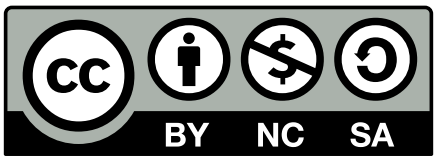
# Digital Currencies: Algorithms and Protocols

Élise Alfieri
<elise.alfieri@gmail.com>,
Emmanuel Fleury
<emmanuel.fleury@u-bordeaux.fr>

LaBRI, Université de Bordeaux, France

March 7, 2017



# Plan

# Plan

# Types of Digital Currency

université de BORDEAUX

## Electronic Currency

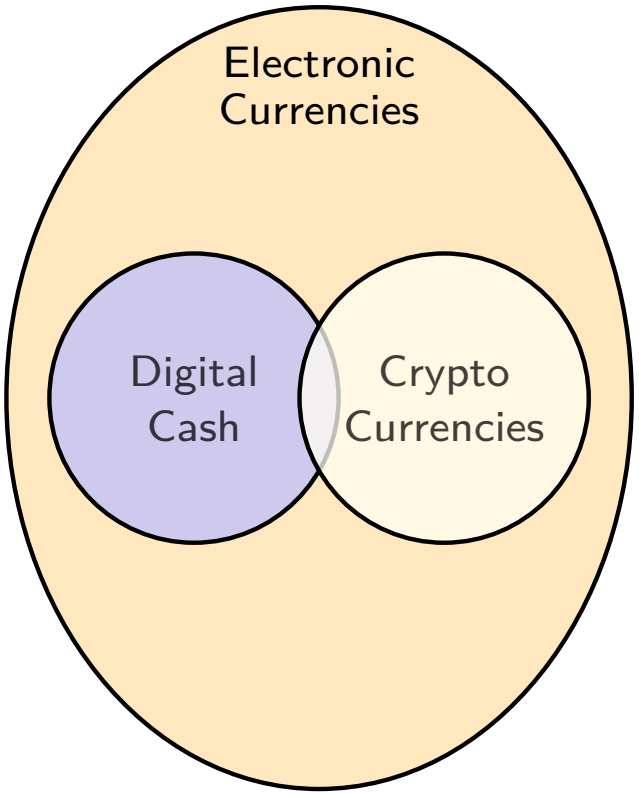
Each user has its money balance recorded electronically on a computer system (central server) or an electronic device (smartcard).

## Digital Cash

A digital version of physical money coins or bank notes. Usually implies anonymity of the transactions.

## Cryptographic Currency

An electronic currency relying on crypto-systems to ensure security properties of the transactions.




## Notes

- Every cryptographic currency is an electronic currency but not the converse.
- These definitions do not follow the ones usually admitted.

# Digital Currency Properties

université de BORDEAUX

- **Secured Ownership**:
  Access to the coins must be securely controlled and unauthorized access banned.
- **Unfalsifiability**:
  Creation of new digital coins must be under strict control of the system.
- **No Double Spending**:
  Once digital coins have been spent for a transaction, they must not be reusable by the original owner in another transaction.

- **Centralized/Distributed** (***Alternative***):
  Transactions using digital currency is done in a centralized/distributed manner.

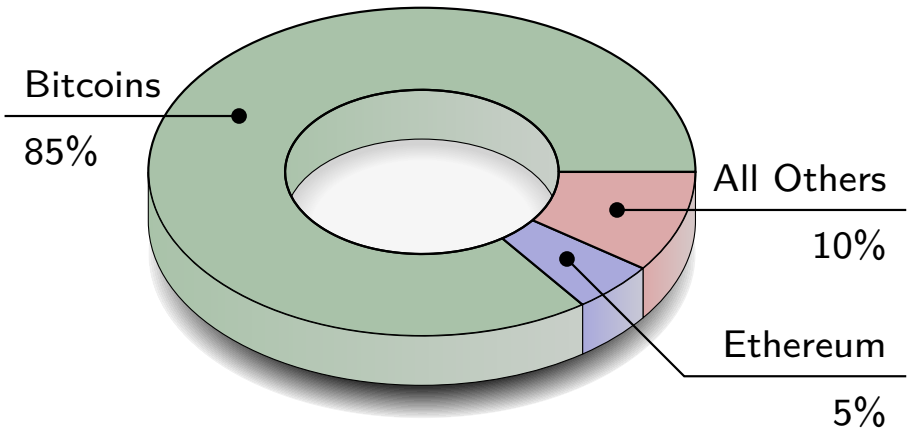
- **Payer Anonymity** (***Optional***):
  A transaction is anonymous during payment.
- **Fraud Tracing** (***Optional***):
  A fraudulent transaction may lead to the unveiling of the original fraudster.

# Crypto-currencies Market

## All Crypto-currencies

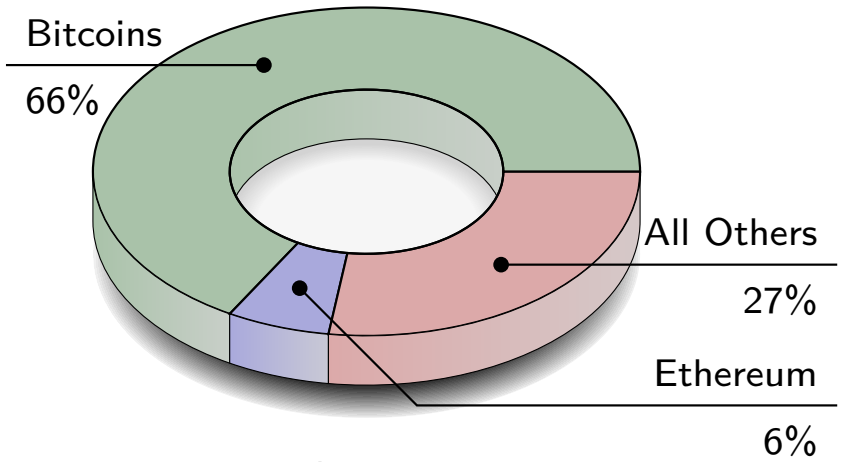
- About 650 crypto-currencies (with some financial activity).
- Total capitalization ≈$17.5 billions.
- Average daily transactions ≈$145 millions.

## Market Capitalization




Total ≈$17.5 billions

## Daily Transactions




Total ≈$145 millions

Data from `http://coinmarketcap.com` on January 31, 2017.

# Plan

# Academic Timeline

université de BORDEAUX

**1982** · · · '*Blind Signatures for Untraceable Payments*' by David Chaum.

**1996** · · · '*How to Make a Mint: The Cryptography of Anonymous Electronic Cash*' by Laurie Law, Susan Sabett and Jerry Solinas.

**1998** · · · '*B-Money*' by Wei Dai (first attempt of distributed crypto-money).

**2002** · · · '*HashCash: A Denial of Service Counter-Measure*' by Adam Back.

**2004** · · · '*RPOW: Reusable Proofs of Work*' by Hal Finney.

**2005** · · · '*Bit Gold*' by Nick Szabo.

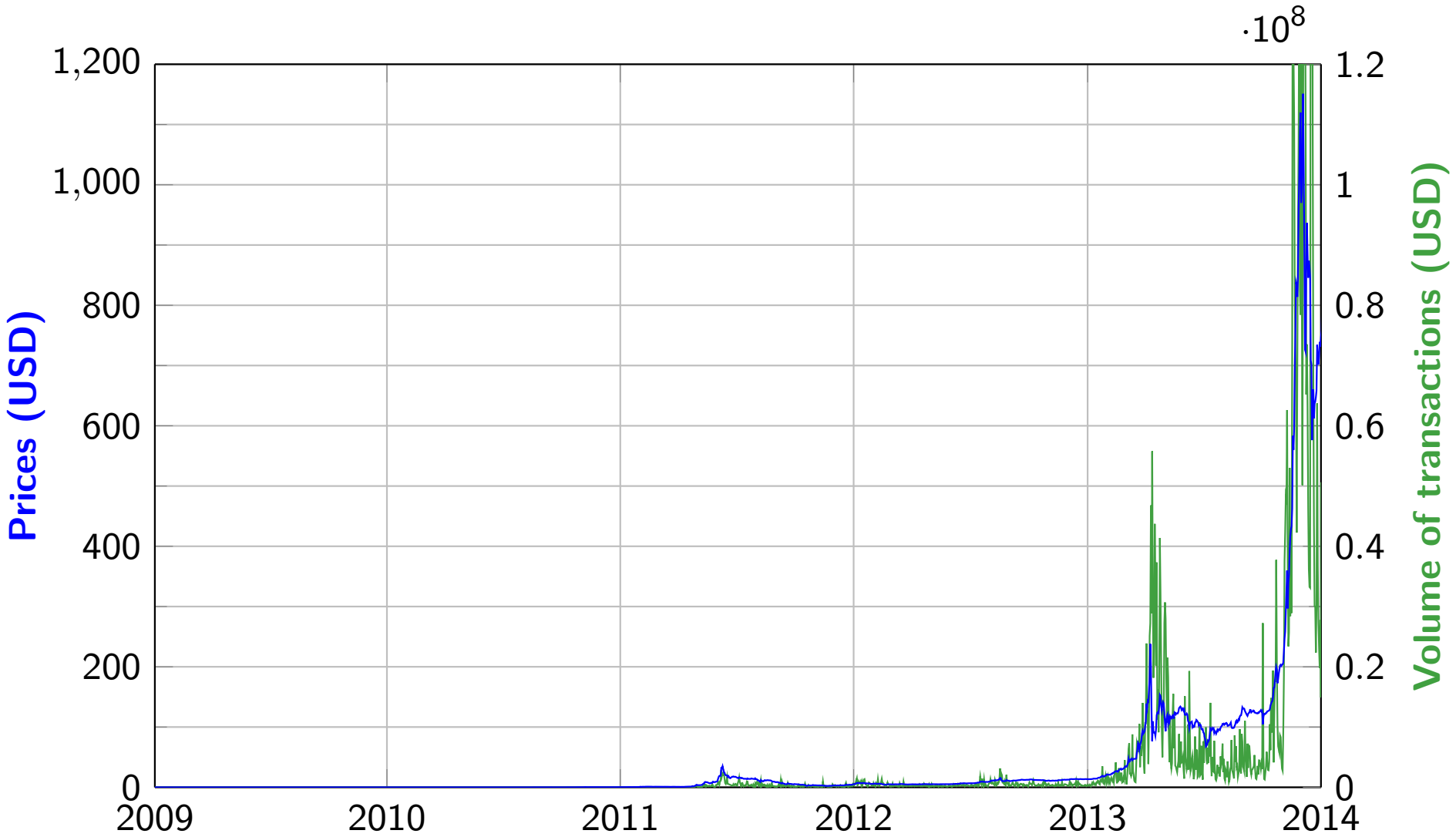
**2008** · · · '*Bitcoin: A Peer-to-Peer Electronic Cash System*' by Satoshi Nakamoto.

**2009** · · · '*Bitcoin software toolkit 1.0*' by Satoshi Nakamoto and Hal Finley.

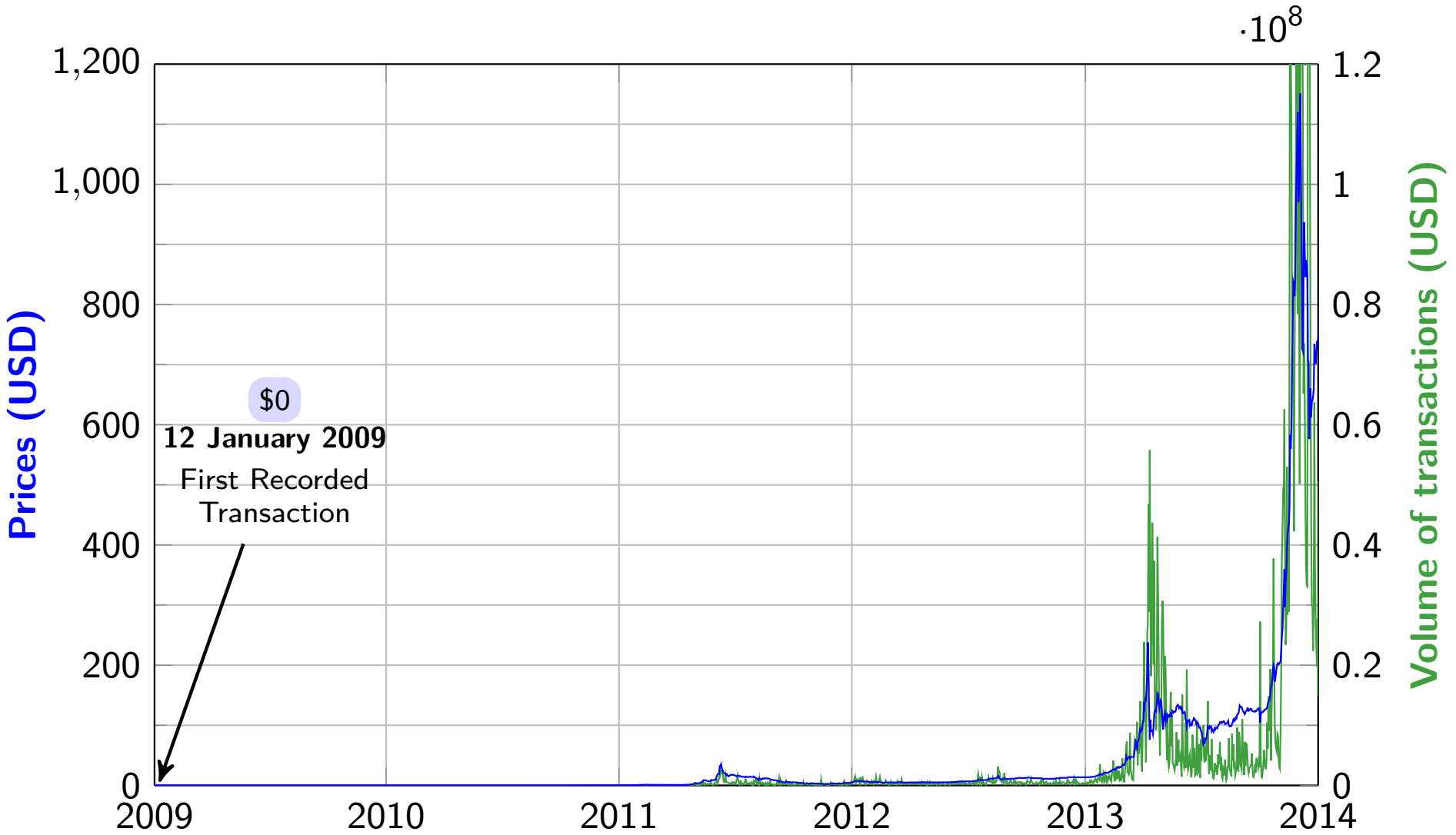
**2011** · · · '*Proof of stake instead of proof of work*' by Quantum Mechanic.

**2014** · · · '*Ethereum: A Secure Decentralized Generalised Transaction Ledger*' (Homestead Draf) by Gavin Wood.

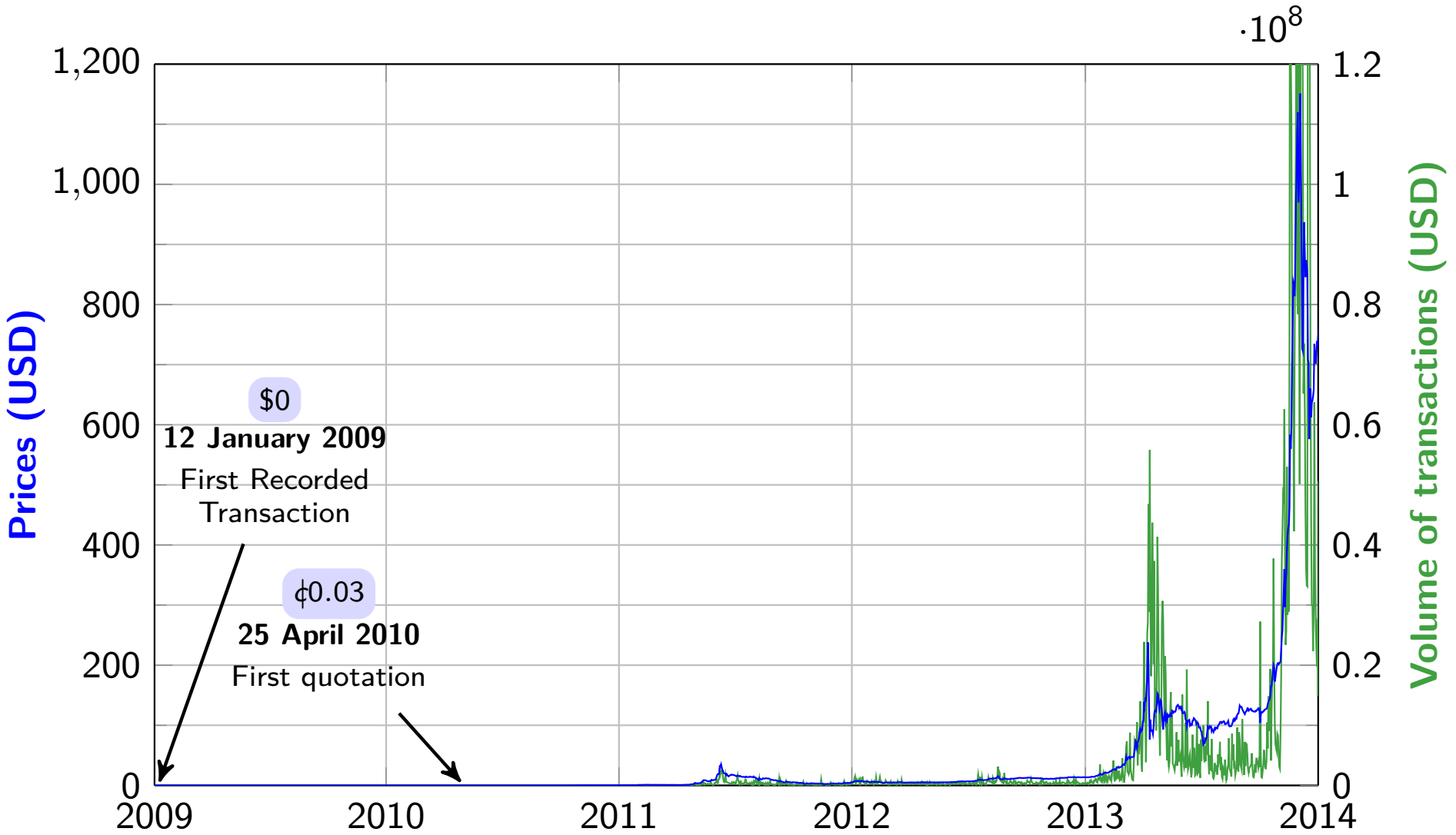
# Bitcoin Market Timeline (2009-2013)



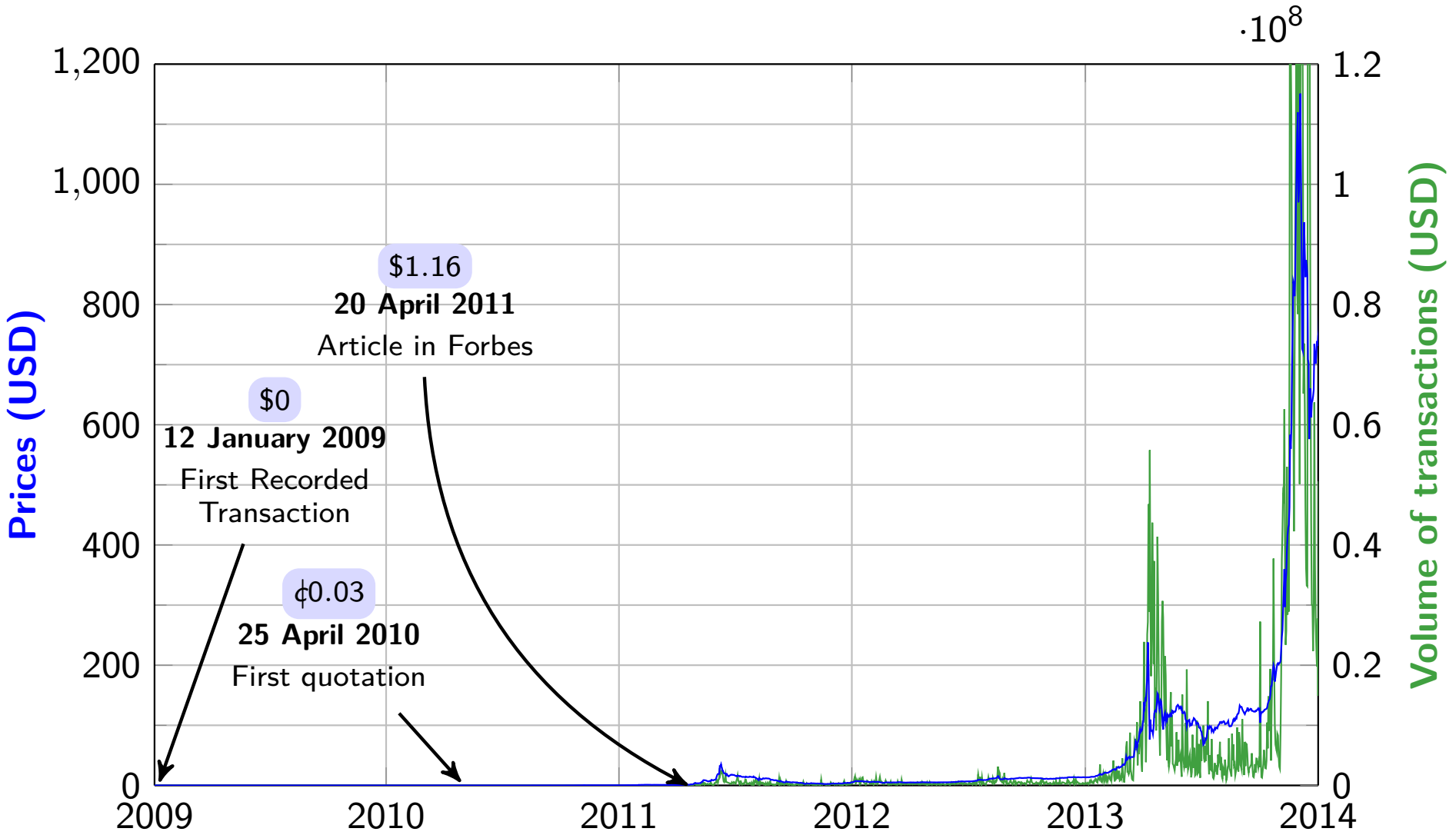
# Bitcoin Market Timeline (2009-2013)



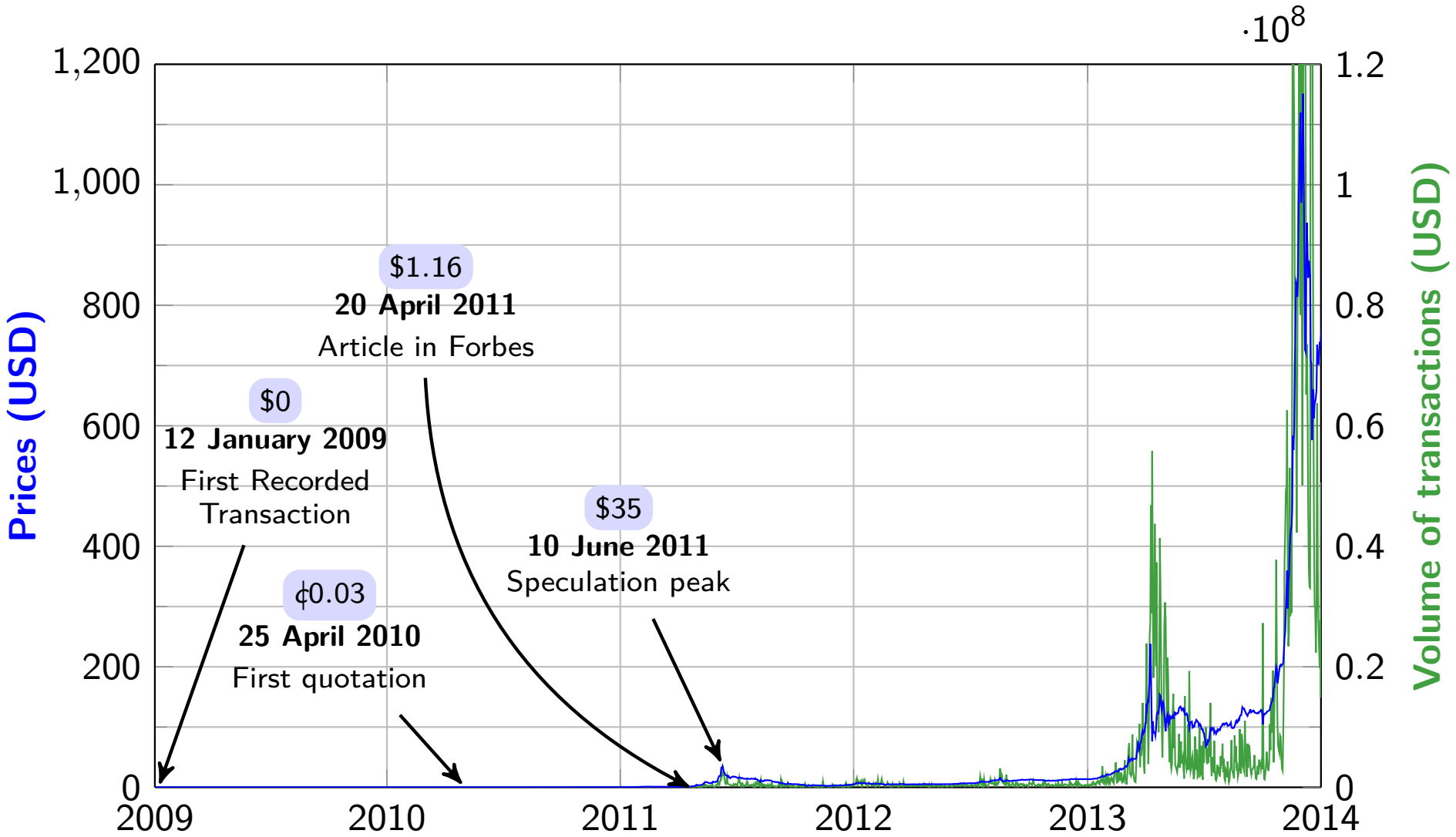
# Bitcoin Market Timeline (2009-2013)



# Bitcoin Market Timeline (2009-2013)

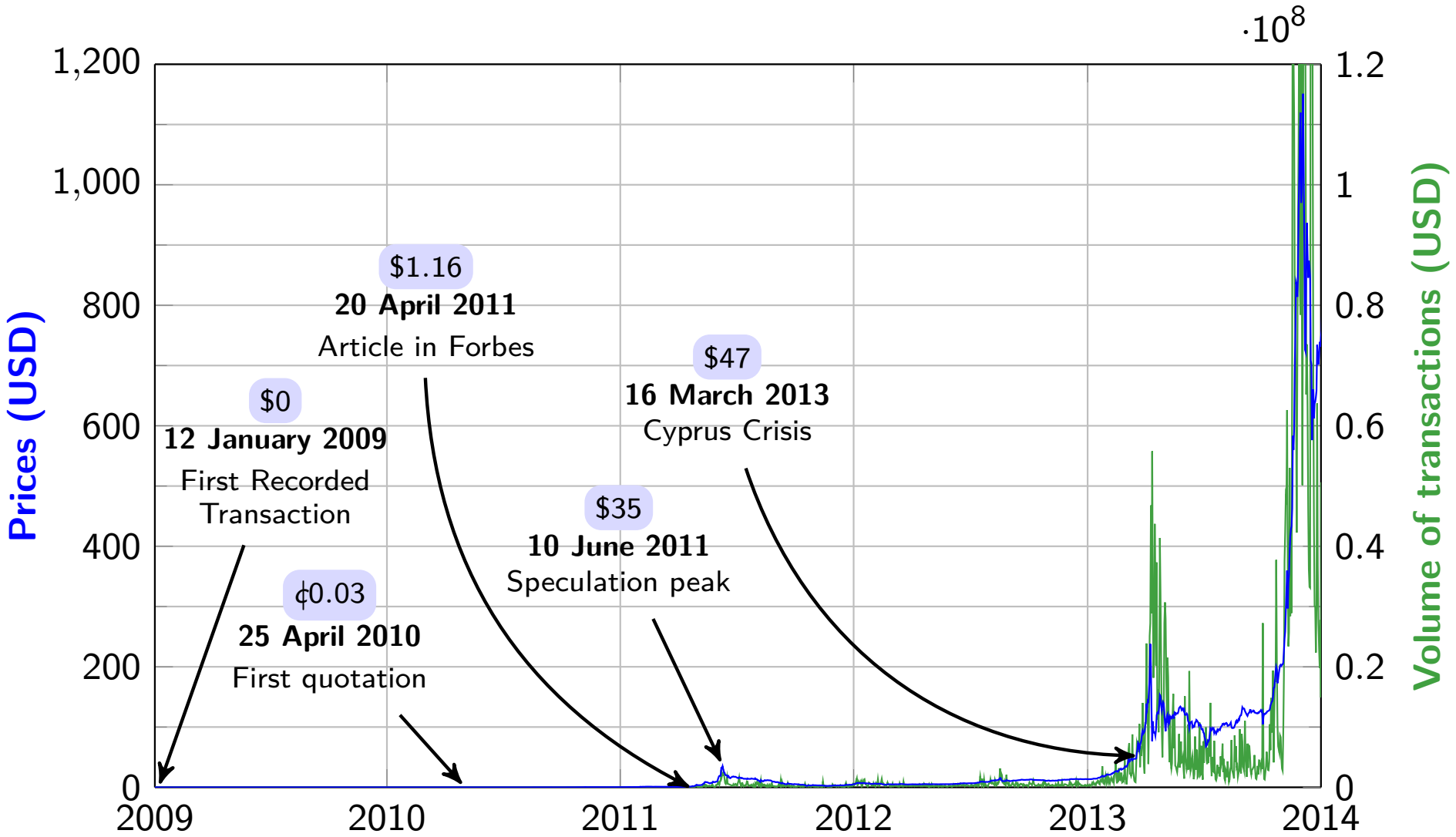


# Bitcoin Market Timeline (2009-2013)



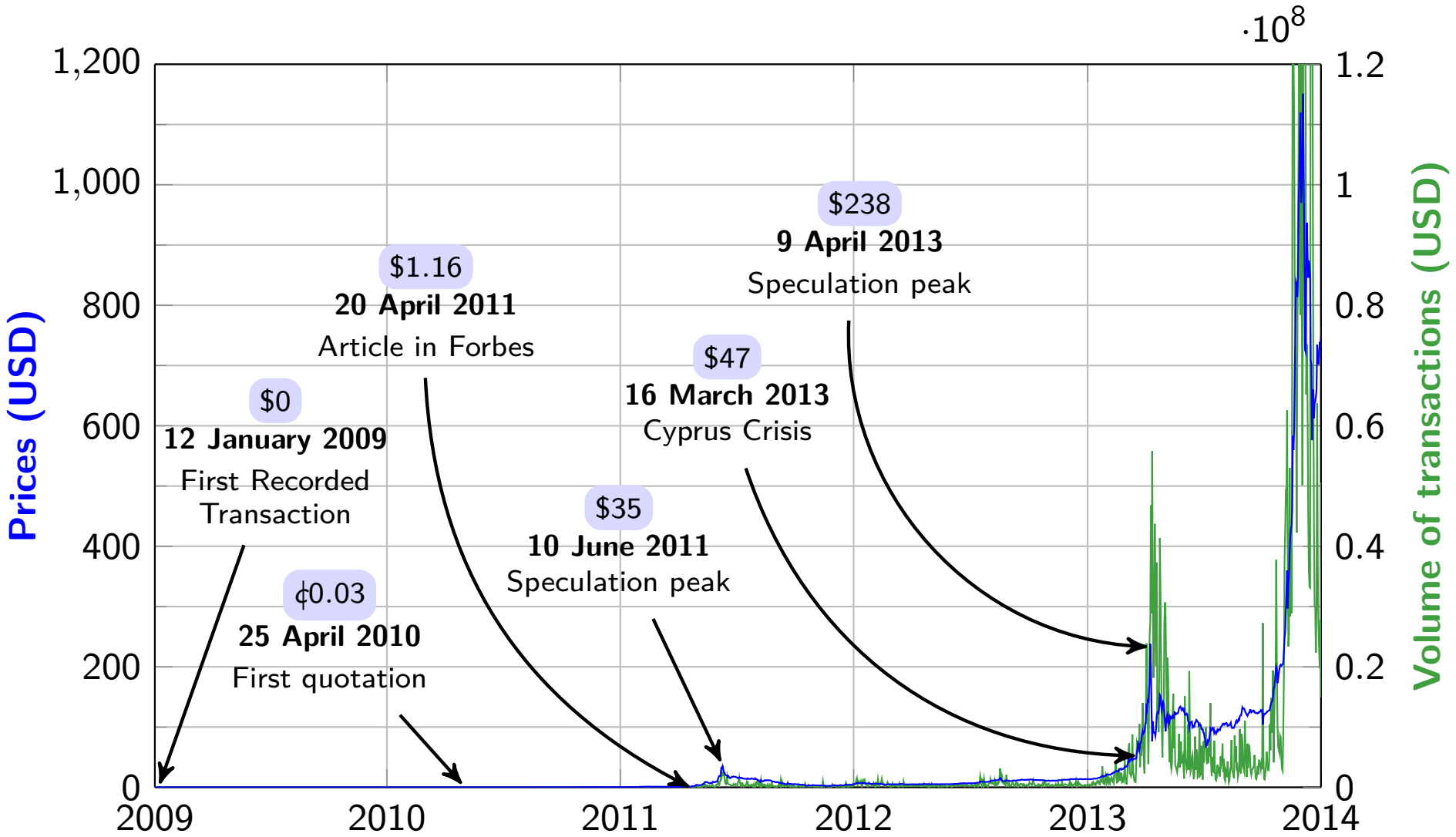
# Bitcoin Market Timeline (2009-2013)

université de BORDEAUX



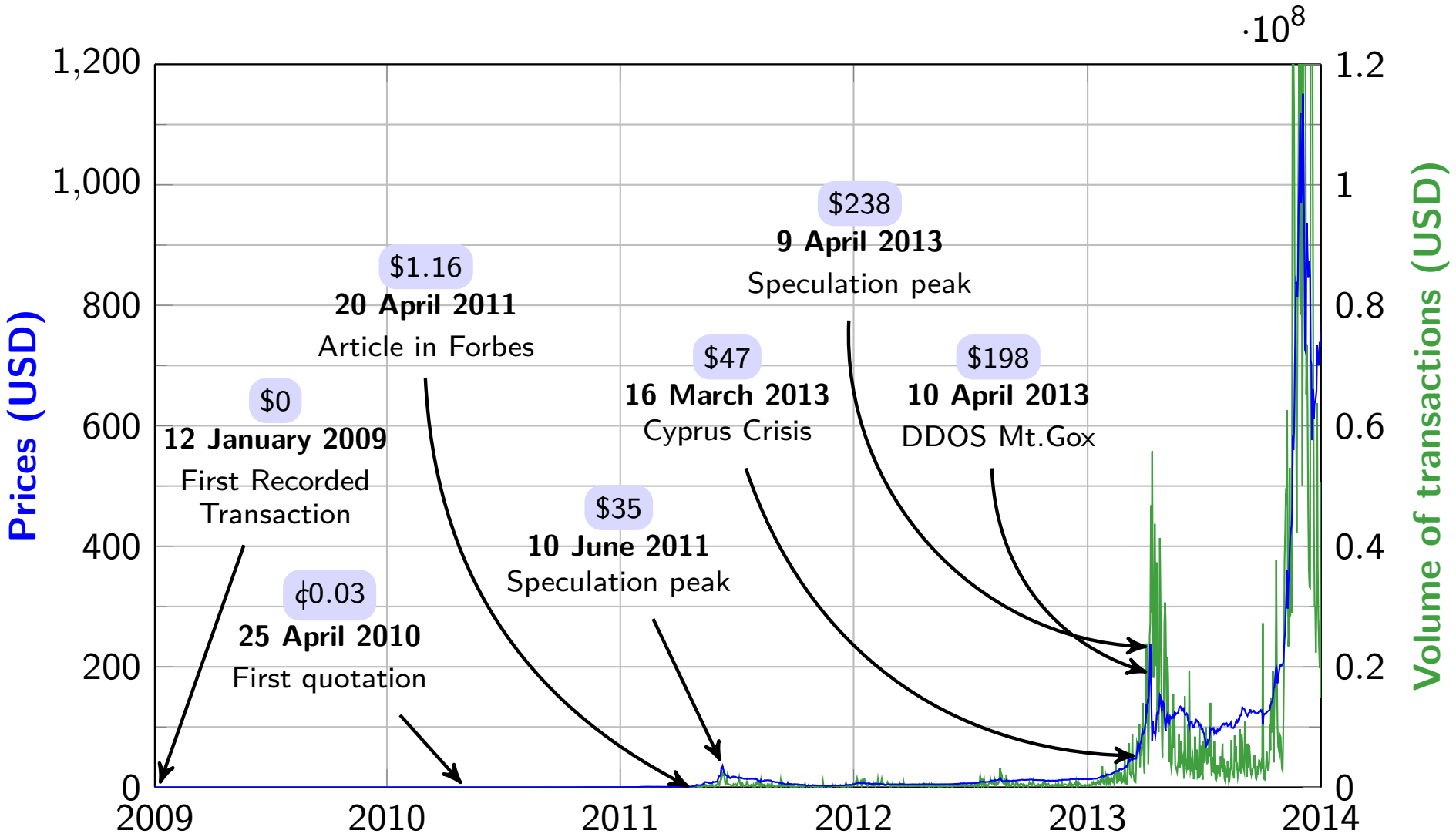
# Bitcoin Market Timeline (2009-2013)

université de BORDEAUX

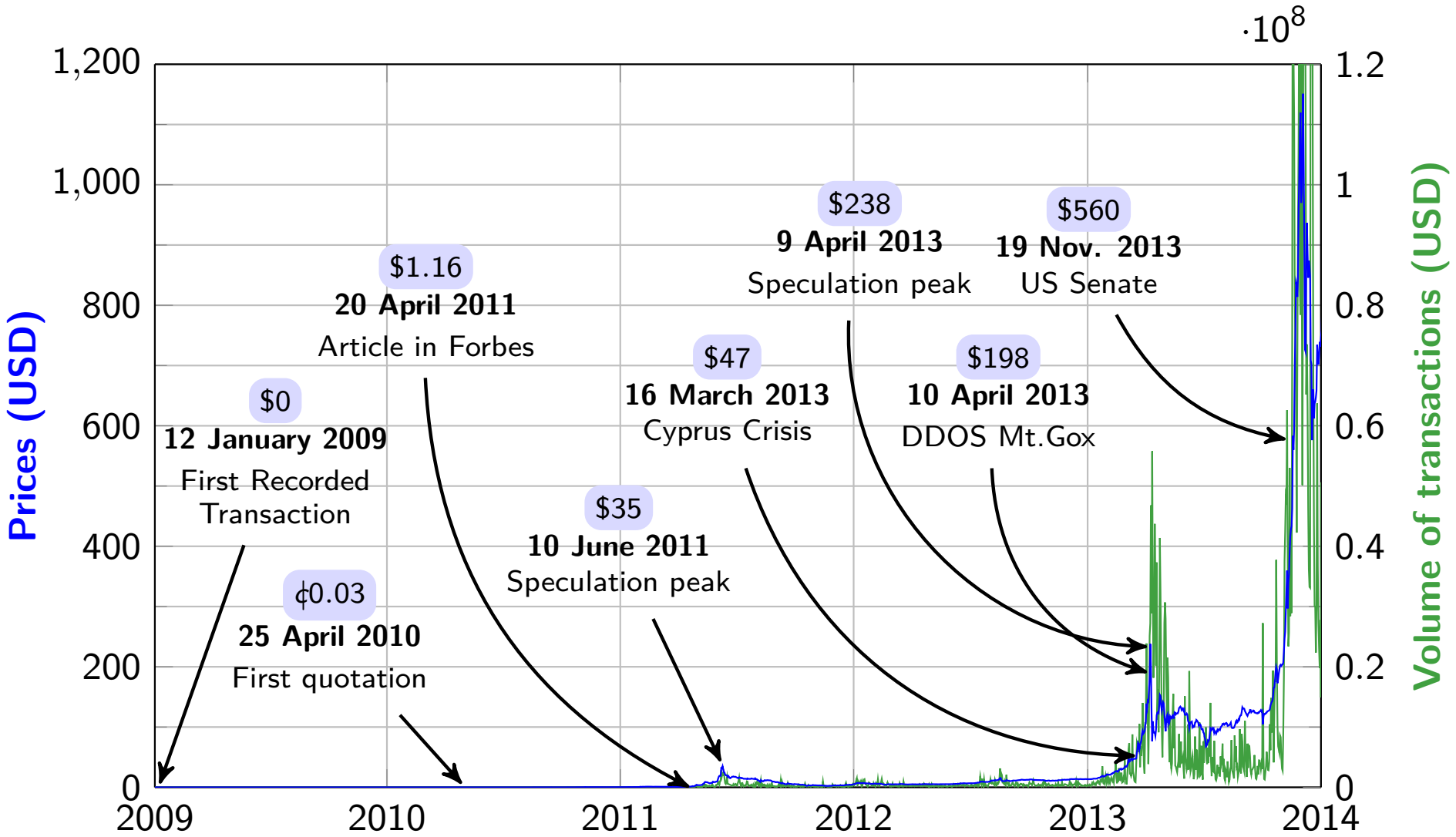


# Bitcoin Market Timeline (2009-2013)

université de BORDEAUX

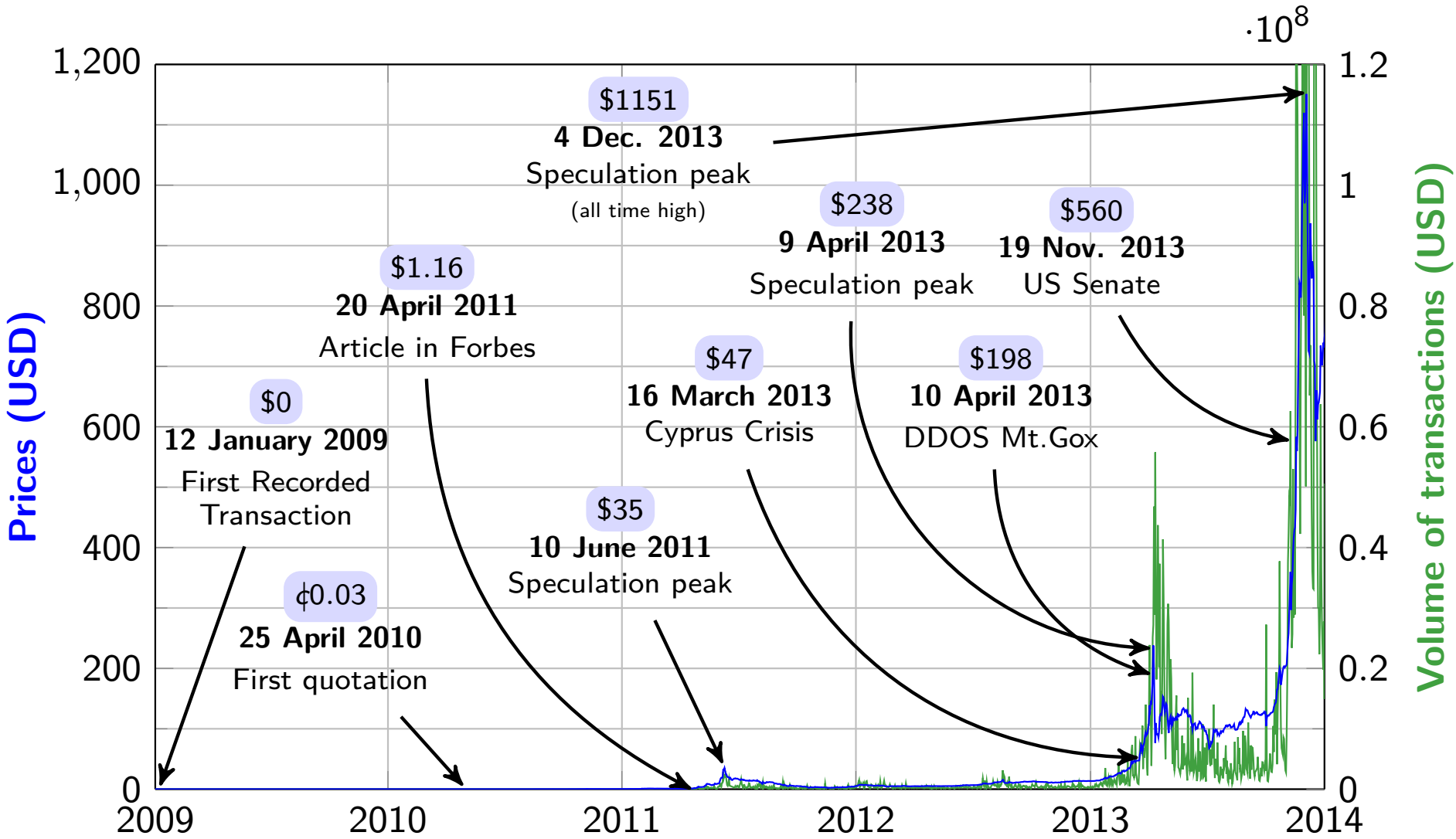


# Bitcoin Market Timeline (2009-2013)

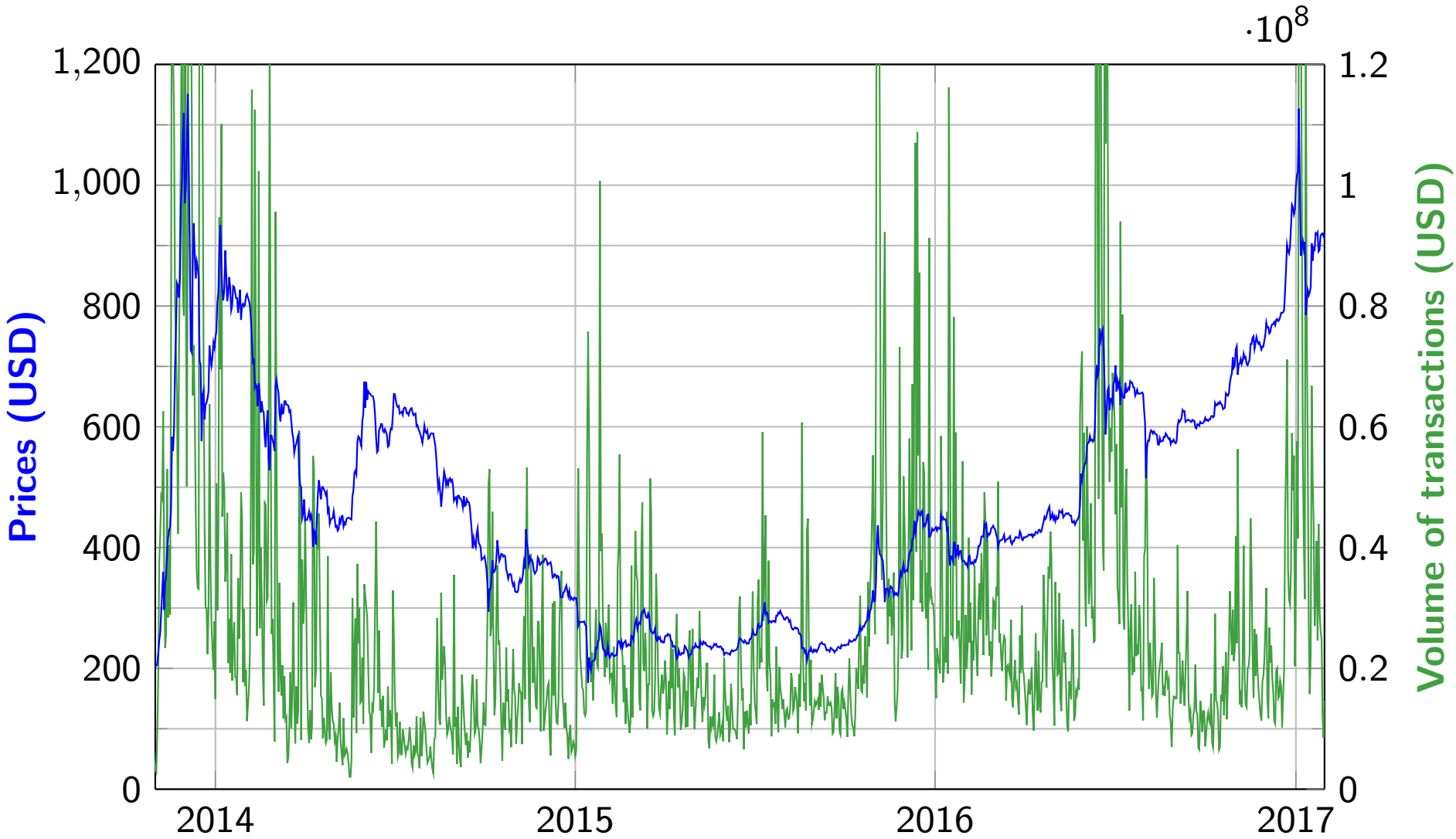


# Bitcoin Market Timeline (2009-2013)

université de BORDEAUX

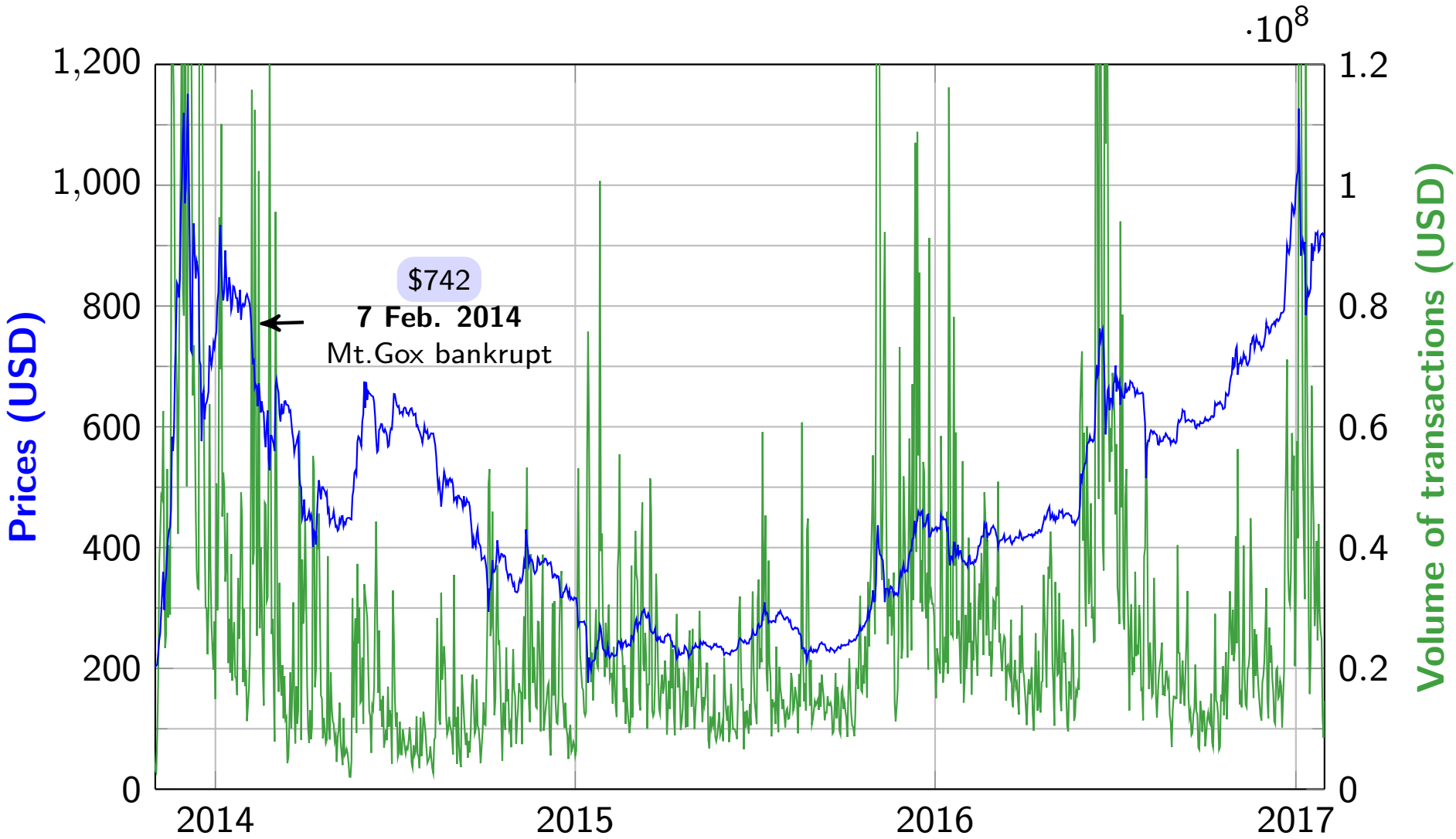


# Bitcoin Market Timeline (Since 2013)



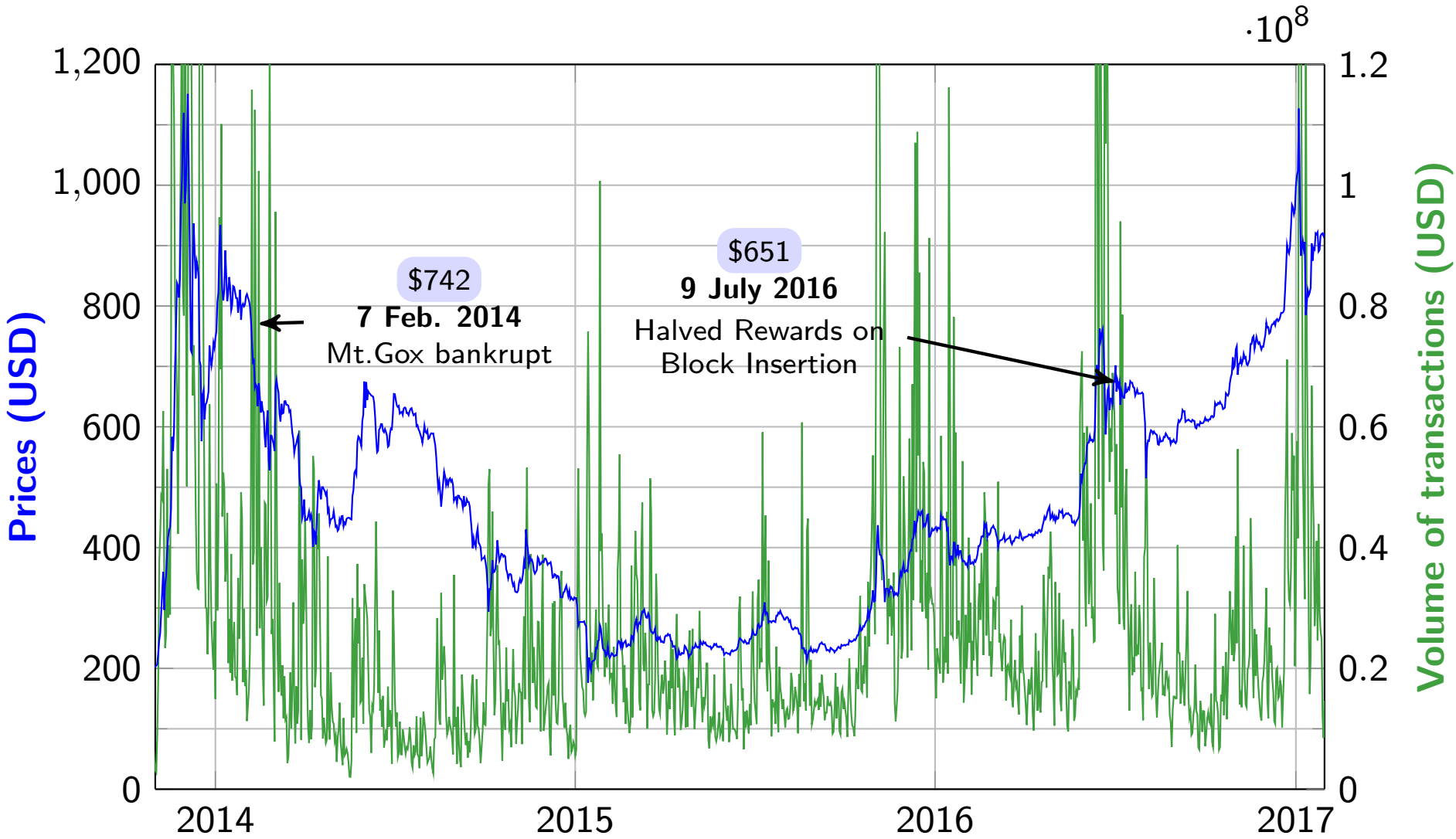
# Bitcoin Market Timeline (Since 2013)

université de BORDEAUX



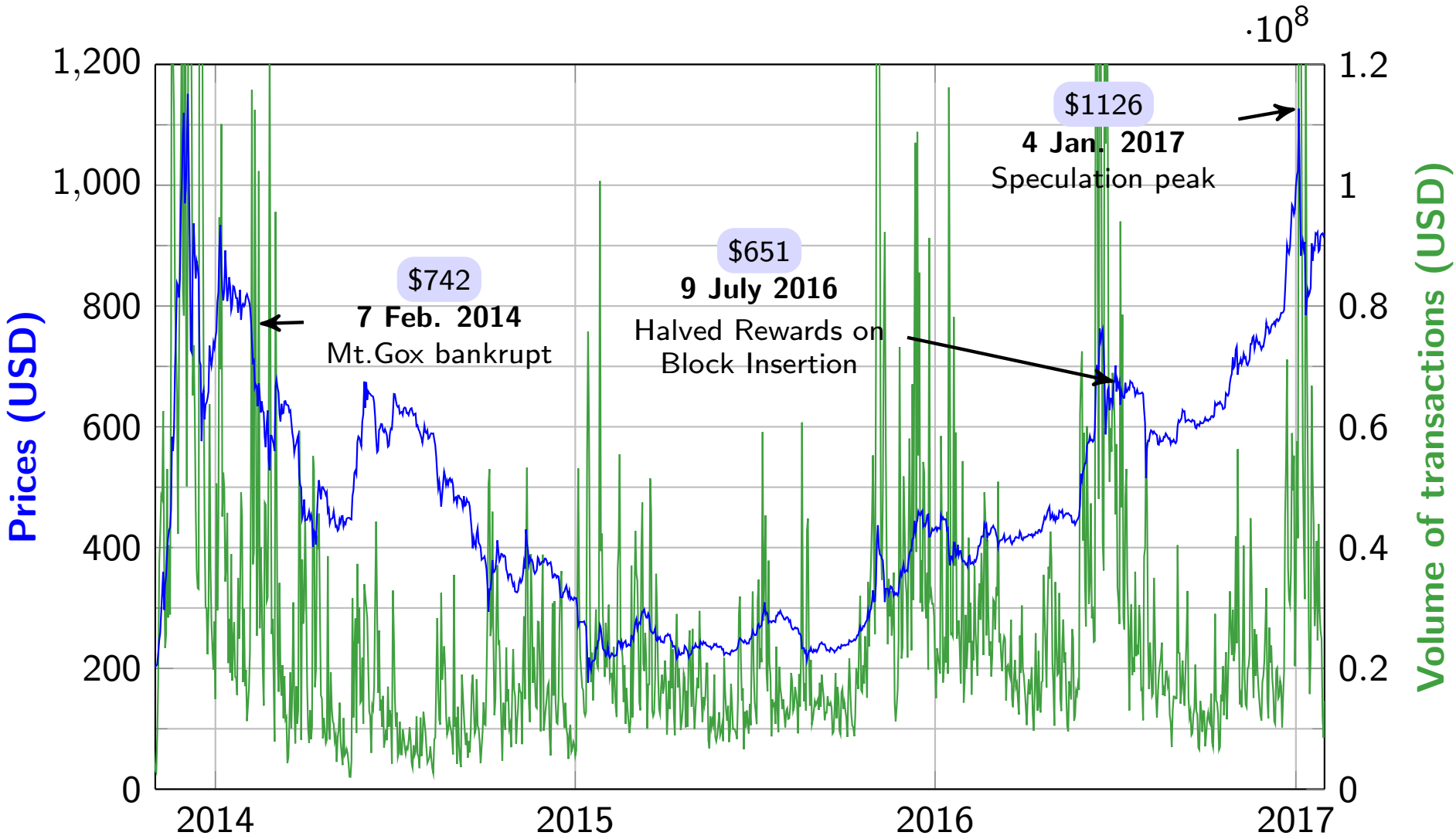
# Bitcoin Market Timeline (Since 2013)

université de BORDEAUX



# Bitcoin Market Timeline (Since 2013)

université de BORDEAUX



# Governmental and Institutional Timeline

université de BORDEAUX

**Mar 2013** — US Department of the Treasury states that Bitcoins is a "*commodity*" and can be used freely.

**Aug 2013** — German government recognizes Bitcoin as a form of private money.

**Dec 2013** — Croatian National Bank states that Bitcoin is not illegal in Croatia.

**Dec 2013** — People's Bank of China allows private usage of Bitcoin.
But, firms and banks cannot use it.

**Dec 2013** — German's Federal Financial Supervisory Authority states that Bitcoins are legally binding financial instruments that fall into the category of units of account.

**Dec 2013** — The Reserve Bank of Australia says that they will not take any legal action against their citizens if they use Bitcoins.

**Jul 2014** — French Senate issues a report on "*Les enjeux liés au développement du Bitcoin et des autres monnaies virtuelles*" which states that "*we must support innovation and, at the same time, keep an eye on it to avoid taking the wrong way*". Bitcoin is considered as a commodity (*bien meuble*) and can be used as such.

# Is Bitcoin a Real Currency?

université de BORDEAUX

## Currency (usual definition)

- **A medium of exchange**: An intermediary used in trade to obtain good and service;
- **A store of value**: Can be saved, retrieved and exchanged at a later time, and be predictably useful when retrieved;
- **A unit of account**: A nominal monetary unit of measure used to value/cost goods, services, assets, liabilities, income, expenses.

## Bitcoin is:

- A medium of exchange:
- A store value:
- A unit of account:

# Is Bitcoin a Real Currency?

université de BORDEAUX

## Currency (usual definition)

- **A medium of exchange**: An intermediary used in trade to obtain good and service;
- **A store of value**: Can be saved, retrieved and exchanged at a later time, and be predictably useful when retrieved;
- **A unit of account**: A nominal monetary unit of measure used to value/cost goods, services, assets, liabilities, income, expenses.

## Bitcoin is:

- A medium of exchange: **Yes!**
- A store value:
- A unit of account:

# Is Bitcoin a Real Currency?

université de BORDEAUX

## Currency (usual definition)

- **A medium of exchange**: An intermediary used in trade to obtain good and service;
- **A store of value**: Can be saved, retrieved and exchanged at a later time, and be predictably useful when retrieved;
- **A unit of account**: A nominal monetary unit of measure used to value/cost goods, services, assets, liabilities, income, expenses.

## Bitcoin is:

- A medium of exchange: **Yes!**
- A store value: **Controversial!**
- A unit of account:

# Is Bitcoin a Real Currency?

université de BORDEAUX

## Currency (usual definition)

- **A medium of exchange**: An intermediary used in trade to obtain good and service;
- **A store of value**: Can be saved, retrieved and exchanged at a later time, and be predictably useful when retrieved;
- **A unit of account**: A nominal monetary unit of measure used to value/cost goods, services, assets, liabilities, income, expenses.

## Bitcoin is:

- A medium of exchange: **Yes!**
- A store value: **Controversial!**
- A unit of account: **Controversial!**

# Plan

# A Brave New World

université de BORDEAUX

## Lets, first, imagine a perfect World!
## No failure, no cheating!

- **Alice**, **Bob**, **Carol** and **Oscar** want to create their own electronic currency for their private usage.
- They use their own **Peer-to-Peer (P2P) network** to exchange information in a safe and quick way.
- The protocol handling the electronic currency must be **distributed** and may use **cryptography**.

**Peer-to-Peer (P2P) Network**

A gathering of computer systems exchanging through a network to share their resources, such as processing power, disk storage or network bandwidth, . . . Peers are equally privileged in the protocol, there is no need for central authority among them.

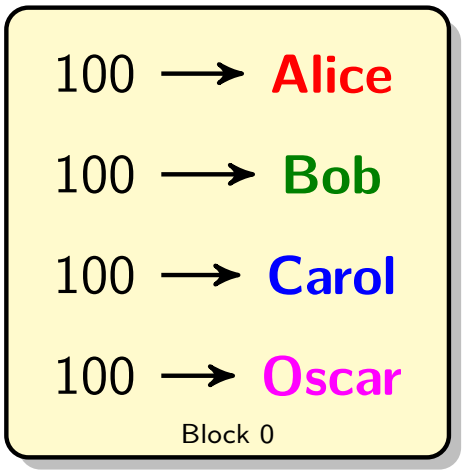
# Simplified Protocol

université de BORDEAUX

1. **Transactions are simple.**
   A transaction is just moving units from one account to another one.
2. **A ledger is distributed to all participants.**
   It contains all the transactions from start and each user has a copy of it.
3. **There will be no coin!**
   The balance of an account is the sum of all its transactions in the ledger.
4. **Every transaction is broadcasted to the others.**
   Each participant can transfert units from their own account to somebody else account by publishing the transaction on the P2P network.
5. **Adding a transaction to the ledger make it valid.**
   From time to time, one of the users collects a set of pending transactions and creates a new page in the ledger (a page in the ledger is called a **block**). Once added, it validates the transactions added to the ledger.

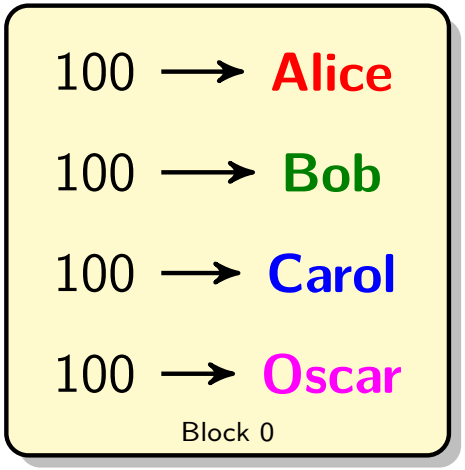
université de BORDEAUX

## 1. Ledger Initialization

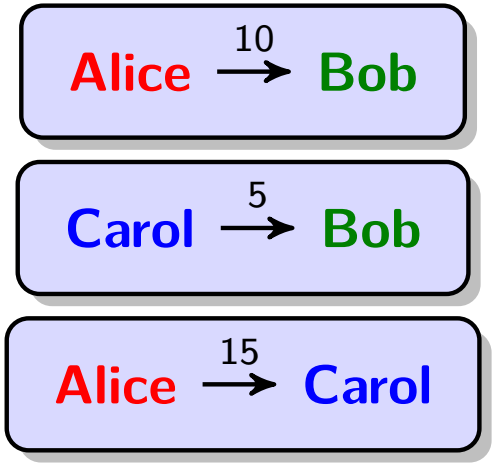


# Simplified Protocol in Action

## 1. Ledger Initialization

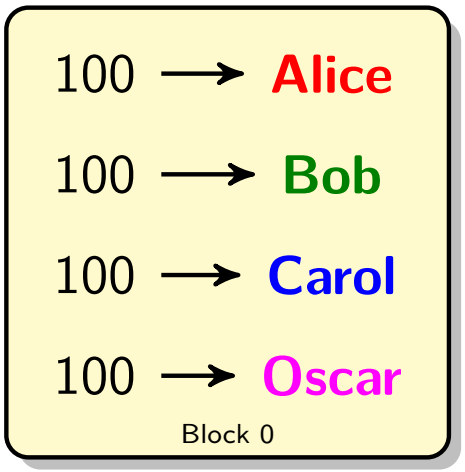



## 2. Collecting Transactions

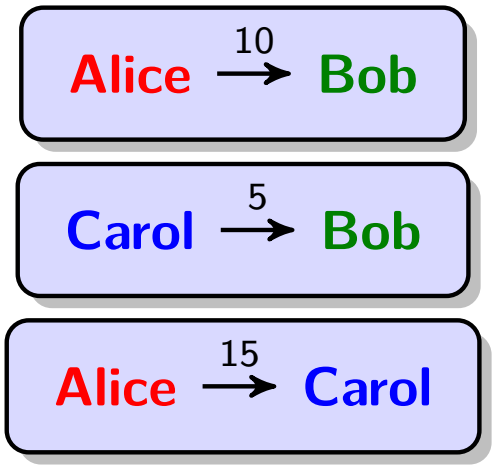


# Simplified Protocol in Action

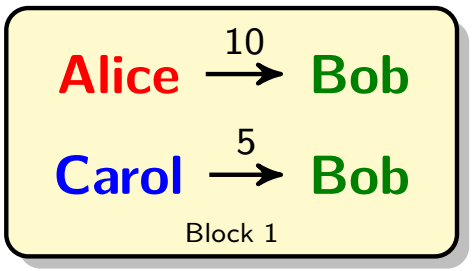
## 1. Ledger Initialization




## 2. Collecting Transactions

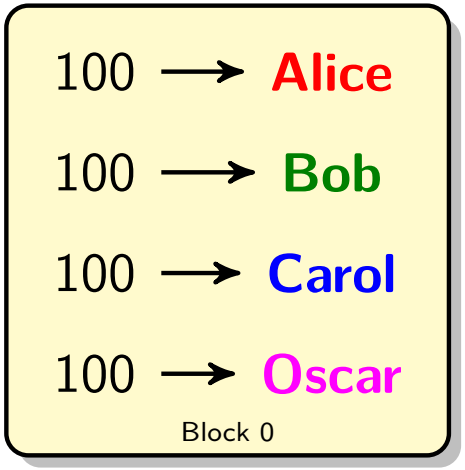



## 3. New Block in Ledger

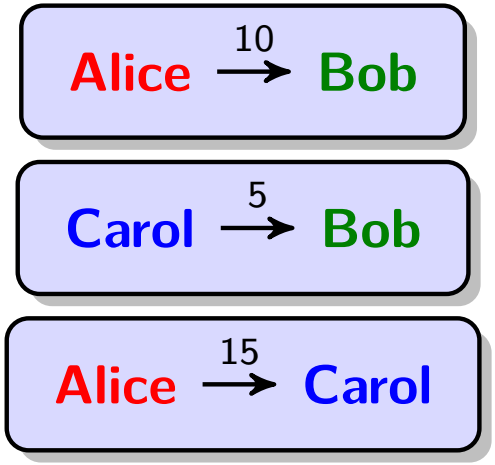


# Simplified Protocol in Action

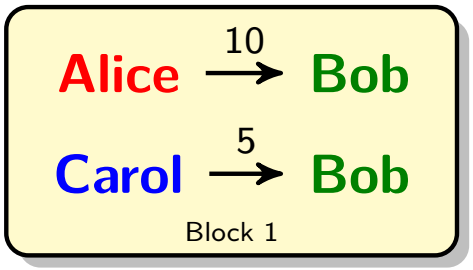
## 1. Ledger Initialization




## 2. Collecting Transactions




## 3. New Block in Ledger




## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **115**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Back to Reality!

université de BORDEAUX

We need to enforce the three following properties in a "*real life*" context and **in a distributed manner**:

1. **Secured Ownership**
2. **Unfalsifiability**
3. **No Double Spending**

**Oscar** will be our opponent, he will try to find all the weaknesses of our simplified protocol and we will try to fix it.

# How to break Secured Ownership?

# Secured Ownership

**Oscar** can **send a transactions on the network to transfer money from Alice to his account** as there is no authentication on who is issuing the transaction.

# Secured Ownership

**Oscar** can **send a transactions on the network to transfer money from Alice to his account** as there is no authentication on who is issuing the transaction.

## Using Asymmetric Cryptography (Public/Private Keys)

Each user gets a **signing key** (private key) and **sign its transactions** with it to ensure the origin of the transaction.

Any other user can take the **verification key** (public key) and **check the signature of the transaction** (which must be enclosed in the transaction block).

# Secured Ownership

université de BORDEAUX

**Oscar** can **send a transactions on the network to transfer money from Alice to his account** as there is no authentication on who is issuing the transaction.

## Using Asymmetric Cryptography (Public/Private Keys)

Each user gets a **signing key** (private key) and **sign its transactions** with it to ensure the origin of the transaction.

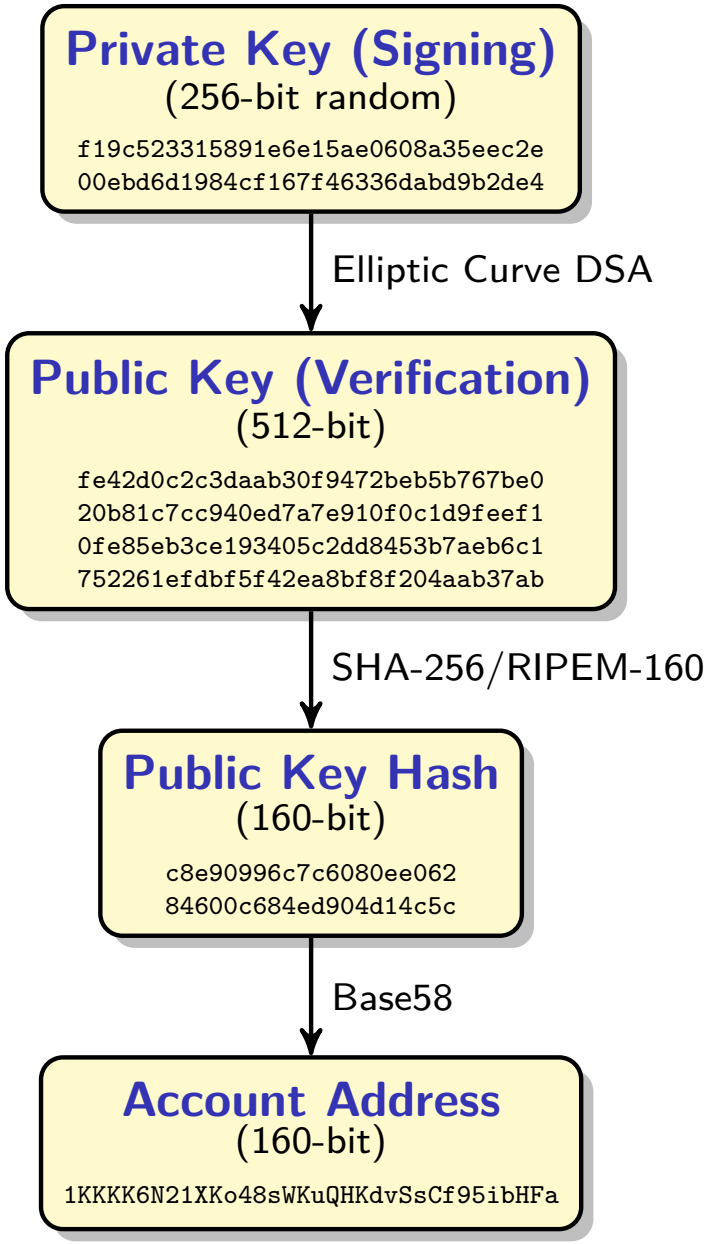
Any other user can take the **verification key** (public key) and **check the signature of the transaction** (which must be enclosed in the transaction block).

**But, we want a distributed protocol!**

The problem, here, is that when verifying the transaction, **you cannot trust a public key that you get through the P2P network**. We need a way to associate trustfully an account and a verification key in a distributed manner.

# Link Public Key and Account Number

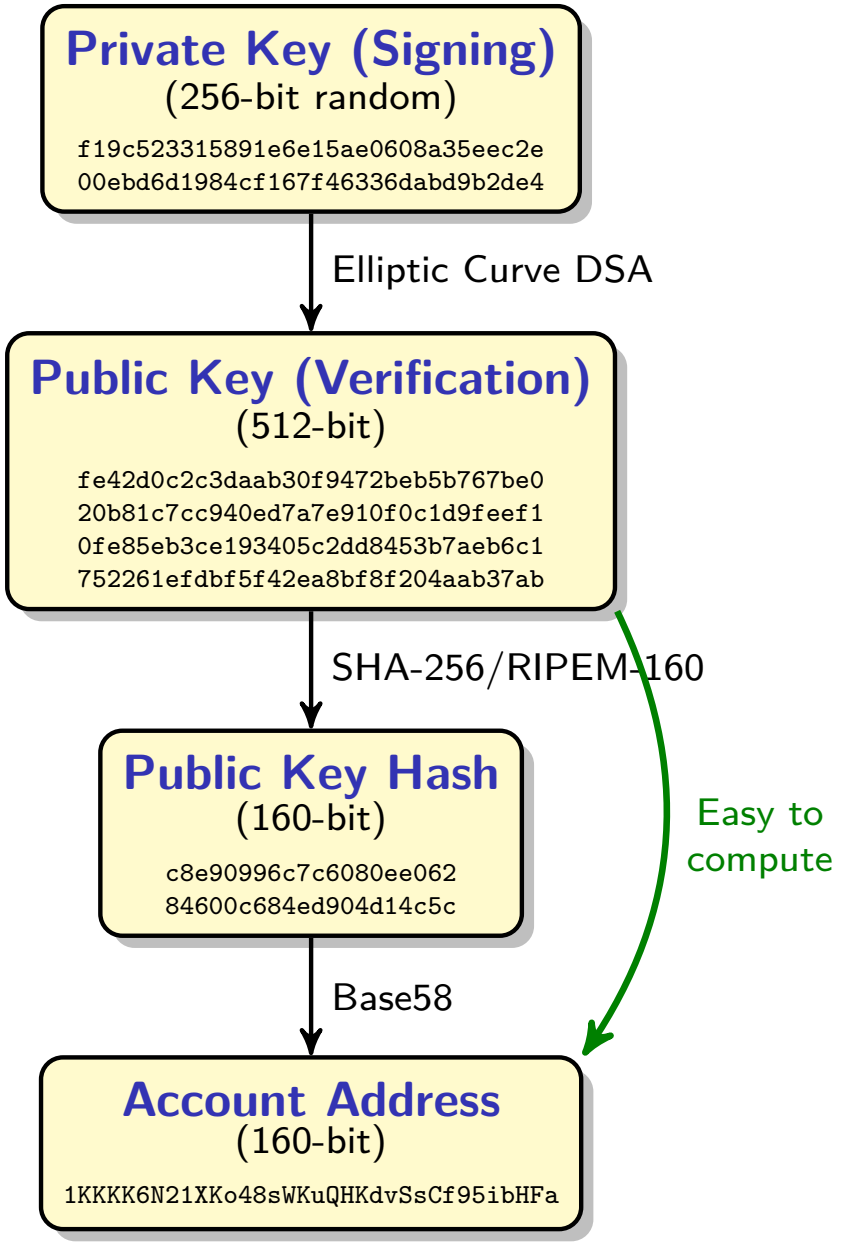
We link the private/public keys and an account number in an asymmetric way that cannot be undone.



# Link Public Key and Account Number

We link the private/public keys and an account number in an asymmetric way that cannot be undone.

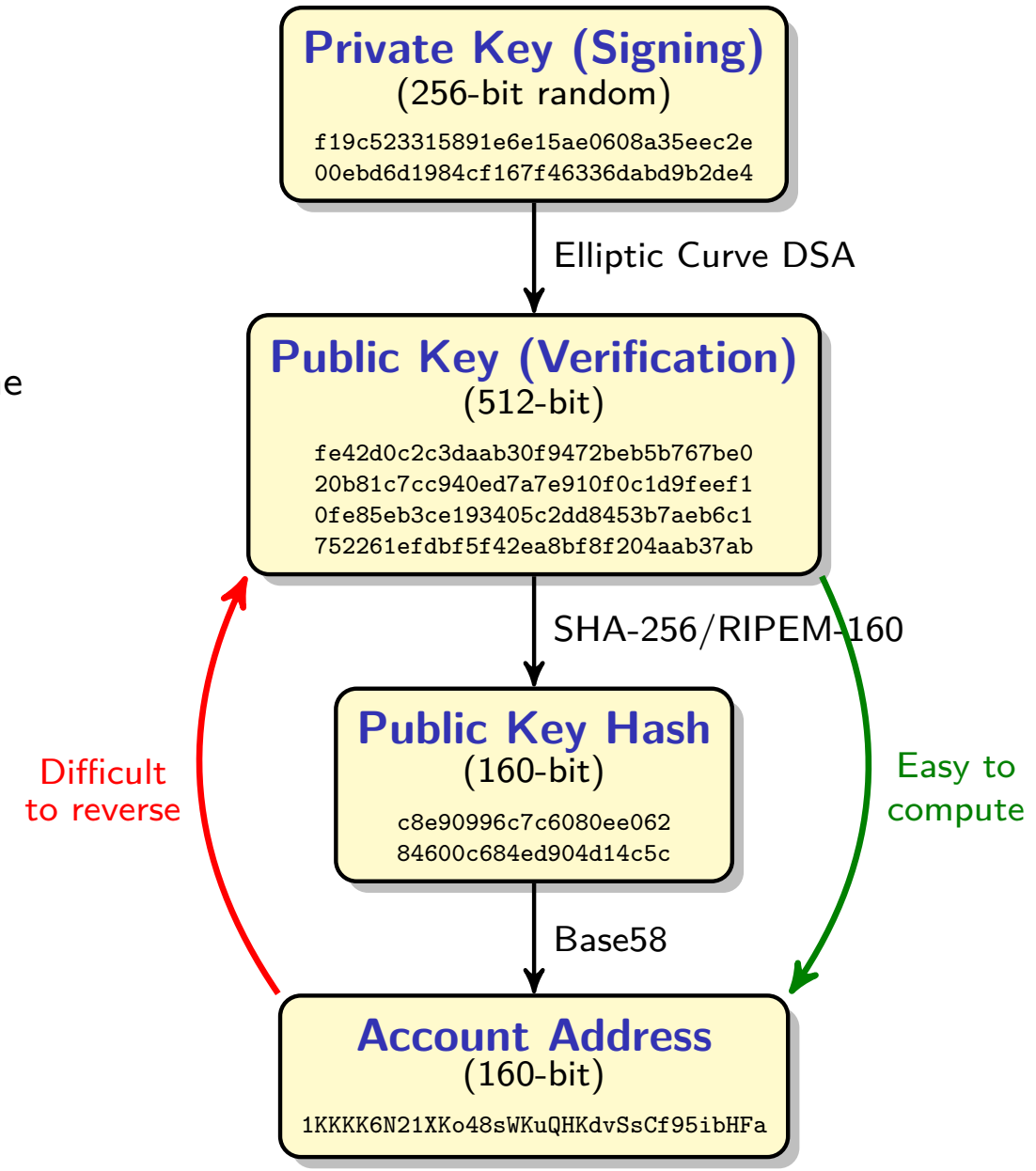
- Finding the account number from the public key is easy;



# Link Public Key and Account Number

We link the private/public keys and an account number in an asymmetric way that cannot be undone.

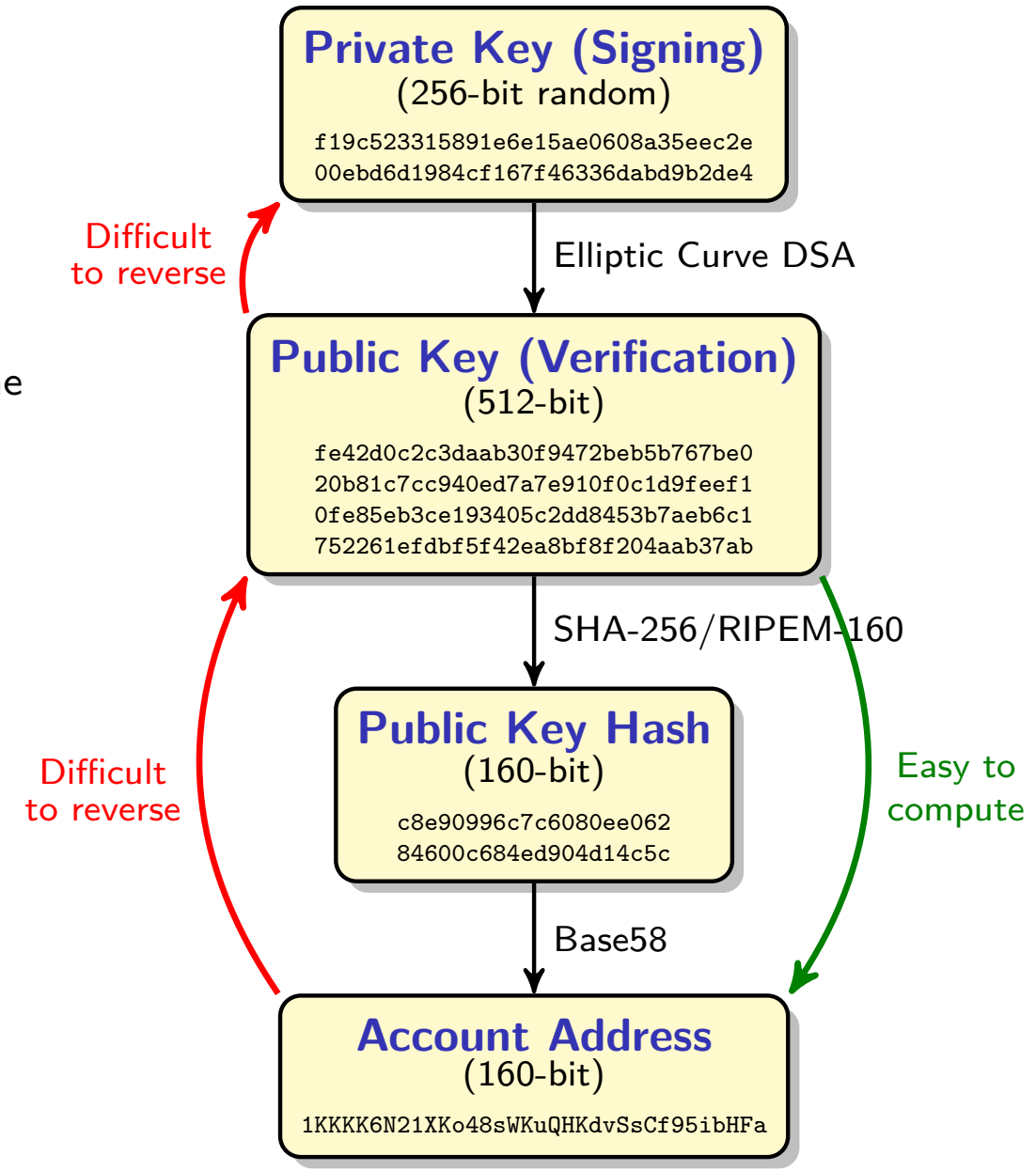
- Finding the account number from the public key is easy;
- Get the public key from the account number is kept difficult;



# Link Public Key and Account Number

We link the private/public keys and an account number in an asymmetric way that cannot be undone.

- Finding the account number from the public key is easy;
- Get the public key from the account number is kept difficult;
- Secret key must be kept secret.



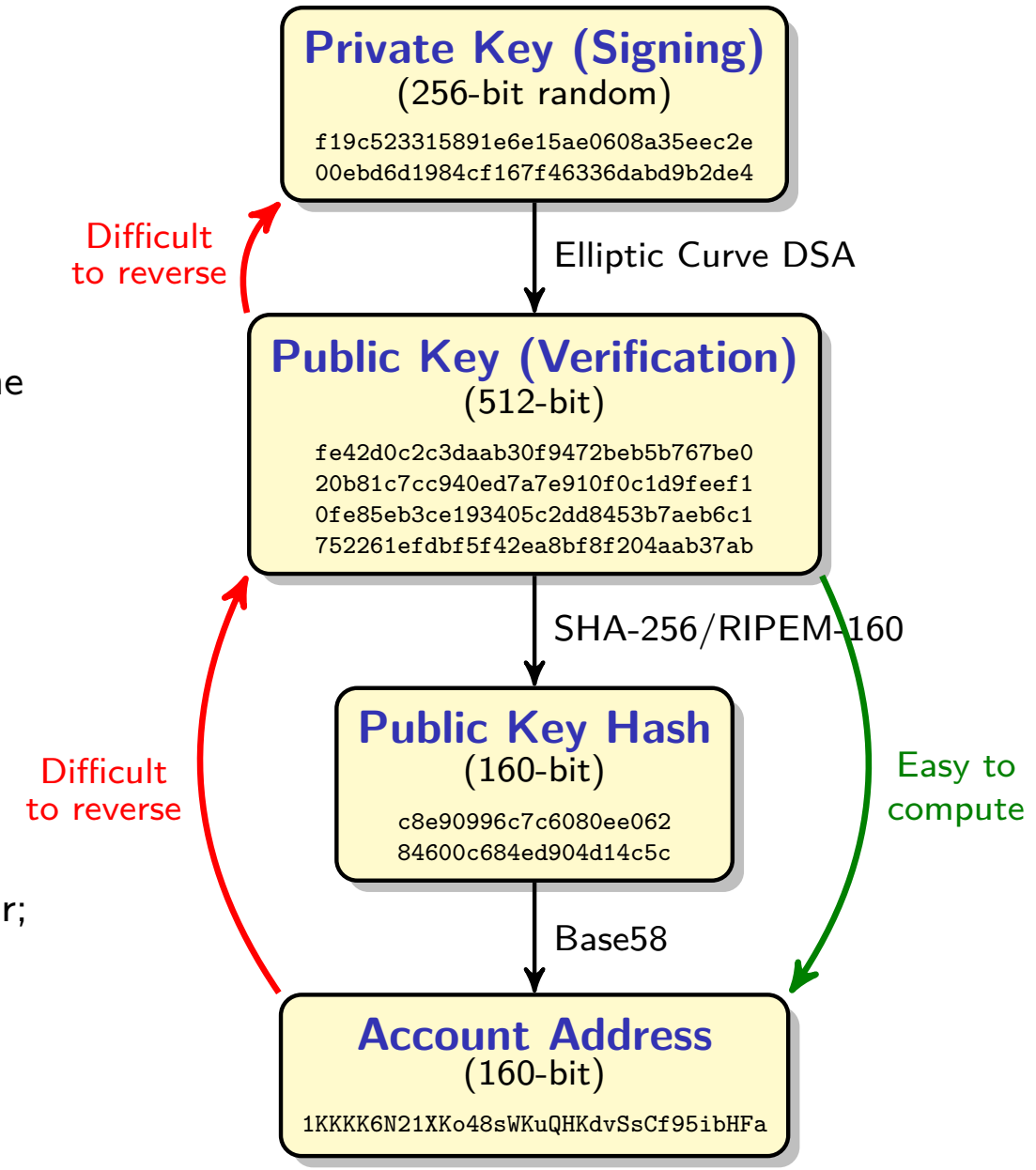
# Link Public Key and Account Number

We link the private/public keys and an account number in an asymmetric way that cannot be undone.

- Finding the account number from the public key is easy;
- Get the public key from the account number is kept difficult;
- Secret key must be kept secret.

Now, a transaction enclose:

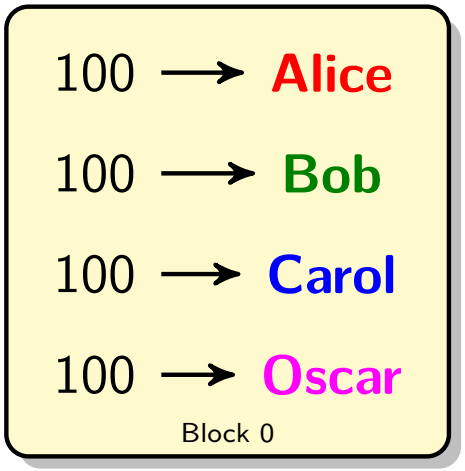
- Details of the transfer (as before);
- Signature of the transfer by the payer;
- Verification key of the payer.



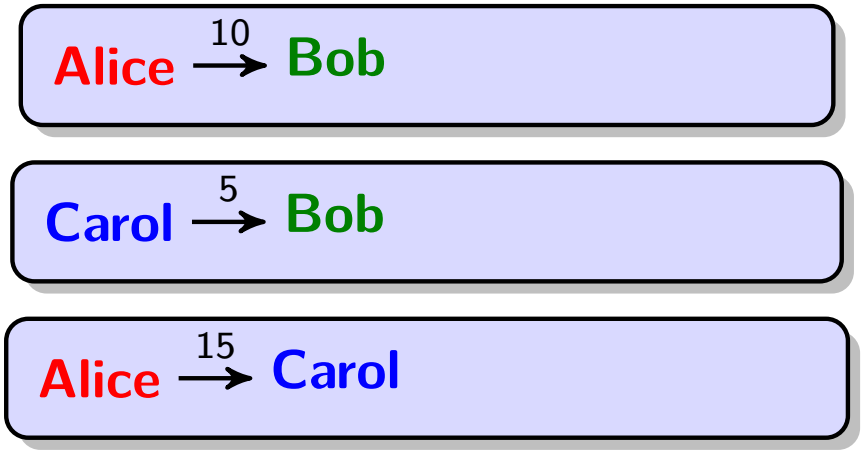
# Simplified Protocol in Action (Take 2)

université de BORDEAUX

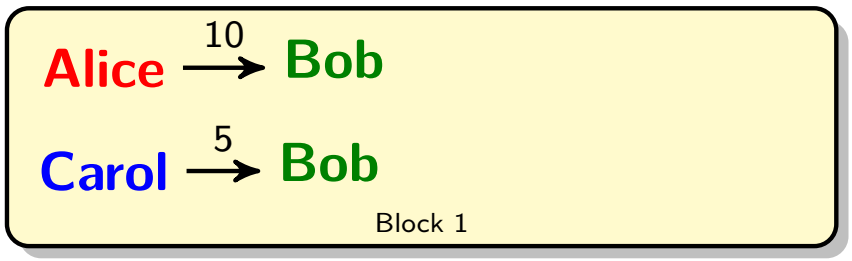
## 1. Ledger Initialization




## 2. Collecting Transactions




## 3. New Block in Ledger




## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **115**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 2)

université de BORDEAUX

## 1. Ledger Initialization

100 → Alice

100 → Bob

100 → Carol

100 → Oscar

Block 0

## 2. Collecting Transactions

Alice $\xrightarrow{10}$ Bob, $\text{Sign}_{\text{Alice}}$, $V_{K_{\text{Alice}}}$

Carol $\xrightarrow{5}$ Bob, $\text{Sign}_{\text{Carol}}$, $V_{K_{\text{Carol}}}$

Alice $\xrightarrow{15}$ Carol, $\text{Sign}_{\text{Alice}}$, $V_{K_{\text{Alice}}}$

## 3. New Block in Ledger

Alice $\xrightarrow{10}$ Bob

Carol $\xrightarrow{5}$ Bob

Block 1

## 4. Account Balances

- Alice has **90** (with a transaction pending).
- Bob has **115**.
- Carol has **95** (with a transaction pending).
- Oscar has **100**.

# Simplified Protocol in Action (Take 2)

université de BORDEAUX

## 1. Ledger Initialization

100 → **Alice**

100 → **Bob**

100 → **Carol**

100 → **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **115**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 2)

université de BORDEAUX

## 1. Ledger Initialization

100 → Alice

100 → Bob

100 → Carol

100 → Oscar

Block 0

## 2. Collecting Transactions

$\text{Alice} \xrightarrow{10} \text{Bob}, \text{Sign}_{\text{Alice}}, V_{K_{\text{Alice}}}$

$\text{Carol} \xrightarrow{5} \text{Bob}, \text{Sign}_{\text{Carol}}, V_{K_{\text{Carol}}}$

$\text{Alice} \xrightarrow{15} \text{Carol}, \text{Sign}_{\text{Alice}}, V_{K_{\text{Alice}}}$

## 3. New Block in Ledger

$\text{Alice} \xrightarrow{10} \text{Bob}, \ldots, V_{K_{\text{Alice}}}$

$\text{Carol} \xrightarrow{5} \ldots, V_{K_{\text{Carol}}}$

Block 1

Verified before each local insertion!

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **115**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

université de BORDEAUX

# How to break Unfalsifiability?

# Unfalsifiability

**Oscar** can simply **create new coins by crediting his account in the ledger** as it is done in the initialization phase. Nobody seems to check that...

# Unfalsifiability

université de BORDEAUX

**Oscar** can simply **create new coins by crediting his account in the ledger** as it is done in the initialization phase. Nobody seems to check that...

A quick fix could be to **forbid the issue of new coins except in the very first block of the ledger**. But, it would be impossible to regulate the monetary base!

# Unfalsifiability

université de BORDEAUX

**Oscar** can simply **create new coins by crediting his account in the ledger** as it is done in the initialization phase. Nobody seems to check that...

A quick fix could be to **forbid the issue of new coins except in the very first block of the ledger**. But, it would be impossible to regulate the monetary base!

## Chain of Ownerships

We want to build **a chain of ownerships** from the birth of the coin to the current owner in order to be able to follow each coin spent and establish if it is a legal coin or not.

# Unfalsifiability

université de BORDEAUX

**Oscar** can simply **create new coins by crediting his account in the ledger** as it is done in the initialization phase. Nobody seems to check that...

A quick fix could be to **forbid the issue of new coins except in the very first block of the ledger**. But, it would be impossible to regulate the monetary base!

## Chain of Ownerships

We want to build **a chain of ownerships** from the birth of the coin to the current owner in order to be able to follow each coin spent and establish if it is a legal coin or not.

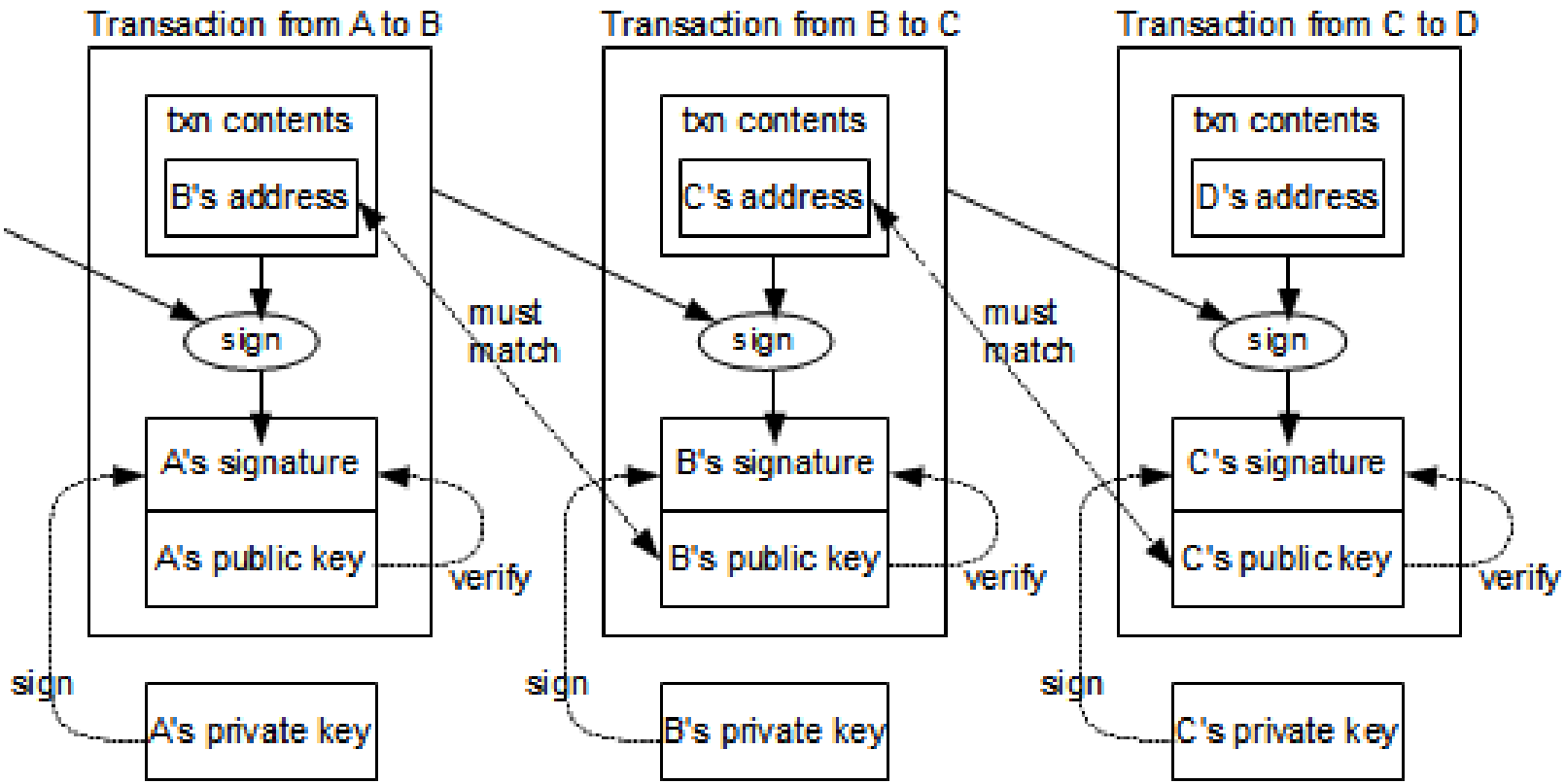
## Creating New Coins

**Creating new coins** cannot be done "*on user demand*" without risk of abuse. In fact, **each coin creation should be agreed by all the participants and distributed in a fair way among users**.

One way to do it is to **issue new coins with a predictable rule** that everybody should be aware off. It can also be **used as an incentive to build and check new ledger blocks.**

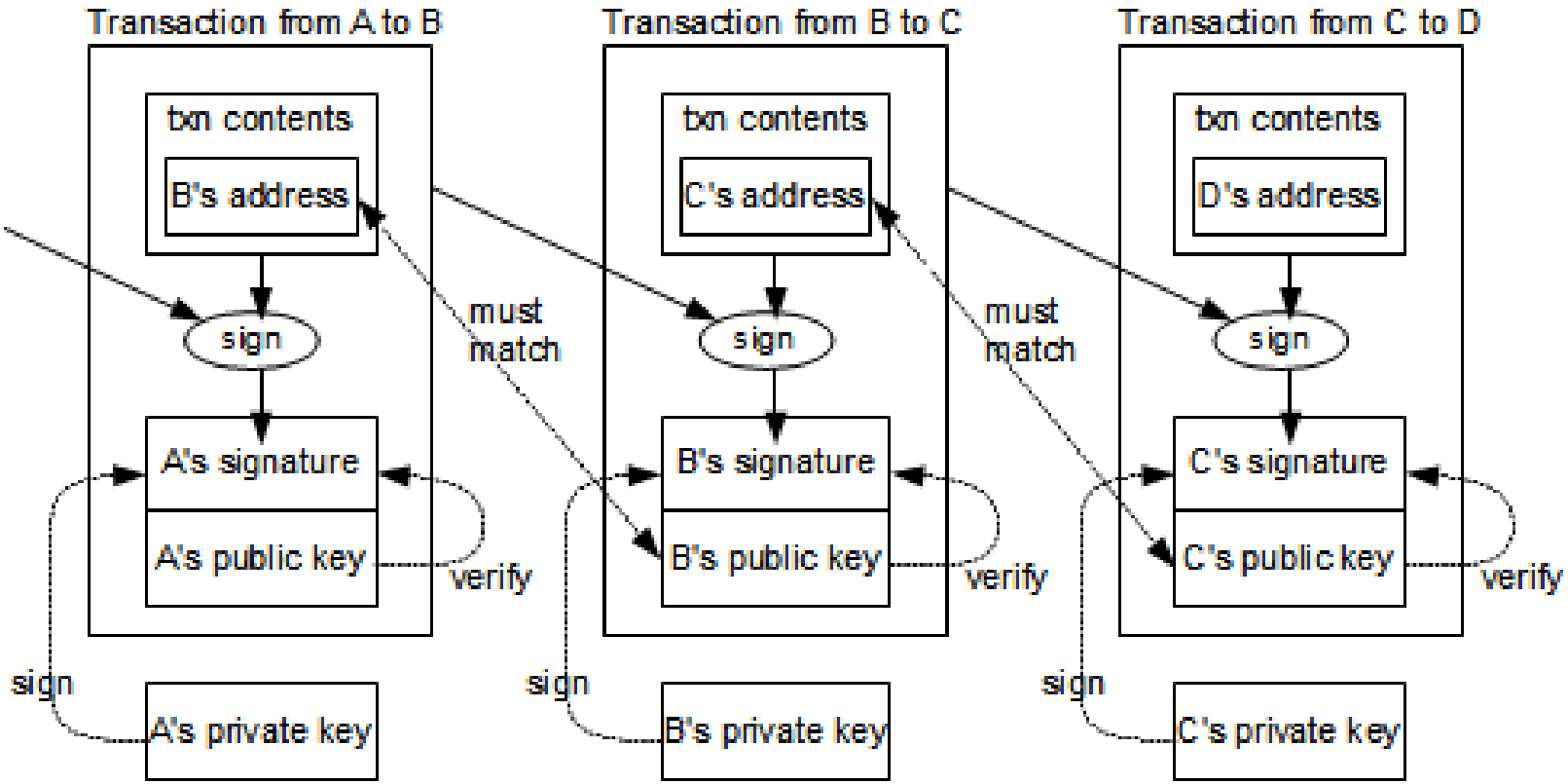
# Build a Chain of Ownerships

Build a chain of transactions in a way that cannot be undone!



# Build a Chain of Ownerships

Build a chain of transactions in a way that cannot be undone!




**But, lets dig into the Bitcoin transactions!**

# Types of Transactions

Three Types of Transactions:

- **Generation**;
- **Simple**;
- **Multiple**.

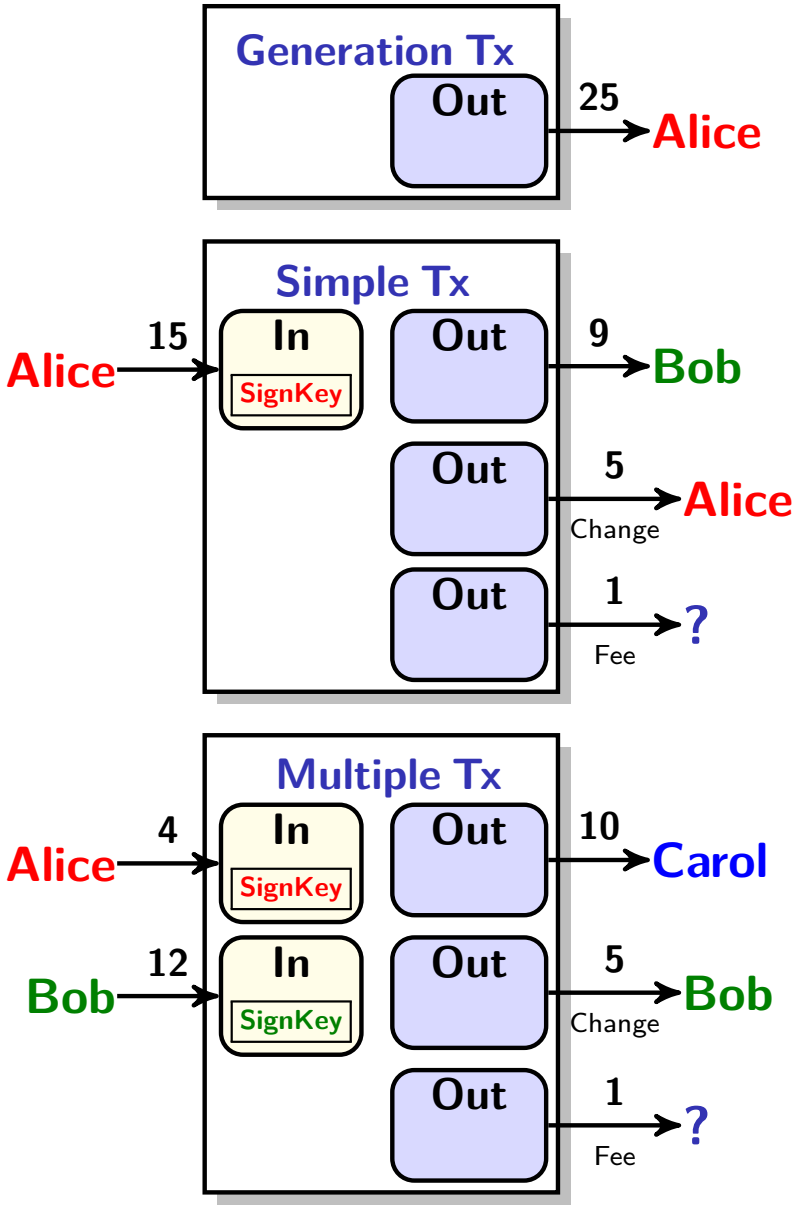
A **transaction** is a set of **In**s and **Out**s.

The sum of **In** amounts must be equal to the sum of **Out** amounts (except for generation).

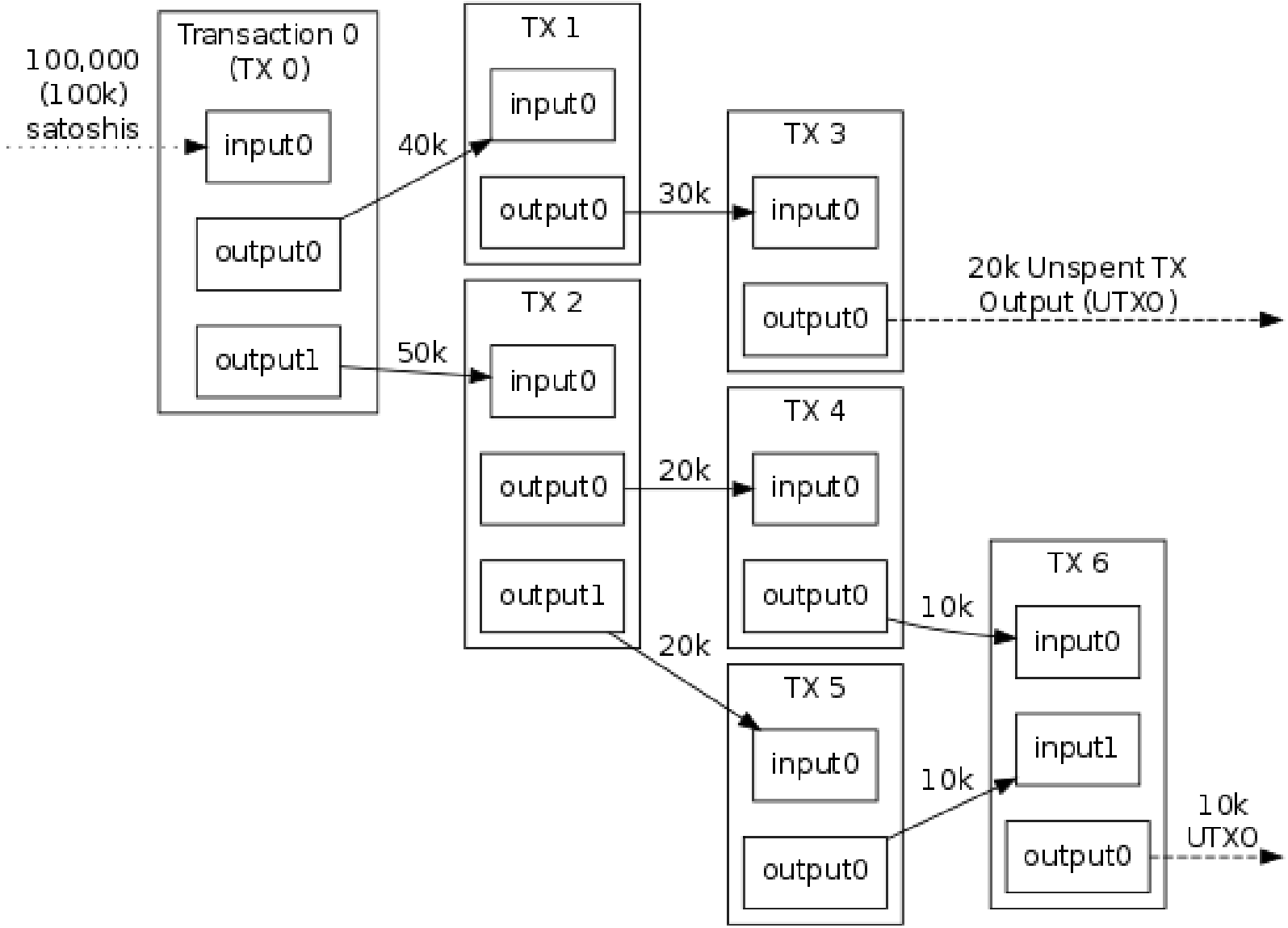
All the **In** boxes must retain the **Sign Key** of the payer (chain of ownerships).

The whole amount of the previous transactions is used, so some **change** is returned back to the payer.

Part of the transaction can be left as **fees** for the user that will insert the transaction in the ledger.



# Transaction Propagation




Triple-Entry Bookkeeping (Transaction-To-Transaction Payments) As Used By Bitcoin

# Control the Generation of New Coins

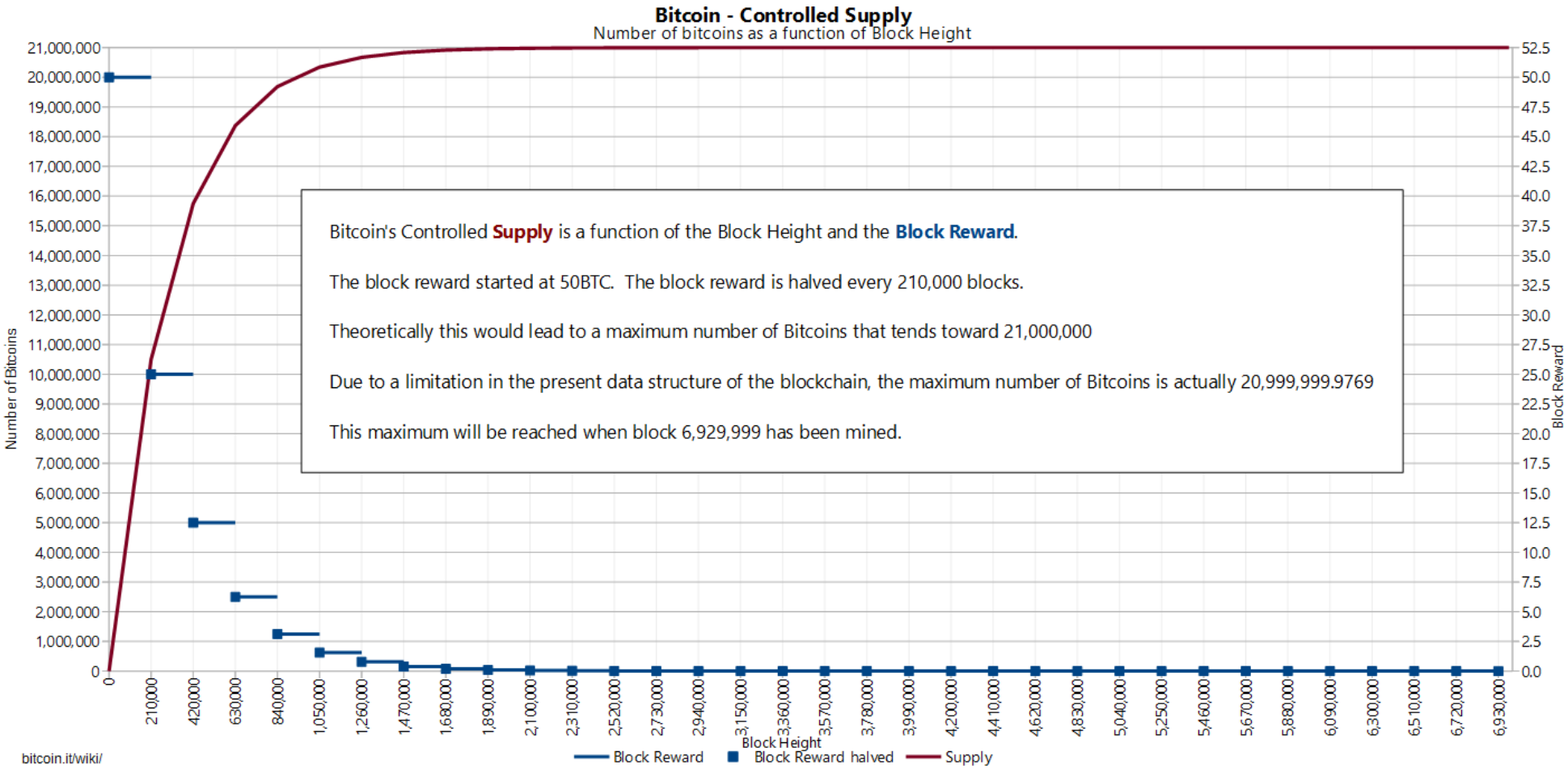
université de BORDEAUX

Creating **new coins** in a **decentralized way** must follow a **predetermined rule agreed by all participants**.

And, these new coins can be an **incentive to update the ledger**.

- We aim to generate **21,000,000 coins** (almost arbitrarily chosen).
  Due to a limitation in the present data-structure of the blockchain, the maximum number of Bitcoins is actually 20,999,999.9769.
- Each new block starts with a **reward in favor of the block creator**.
- Amount of reward **starts at 50 and is divided by two every 210,000 blocks** (50 for the first 210,000, 25 after 210,000, 12.5 after 420,000, ...).
- A coin can be broken into $10^{-8}$ sub-units (Satoshi).

# Control the Generation of New Coins




Bitcoin's Controlled **Supply** is a function of the Block Height and the **Block Reward**.

The block reward started at 50BTC. The block reward is halved every 210,000 blocks.

Theoretically this would lead to a maximum number of Bitcoins that tends toward 21,000,000

Due to a limitation in the present data structure of the blockchain, the maximum number of Bitcoins is actually 20,999,999.9769

This maximum will be reached when block 6,929,999 has been mined.

# Fair Distribution of New Coins

université de BORDEAUX

Giving away new coins for building blocks must match properties:

- We need to ensure **fairness** between all participants;
- We need to ensure that **people getting it are really involved**;
- We need to do this in **a distributed manner**;
- We need **a process that self-adjusts with the number of participants**;
- Finally, we want to **limit blocks computation by one every 10 minutes maximum** (to ensure a fair propagation over the P2P network).

# Fair Distribution of New Coins

université de BORDEAUX

Giving away new coins for building blocks must match properties:

- We need to ensure **fairness** between all participants;
- We need to ensure that **people getting it are really involved**;
- We need to do this in **a distributed manner**;
- We need **a process that self-adjusts with the number of participants**;
- Finally, we want to **limit blocks computation by one every 10 minutes maximum** (to ensure a fair propagation over the P2P network).

## Proof of Work (Hashcash protocol)

1. Choose a **level of difficulty** $n$.
2. We add a **header** to each block with **the hash of each transaction that is enclosed in the block** and **a nonce**.
3. **Choose a value for this nonce** and **hash the header with SHA-256**.
4. If **the hash begins with $n$ zeros**, you have **a valid block**. If not, **try another nonce**.
5. **Every 2,016 blocks**, check if the average computational time for a block is **10 minutes**. **Adjust the difficulty** depending on this average time (make it harder or easier).

# Mining a Block: An example

| Block Header | SHA-256 |
|---|---|
| "Hello, world:0" | f669d19d4e9164a966c148d82730bbfa92834c9966ee1fa38181e51630de3e16 |
| "Hello, world:1" | f1fadff933b9d7aeb47e94366aa4dcf35681a7c8ff0f9bf16f23bdff6606993b |
| "Hello, world:2" | 987cf12b3d173f6eddf5cbb22da4974ab4a41f92d3e6ec1e4be9415d70f4dc88 |
| ... | ... |
| "Hello, world:4" | 06b4c68f2cc52963e49aba36b7c7be87bd1ca1222b475fa6a33ce1a35d9b4116 |
| ... | ... |
| "Hello, world:24" | 00a082f172ae174a91ef5999dbd2c98b066116cae902a7d907bf20f01e7b2a35 |
| ... | ... |
| "Hello, world:2062" | 000b59a5cbe48154de5ecc492fdadc5e346b08f4e353fecd36c8f2803134371f |
| ... | ... |
| "Hello, world:54049" | 0000ba63655fd895c461cf9eb8b12d84bb6ee67617f03718c9af58b3529eabeb |
| ... | ... |

## What does that mean?

- Cryptographic hash functions are designed in a way that make this protocol equivalent to a (fair) random choice among the participants;
- The more computing power you invest in the process, the more you get rewarded;
- You can do it in a distributed way, even if you do not trust the other participants because once the nonce has been found, everybody can check if this is true;
- The protocol self-adjusts every 2,016 blocks.

# Simplified Protocol in Action (Take 3)

université de BORDEAUX

## 1. Ledger Initialization

100 → **Alice**

100 → **Bob**

100 → **Carol**

100 → **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **115**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 3)

université de BORDEAUX

## 1. Ledger Initialization

100 ⟶ **Alice**

100 ⟶ **Bob**

100 ⟶ **Carol**

100 ⟶ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

50 ⟶ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 3)

université de BORDEAUX

## 1. Ledger Initialization

100 ⟶ **Alice**

100 ⟶ **Bob**

100 ⟶ **Carol**

100 ⟶ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

50 ⟶ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 3)

université de BORDEAUX

## 1. Ledger Initialization

100 ⟶ **Alice**

100 ⟶ **Bob**

100 ⟶ **Carol**

100 ⟶ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

BlockID,

50 ⟶ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 3)

## 1. Ledger Initialization

100 $\longrightarrow$ **Alice**

100 $\longrightarrow$ **Bob**

100 $\longrightarrow$ **Carol**

100 $\longrightarrow$ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

BlockID,Hsh($Tx_0$),Hsh($Tx_1$),Hsh($Tx_2$),

50 $\longrightarrow$ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 3)

université de BORDEAUX

## 1. Ledger Initialization

100 $\longrightarrow$ **Alice**

100 $\longrightarrow$ **Bob**

100 $\longrightarrow$ **Carol**

100 $\longrightarrow$ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

BlockID,Hsh($Tx_0$),Hsh($Tx_1$),Hsh($Tx_2$),**Nonce**

50 $\longrightarrow$ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 3)

université de BORDEAUX

## 1. Ledger Initialization

100 ⟶ **Alice**

100 ⟶ **Bob**

100 ⟶ **Carol**

100 ⟶ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

BlockID,Hsh($Tx_0$),Hsh($Tx_1$),Hsh($Tx_2$),**Nonce**

50 ⟶ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

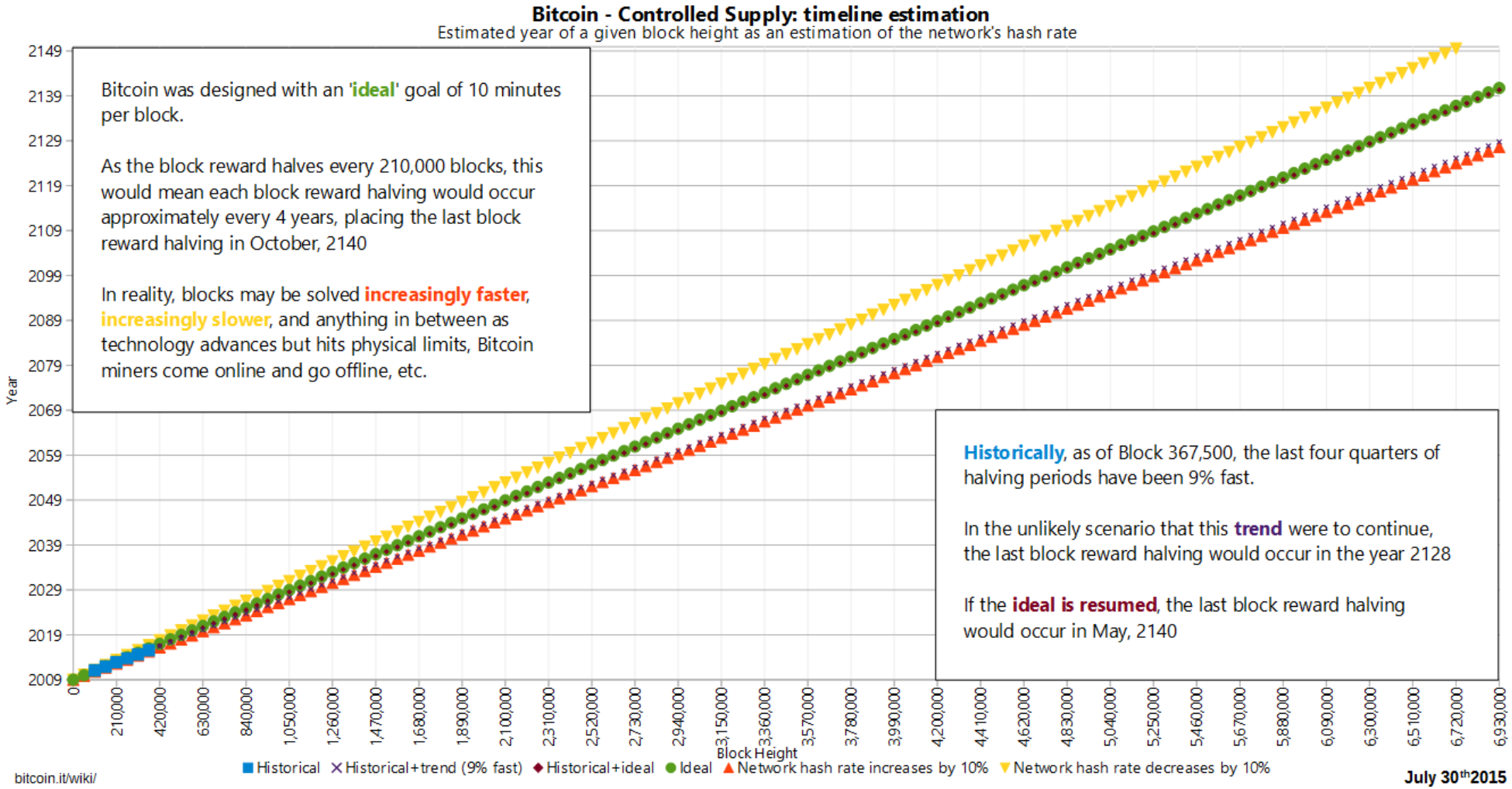
Verified before each local insertion!

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# When Will Be Issued the Last Bitcoin?

université de BORDEAUX



# Bitcoin Price / Mining Difficulty

# Alternate Algorithm: Proof of Stake

université de BORDEAUX

How to reach a **distributed consensus** using **less resources** than with the **proof of work**?

## Proof of Stake (PoS)/Proof of Work (PoW)

In *PoS*, we replace the CPU power used in *PoW* by the amount of Bitcoins detained by the participant. If the participant own 17% of the whole amount of all the participants to decide who is inserting the new block, then he has 17% of chances to insert his block. The choice among the participants is then made by using a known pseudo-random function based on a predictable seed (*e.g.* current time).

### Pros

- No consumption of resources;
- Forks are costless;
- Render attacks much more costly.

### Cons

- Monopoly problem;
- Forks can be a good choice;
- No more incentive to create blocks.

université de BORDEAUX

# How to do Double-spending?

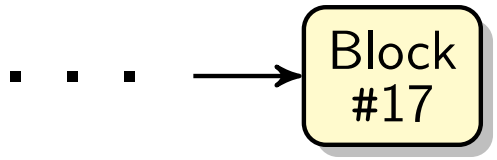
# No Double-spending

université de BORDEAUX

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.

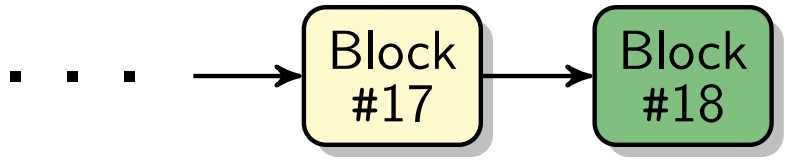
# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



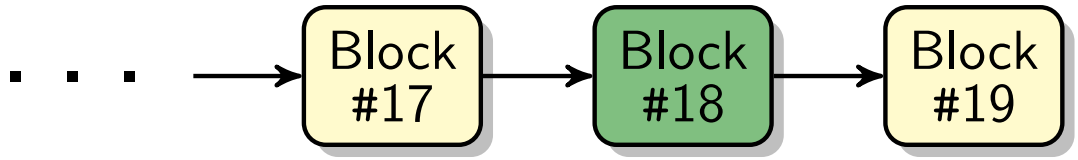
# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



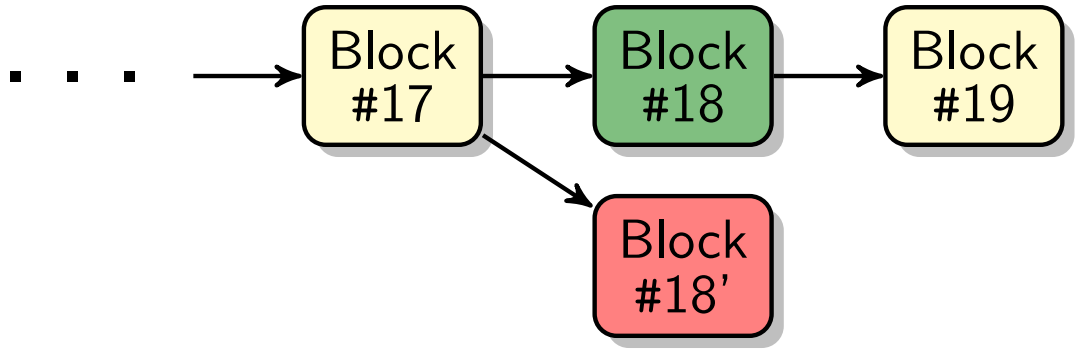
# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



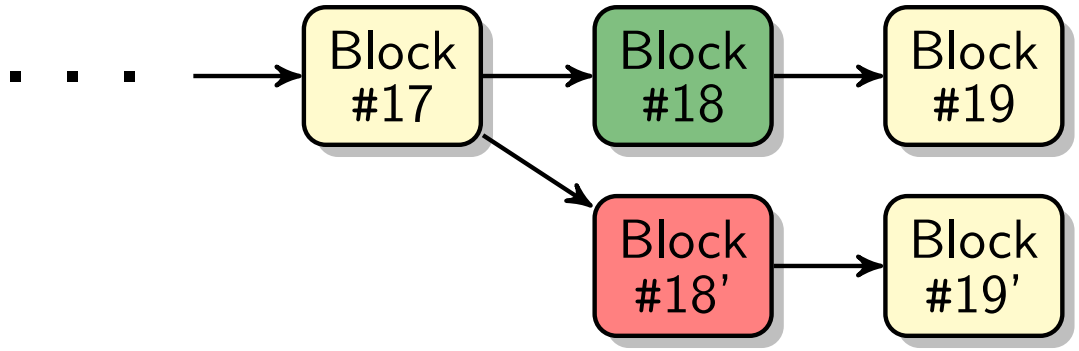
# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



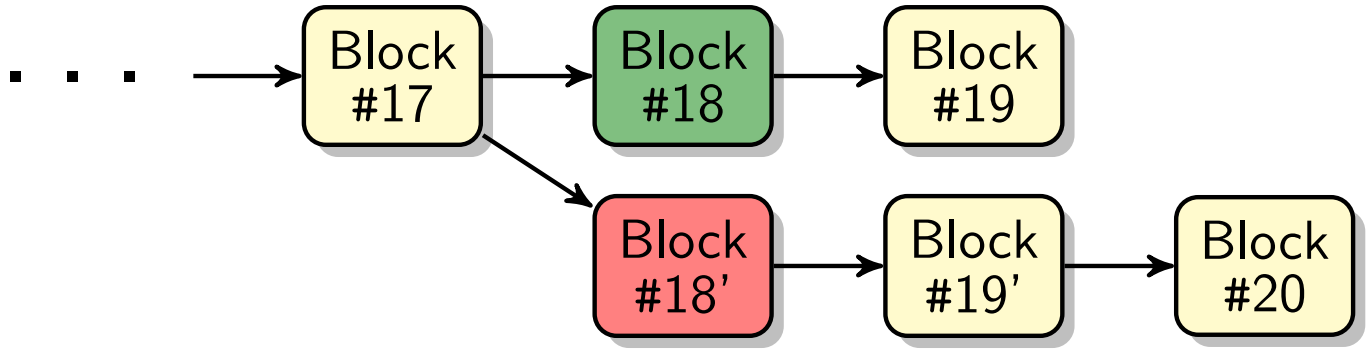
# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



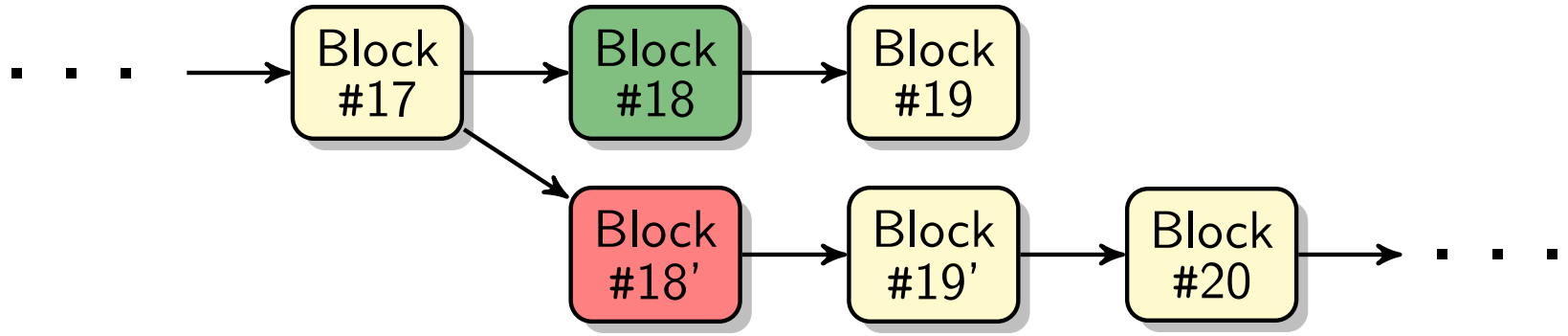
# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



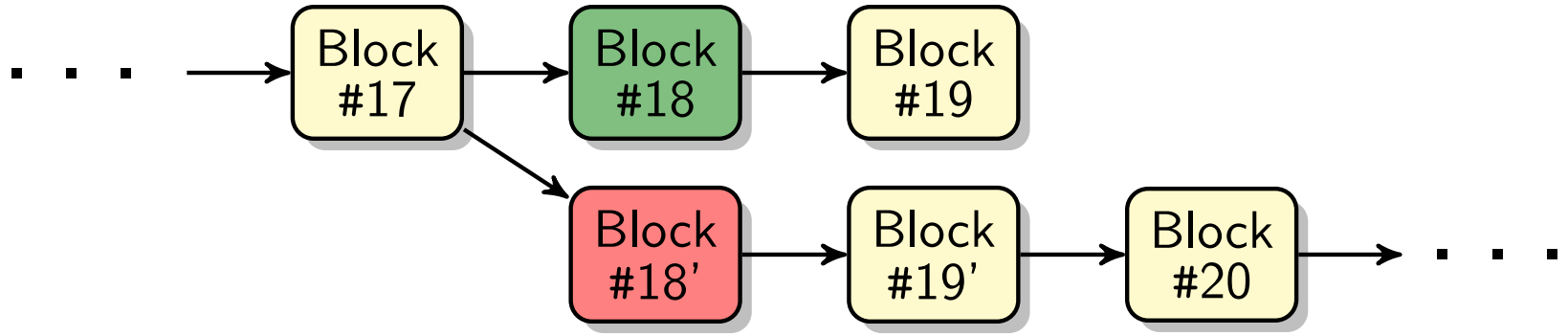
# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.

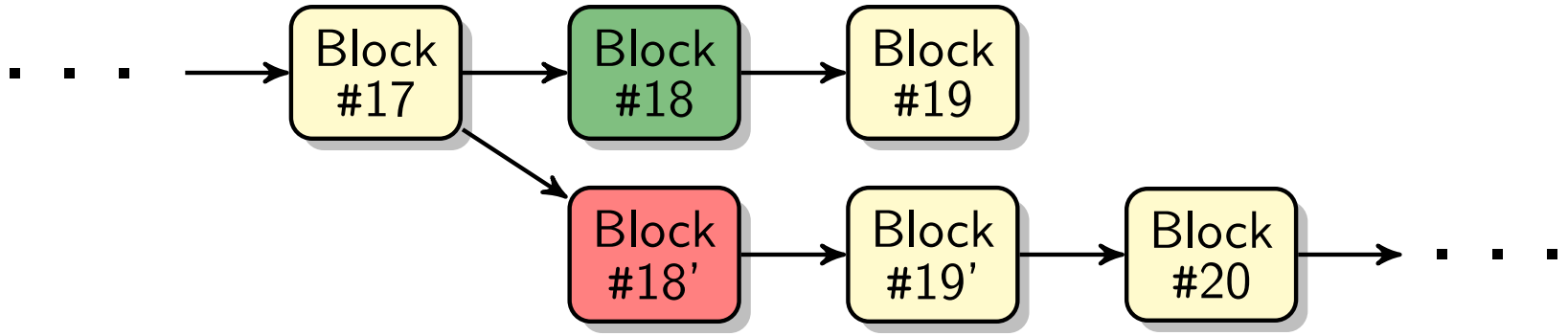



**Longest Chain Principle**

All participants should make sure to always build on the latest block.

# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.



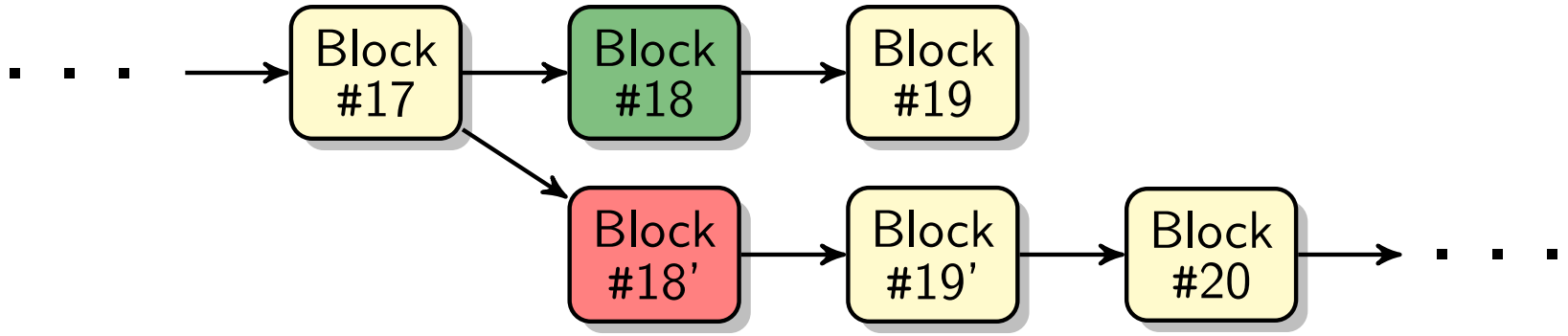

**Longest Chain Principle**

All participants should make sure to always build on the latest block.

The **longest chain principle**, together with the **proof of work**, makes it unlikely for an attacker to take-over the blockchain.

# No Double-spending

To do double-spending **Oscar** needs to **fork the blockchain**, **rewrite part of its past** and **force the others use his fake blocks**.




**Longest Chain Principle**

All participants should make sure to always build on the latest block.

The **longest chain principle**, together with the **proof of work**, makes it unlikely for an attacker to take-over the blockchain.

**The 51% Attack**

An attacker controlling 51% of the computing power of the blockchain network has a chance to take-over the blockchain and arbitrarily choose which transaction to validate.

université de BORDEAUX

## Two types of forks on the Blockchain!

### Soft Forks

A fork on the consensus of the last block in the blockchain.

- Network splits;
- Accidental double finding;
- Malicious insertions in the blockchain.

### Hard Fork

A fork on the consensus of the rules of the blockchain.

- Update of the P2P network software;
- Changing consensus rules in the network.

# Simplified Protocol in Action (Take 4)

université de BORDEAUX

## 1. Ledger Initialization

100 ⟶ **Alice**

100 ⟶ **Bob**

100 ⟶ **Carol**

100 ⟶ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

BlockID,Hsh($Tx_0$),Hsh($Tx_1$),Hsh($Tx_2$),Nonce

50 ⟶ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 4)

université de BORDEAUX

## 1. Ledger Initialization

$100 \longrightarrow$ **Alice**

$100 \longrightarrow$ **Bob**

$100 \longrightarrow$ **Carol**

$100 \longrightarrow$ **Oscar**

Block 0

## 2. Collecting Transactions

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

**Alice** $\xrightarrow{15}$ **Carol**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

## 3. New Block in Ledger

**Build on the latest block you can get!**

BlockID,Hsh($Tx_0$),Hsh($Tx_1$),Hsh($Tx_2$),Nonce

$50 \longrightarrow$ **Bob**, Generation Tx

**Alice** $\xrightarrow{10}$ **Bob**, $\mathbf{Sign}_{\mathbf{Alice}}$, $V_{K_{\mathbf{Alice}}}$

**Carol** $\xrightarrow{5}$ **Bob**, $\mathbf{Sign}_{\mathbf{Carol}}$, $V_{K_{\mathbf{Carol}}}$

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Simplified Protocol in Action (Take 4)

université de BORDEAUX

## 1. Ledger Initialization

100 ⟶ Alice

100 ⟶ Bob

100 ⟶ Carol

100 ⟶ Oscar

Block 0

## 2. Collecting Transactions

Alice $\xrightarrow{10}$ Bob, $\text{Sign}_{\text{Alice}}$, $V_{K_{\text{Alice}}}$

Carol $\xrightarrow{5}$ Bob, $\text{Sign}_{\text{Carol}}$, $V_{K_{\text{Carol}}}$

Alice $\xrightarrow{15}$ Carol, $\text{Sign}_{\text{Alice}}$, $V_{K_{\text{Alice}}}$

## 3. New Block in Ledger

**Build on the latest block you can get!**

BlockID,Hsh($Tx_0$),Hsh($Tx_1$),Hsh($Tx_2$),Nonce

50 ⟶ Bob, Generation Tx

Alice $\xrightarrow{10}$ Bob, $\text{Sign}_{\text{Alice}}$, $V_{K_{\text{Alice}}}$

Carol $\xrightarrow{5}$ Bob, $\text{Sign}_{\text{Carol}}$, $V_{K_{\text{Carol}}}$

Verified before each local insertion!

Block 1

## 4. Account Balances

- **Alice** has **90** (with a transaction pending).
- **Bob** has **165**.
- **Carol** has **95** (with a transaction pending).
- **Oscar** has **100**.

# Plan

# Smart Contracts

université de BORDEAUX

A **Smart Contract** is a term coined by Nick Szabo in 1994. It can be seen as "*a computerized transaction protocol that executes the terms of a contract*".

In other words, the **maker(s) of the contract attach a program to the agreement** which is inserted in the blockchain. The **program is executed** each time the blockchain is read through **to evaluate the balance of an account**.

## Examples

- Complex certifications in multi-parties payment in Bitcoin protocol;
- Assurance contract for a farmer based on weather conditions;
- Certified diploma delivered based on final exam marks;
- ...

## Problem

Smart contracts are programs like others, they may have **bugs**.

# Plan

# Conclusion

université de BORDEAUX

- This was the short version of Bitcoin protocol!
  There is much more under the hood!
- Is there a way to control or to cheat the system?
- Will it last forever? Or, can we predict its evolution?
- Can we reproduce the system with different choices?
- Can we adapt the protocol to other domains?

# Questions?

# References


Adam Back.

Hashcash - a denial of service counter-measure, 2002.

David Chaum.

Blind signatures for untraceable payments.

In *Advances in Cryptology: Proceedings of CRYPTO'82*, pages 199–203, 1982.

Chris Clark.

Bitcoin internals - a technical guide to bitcoin, 2013.

Wei Dai.

B-money.

http://www.weidai.com/bmoney.txt, 1998.

Hal Finney.

RPOW: Reusable Proofs of Work.

http://diyhpl.us/~bryan/papers2/bitcoin/finney.org/rpow/rpow-index.html, 2004.

Laurie Law, Susan Sabett, and Jerry Solinas.

How to make a mint: The cryptography of anonymous electronic cash.

*Journal of the American University Law Review*, 46(4):1131–1162, April 1997.

# References


Quantum Mechanic.

Proof of stake instead of proof of work.

https://bitcointalk.org/index.php?topic=27787.0, 2011.

Satoshi Nakamoto.

Bitcoin source code.

https://github.com/bitcoin/bitcoin.

Satoshi Nakamoto.

Bitcoin: A peer-to-peer electronic cash system, 2008.

Nick Szabo.

Bit gold.

http://unenumerated.blogspot.fr/2005/12/bit-gold.html, December 2008.

Florian Tschorsch and Björn Scheuermann.

Bitcoin and beyond: A technical survey on decentralized digital currencies.

*IEEE Communications Surveys Tutorials (COMST)*, (99), 2016.

Gavin Wood.

Ethereum: A secure decentralized generalised transaction ledger.

http://gavwood.com/paper.pdf, 2014.