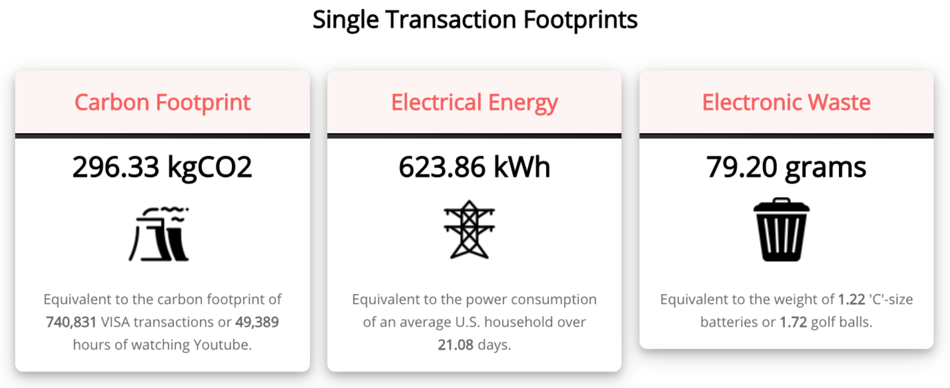

Figure 13: Digiconomist, Bitcoin Energy Consumption in 2019. https://digiconomist.net/bitcoin-energy-consumption

Cryptocurrencies and Consensus
Proof of Work: Energy Consumption

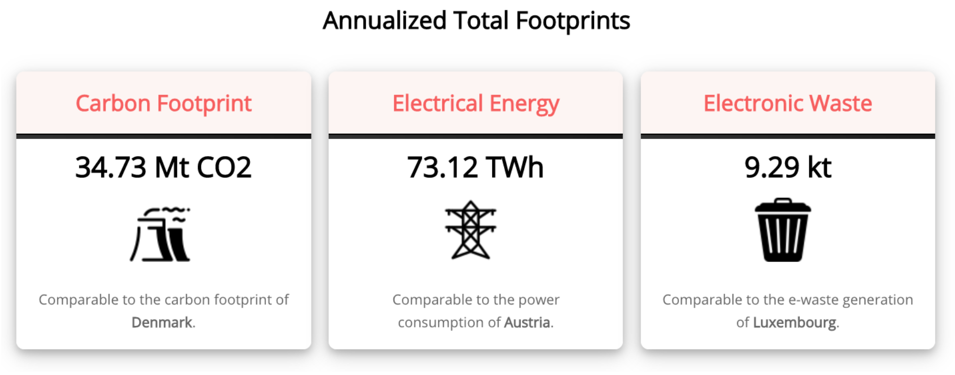

Figure 14: Digiconomist, Bitcoin Energy Consumption in 2019. https://digiconomist.net/bitcoin-energy-consumption

Figure 15: GPSLeo: FridaysForFuture, Wikimedia Commons, Used by CC0 1.0

This power consumption of blockchains is **not** a justification for

- skipping school, university – or Blossom lectures ;-)
- promoting panic
- turning irrational

but it **is** a justification for research towards better efficiency.

Idea: Higher risk / stake ⇒ higher interest in correct functioning of the system

- Miners bet tokens on valid parts of tree; bet lost if majority votes differently

Examples: Peercoin, Decred, Ethereum 2.0 (since May 2019)

Pro: Better ressource situation

- Less energy costs (thus: better CO_2 footprint, reduced incentive for pool formation, ...)
- Bad behavior more costly (lose placed bet vs. waste CPU cycles on wrong branch)

Con: Nothing at Stake problem

- Validators vote for and work on both sides of a fork
- Risk by participants with irrational behavior not caring about costs
- Risk by participants wanting to ruin the chain at all costs

Attempts to repair *nothing at stake problem*:

Punish voting on both variants
Additional penalty for voting on what finally is wrong chain

See also: Detailed FAQ on PoS, Ethereum Casper 101, and Casper White Paper

Idea: TX validation by authorities (i.e. approved, well-known, identified nodes)
Examples: POA.Network, Kovan@Ethereum, R3 (fintech, digital assets), EWF (energy), b3i (insurance)
Pro: Ressource Usage

- High throughput and scalability
- Soft on all kinds of resources

Con: Centralization & Needs Trusted Legal System

- Small number of powerful nodes
- Need backing of a legal system in case of authority fraud
- Needs trustworthy mechanism for establishing authority identity (PKI)
- No protection against discrimination by the authority

See also: PoA Network Whitepaper and De Angelis et al, PBFT vs Proof-of-Authority

Idea: Probability of minting next block proportional to some relatively weighted value, not necessarily coupled to system tokens as in PoS.

Examples: IPFS – Inter-Planetary-File-System (weight = amount of storage provided)

Pro:

- Customizable scalability

Con:

- Incentivation difficult as it is not coupled to tokens

See also: Algorand Whitepaper, Filecoin Whitepaper

Idea: Use byzantine fault tolerance algorithms in different versions.

- Classical
- Federated
- Signed

Examples: Hyperledger, Ripple, Stellar
Pro: Dependent on specific algo features
Con: Dependent on specific algo features

- Classical: Only small n
- Federated: Attacks by delegates possible.
- Signed: Need a PKI, not fully distributed

See also: Castro, Liskov: Practical Byzantine Fault Tolerance Mazieres, The Stellar Consensus Protocol
Federated Byzantine Agreement

Idea:

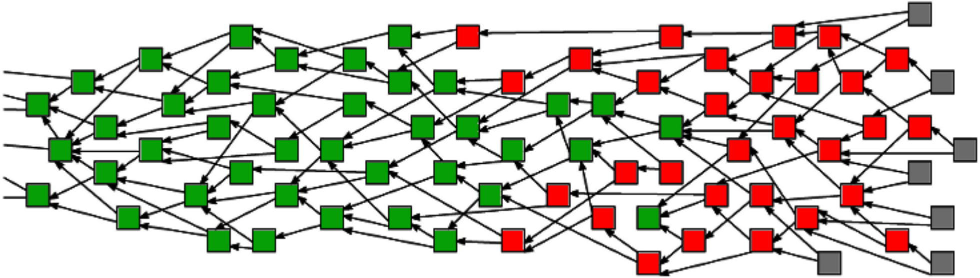

Figure 16: DAGs focused on front covering instead of trees focused on a single valid chain.

- Various mechanisms to grow the DAG and validate new nodes

Examples: IOTA, Hashgraphs, Nano

Pro:

- Higher transaction rates
- Better scalability
- Maybe even suitable for IoT devices

Con: Highly dependent on specific implementation

- Rumors of loopholes (IOTA)
- Some degree of centralization might be necessary
- Trading speed for security

Idea: Delegation

Examples: Various forms of side chain currencies.

Pro:

- Much softer on resources
- Much better scalability
- Much faster transaction clearance (up to 1 block/sec)

Con:

- Centralized concept

Warning

Most[a]

- **proof of authority** blockchains
- **private** blockchains
- **commissioned** blockchains
- **cloud provisioned** blockchains
- **delegated consensus** blockchains
- **directed acyclic graph** blockchains

are not blockchains in the orthodox sense.

[a] Generally speaking; the mileage may vary depending on the specific chain in question.

An orthodox blockchain

Warning: Centralized Institution

- PKI or CA for establishing identity of authorities (cf. most PoA)
- Coordinator node for ensuring proper top selection (cf. IOTA)
- Directory servers or registry for onboading (cf. TOR directory server)
- Instance for norming and standardizing protocols and APIs

Warning

- There is no centralized institution which

Cryptocurrency Freedom Manifesto (1)

(1) Right to an Account

No Bank or financial institution shall deny an account or close an account for a Person, Private Entity or Public Entity simply because of their possession of, sale of, or transactions based on cryptocurrency.

(2) Right to Unrestricted Anonymous Token Transfer

No Government Organization or Quasi Public Entity or Private Entity shall interfere or restrict the ability of a person or Public Entity or Private Entity to anonymously conduct cryptocurrency transfers from one wallet to another.

(3) Right to Token Convertibility

No Government Organization or Private Entity or Quasi Public Entity shall restrict the ability of a Person or Private Entity or other Public Entity to exchange fiat currency for cryptocurrency or vice versa (i.e. buying or selling crypto via fiat).

(4) Freedom of Token Transfer from Taxation

No Government Organization or Quasi Public Entity shall, or cross-governmental impose taxation on any individual or private entity conducting crypto to crypto transactions.

(5) Freedom from Duress

No Government Organization shall threaten to or actually imprison or fine a Person solely on the basis of buying, possessing, selling, trading, or transferring cryptocurrency (including tokens).

(6) Freedom from Registration

No Government Organization, Private Entity or Quasi Public Entity shall restrict the right of any Person or Private Entity or Government Organization citizen to purchase or sell cryptocurrency for fiat by requiring registration of any kind; this includes the need to present identification or proof of citizenship or other registration in order to conduct fiat to crypto transactions.

Source: Riz Virk, A Cryptocurrency Freedom Manifesto – Is it Too Late for Bitcoin?

Keep in mind:

Manifestos provide important stimuli but not always are the optimal approach.

Prinzipien erzeugen Brüche (principles produce cracks).

tweedback

Session: blossom19

Quiz: Which of the following rights and freedoms of the cryptocurrency manifesto are (more or less) guaranteed by Bitcoin, Monero or ZCash?

1. Right to an Account
2. Right to Unrestricted Anonymous token Transfer
3. Right to Token Convertibility
4. Freedom of Token Transfer from Taxation
5. Freedom from Duress
6. Freedom from Registration

Orthodox blockchains are blockchains which *implement* the rights and freedoms of the cryptocurrency manifesto *in code*.

For the "*Code is Law*" metaphora see also:Lawrence Lessig, Harvard Magazine, 1. 1. 2000.

tweedback

Session: blossom19

Quiz: Which of the following rights and freedoms of the cryptocurrency manifesto are taken away by a PoA blockchains?

1. Right to an Account
2. Right to Unrestricted Anonymous token Transfer
3. Right to Token Convertibility
4. Freedom of Token Transfer from Taxation
5. Freedom from Duress
6. Freedom from Registration

References

- Blockchain Working Group (124)
  - Main Topics (123)
    - Bitcoin (29)
    - Blockchain (91)
    - Crypto Currency (30)
    - Distributed Systems (24)
    - Economy (47)
    - Ethereum (22)
    - IOTA (1)
    - Internet of Things (13)
    - Law (12)
    - Monero (6)
    - Privacy (24)
    - Security (40)
    - Smart Contracts (36)
    - Theory (12)
  - Projects (124)
    - BloSSom 2019 (124)

Figure 17: Check out the 100+ papers on the ePrints repository for the summer school