A model of Bitcoin in the Theorem Prover Agda

Anton Setzer
Swansea University, Swansea UK
http://www.cs.swan.ac.uk/∼csetzer/

South Wales Cyber Security Seminars
Cardiff, Wales, UK

11 January 2019

Need for Models of Cryptocurrencies/Blockchain

- Cryptocurrencies very young subject.
- Seemingly convincing hand waving arguments.
- But very little known about its mathematical structure.
- Smart contracts = contracts governed entirely by algorithms
  - Lots of mistakes have occurred with huge financial implications.
  - In order to verify a good approach is to have a model against one can verify it.

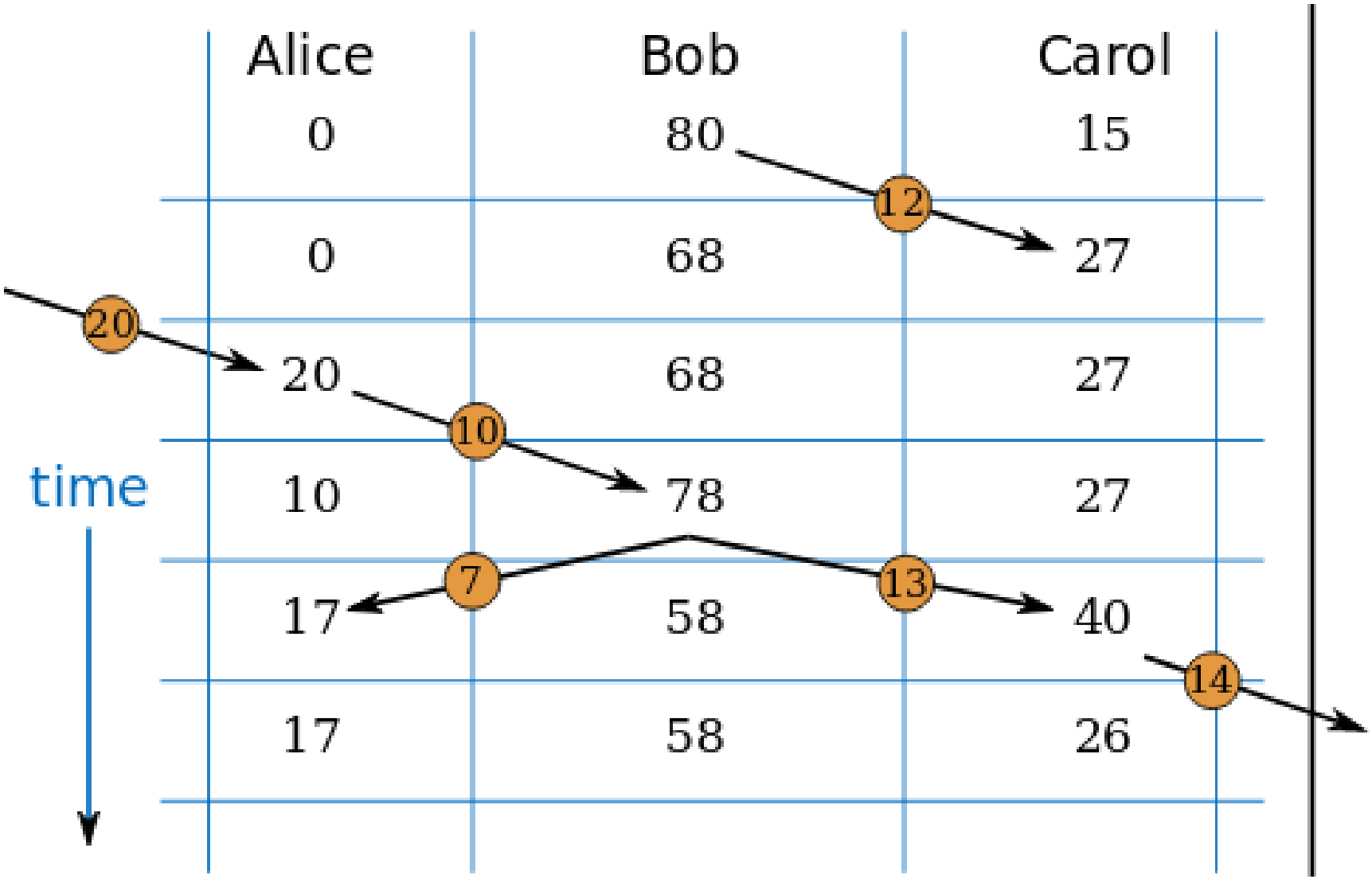
Components of the Blockchain

- Blockchain is a combination of 3 concepts:
  1. A distributed database with a consensus model.
  2. An immutable digital ledger.
  3. A representation of an immutable ledger as a linked list of blocks of transactions.
- 3. is a minor implementation detail.
- 1. and 2. are orthogonal.
- Here focus on 2., model includes 3.

Agda

- Agda interactive theorem prover and dependently typed functional programming language.
- Haskell like syntax.
- Potential of executing a blockchain in Agda, proving its properties and verifying (certifying) the correctness of smart contracts in Agda.
- Agda supports induction-recursion which is a key part for defining transactions dags.

Model of Bank (from Talk by Warner)



Source: [2]

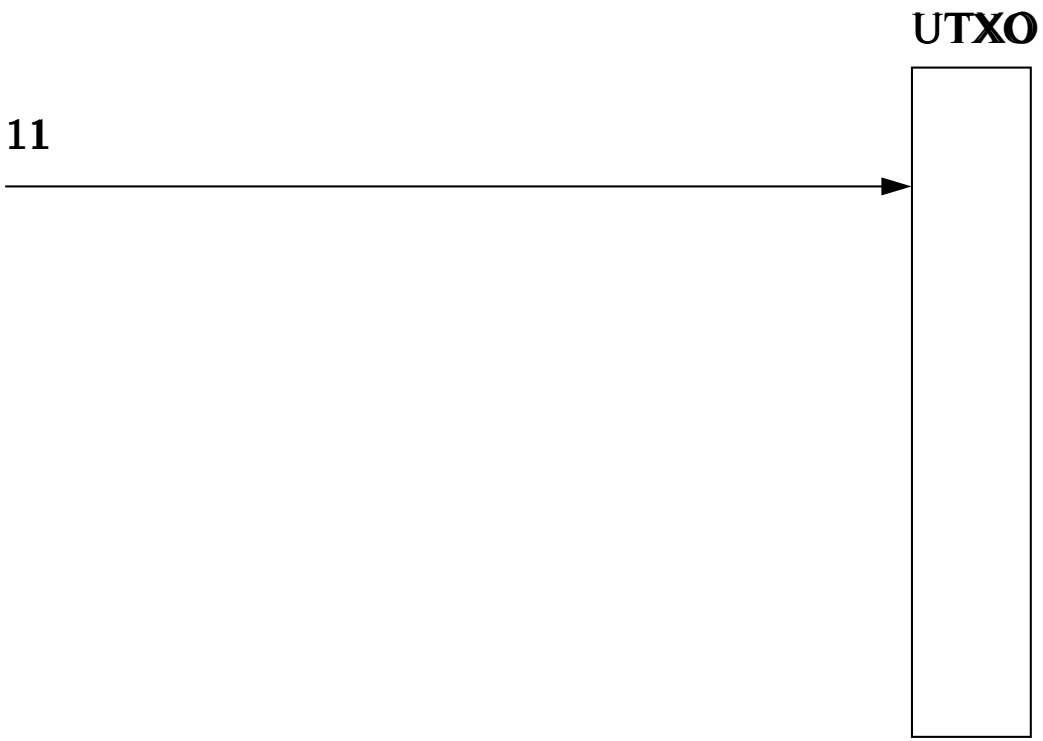
Ledger Model

- Ledger model based on this approach.
- Relatively easy to formalise.
- Ethereum based on the ledger model.
- What is a (Bit)coin in the ledger model?

Transaction Dags

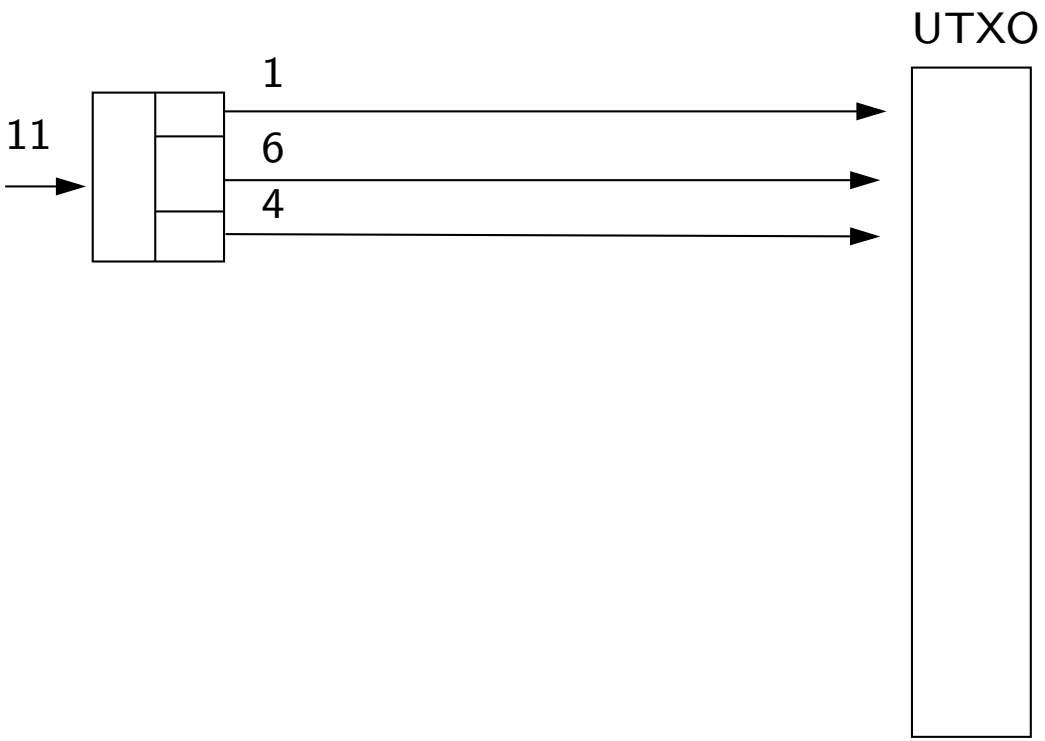
- In ledger model problem of replay attack (replay same transaction if enough money left).
- Solution in Bitcoin: transactions refer to previous unspent transaction outputs (UTXO) instead of an amount.
  - We call the corresponding model the **transaction model of Bitcoin**.
- Ethereum solves this problem easier by adding a sequence number to transactions.
- Advantage of transaction model.
  - Can be used for modelling tracing goods.
    - E.g. tracing food to the originator (e.g. organic food).
- **Dag** = **d**irected **a**cyclic **g**raph.
  Like a tree but branches can merge into one.

Transaction Dag Step 0 - Coinbase



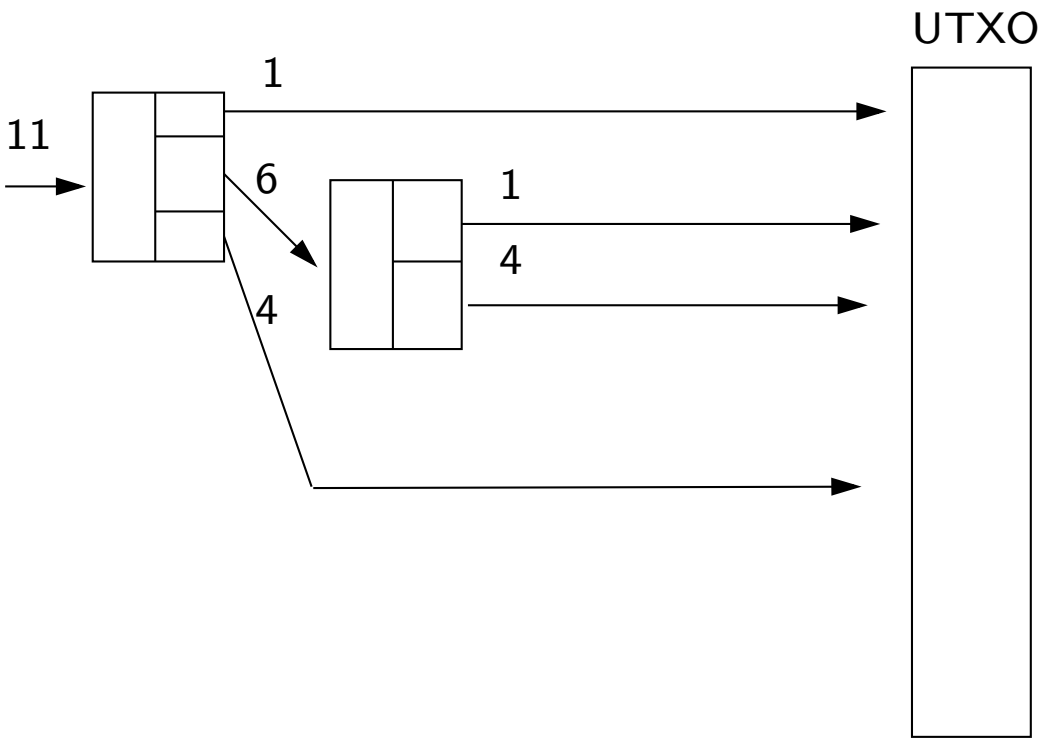
UTXO = unspent transaction outputs.

Transaction Dag Step 1



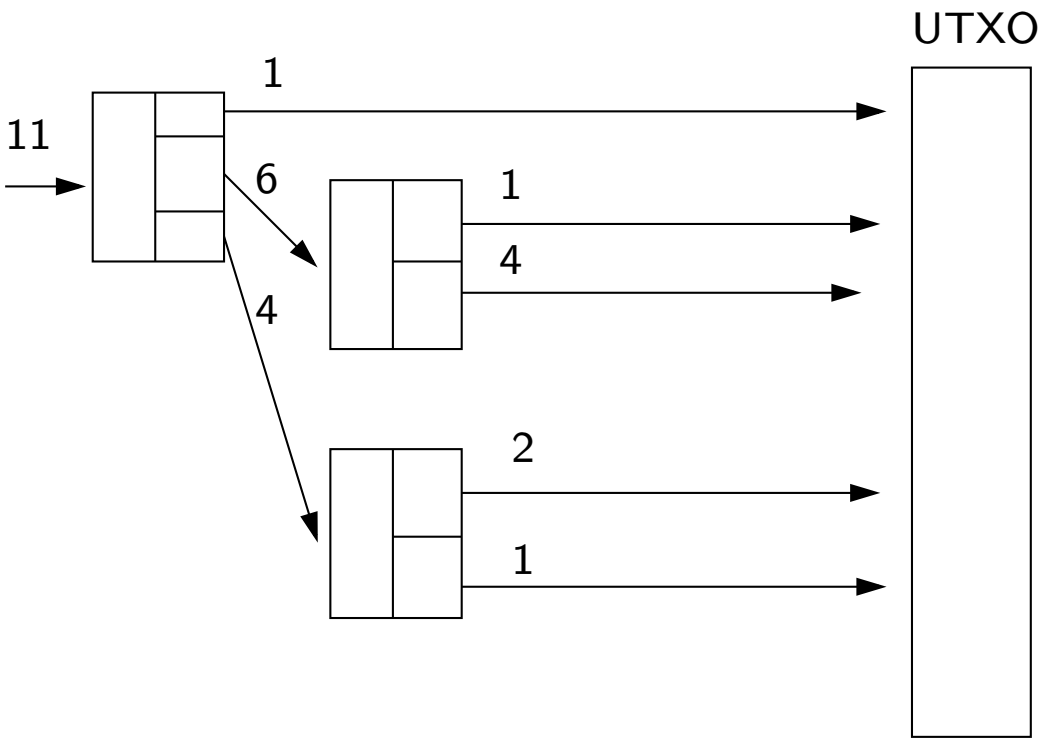
UTXO = unspent transaction outputs.

Transaction Dag Step 2



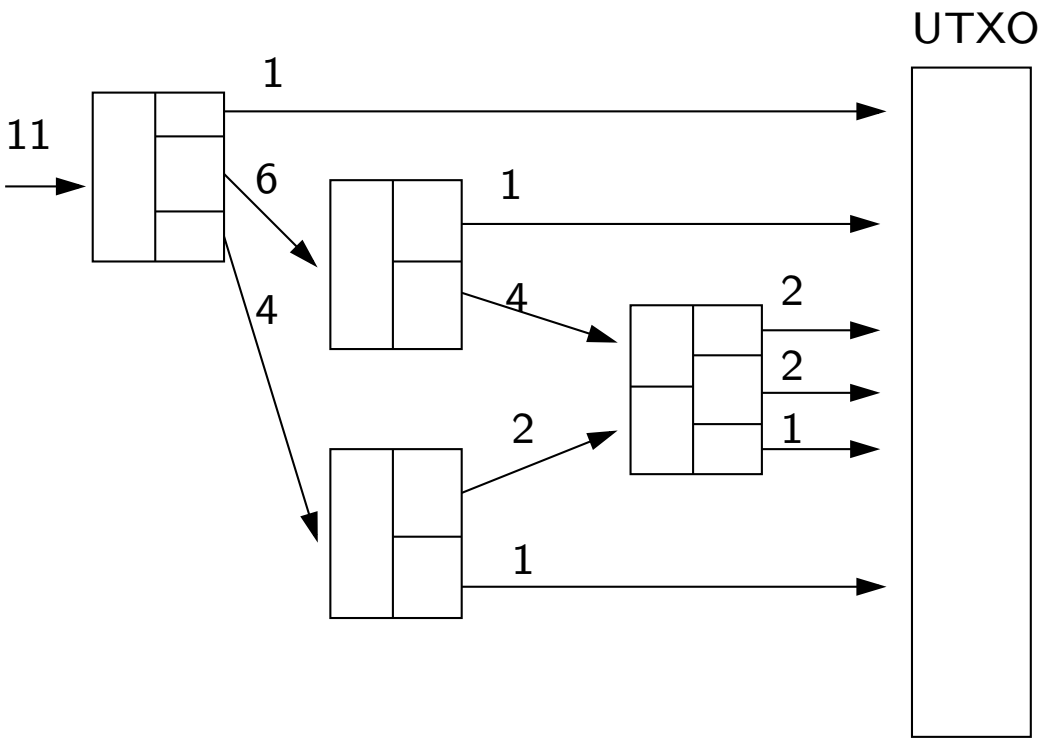
UTXO = unspent transaction outputs.

Transaction Dag Step 3



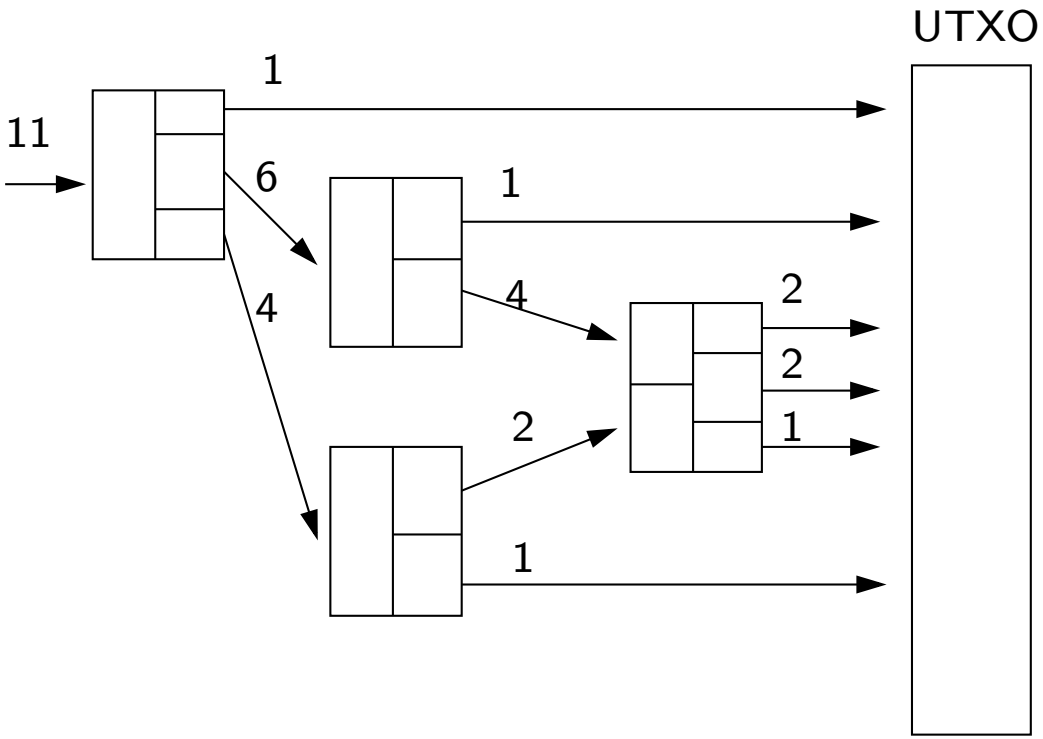
UTXO = unspent transaction outputs.

Transaction Dag Step 4



UTXO = unspent transaction outputs.

Transaction Dag



Need to define the transaction dag, transactions, and unspent transactions (UTXO) simultaneously.

Inductive-Recursive Definition

- We define
  - **inductively** transaction dag and transactions
  - while **recursively** defining the set of unspent transactions

Inductive-Recursive Definition – TXDag, TX

```
data TXDag : Set where
  genesisDag : TXDag
  txdag      : (dag : TXDag)(tx : TX dag) → TXDag

data TX (tr : TXDag) : Set where
  normalTX : (inputs : TxInputs tr) (outputs : List TXOutputfield)
             → TX tr
  coinbase : (time : Time)          (outputs : List TXOutputfield)
             → TX tr
```

Inductive-Recursive Definition – TXOutput

```
record TXOutput : Set where
  inductive
  constructor txOutput
   trDag : TXDag
   tx    : TX trDag
   output : Fin (nrOutputs trDag tx)
```

Inductive-Recursive Definition – utxo

```
utxoMinusNewInputs : (tr : TXDag)(tx : TX tr) → List TXOutput
utxo :               (tr : TXDag) → List TXOutput

utxoMinusNewInputs tr (normalTX inputs outputs)
  = listMinusSubList+ (utxo tr) inputs
utxoMinusNewInputs tr (coinbase time outputs)
  = utxo tr

utxo genesisDag    = []
utxo (txdag tr tx) = utxoMinusNewInputs tr tx ++ tx2TXOutputs tr tx
```

Correctness

- Probably easy:
  amount of bitcoins = sum of the block rewards up to now
- More challenging properties:
  Assume **injectivity of hashing** (which is only true up to high probability).
  Prove
  - **Signatures of transactions** are **unique**.
  - If a transaction has an output, the **output is not used in transactions** up to now, then **it can be spent**.
  - If a **transaction is spent**, it **cannot be spent again**.
  - Any **input of a transaction** was the **unspent output of a transaction**.

Correctness – Main Lemma

- All the above depends on the main property:
  - Show that the **transaction ids are unique**.
- Uniqueness **didn't hold originally** because of coinbase transactions being non-unique.
  Solved by adding block number to coinbase transactions.

What is a Bitcoin?

- Bitcoins are
  - the unspent transaction outputs in the current blockchain,
  - which can be computed from the blockchain,
  - where the blockchain is stored in a peer-to-peer network
  - with consensus obtained by the mining protocol.

Conclusion

- Introduction to the Bitcoin Protocol.
- Development of two models of the Bitcoin protocol in Agda.
  - First model based on ledger allows still replay attacks.
  - Second model based on transaction trees.
- Use of induction-recursion.

Bibliography


A. Setzer.
Modelling Bitcoin in Agda.
Arxiv, arXiv:1804.06398:27, 17 April 2018.


B. Warner.
Bitcoin: A technical introdution.
Available from
http://www.lothar.com/presentations/bitcoin-brownbag/, July 2011.