Blockchain (de)constructed

Fritz Henglein
Department of Computer Science, University of Copenhagen (DIKU)

DIKU Business Club meeting on blockchain
January 11th, 2016
Fritz Henglein

- Professor of programming languages and systems
- Foundations, techniques, algorithmics
- Enterprise systems, healthcare, finance
- Director, Research center for high-performance computing for finance (HIPERFIT.dk)
- Steering committee chair, Innovation network for Finance IT (CFIR.dk)
- Mostly academic, some industrial lab/start-up experience
Resources, Events, Agents

- Accounting model
  - Models business events
  - *No prima facie* accounting artifacts (debit, credit, liability, ...)

- **Resource**: Goods, services, currencies

- **Agent**: Company, division, person

- **Event**:  
  - **Transfer**:  
    A1 has transferred R to A2.
  - **Production**:  
    A has produced R2 from R1.

McCarthy, 1982
Agents

• Real identity: Name, address, state-issued id, central registry number, ...
  • 1 real identity per 1 `real’ person

• Pseudonymous identity:
  • Public key: some sequence of bits, stored openly on blockchain
  • Private key: For proving ownership of public key
  • $n$ public/private key pairs per 1 `real’ person

• Bitcoin: If the public key can be associated with owner’s identity, all past transactions are deanonymized
Resources

- **Linear**: Can be moved, transferred, produced ("transformed"), but not duplicated.
  - Real things, economically scarce resources, (digital) assets
- **Nonlinear**: All of the above, and can be freely duplicated
  - Information (invoices, manuals, rules, …)
Events

- Event = Something that has happened; e.g.
  - Transfer, producing, issuing, transforming, moving
  - of resources
  - by agents
- Event log = (Date: Event)*
  - More generally: Acyclic graph of events
Contracts

• Specification of (sets of) sequences of permissible and obligatory events

• Bitcoin:
  • Only owner may transfer amount
  • Transfer amount must be less than owner’s balance
  • No double spending (= bitcoin is linear resource)
Business event log

Example:

- 2016-08-15T09:20+02:00: U transfers R30 to S
- 2016-08-15T09:24+02:00: S transfers R30 to P
- 2016-08-15T09:25+02:00: S transfers R30 to Q

Not valid

Valid
Example 2 (FX American Option). Party X may, within 90 days, decide whether to (immediately) buy 100 US dollars for a fixed rate 6.5 of Danish kroner from party Y.

\[
\text{if } \text{obs}(X \text{ exercises option, 0) within 90} \\
\text{then } 100 \times (\text{USD}(Y \rightarrow X) \& 6.5 \times \text{DKK}(X \rightarrow Y)) \\
\text{else } \emptyset
\]
Distributed system

• Network of
  
  • **actors** following instructions
  
  • **connections** for sending and receiving messages
  
  • to collectively offer
  
  • coherent service to users

• Nodes may be **reliable** (following instructions) or **unreliable** (crashing or faulty/lying)

• Connections may be **reliable** or **unreliable** (arbitrarily slow)
What is “blockchain”? 

- Blockchain: database ("distributed ledger")
  - containing a valid, tamper-proof (event) log
  - admitting append-only updating
  - distributed peer-to-peer system
  - enforcing fixed or user-defined contracts
  - anarchistic:
    - self-governing
    - resisting authority
Consensus

- Contract execution requires consensus on a single sequence of unalterable events by multiple parties who may not even be known from the outset and do not (always) trust each other.

Solution:

- Trusted third party (e.g. tinglysning) or
- Distributed ledger
Distributed human system

- Network of
  - people following instructions
  - connections for sending and receiving messages
- to collectively offer
  - coherent service to customers
Distributed computer system

- Network of
  - computers, each running a specific program
  - connections for sending and receiving data
- to collectively offer
  - coherent interface to client computers
Distributed system properties

- **Consistency**: Clients get the same response, independent of the node accessed.
- **Availability**: All clients get a response eventually (fast enough).
- **Partition tolerance**: All accessible nodes respond, even when internal communication is impossible (slow).
CAP Theorem

- Assume: All nodes reliable; connections unreliable

- Cannot simultaneously have
  - Consistency
  - Availability
  - Partition tolerance

- Inherent trade-off:
  - More consistency, less availability or less partition tolerance
  - More availability, less consistency or less partition tolerance
  - More partition tolerance, less consistency or less availability

Brewer, 2000; Gilbert, Lynch 2002
Byzantine consensus

- Assume: $n$ reliable and $m$ unreliable nodes, connections reliable
- Cannot guarantee
  - Consistency (consensus) amongst the reliable nodes
- ... unless $n > 2m$
- Inherent trade-off:
  - More liars, less/slower consistency

Lamport, Shostak, Pease, 1982

Basic approach: Nodes vote
Sybil attack

- Assume: Reliable and unreliable nodes (unknown number), connections reliable

- Cannot guarantee
  - consistency among reliable nodes

- if lying nodes can be manufactured efficiently
  - ballot stuffing
Sybil countermeasures

- Node authentication
- Proof of (real-world) identity
- Proof of stake
- Reputation system
- Peer review
- Impose costs on node
  - Proof of (useless) work
- Charge money for joining
Distributed ledger

Distributed ledgers are **systems** that enable parties who *don’t fully trust each other* to form and maintain **consensus** about the **existence, status and evolution** of a set of **shared facts**

Richard Brown, R3 CTO
Distributed ledgers — what for?

There is a huge numbers of situations where we need to maintain accurate, shared records with our counterparts. Indeed, a vast amount of the cost and inefficiency in today’s financial markets stems from the fact that it has been so difficult to achieve this.

Richard Brown, R3 CTO
DLT: applications

- Payments
- Letter of Credit, trade financing
- Factoring (invoice trading)
- Financial contract management (vanilla derivatives)
- Virtual currency transactions
DLT: application potential

- Inter-organization record keeping (automatic reconciliation)
- Identity management and verification (KYC, AML)
- Digital cash
- Distributed bank accounts
- Decentralized autonomous organizations
- License and ownership management
DLT: application potential

- Insurance claim settlement
- Stock trade settlement
- Payments
- Deposits, lending, capital raising
- Investment management
- Asset rehypothecation
DLT-based systems

- Bitcoin
- Ethereal
- Hyperledger
  - Corda
- ...
<table>
<thead>
<tr>
<th>Characteristics of high-potential use cases</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Shared repository</strong></td>
</tr>
<tr>
<td>A shared repository of information is used by multiple parties</td>
</tr>
<tr>
<td><strong>Multiple writers</strong></td>
</tr>
<tr>
<td>More than one entity generates transactions that require modifications to the shared repository</td>
</tr>
<tr>
<td><strong>Minimal trust</strong></td>
</tr>
<tr>
<td>A level of mistrust exists between entities that generate transactions</td>
</tr>
<tr>
<td><strong>Intermediaries</strong></td>
</tr>
<tr>
<td>One (or multiple) intermediary or a central gatekeeper is present to enforce trust</td>
</tr>
<tr>
<td><strong>Transaction dependencies</strong></td>
</tr>
<tr>
<td>Interaction or dependency between transactions is created by different entities</td>
</tr>
</tbody>
</table>
Where Are We Going?

Design parameters for DLTs

- Efficiency (energy, time, space)

- Privacy vs. transparency:
  - Should the full event log be readable by everybody?
  - Query interface?

- Expressiveness vs. predictability/safety:
  - Built-in contract (Bitcoin)
  - Declarative contract/protocol specification (POETS)
  - Smart contract/executable program (Ethereum)

- Economics:
  - Mechanism design (amongst contract parties, amongst node operators)

Bitcoin: >16 kWh per transaction (!)
Take-home message

• Blockchain = (distributed system for) valid, append-only event logs

• Desirable properties: Predictable, expressive; private, transparent; available, consistent, partition tolerant; anonymous, authenticated

• Not possible simultaneously

• Which properties are (most) important in your application?

• Use/build a corresponding blockchain or similar system

• Contracts: Key enabler for enforcing and analyzing agreements

RDBMS
Temporal database
Federated database system
Cloud database
Structured P2P system
Unstructured P2P system
Multi-agent system
Replicated state machine
Thank you!