Introduction to
Distributed Ledger Technologies
(the classy name of blockchains)

Quentin Bramas bramas@unistra.fr

May, 25th 2018, ICUBE

What are we ?

What are we ?

Computers

What are we ?

What do we want ?

Computers

What are we ?

What do we want ?

Computers

Distributed data

What are we ?

What do we want ?

How we want it ?

Computers

Distributed data

What are we ?

What do we want ?

How we want it ?

Computers

Distributed data

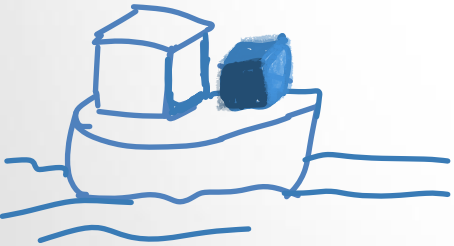
Consistent

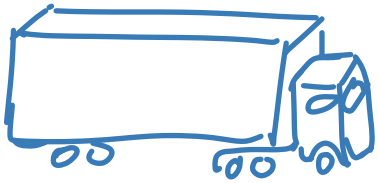
Efficient

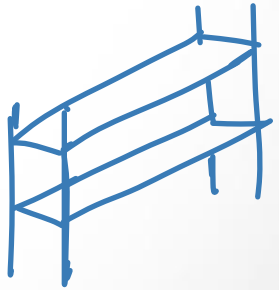
Secure

Scalable

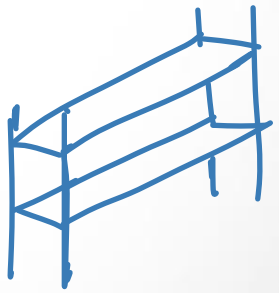
Blockchain



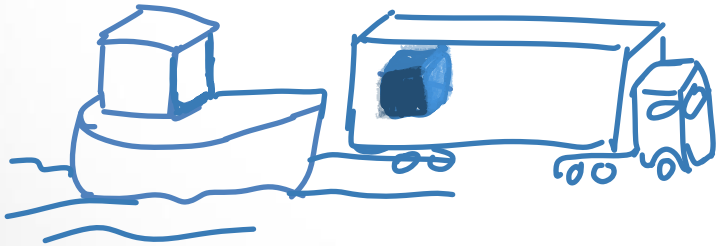


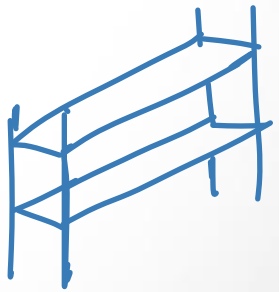


Blockchain



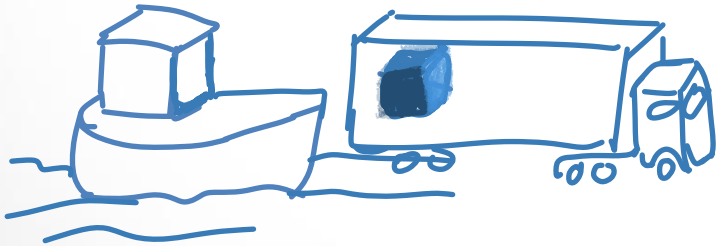
Blockchain

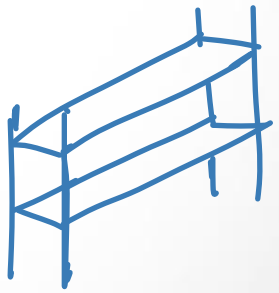




Blockchain

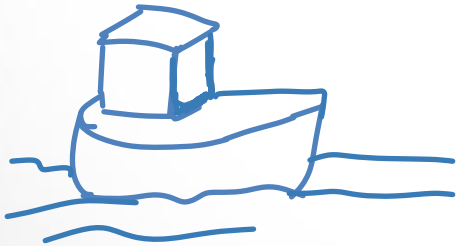
X is delivered
from A to B

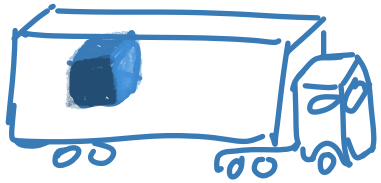


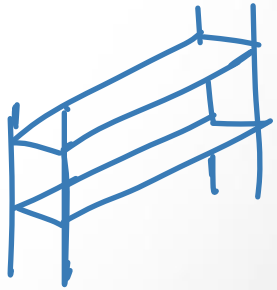


Blockchain

X is delivered
from A to B

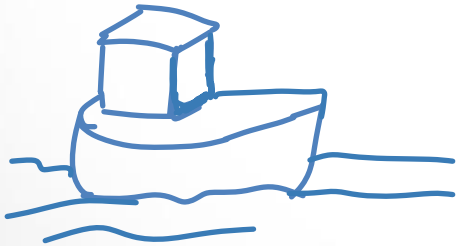


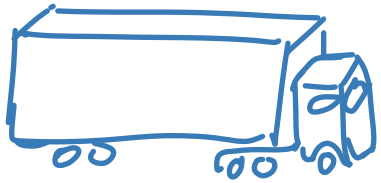




Blockchain

X is delivered
from A to B

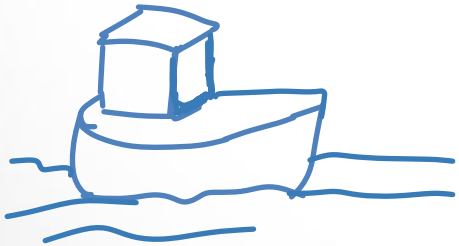


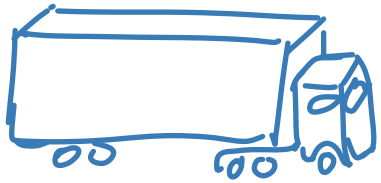


Blockchain

X is delivered
from A to B

X is delivered
from B to C



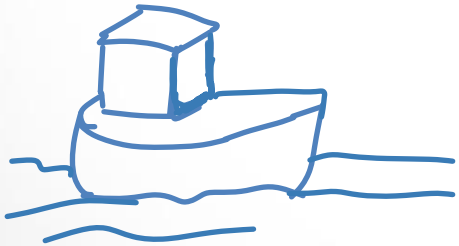


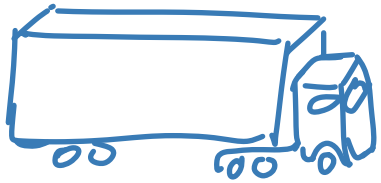
Blockchain

X is delivered from A to B

X is delivered from B to C

X is in the final product



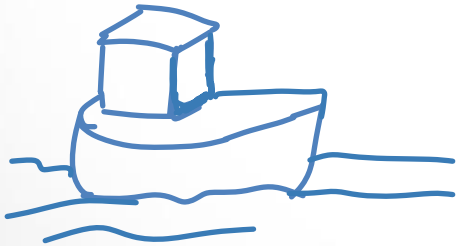


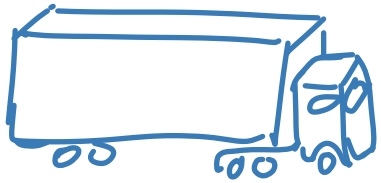
Blockchain

X is delivered from A to B

X is delivered from B to C

X is in the final product





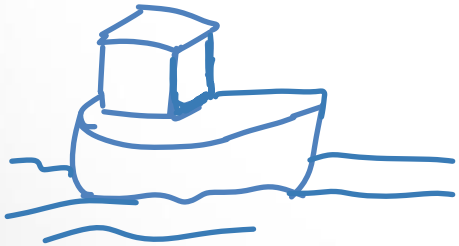
X is delivered from B to D

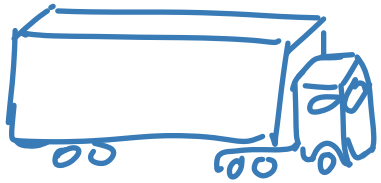
Blockchain

X is delivered from A to B

X is delivered from B to C

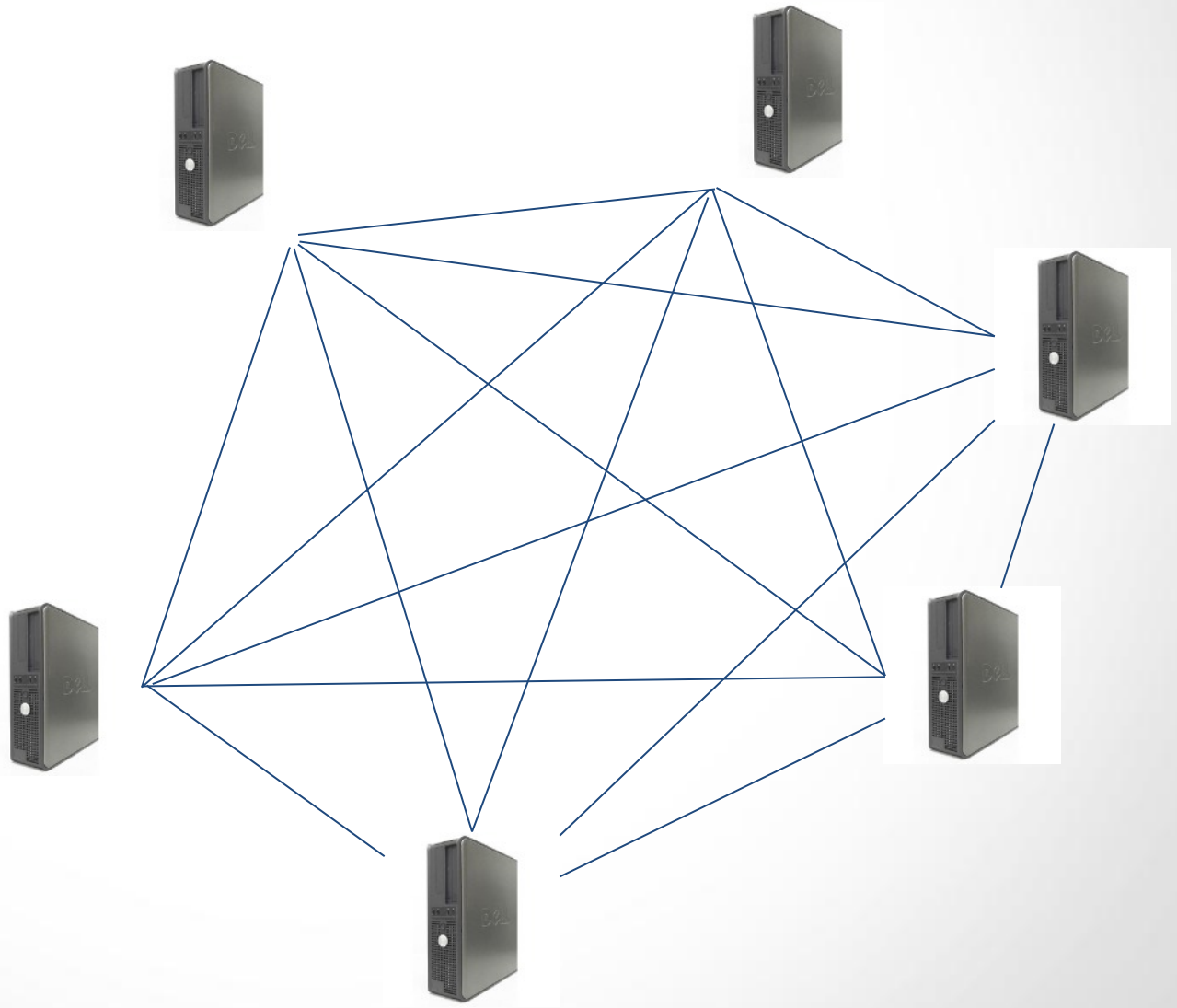
X is in the final product





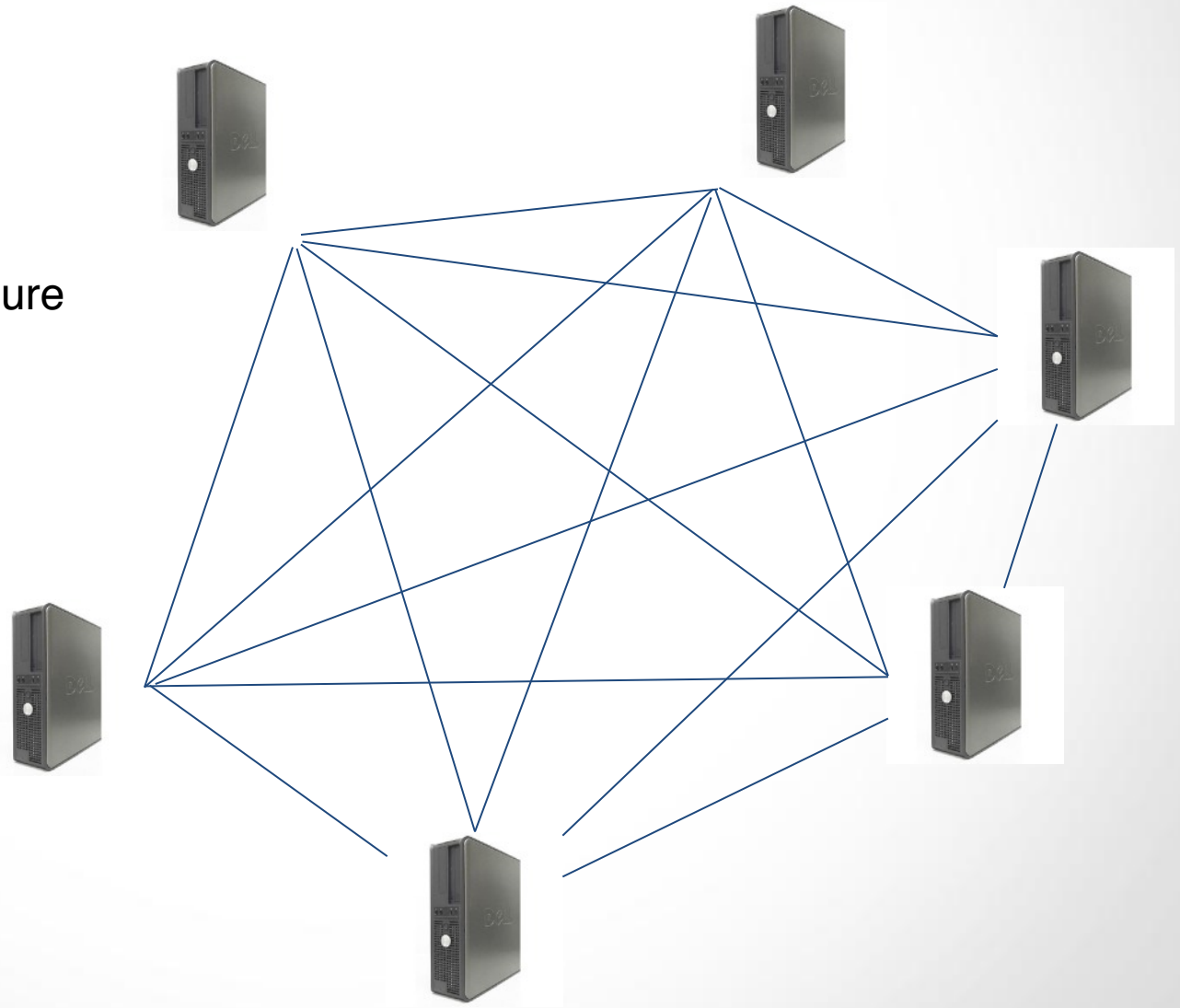
X is delivered from B to D

Data is distributed :



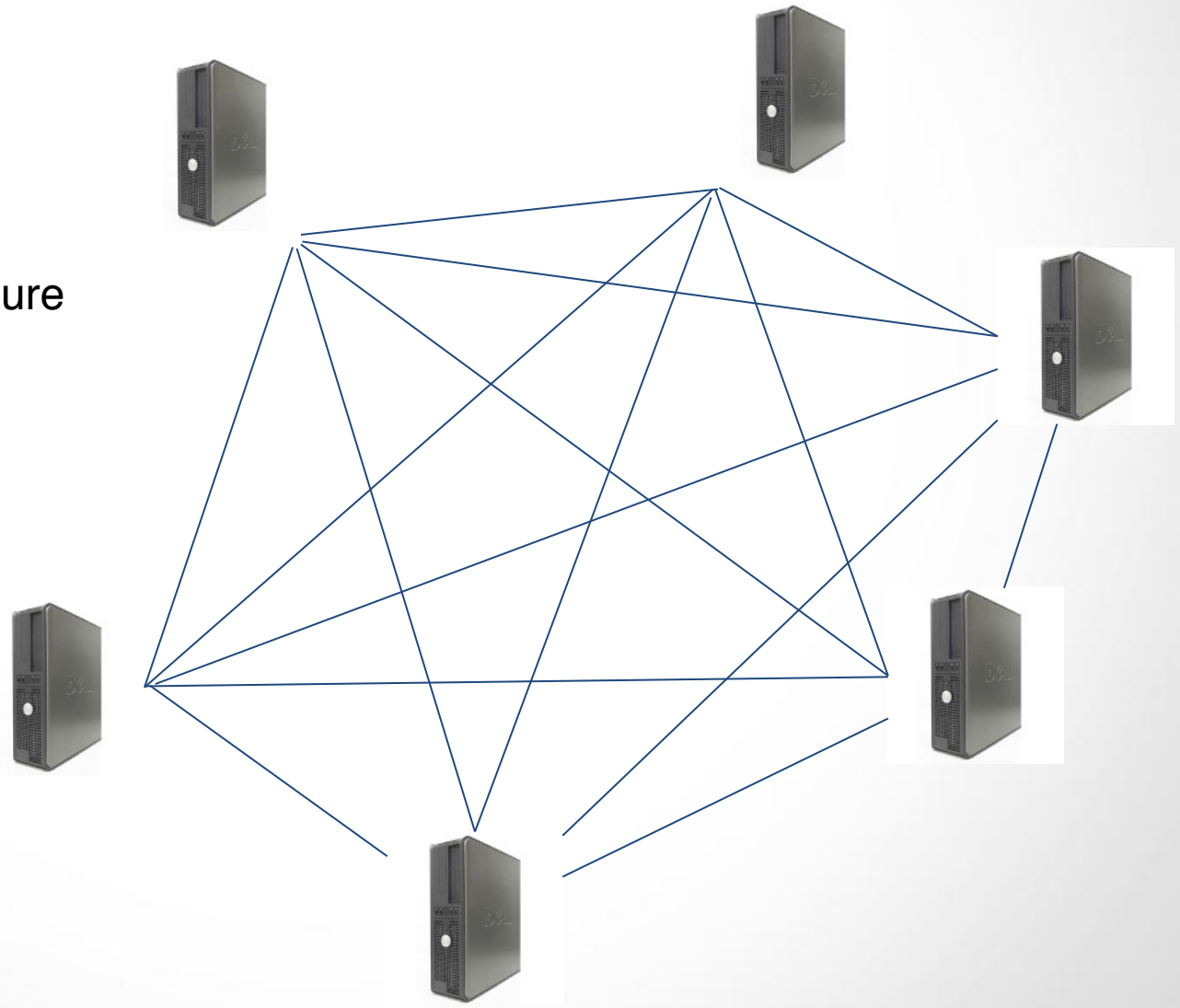
Data is distributed :

- no single point of failure



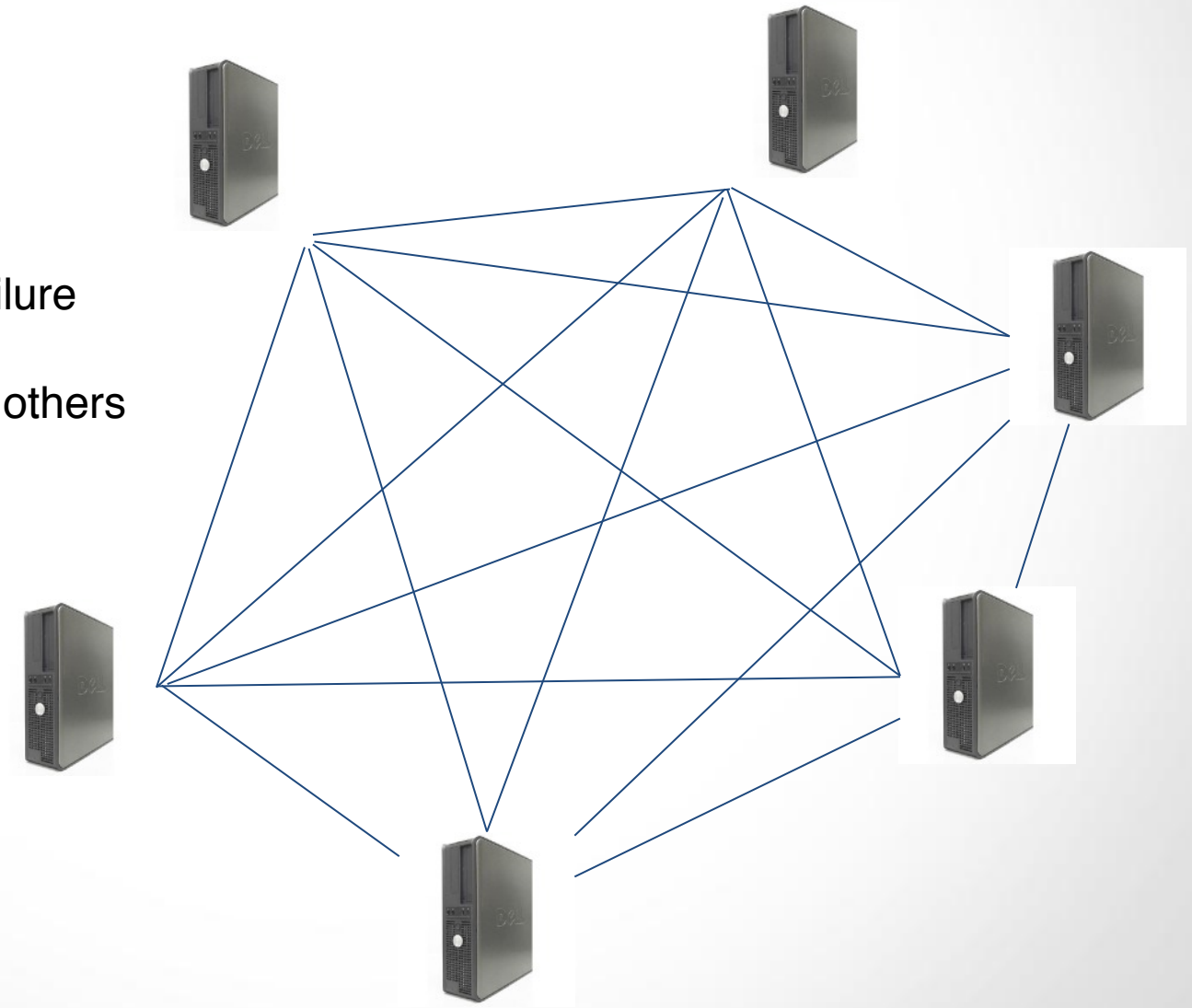
Data is distributed :

- no single point of failure
- no central Authority



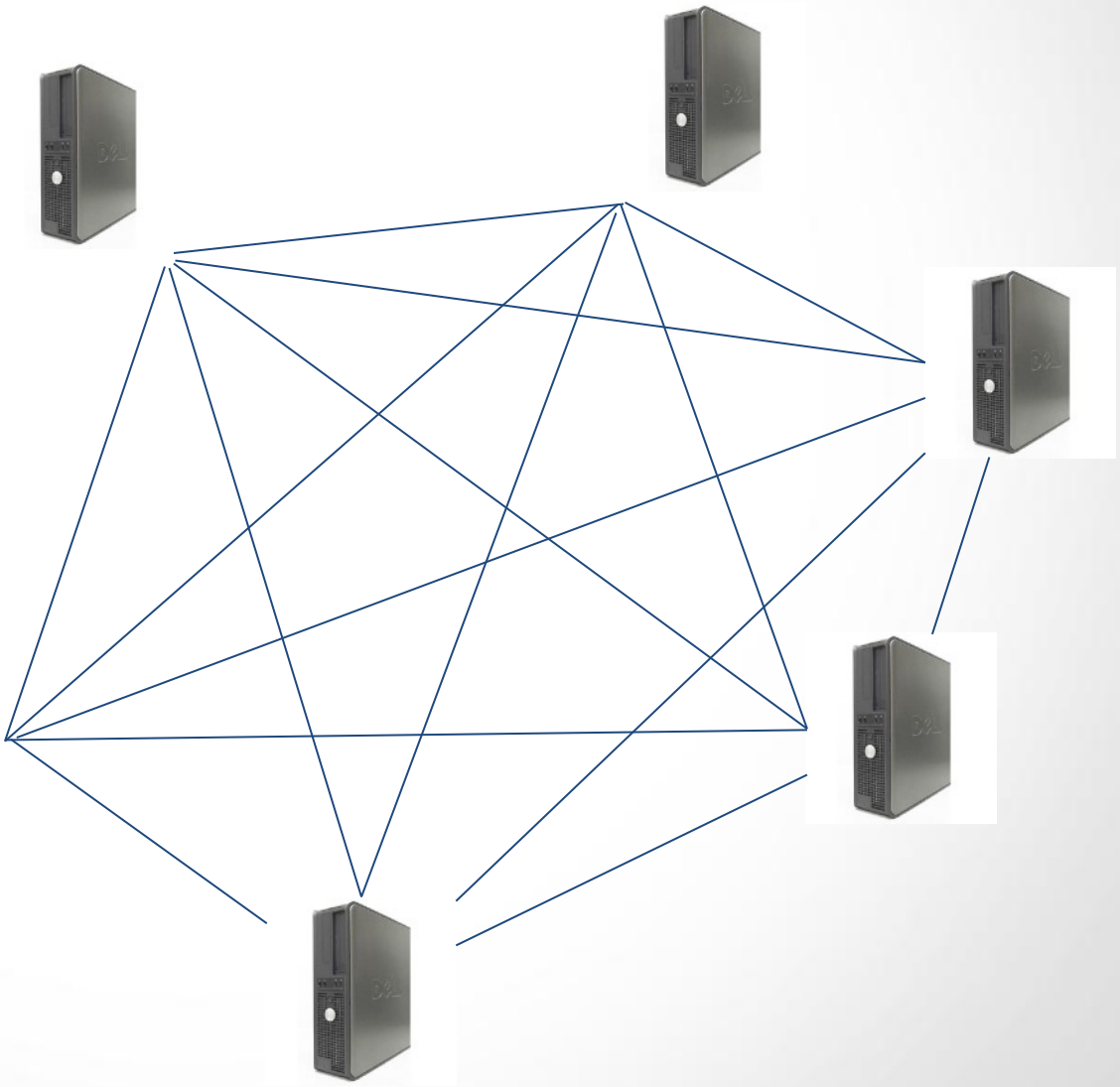
Data is distributed :

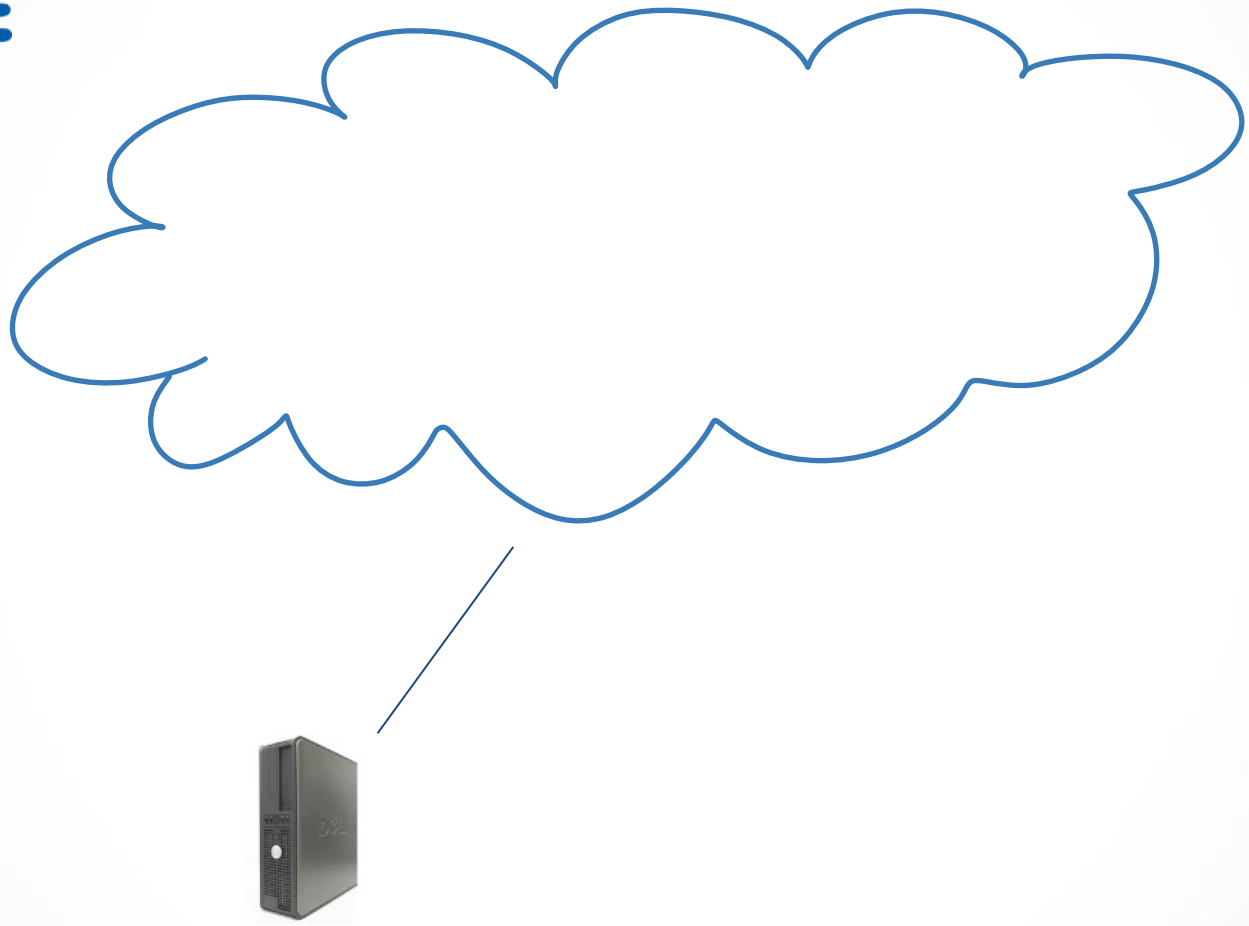
- no single point of failure
- no central Authority
- no need to trust the others

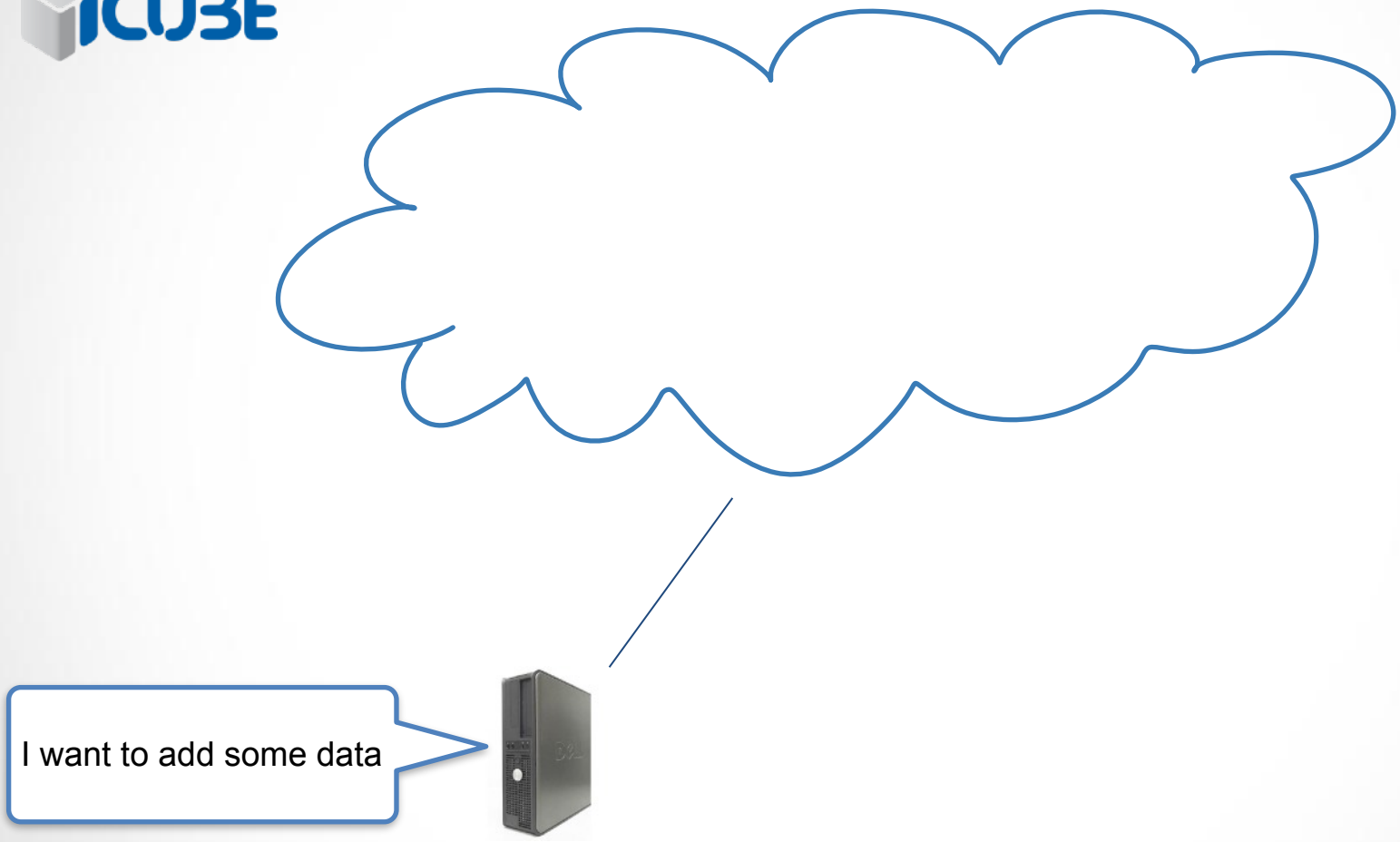


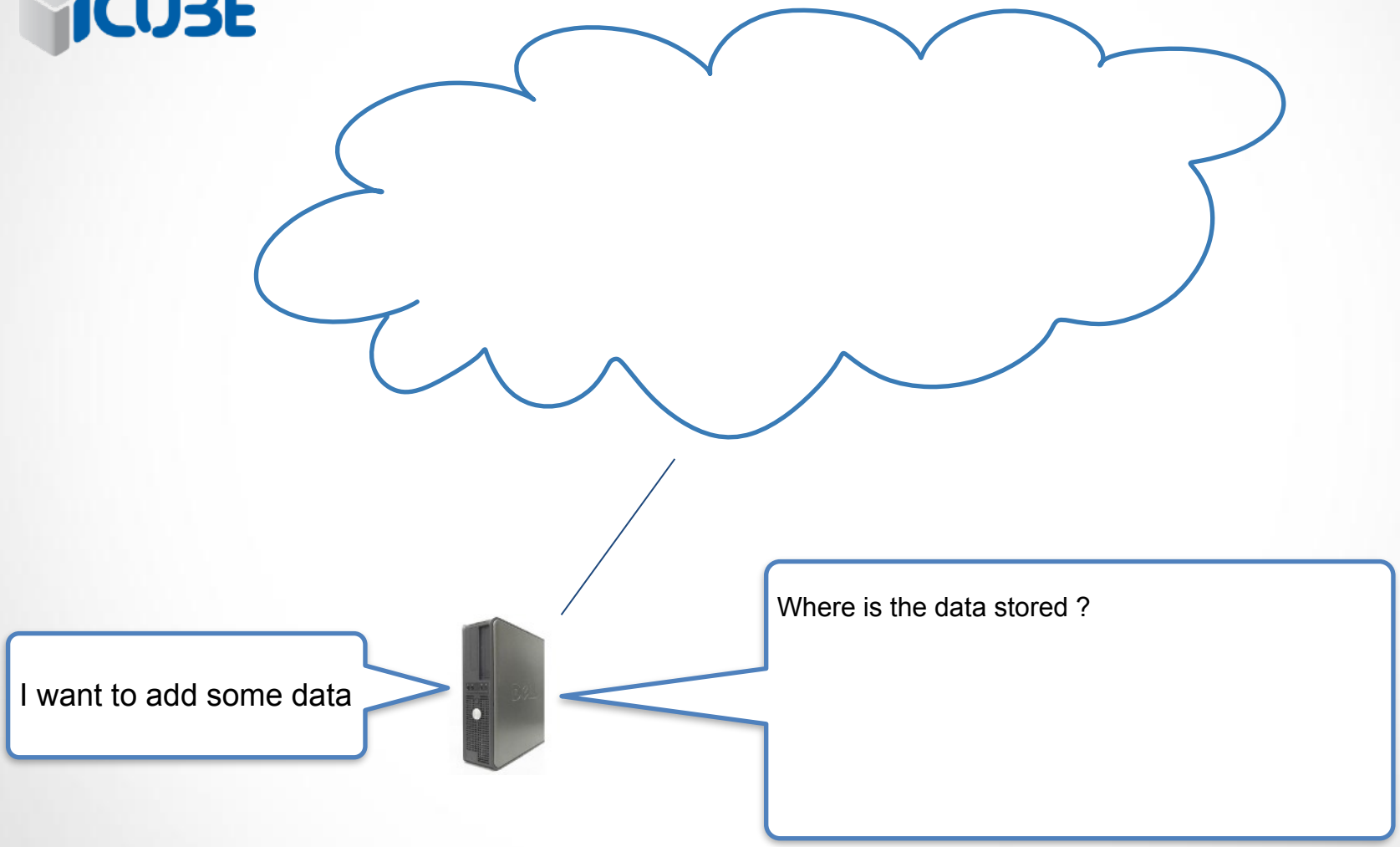
Data is distributed :

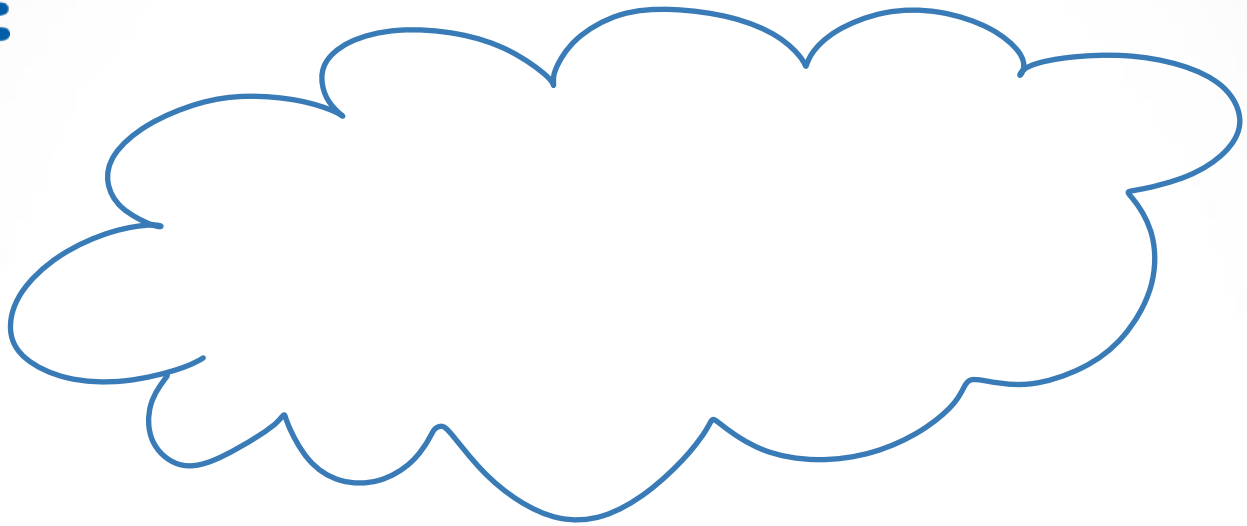
- no single point of failure
- no central Authority
- no need to trust the others





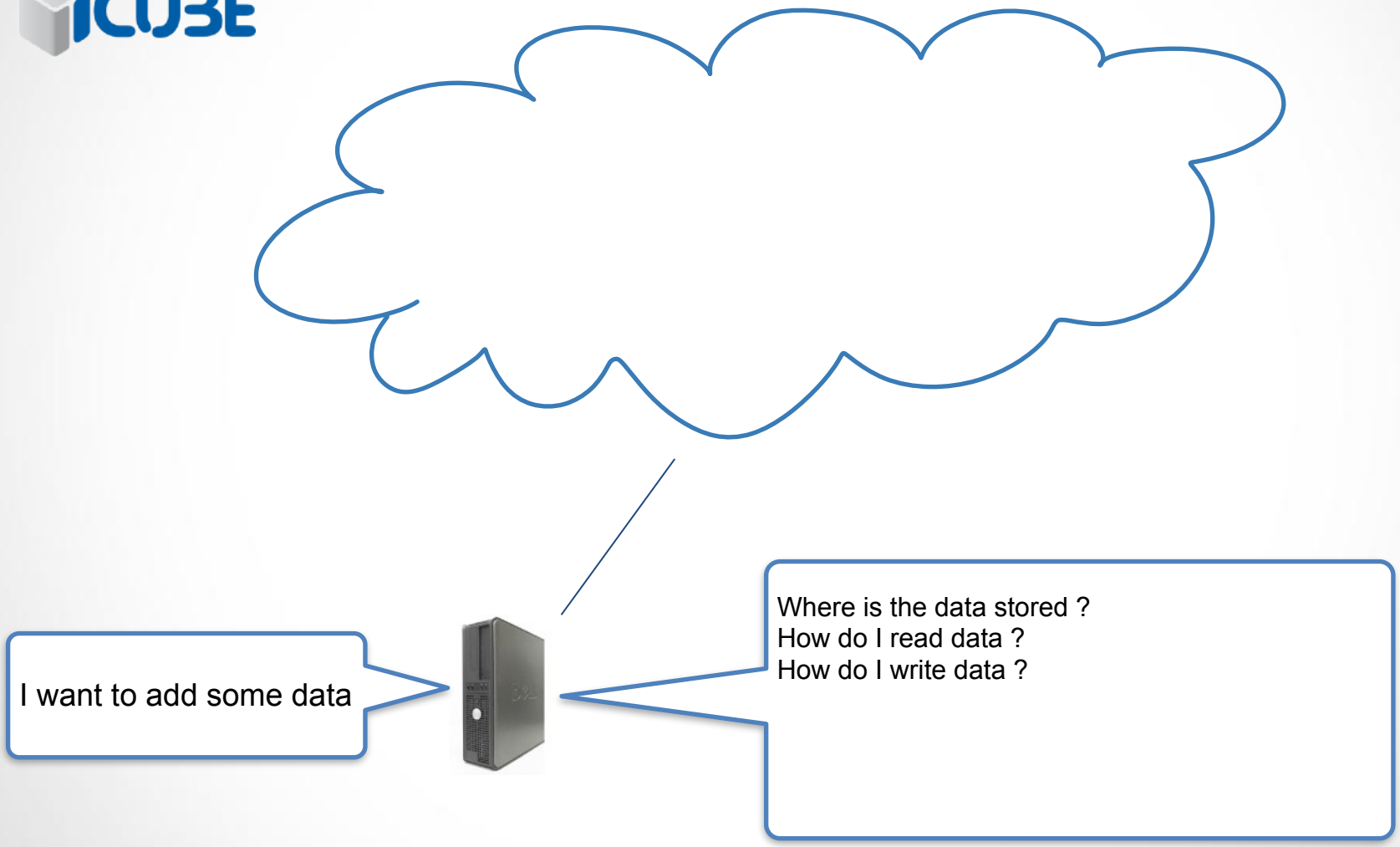

I want to add some data

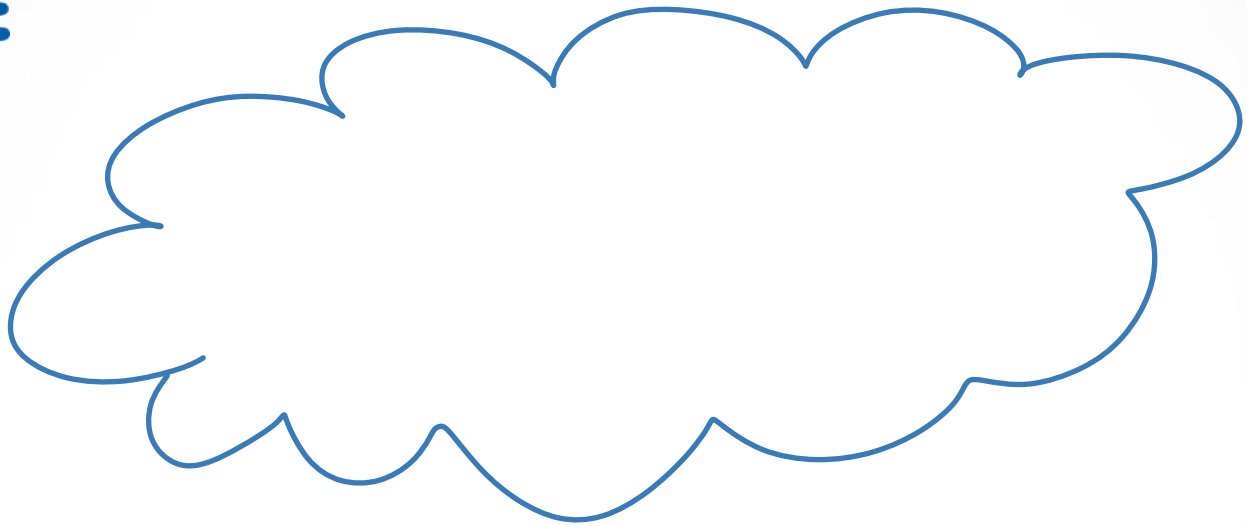

I want to add some data
Where is the data stored ?



I want to add some data

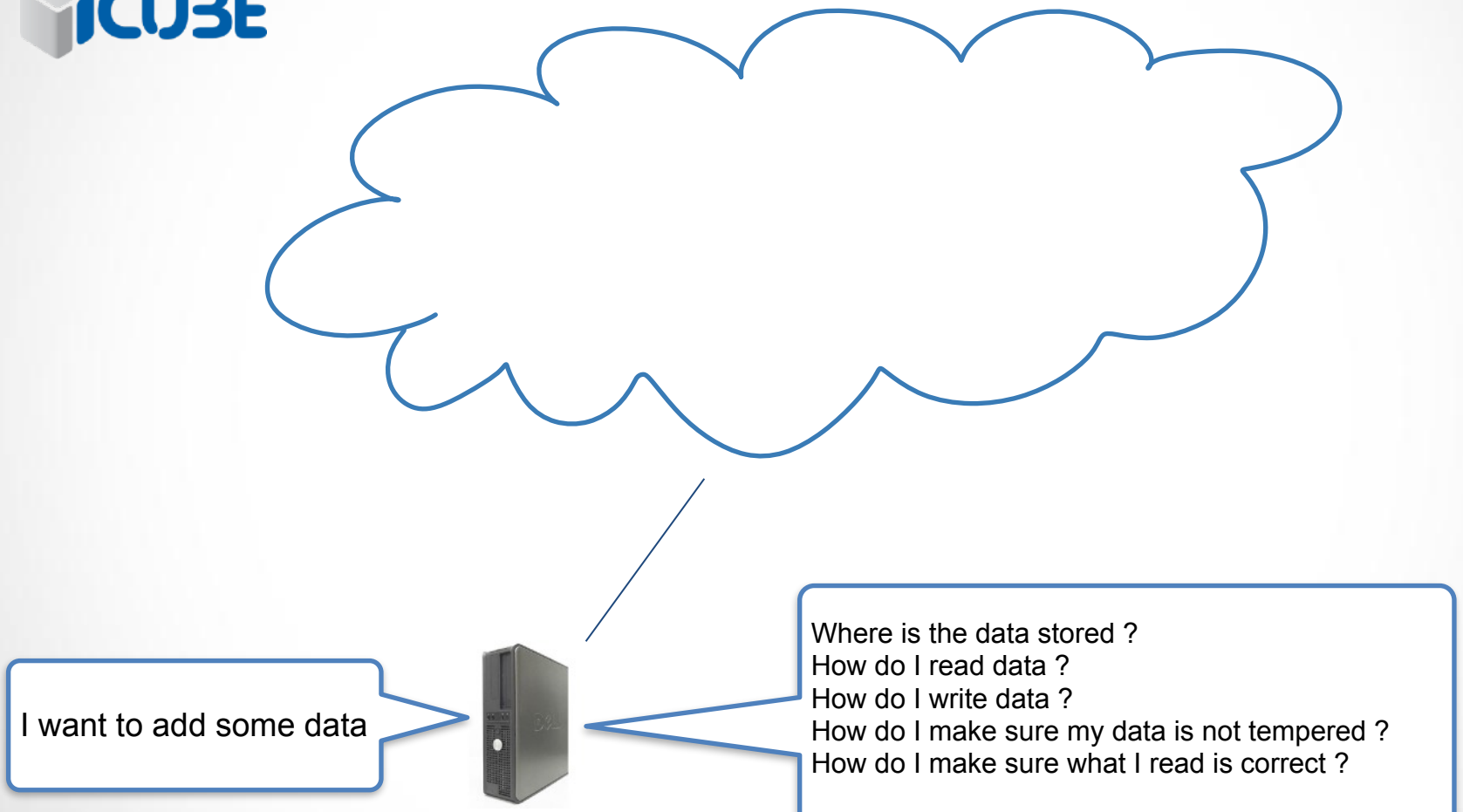
Where is the data stored ?
How do I read data ?

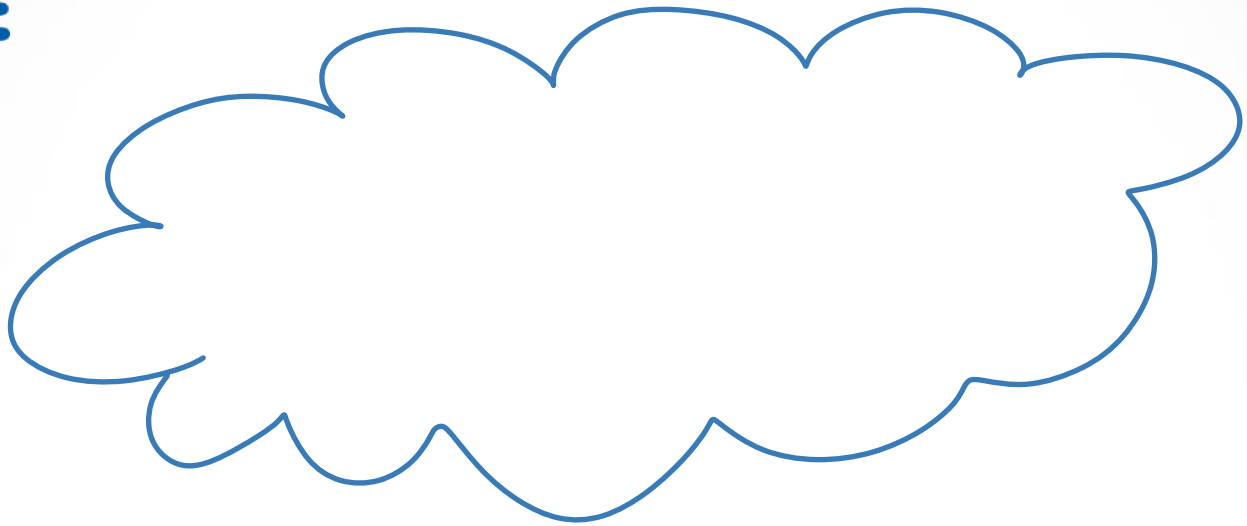

I want to add some data
Where is the data stored ?
How do I read data ?
How do I write data ?



I want to add some data

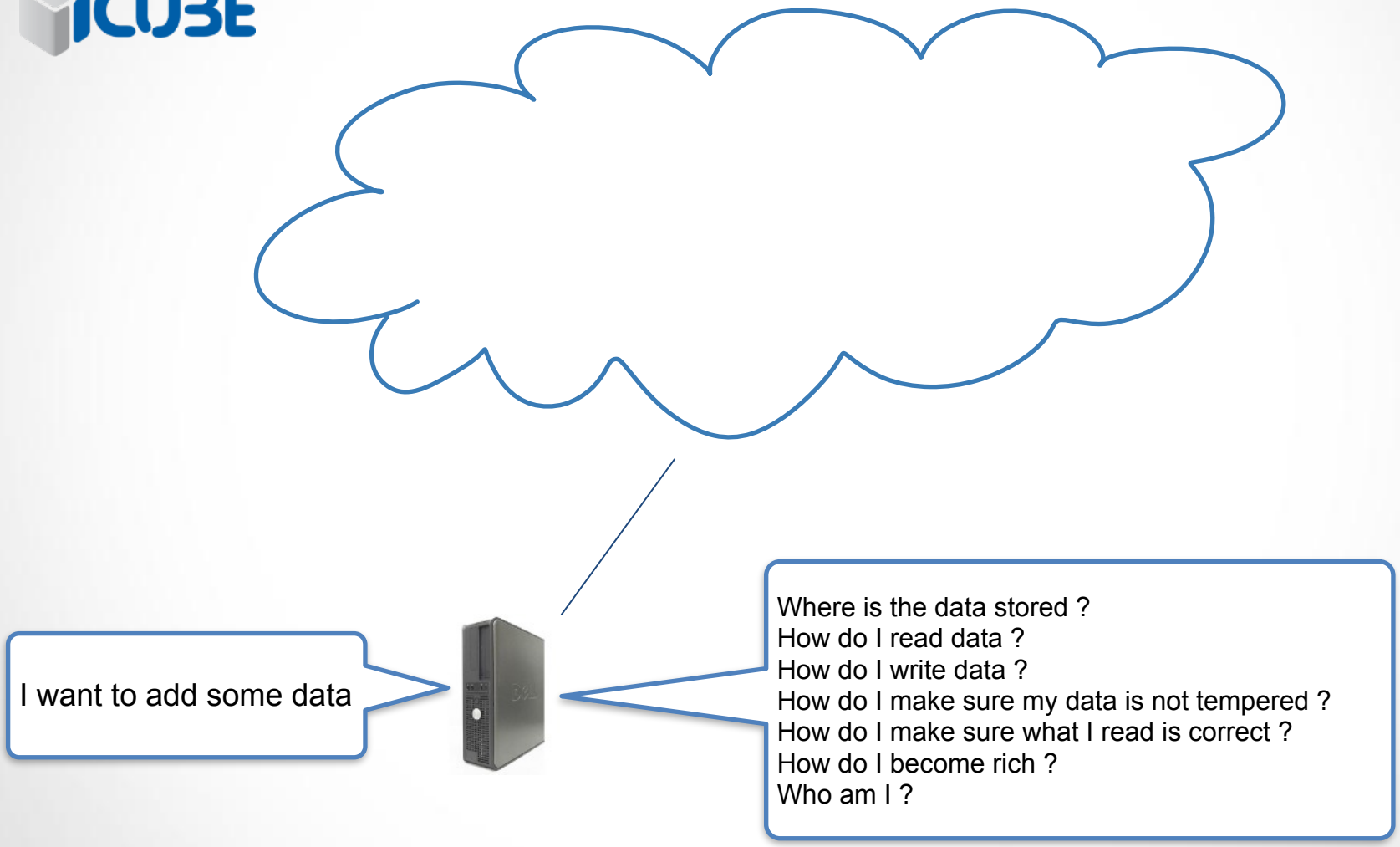
Where is the data stored ?
How do I read data ?
How do I write data ?
How do I make sure my data is not tempered ?


I want to add some data
Where is the data stored ?
How do I read data ?
How do I write data ?
How do I make sure my data is not tempered ?
How do I make sure what I read is correct ?

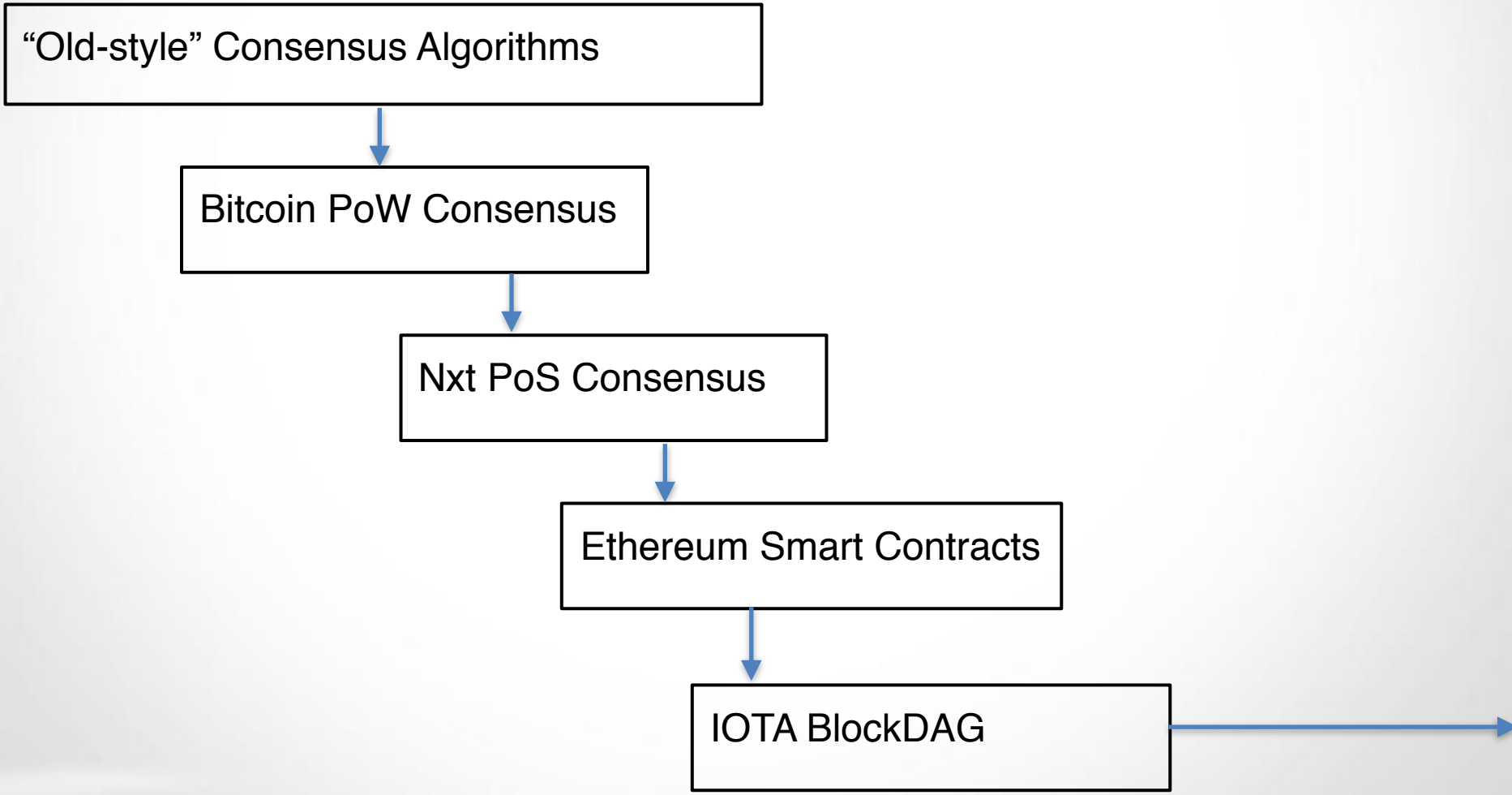


I want to add some data

Where is the data stored ?
How do I read data ?
How do I write data ?
How do I make sure my data is not tempered ?
How do I make sure what I read is correct ?
How do I become rich ?

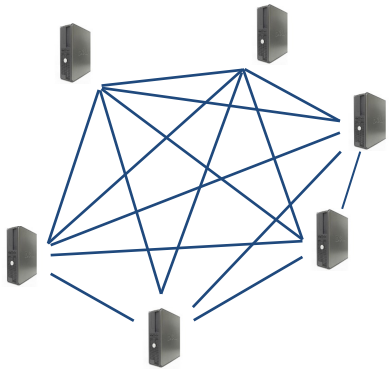

I want to add some data
Where is the data stored ?
How do I read data ?
How do I write data ?
How do I make sure my data is not tempered ?
How do I make sure what I read is correct ?
How do I become rich ?
Who am I ?

Talk Chain

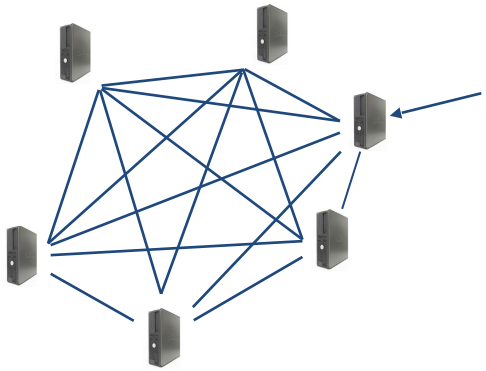

“Old-style” Consensus Algorithms
Bitcoin PoW Consensus
Nxt PoS Consensus
Ethereum Smart Contracts
IOTA BlockDAG

“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

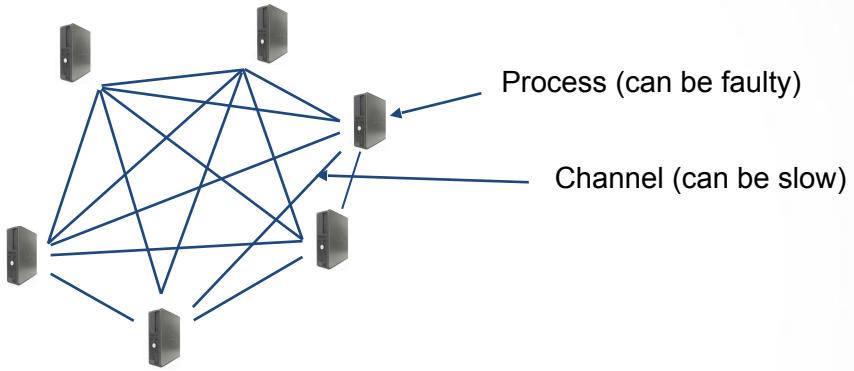
“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

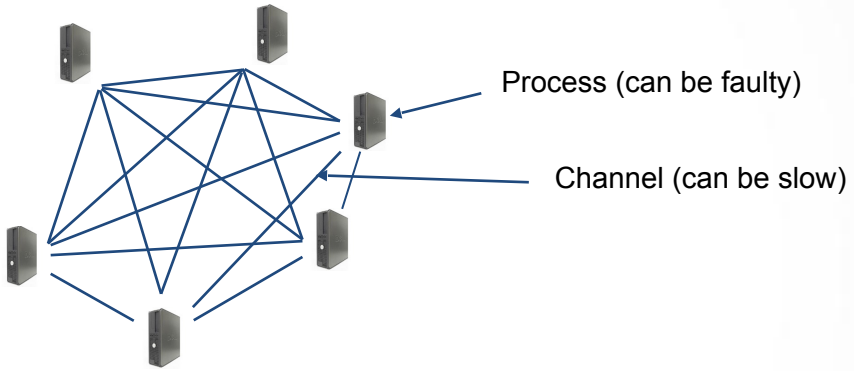


“Old-style” Consensus Algorithms → Bitcoin PoW Consensus



“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

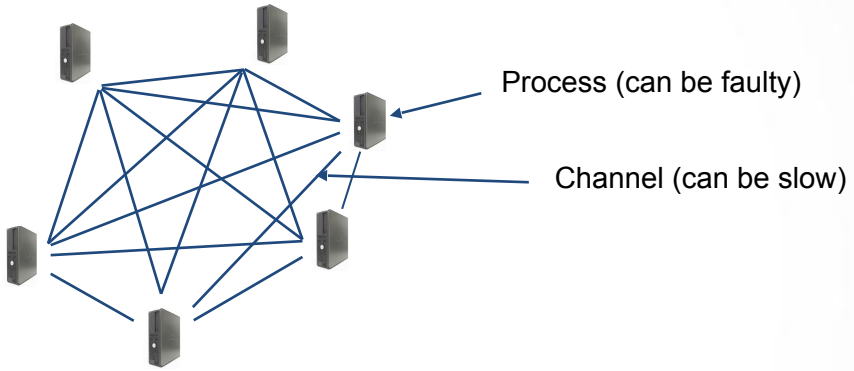




Definition of Consensus:

“Old-style” Consensus Algorithms

Bitcoin PoW Consensus

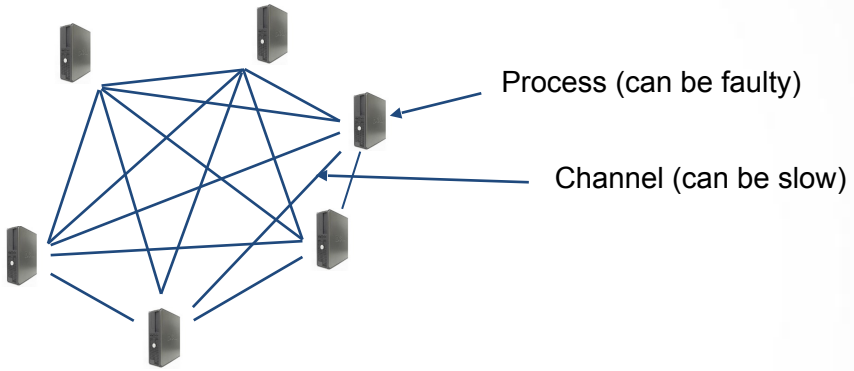


Definition of Consensus:

Each process starts with an input value

“Old-style” Consensus Algorithms

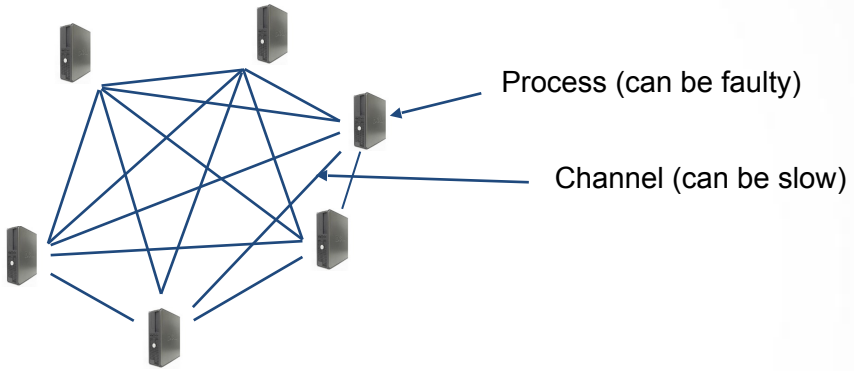
Bitcoin PoW Consensus



Definition of Consensus:

Each process starts with an input value

“Old-style” Consensus Algorithms → Bitcoin PoW Consensus



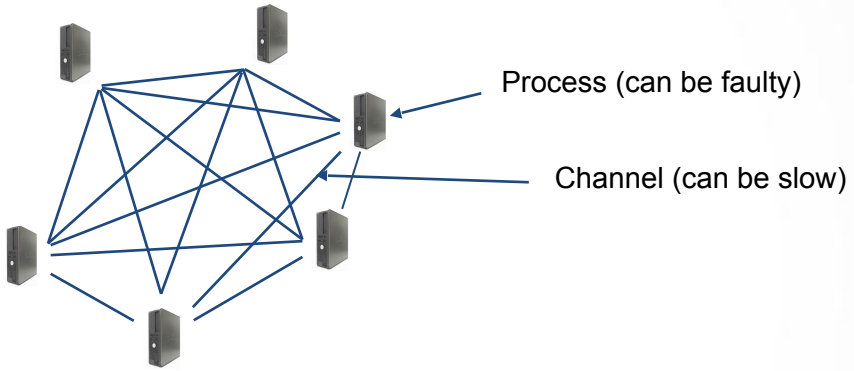
Definition of Consensus:

Each process starts with an input value

Agreement

“Old-style” Consensus Algorithms

→ Bitcoin PoW Consensus

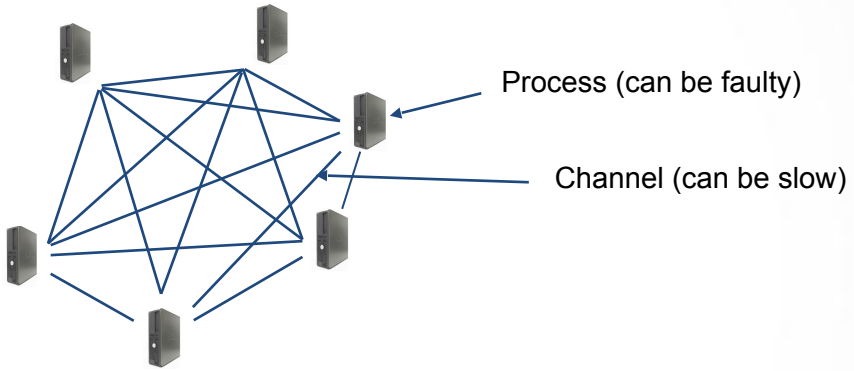


Definition of Consensus:

Each process starts with an input value

Agreement

Every correct process must agree on the same value.



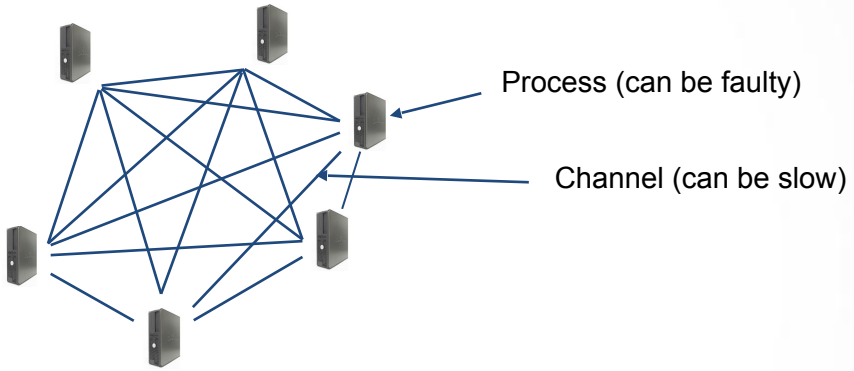
Definition of Consensus:

Each process starts with an input value

Agreement

Every correct process must agree on the same value.

Validity



Definition of Consensus:

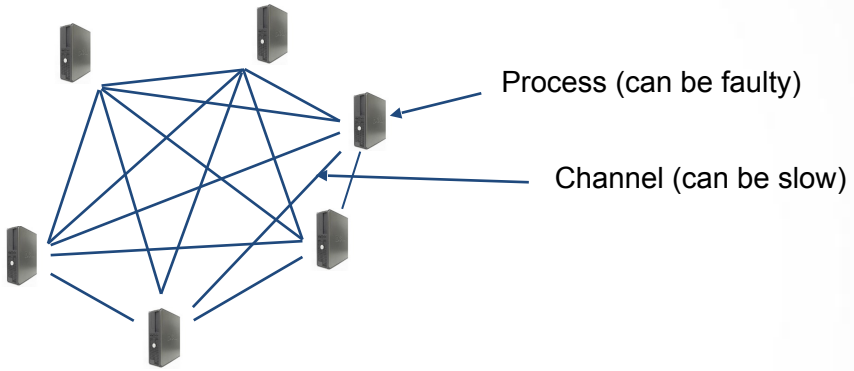
Each process starts with an input value

Agreement

Every correct process must agree on the same value.

Validity

If all processes propose the same value *v*, then all correct processes decide *v*.



Definition of Consensus:

Each process starts with an input value

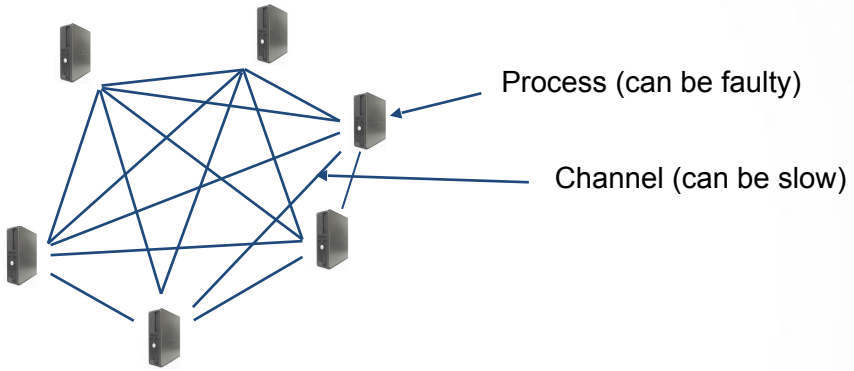
Agreement

Every correct process must agree on the same value.

Validity

If all processes propose the same value *v*, then all correct processes decide *v*.

Termination



Definition of Consensus:

Each process starts with an input value

Agreement

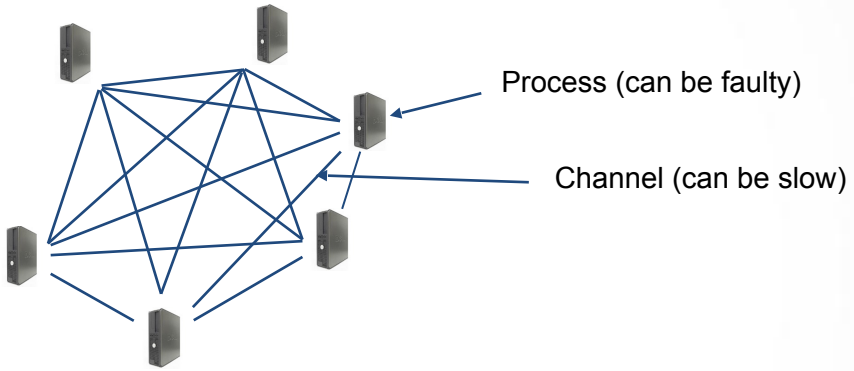
Every correct process must agree on the same value.

Validity

If all processes propose the same value *v*, then all correct processes decide *v*.

Termination

Every correct process decides some value.



Definition of Consensus:

Each process starts with an input value

Agreement

Every correct process must agree on the same value.

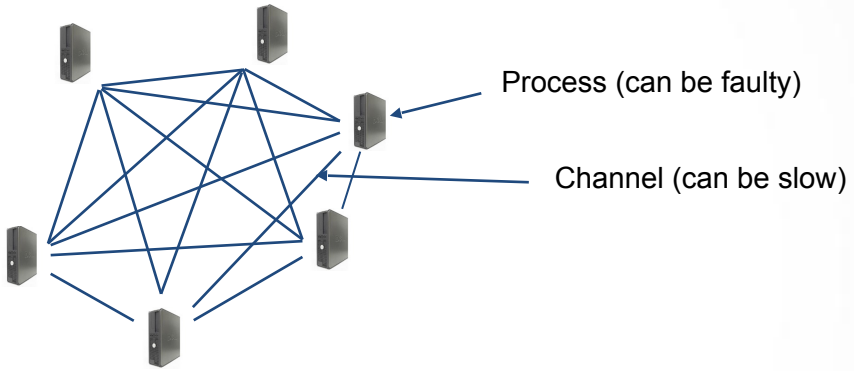
Validity

If all processes propose the same value *v*, then all correct processes decide *v*.

Termination

Every correct process decides some value.

Integrity



Definition of Consensus:

Each process starts with an input value

Agreement
Every correct process must agree on the same value.

Validity
If all processes propose the same value *v*, then all correct processes decide *v*.

Termination
Every correct process decides some value.

Integrity
Every correct process decides at most one value, and if it decides some value *v*, then *v* must have been proposed by some process.

Bitcoin PoW Consensus

Impossibility results

Impossibility results [FLP 1985] "Impossibility of distributed consensus with one faulty process"

Impossibility results [FLP 1985] "Impossibility of distributed consensus with one faulty process"

Costly algorithms

“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

Impossibility results	[FLP 1985] "Impossibility of distributed consensus with one faulty process"
Costly algorithms	Every node should be connected to every node, broadcast everything, and waits for n/2 responses

“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

Impossibility results	[FLP 1985] "Impossibility of distributed consensus with one faulty process"
Costly algorithms	Every node should be connected to every node, broadcast everything, and waits for n/2 responses
Still puzzle researchers’ minds	

“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

Impossibility results	[FLP 1985] "Impossibility of distributed consensus with one faulty process"
Costly algorithms	Every node should be connected to every node, broadcast everything, and waits for n/2 responses
Still puzzle researchers’ minds	[Bramas 2016] Packet efficient implementation of the **omega** failure detector

“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

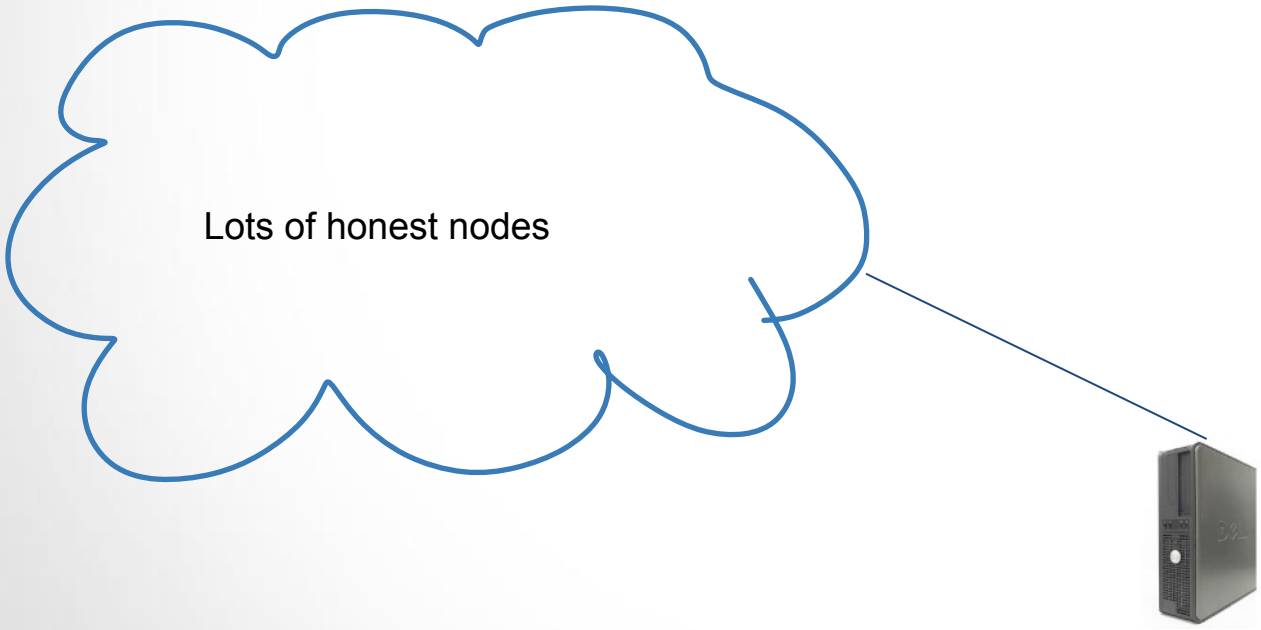
Impossibility results	[FLP 1985] "Impossibility of distributed consensus with one faulty process" [CAP theorem] "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services"
Costly algorithms	Every node should be connected to every node, broadcast everything, and waits for n/2 responses
Still puzzle researchers’ minds	[Bramas 2016] Packet efficient implementation of the **omega** failure detector

“Old-style” Consensus Algorithms → Bitcoin PoW Consensus

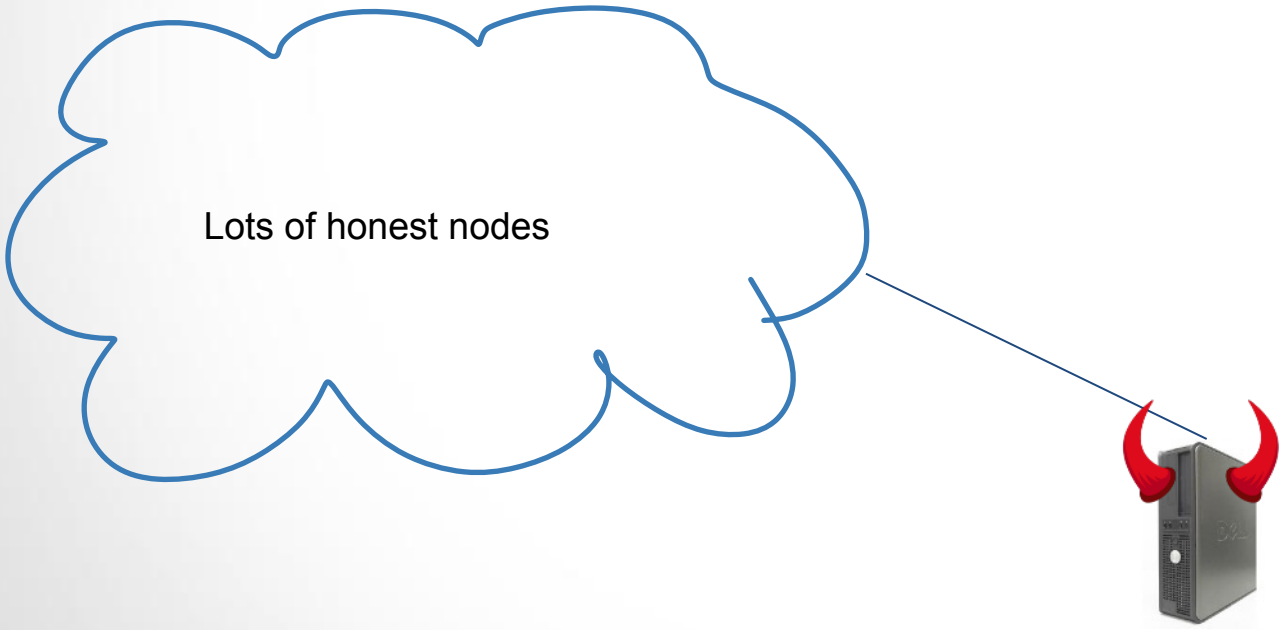
Impossibility results	[FLP 1985] "Impossibility of distributed consensus with one faulty process" [CAP theorem] "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services"
Costly algorithms	Every node should be connected to every node, broadcast everything, and waits for n/2 responses
Still puzzle researchers’ minds	[Bramas 2016] Packet efficient implementation of the **omega** failure detector

There exists consensus algorithms that tolerates up to n/3 Byzantins processes

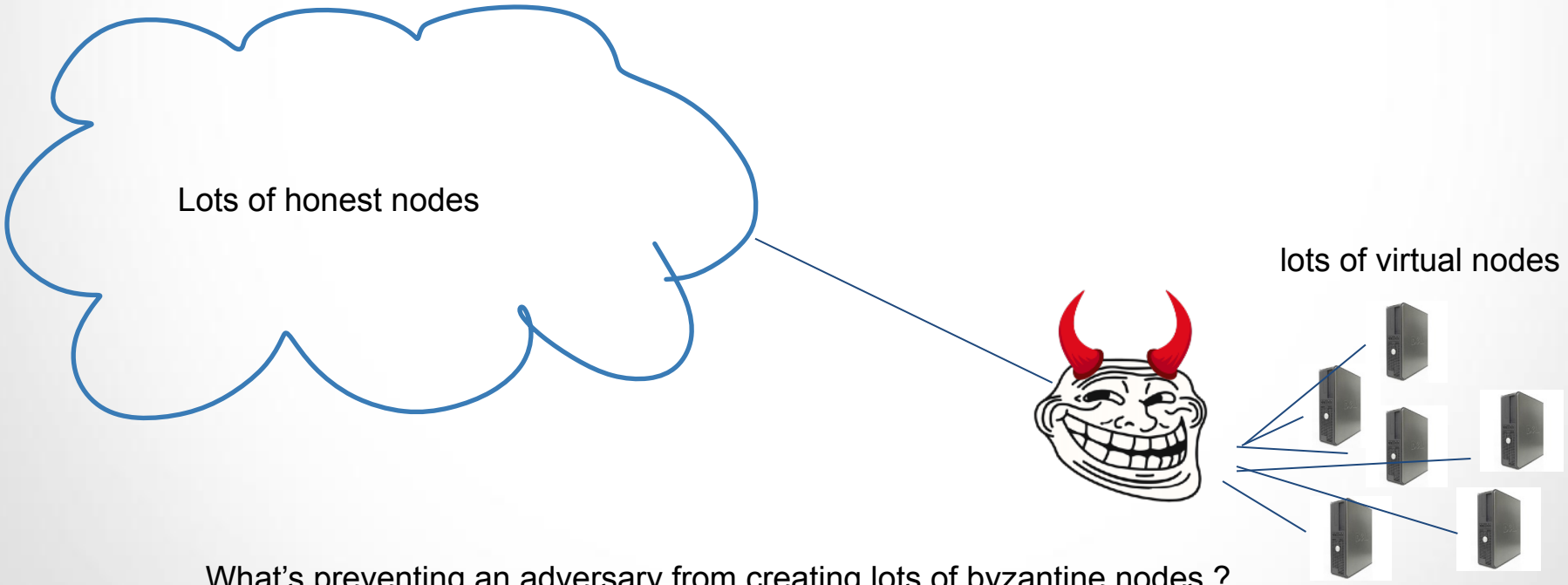
So what’s the problem with an algorithm tolerating n/3 byzantin nodes ?



So what’s the problem with an algorithm tolerating n/3 byzantin nodes ?

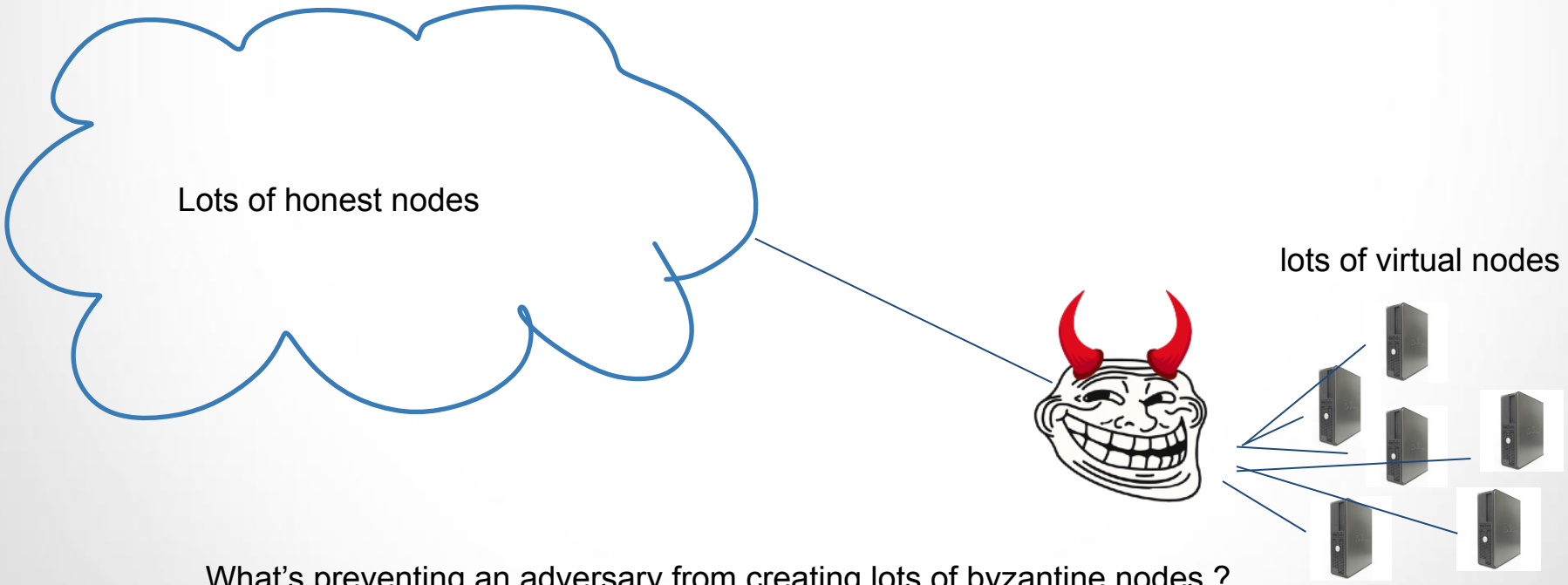


So what’s the problem with an algorithm tolerating n/3 byzantin nodes ?



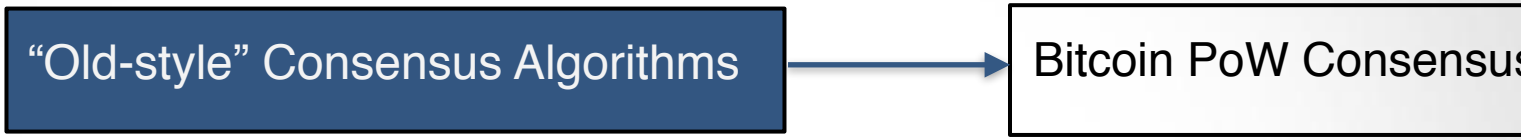
What’s preventing an adversary from creating lots of byzantine nodes ?

So what’s the problem with an algorithm tolerating n/3 byzantin nodes ?



What’s preventing an adversary from creating lots of byzantine nodes ?

“old-style” consensus algorithms work well when the participants are known, and not too many (eg, Hashgraph blockchain, 39 known participants in the “council”)


“Old-style” Consensus Algorithms
Bitcoin PoW Consensus

We need some kind of

We need some kind of **Proof**

We need some kind of **Proof of “existence”**

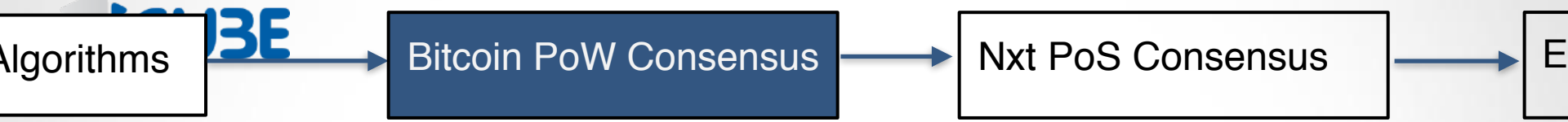
We need some kind of **Proof of “existence”**

that does not depend on how many nodes you control in the network

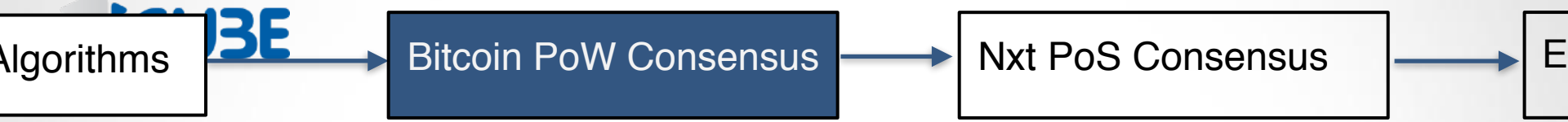
We need some kind of **Proof of “existence”**

that does not depend on how many nodes you control in the network

but depends on ...



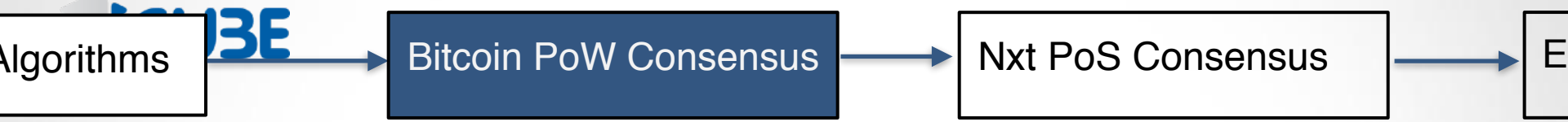
How much work can you do ?

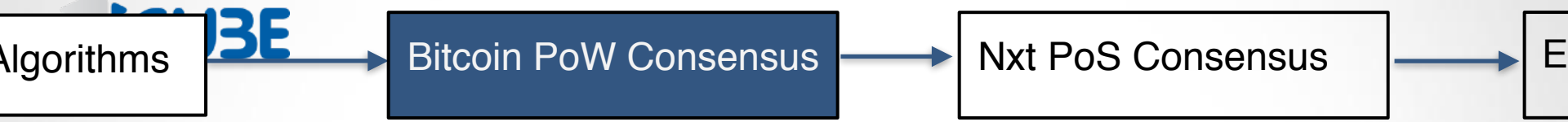


How much work can you do ?

Your importance in the network depends on how many times you can execute a hash function.

(no interest in simulating many nodes, because your hashing power will remain the same)


Algorithms
Bitcoin PoW Consensus
Nxt PoS Consensus
E



Basic principle of the Bitcoin Protocol :

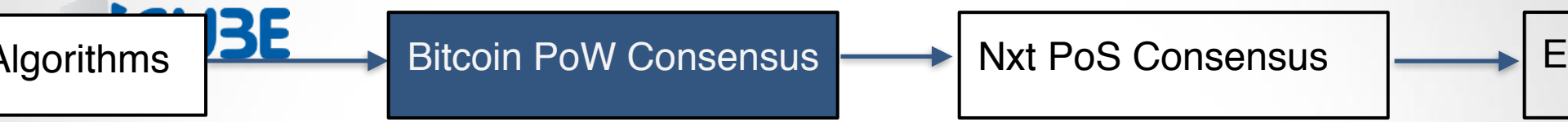
Basic principle of the Bitcoin Protocol :

- Choose randomly one node

Basic principle of the Bitcoin Protocol :

- Choose randomly one node

The more computing power, the more chance you have to be selected



Basic principle of the Bitcoin Protocol :

- Choose randomly one node
- This node decides what to write in the database

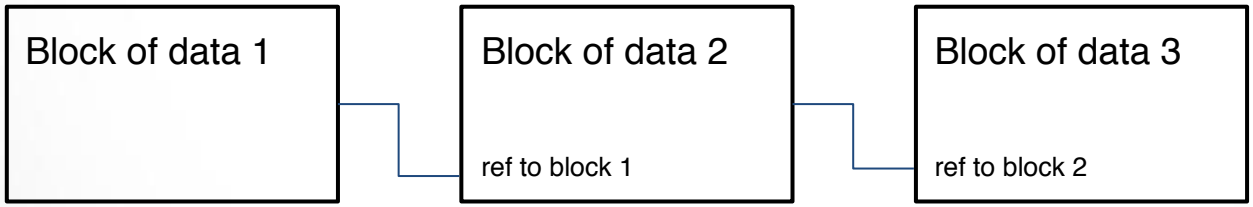
The more computing power, the more chance you have to be selected

In more details

What is the database ?
The blockchain

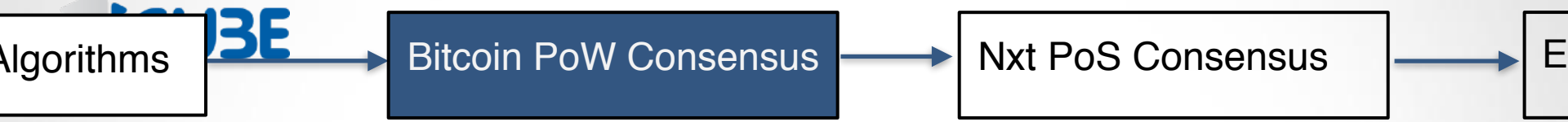
In more details

What is the database ?
The blockchain



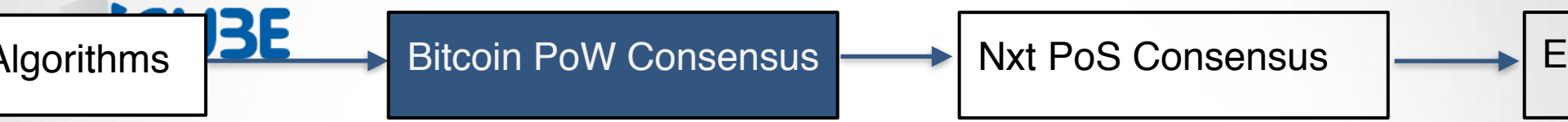
It's a data structure with one function: append(Block)

(you cannot remove a block, so to "remove" a data just overwrite it)



In more details

Who stores the data? Every nodes



In more details

Who stores the data? Every nodes

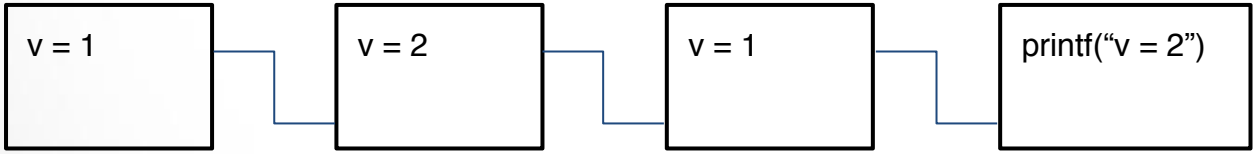
How to read data? just read all the blocks in order

In more details

Who stores the data? Every nodes

How to read data? just read all the blocks in order

Exemple 1: what's the value of v ?

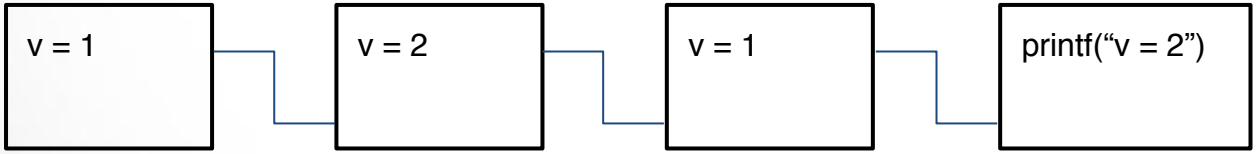


In more details

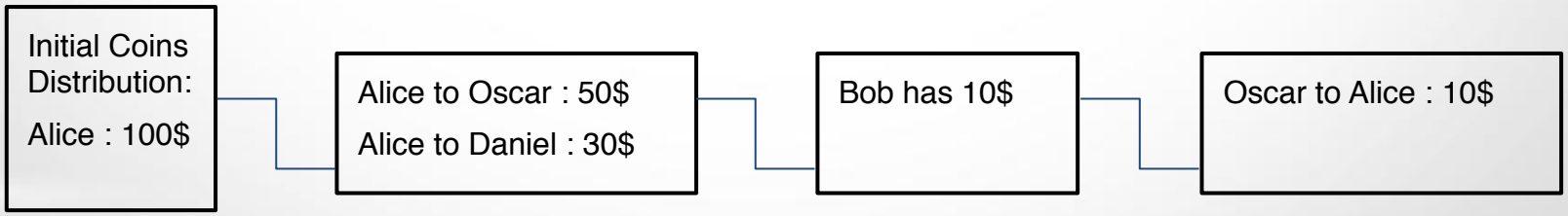
Who stores the data? Every nodes

How to read data? just read all the blocks in order

Exemple 1: what's the value of v ?



Exemple 2: How much does Alice have ? and Bob ?



In more details

How to write data ?

In more details

How to write data ?

- Send my data to everyone

In more details

How to write data ?

- Send my data to everyone
- The node that will be elected will include it in its block (hopefully)

In more details

How to write data ?

- Send my data to everyone
- The node that will be elected will include it in its block (hopefully)

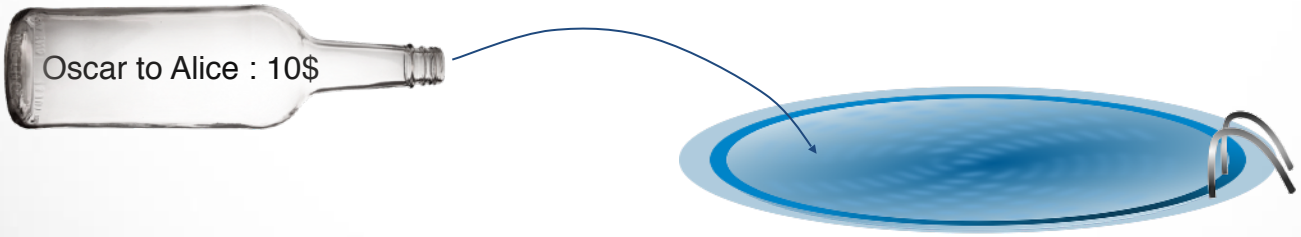
Exemple : Oscar to Alice 10$

In more details

How to write data ?

- Send my data to everyone
- The node that will be elected will include it in its block (hopefully)

Exemple : Oscar to Alice 10$



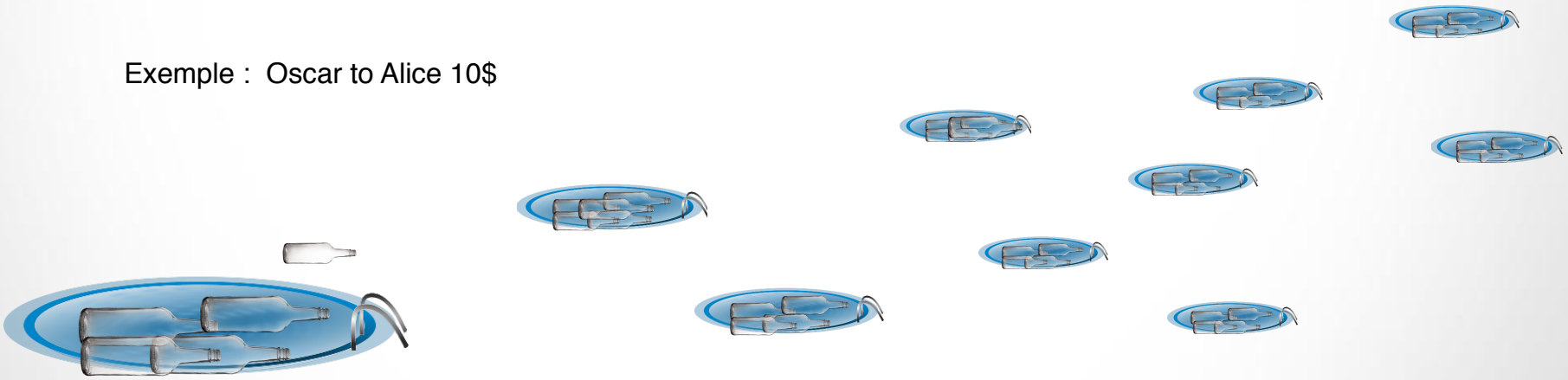
https://blockchain.info/unconfirmed-transactions

In more details

How to write data ?

- Send my data to everyone
- The node that will be elected will include it in its block (hopefully)

Exemple : Oscar to Alice 10$



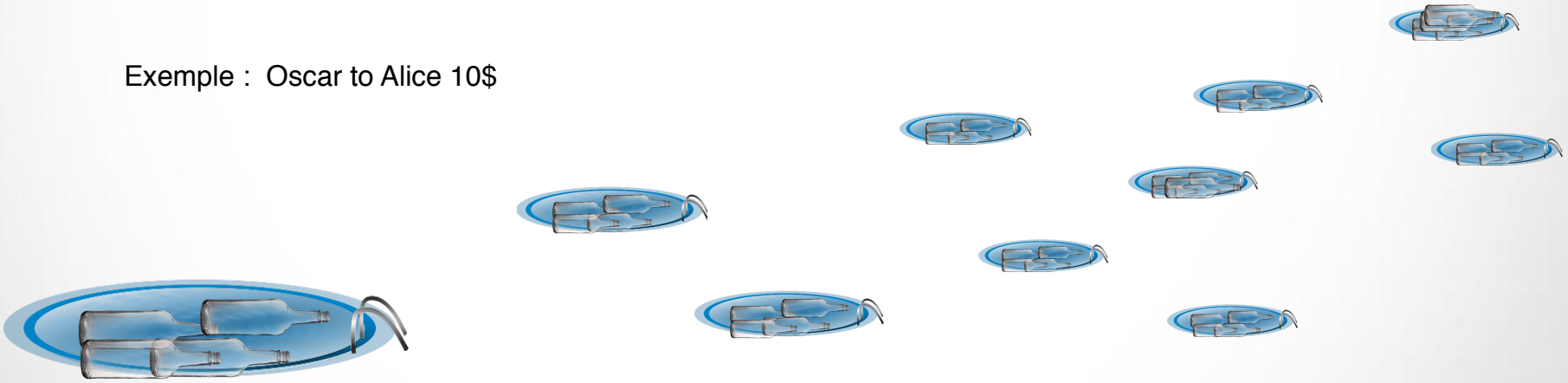
https://blockchain.info/unconfirmed-transactions

In more details

How to write data ?

- Send my data to everyone
- The node that will be elected will include it in its block (hopefully)

Exemple : Oscar to Alice 10$



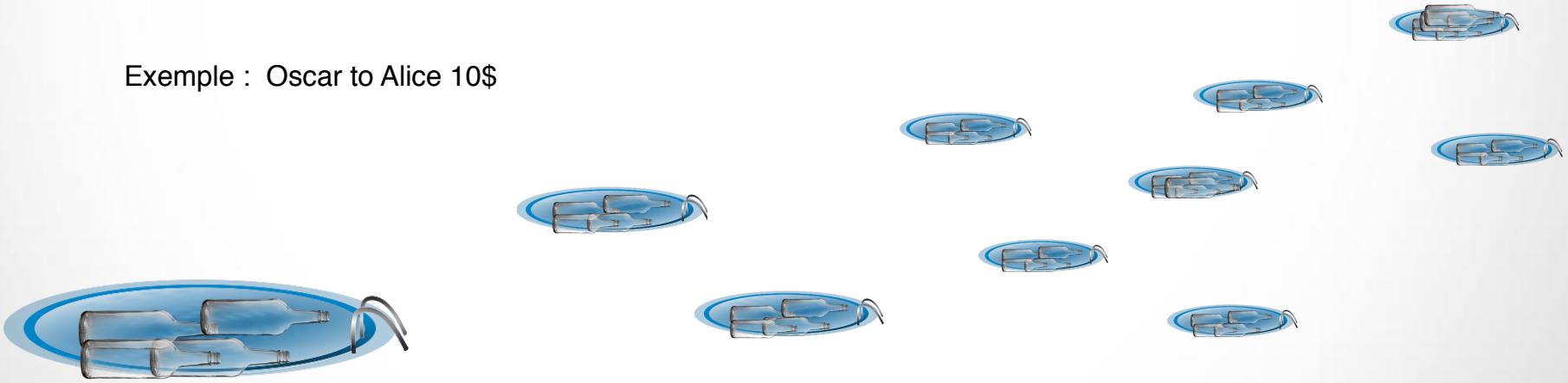
https://blockchain.info/unconfirmed-transactions

In more details

How to write data ?

- Send my data to everyone
- The node that will be elected will include it in its block (hopefully)

Exemple : Oscar to Alice 10$



https://blockchain.info/unconfirmed-transactions

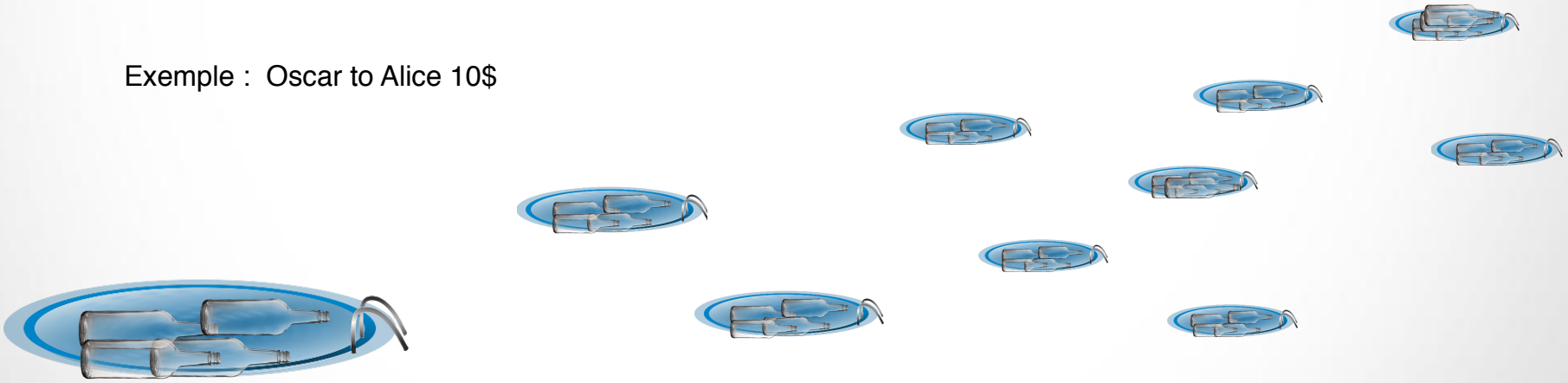
Each node includes “Oscar to Alice : 10$” to their block

In more details

How to write data ?

- Send my data to everyone
- The node that will be elected will include it in its block (hopefully)

Exemple : Oscar to Alice 10$



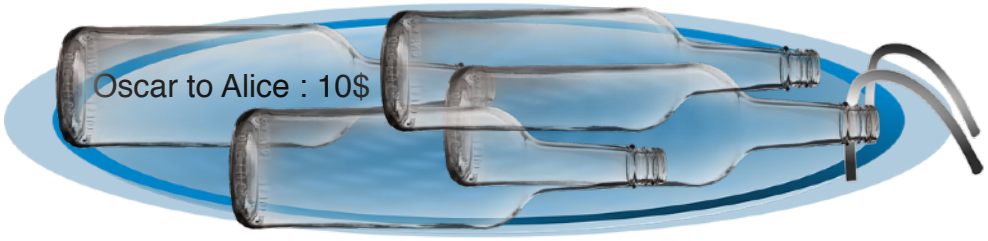
https://blockchain.info/unconfirmed-transactions

Each node includes “Oscar to Alice : 10$” to their block

One Block is “randomly” selected to be appended to the blockchain

In more details

Exemple : Oscar to Alice 10$



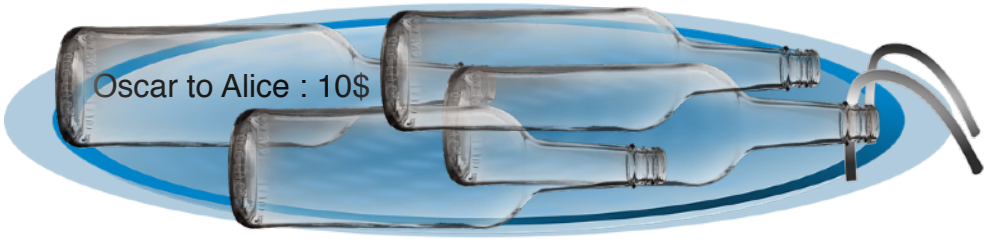
https://blockchain.info/unconfirmed-transactions

Each node include “Oscar to Alice : 10$” to their block

One Block is “randomly” selected to be appended to the blockchain

In more details

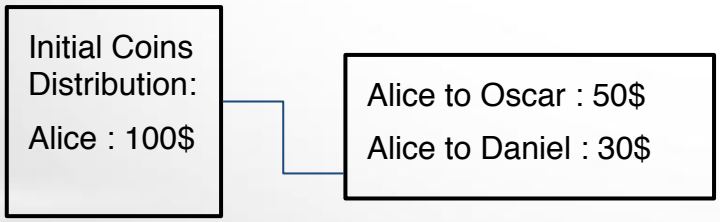
Exemple : Oscar to Alice 10$



https://blockchain.info/unconfirmed-transactions

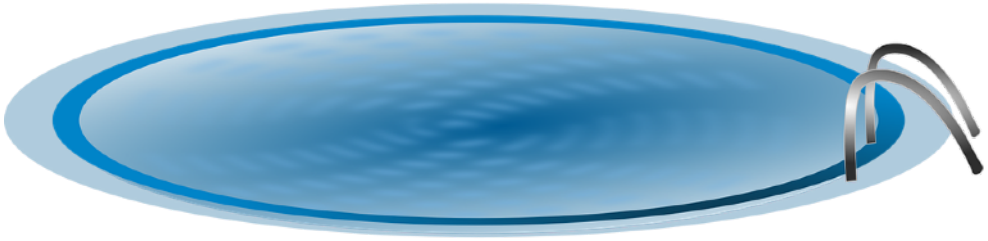
Each node include “Oscar to Alice : 10$” to their block

One Block is “randomly” selected to be appended to the blockchain



In more details

Exemple : Oscar to Alice 10$



https://blockchain.info/unconfirmed-transactions

Each node include “Oscar to Alice : 10$” to their block

One Block is “randomly” selected to be appended to the blockchain

Initial Coins Distribution:

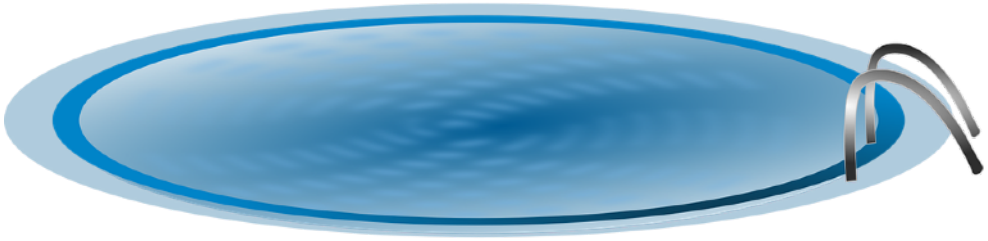
Alice : 100$

Alice to Oscar : 50$

Alice to Daniel : 30$

In more details

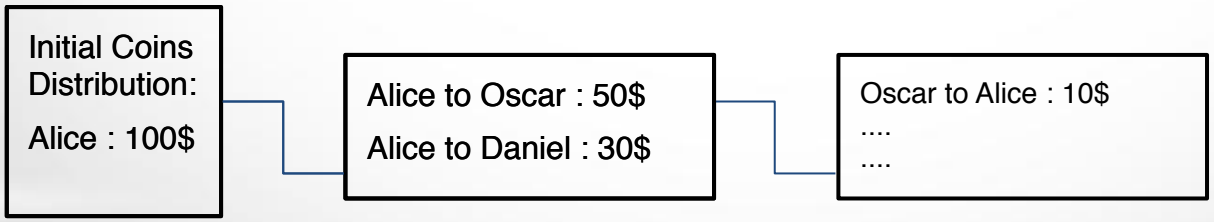
Exemple : Oscar to Alice 10$



https://blockchain.info/unconfirmed-transactions

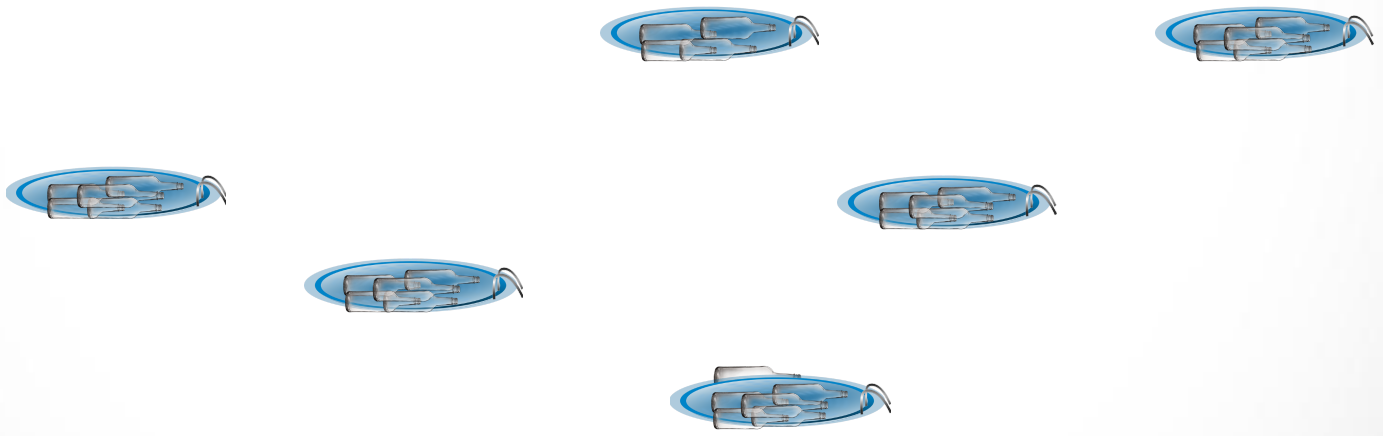
Each node include “Oscar to Alice : 10$” to their block

One Block is “randomly” selected to be appended to the blockchain



In more details

Exemple : Oscar to Alice 10$



Initial Coins Distribution:

Alice : 100$

Alice to Oscar : 50$

Alice to Daniel : 30$

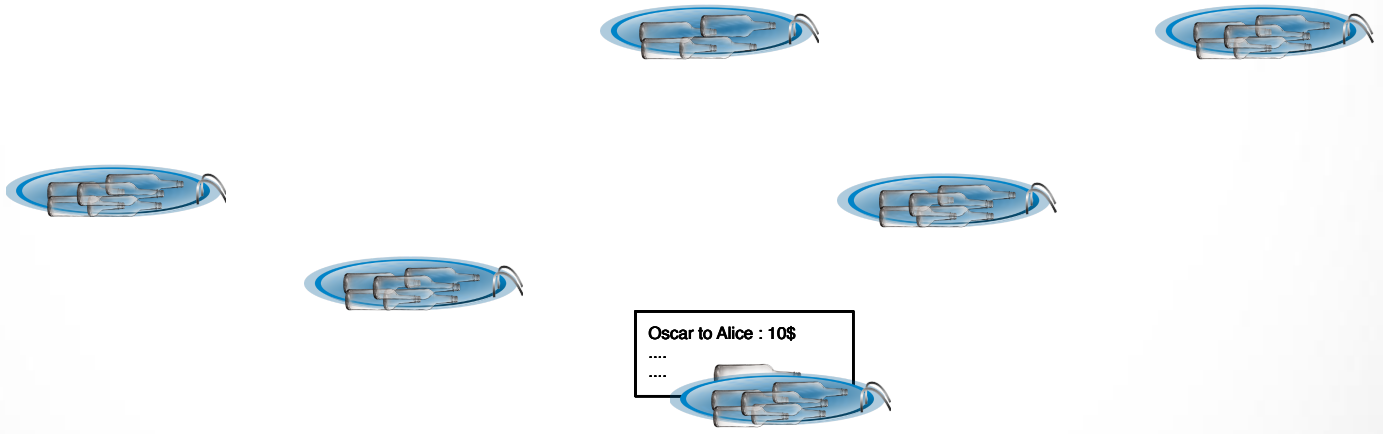
Oscar to Alice : 10$

....

....

In more details

Exemple : Oscar to Alice 10$



Initial Coins Distribution:

Alice : 100$

Alice to Oscar : 50$

Alice to Daniel : 30$

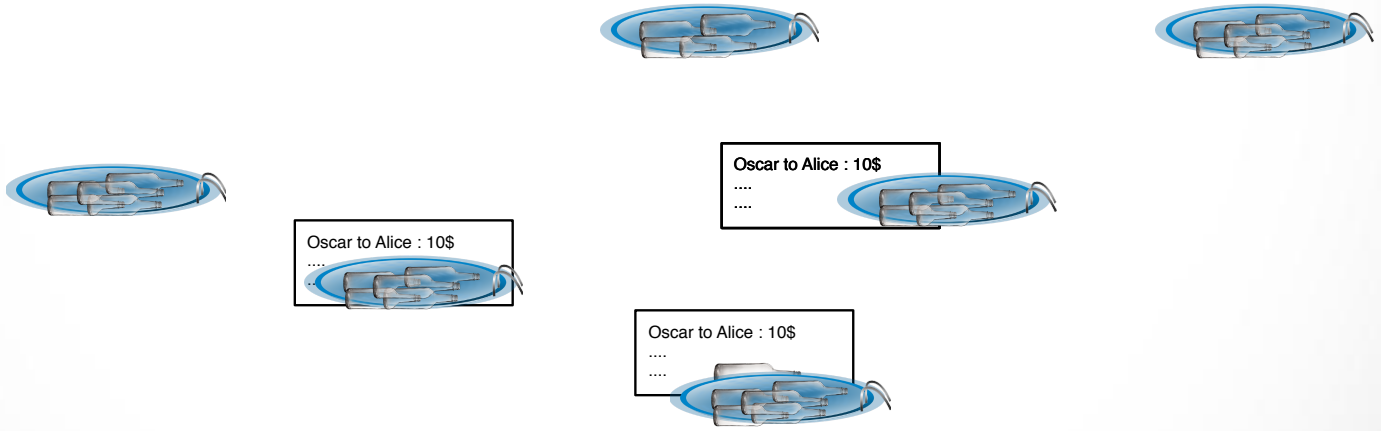
Oscar to Alice : 10$

....

....

In more details

Exemple : Oscar to Alice 10$



Initial Coins Distribution:

Alice : 100$

Alice to Oscar : 50$

Alice to Daniel : 30$

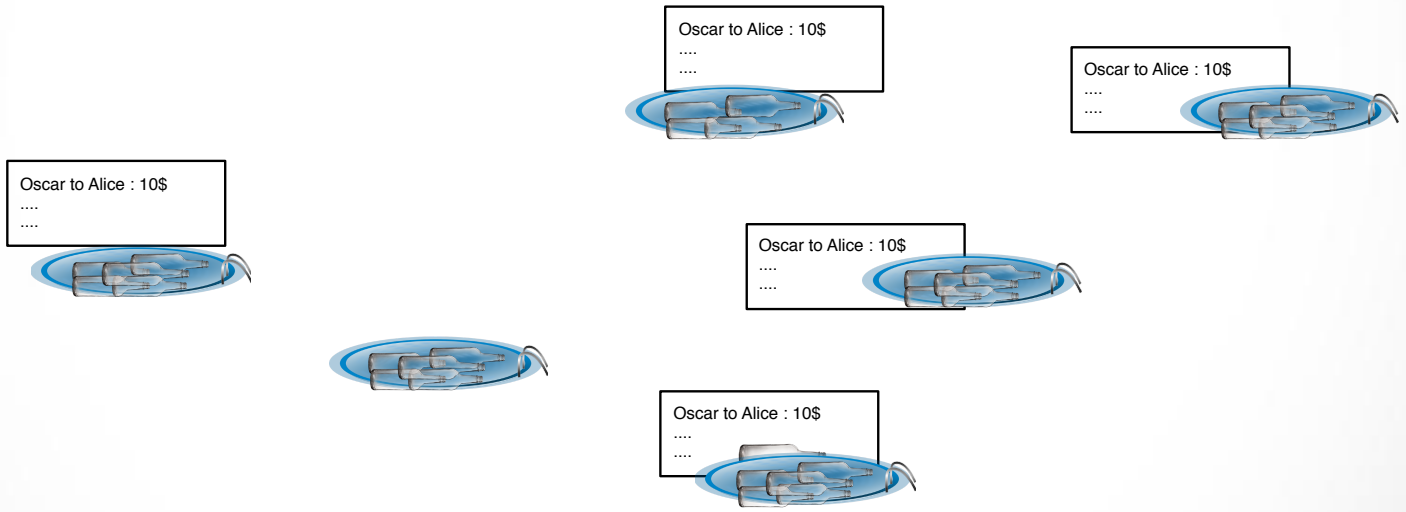
Oscar to Alice : 10$

....

....

In more details

Exemple : Oscar to Alice 10$



Initial Coins Distribution:

Alice : 100$

Alice to Oscar : 50$

Alice to Daniel : 30$

Oscar to Alice : 10$

....

....

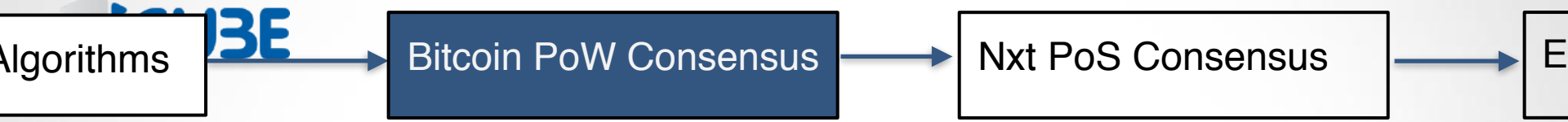
In more details

How to prevent anyone from appending a block?

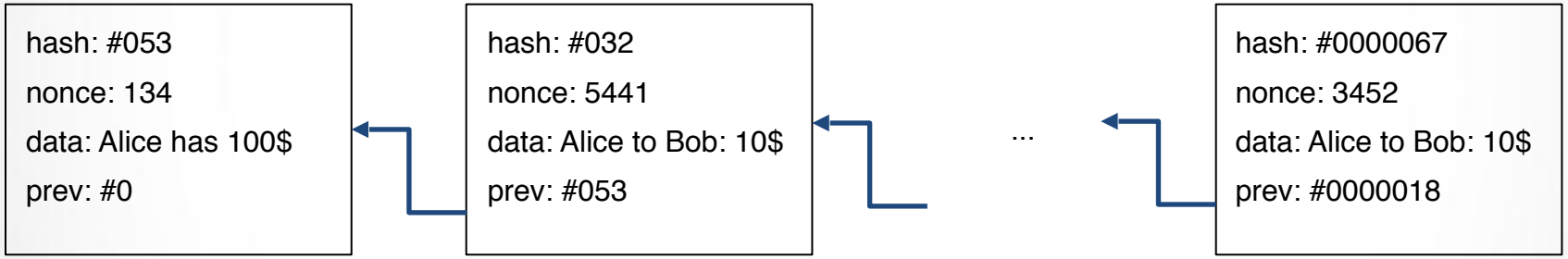
How to make sure data is not removed?

https://anders.com/blockchain/

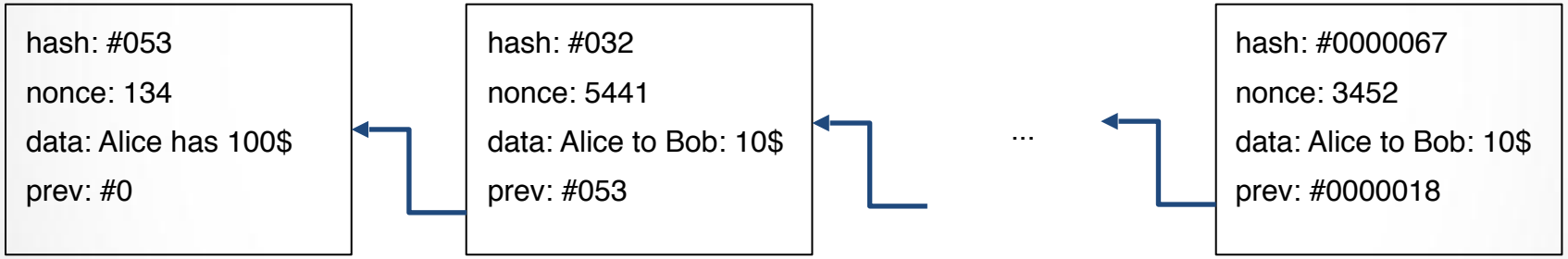
4080782: I worked hard for this presentation



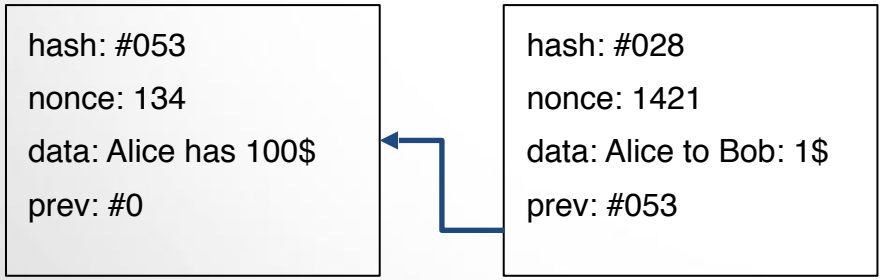
In more details



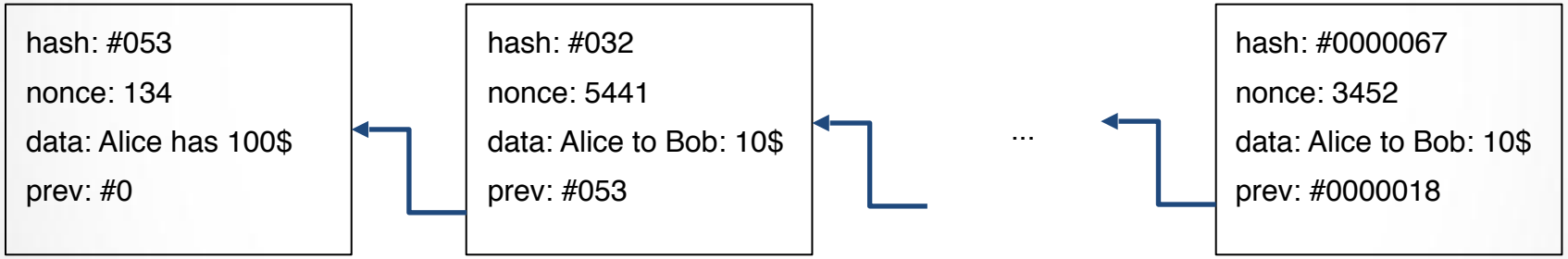
In more details



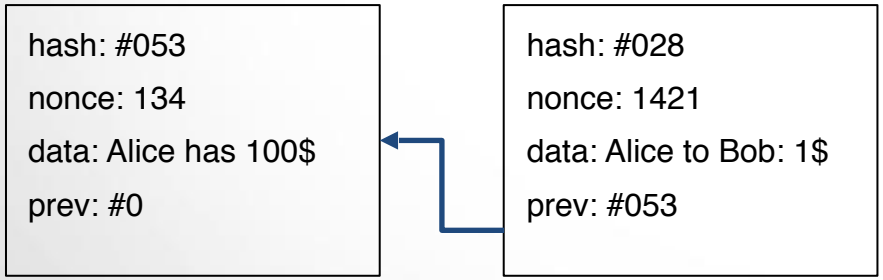
It's easy for Alice to replace a block with another one:



In more details

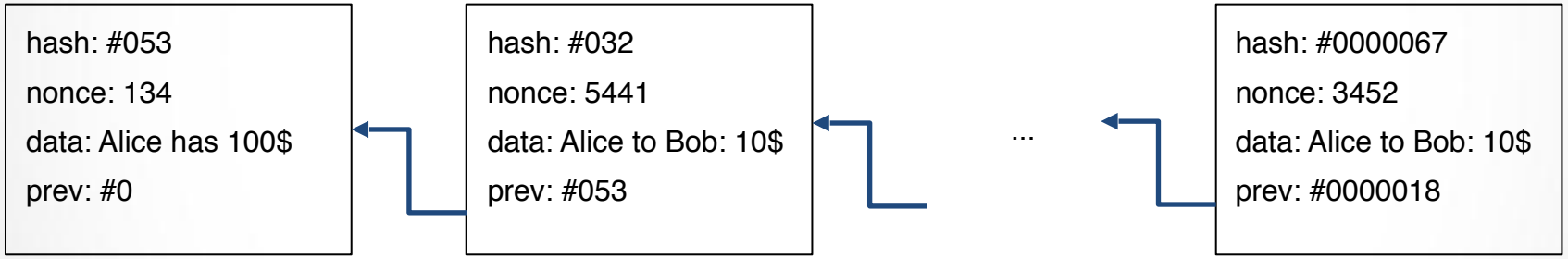


It's easy for Alice to replace a block with another one:

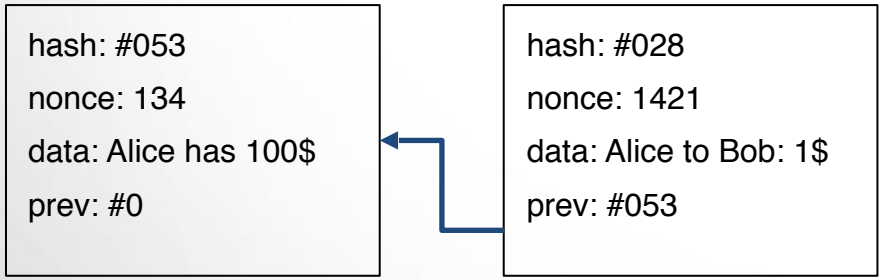


But the other nodes will prefer the longest blockchain

In more details



It's easy for Alice to replace a block with another one:

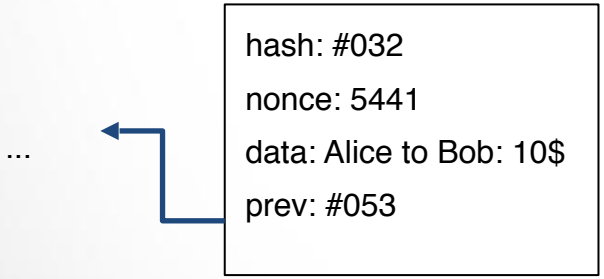


But the other nodes will prefer the longest blockchain

The deeper the block you modify, the harder it gets to generate a blockchain longer than the existing one

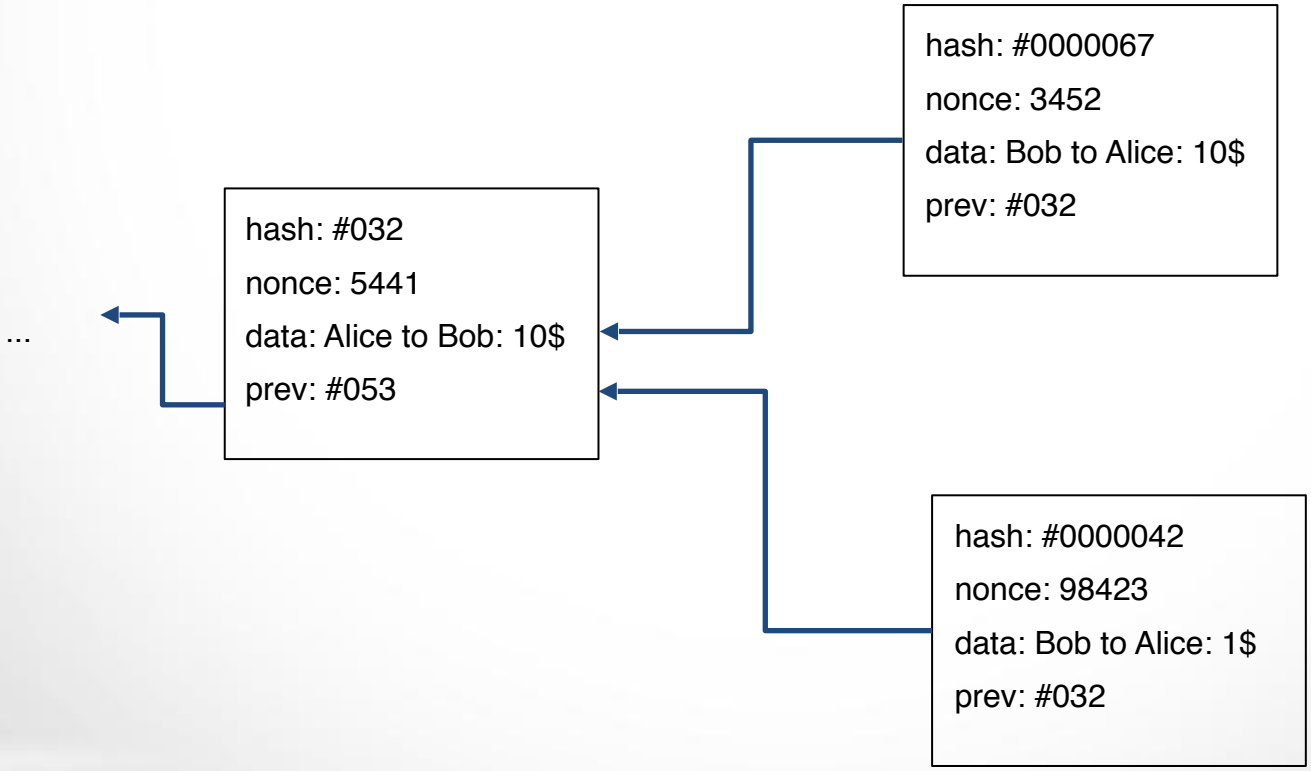
In more details

What if two nodes generates two different block at the same time ?



In more details

What if two nodes generates two different block at the same time ?



In more details

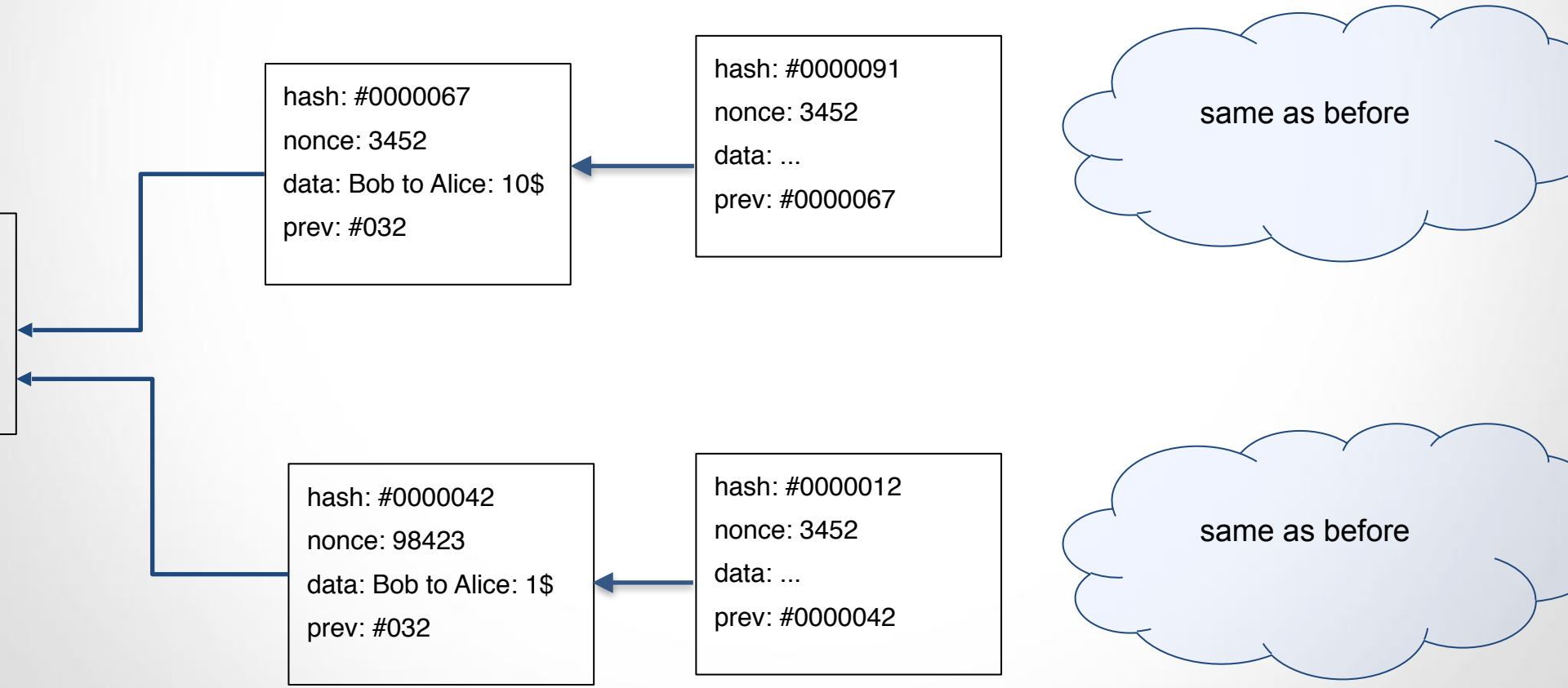
hash: #0000067
nonce: 3452
data: Bob to Alice: 10$
prev: #032

parts of the networks will receive this first and will try to append a block to this
(they don't want to change their choice when they receive the other one because that would mean that they wasted their computing power on this one)

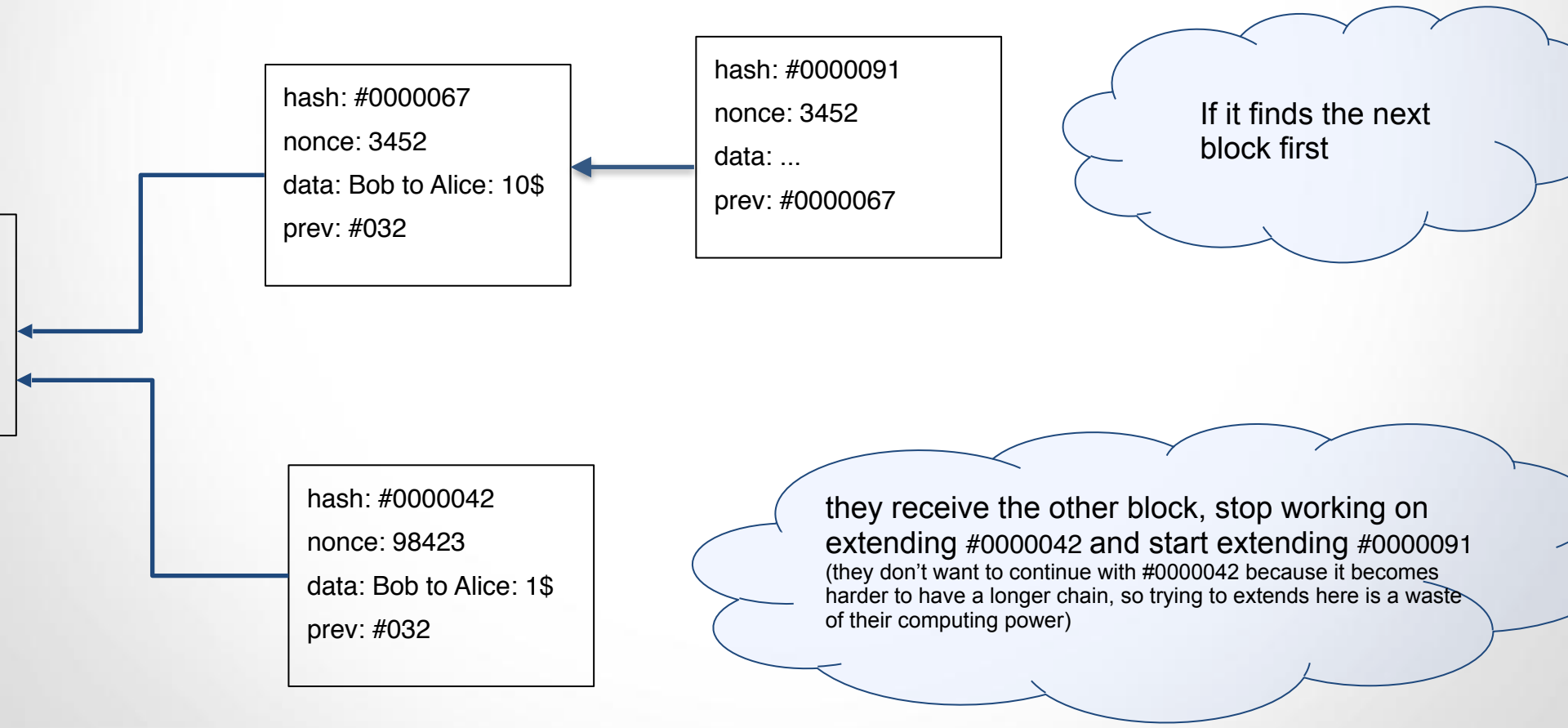
hash: #0000042
nonce: 98423
data: Bob to Alice: 1$
prev: #032

parts of the networks will receive this first and will try to append a block to this
(they don't want to change their choice when they receive the other one because that would mean that they wasted their computing power on this one)

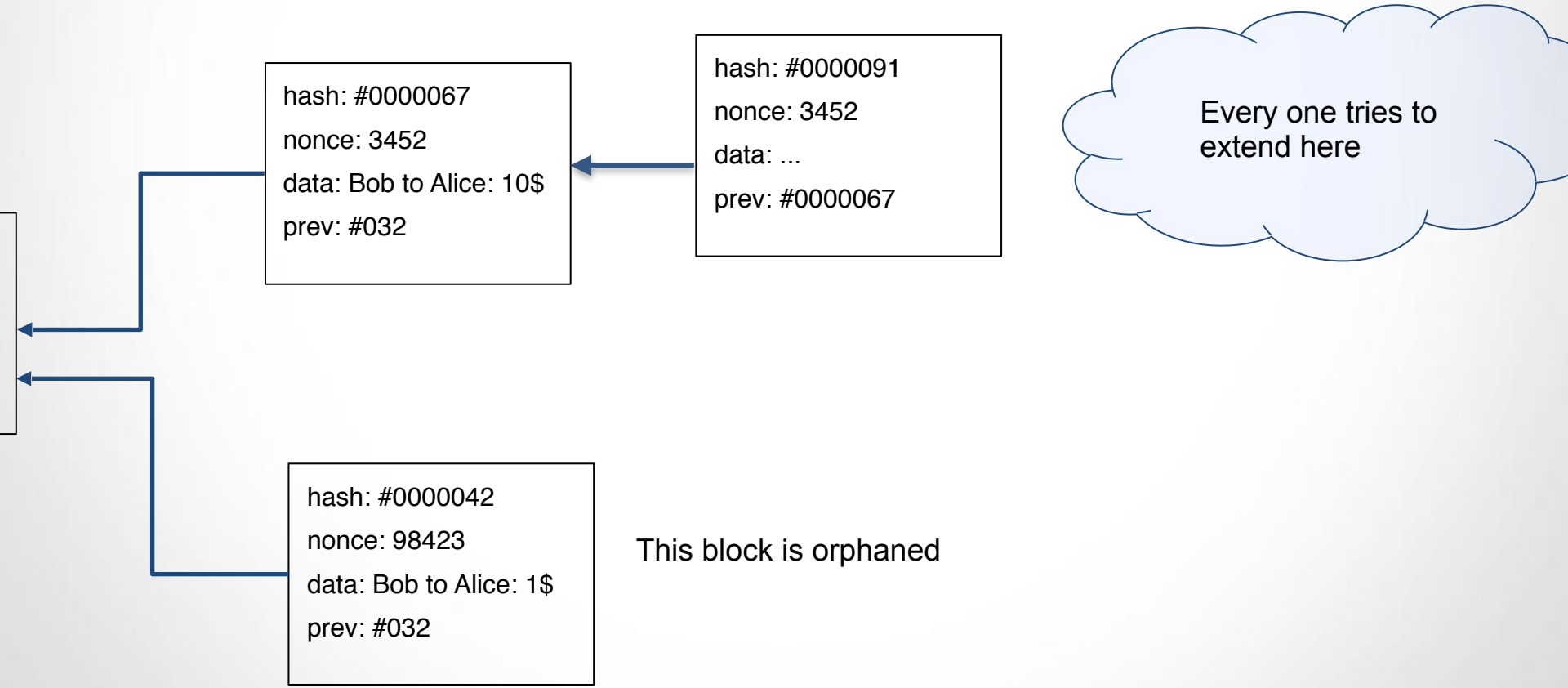
In more details



In more details



In more details

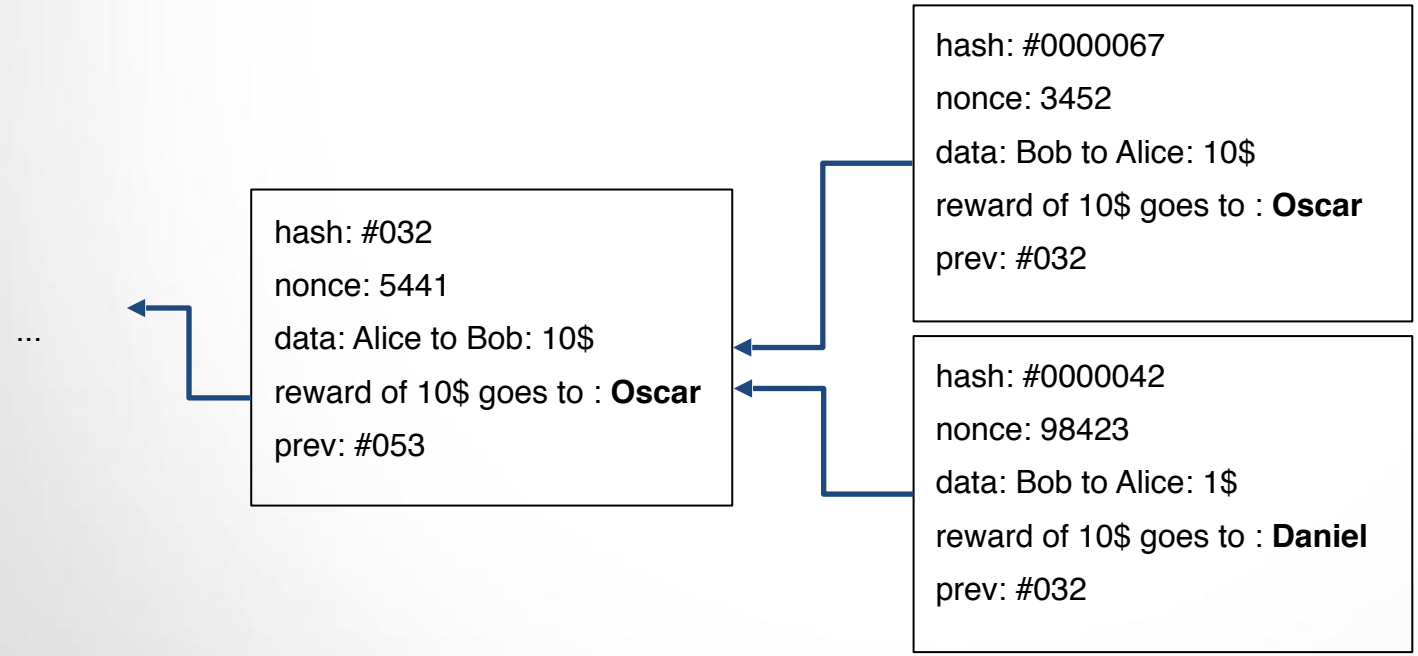


In more details

When finding a block, a node receives some coins
(a fixed reward + transaction fees chosen freely by the sender)

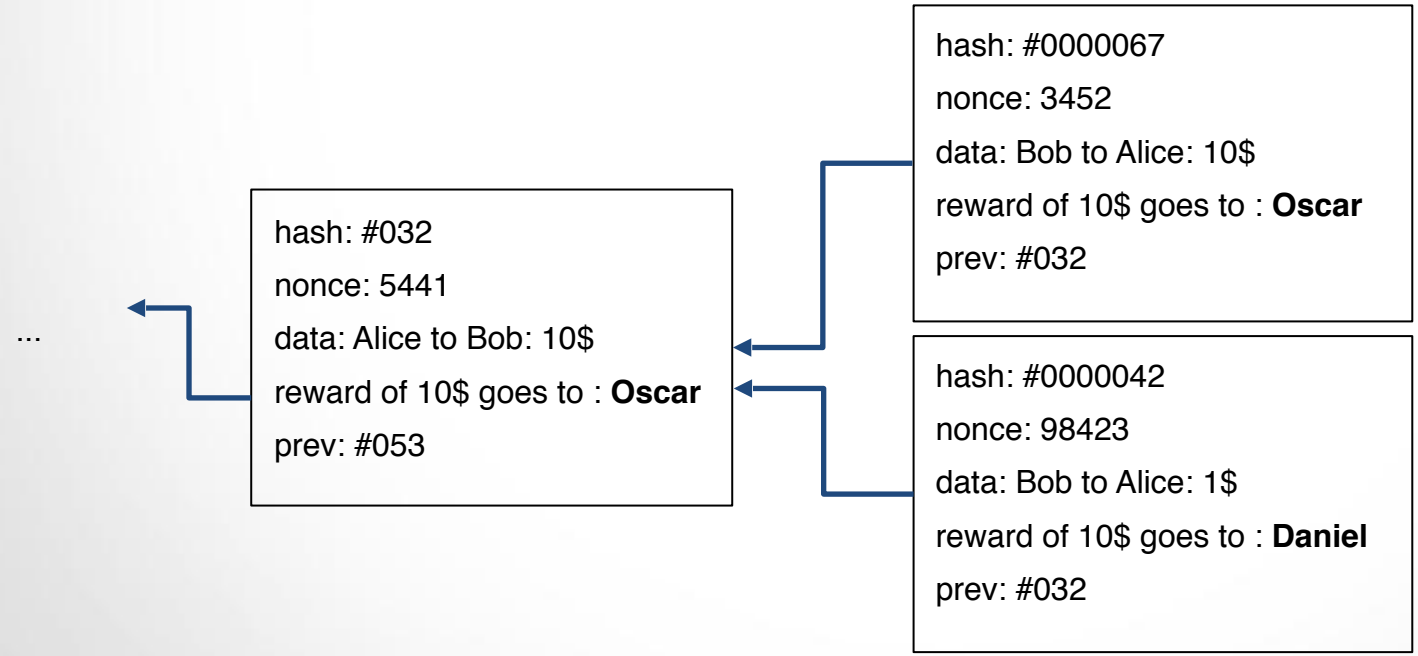
In more details

When finding a block, a node receives some coins
(a fixed reward + transaction fees chosen freely by the sender)



In more details

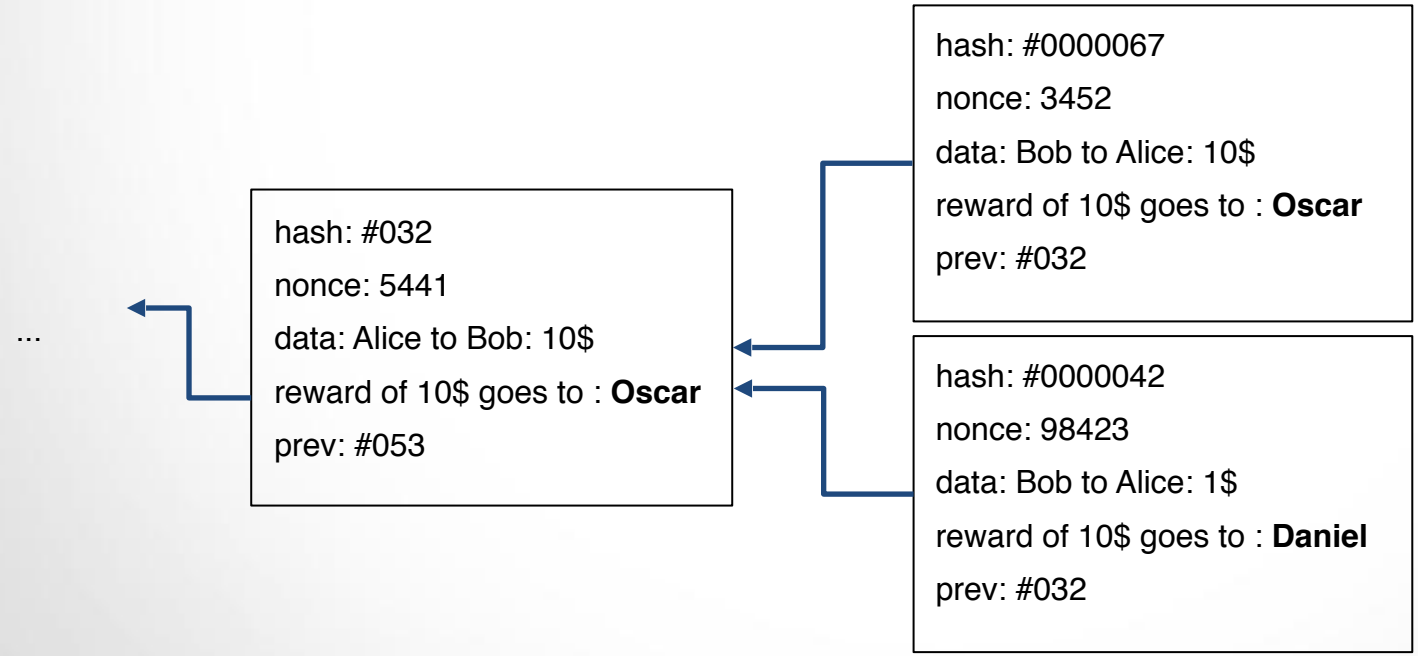
When finding a block, a node receives some coins
(a fixed reward + transaction fees chosen freely by the sender)



Oscar has 10$ in this version of the blockchain

In more details

When finding a block, a node receives some coins
(a fixed reward + transaction fees chosen freely by the sender)

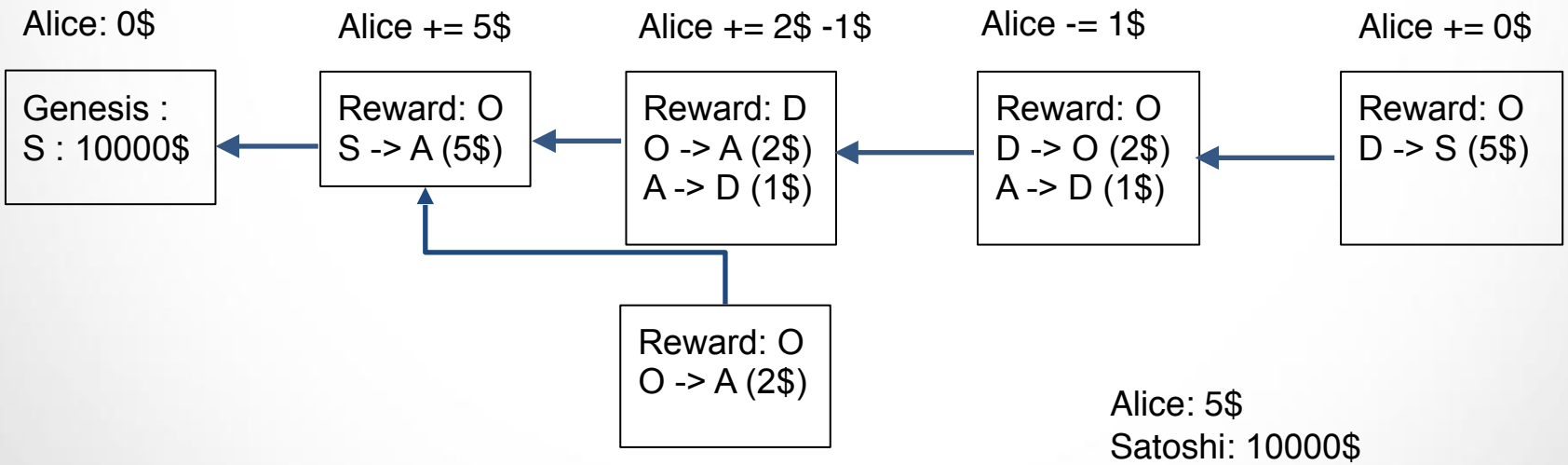


Oscar has 10$ in this version of the blockchain

So the goal of the nodes, is to find a block first, and that this block is not orphaned.
This does not mean “extends only the version where I have the greatest balance” because you want to avoid wasting hashing power on a chain that has great chance to be orphaned

In more details

What is the balance of Alice ?

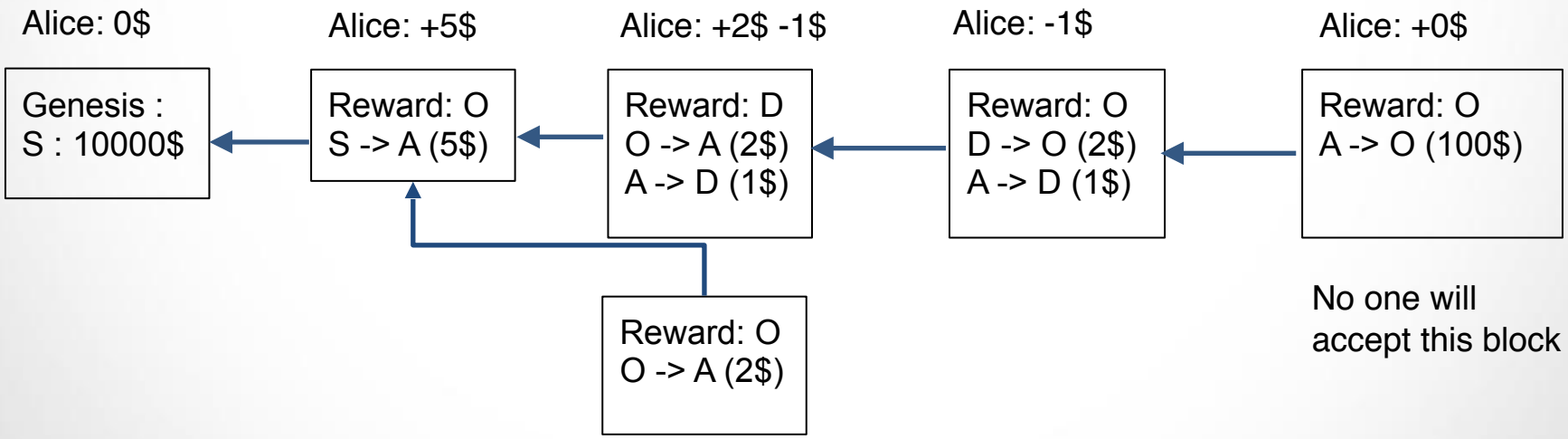


Alice: 5$
Satoshi: 10000$
Oscar: 30$
Daniel: 5$

Sum: 10000 + 4x10

In more details

Block Validation



In more details

Double spending

In more details

Double spending

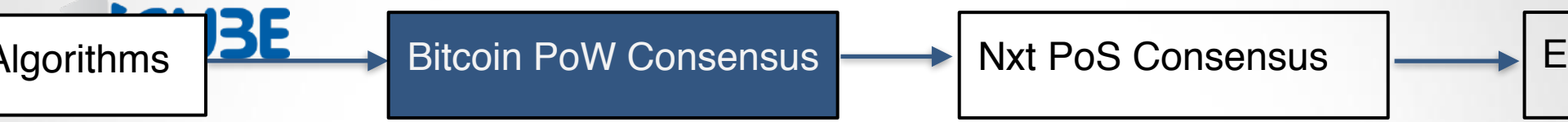
Alice buy a sandwich to Bob and sign A -> B (1$)

In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

but Alice also sign A -> A' (5$) and send this one to the network

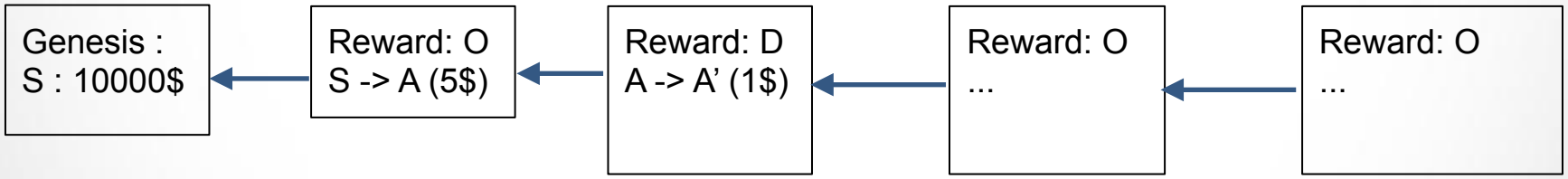


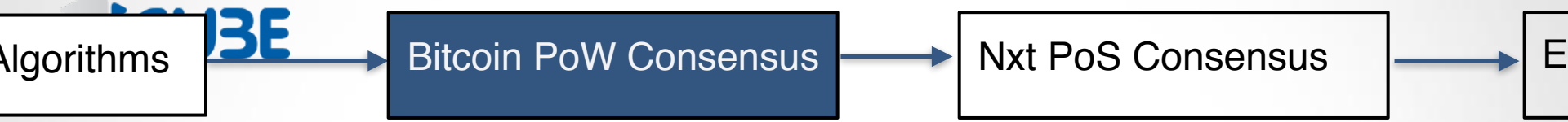
In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

but Alice also sign A -> A' (5$) and send this one to the network



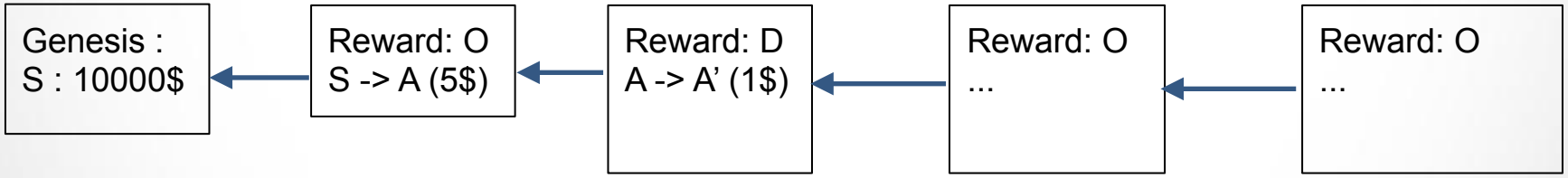


In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

but Alice also sign A -> A' (5$) and send this one to the network



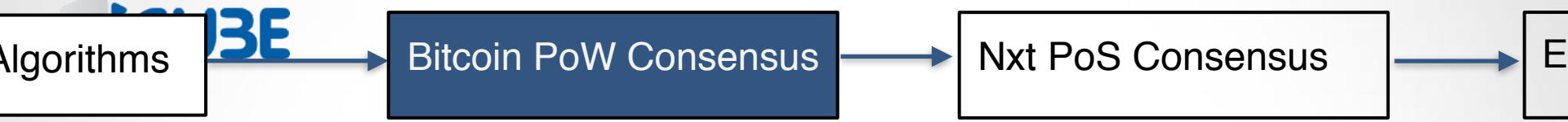
Now, no one will ever include
A -> B (1$) transaction because it's in
conflict with A -> A' (5$)

In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

Bob waits to see this transaction in a block before giving the sandwich

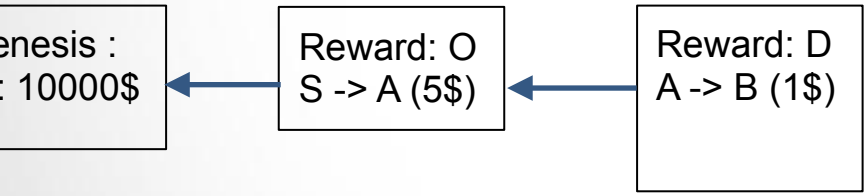


In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

Bob waits to see this transaction in a block before giving the sandwich

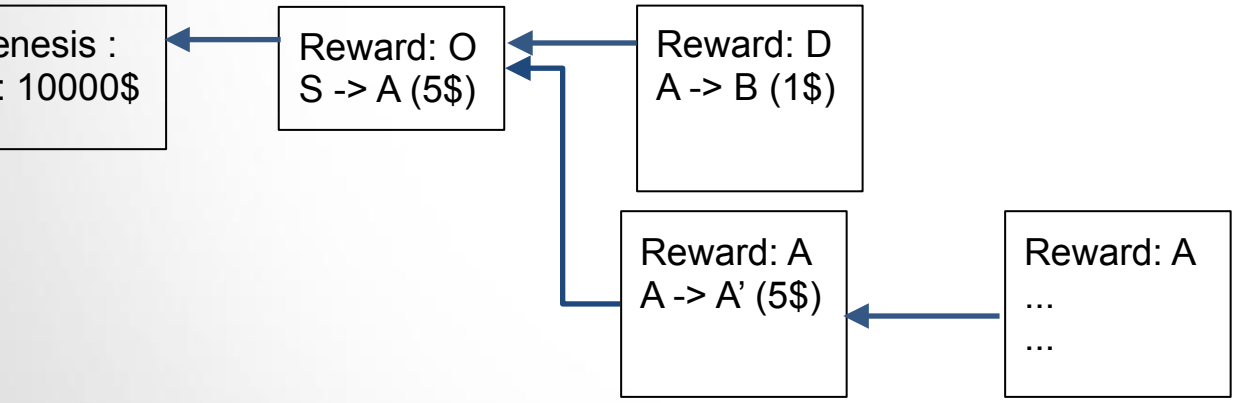


In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

Bob waits to see this transaction in a block before giving the sandwich



Alice also sign A -> A' (5$) and use a lots of power to extends this chain faster than the other

In this chain, B never received the coins and A -> B (1$) cannot be included anymore

In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

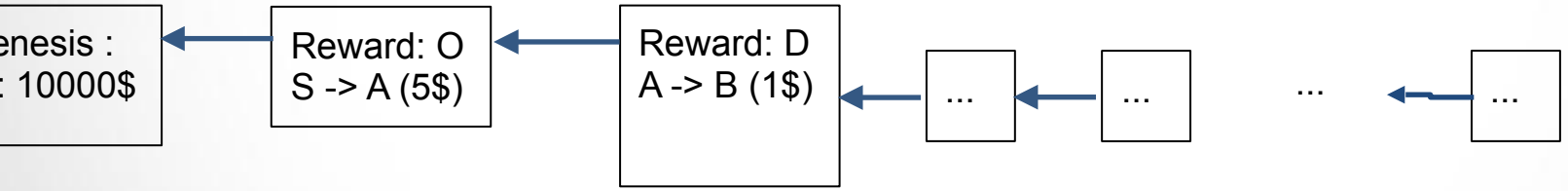
Bob waits to see this transaction in a block and waits for 10 confirmation blocks

In more details

Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

Bob waits to see this transaction in a block and waits for 10 confirmation blocks

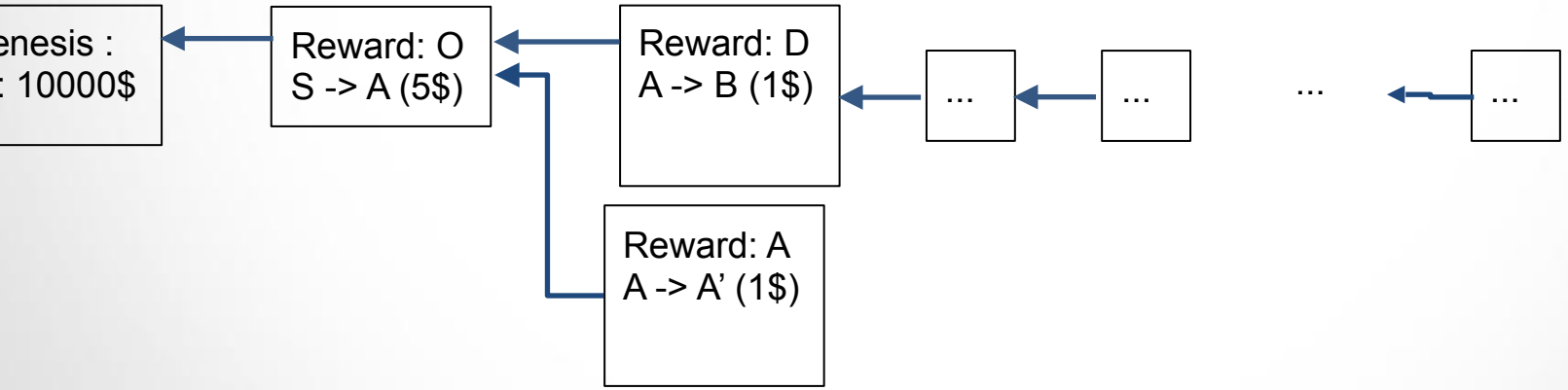


In more details

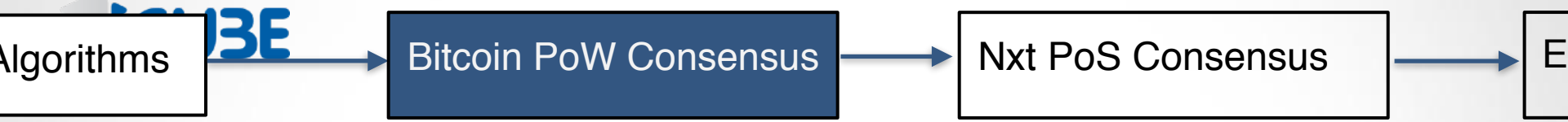
Double spending

Alice buy a sandwich to Bob and sign A -> B (1$)

Bob waits to see this transaction in a block and waits for 10 confirmation blocks



It's too expensive for Alice to generate 10 blocks on her own



In more details

In more details

- Double spending:
  If an adversary owns more than ½ the network hashing power, it can always perform a double spend

In more details

- Double spending:
  If an adversary owns more than ½ the network hashing power, it can always perform a double spend

- Orphaned blocks ⇒ wasted hashing power ⇒ reduce protocol security

In more details

- Double spending:
  If an adversary owns more than ½ the network hashing power, it can always perform a double spend

- Orphaned blocks ⇒ wasted hashing power ⇒ reduce protocol security

- The easier the PoW the more forks will occur.
  (the average time between two nodes to find a block should be much greater than the latency in the network)

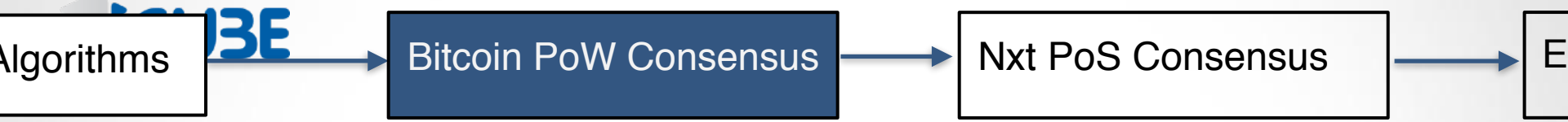
In more details

- Double spending:
  If an adversary owns more than ½ the network hashing power, it can always perform a double spend

- Orphaned blocks ⇒ wasted hashing power ⇒ reduce protocol security

- The easier the PoW the more forks will occur.
  (the average time between two nodes to find a block should be much greater than the latency in the network)

- Bitcoin protocol :
  - new blocks should arrive in average every 10 minutes
  - reward : 12.5BTC (divided by two every 4 years)



In more details

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

p : probability of finding a block with 1 hash

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

p : probability of finding a block with 1 hash

$\frac{1}{p}$: expecting number of hash before finding a block

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

p : probability of finding a block with 1 hash

$\frac{1}{p}$: expecting number of hash before finding a block

$\frac{1}{pH}$: expecting number of seconds before finding a block

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

p : probability of finding a block with 1 hash

$\frac{1}{p}$: expecting number of hash before finding a block

$\frac{1}{pH}$: expecting number of seconds before finding a block

We want $\frac{1}{pH} = 600$ $\qquad p = \frac{1}{H600}$

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

p : probability of finding a block with 1 hash

$\frac{1}{p}$: expecting number of hash before finding a block

$\frac{1}{pH}$: expecting number of seconds before finding a block

We want $\frac{1}{pH} = 600$ $\quad p = \frac{1}{H600}$

$$p = \frac{2^{224}}{D} \times \frac{1}{2^{256}} = \frac{1}{D2^{32}}$$

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

p : probability of finding a block with 1 hash

$\frac{1}{p}$: expecting number of hash before finding a block

$\frac{1}{pH}$: expecting number of seconds before finding a block

We want $\frac{1}{pH} = 600$ $\quad p = \frac{1}{H600}$

$$p = \frac{2^{224}}{D} \times \frac{1}{2^{256}} = \frac{1}{D2^{32}}$$

D is the difficulty. The greater the difficulty, the smaller the probability p

In more details

Bitcoin protocol : new blocks should arrive in average every 10 minutes

H : hashrate, hash/s of the network

p : probability of finding a block with 1 hash

$\frac{1}{p}$: expecting number of hash before finding a block

$\frac{1}{pH}$: expecting number of seconds before finding a block

We want $\frac{1}{pH} = 600$ $\quad p = \frac{1}{H600}$

$$p = \frac{2^{224}}{D} \times \frac{1}{2^{256}} = \frac{1}{D2^{32}}$$

D is the difficulty. The greater the difficulty, the smaller the probability p

$$D = \frac{H600}{2^{32}}$$

Fun facts

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
-
-
-

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)
-
-

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)
- How many BTC I gifted to my cousin in 2013,
-

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)
- How many BTC I gifted to my cousin in 2013,
  but he lost it because it was a paper wallet : 0.022 BTC (170$)
-

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)
- How many BTC I gifted to my cousin in 2013,
  but he lost it because it was a paper wallet : 0.022 BTC (170$)
- Bitcoin's current estimated annual electricity consumption (TWh) : 67.3

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)
- How many BTC I gifted to my cousin in 2013,
  but he lost it because it was a paper wallet : 0.022 BTC (170$)
- Bitcoin's current estimated annual electricity consumption (TWh) : 67.3
  Annualized estimated global mining costs : $3,364,875,703

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)
- How many BTC I gifted to my cousin in 2013,
  but he lost it because it was a paper wallet : 0.022 BTC (170$)
- Bitcoin's current estimated annual electricity consumption (TWh) : 67.3
  Annualized estimated global mining costs : $3,364,875,703
  Electricity consumed per transaction (KWh) : 904

Fun facts

- Estimated Satoshi's balance : 1M BTC = 10 billion USD
- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)
- How many BTC I gifted to my cousin in 2013,
  but he lost it because it was a paper wallet : 0.022 BTC (170$)
- Bitcoin's current estimated annual electricity consumption (TWh) : 67.3
  Annualized estimated global mining costs : $3,364,875,703
  Electricity consumed per transaction (KWh) : 904
  Carbon footprint per transaction (kg of CO2) : 442.72

Fun facts

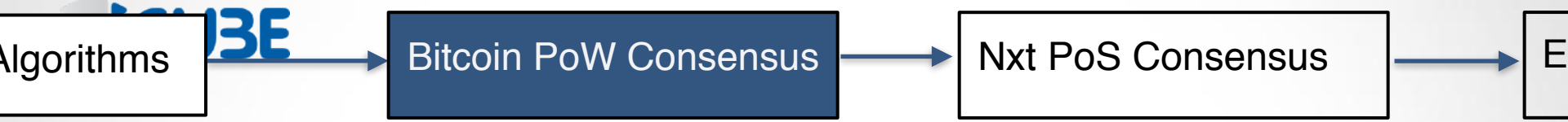
- Estimated Satoshi's balance : 1M BTC = 10 billion USD

- Current difficulty : 4,022,059,196,164
  Current Hashrate : 29,911,310,546 GH/s (3 * 10^16 hash / s)

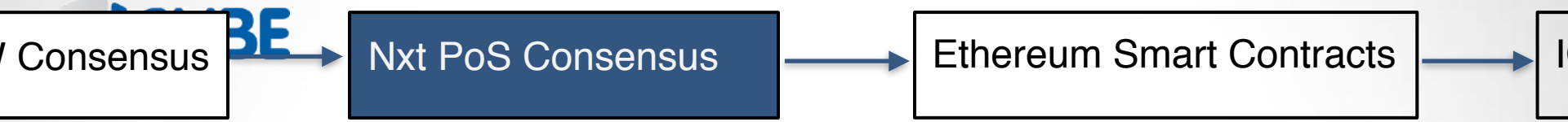
- How many BTC I gifted to my cousin in 2013,
  but he lost it because it was a paper wallet : 0.022 BTC (170$)

- Bitcoin's current estimated annual electricity consumption (TWh) : 67.3
  Annualized estimated global mining costs : $3,364,875,703
  Electricity consumed per transaction (KWh) : 904
  Carbon footprint per transaction (kg of CO2) : 442.72

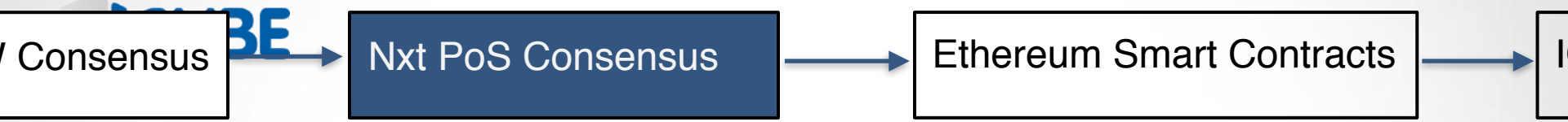
  https://digiconomist.net/bitcoin-energy-consumption



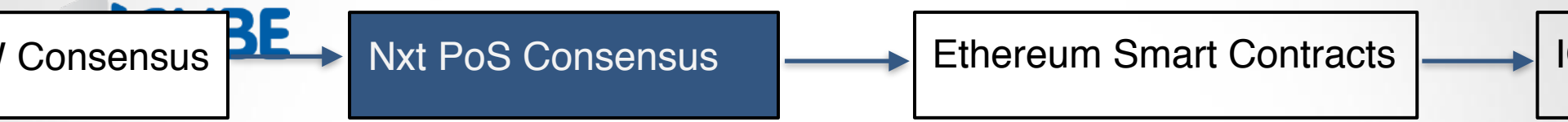
How to avoid wasting energy for leader election ?



How to avoid wasting energy for leader election ?

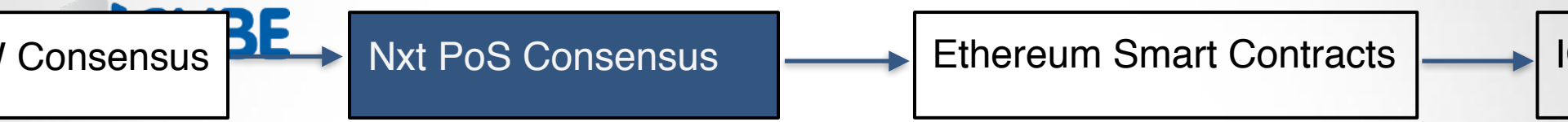


Proof of Stake



Proof of Stake

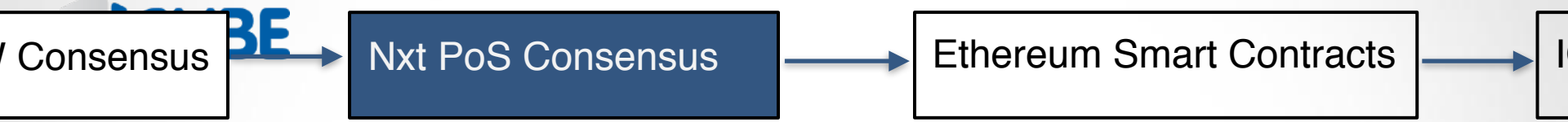
The more coins you own, the more chance you get of being elected



Proof of Stake

The more coins you own, the more chance you get of being elected

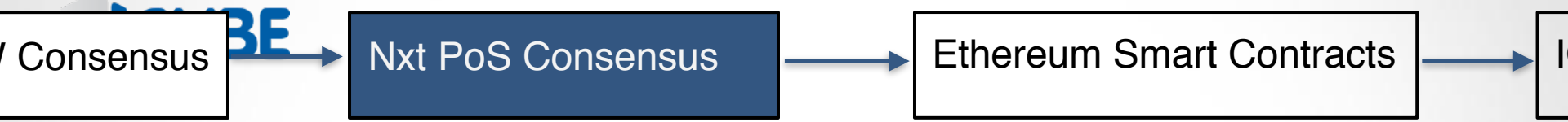
Election :



Proof of Stake

The more coins you own, the more chance you get of being elected

Election :

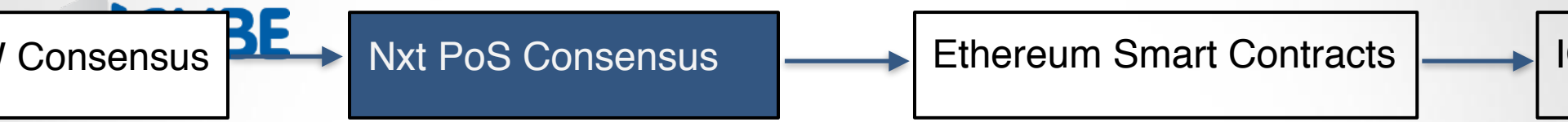


Proof of Stake

The more coins you own, the more chance you get of being elected

Election :

- For each address, take *t* = hash(adr + last block hash) / balance

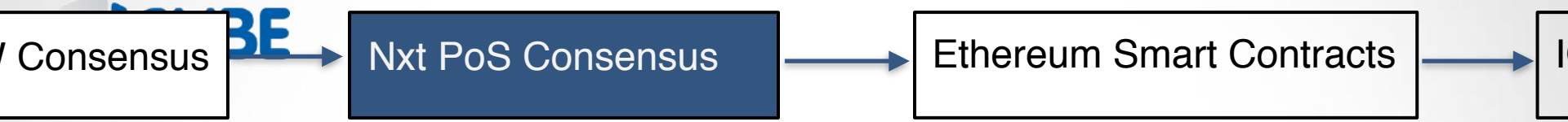


Proof of Stake

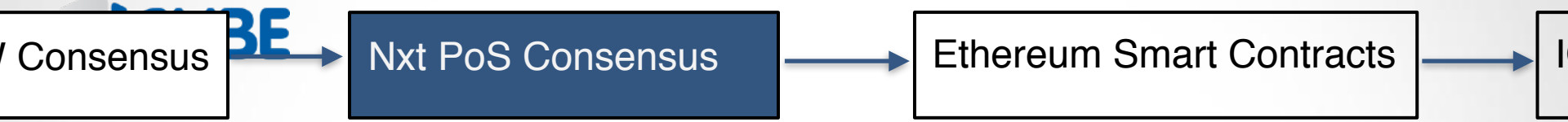
The more coins you own, the more chance you get of being elected

Election :

- For each address, take *t* = hash(adr + last block hash) / balance
- The address with the smallest *t* is elected.



Using “old-style” Consensus algorithm



Using “old-style” Consensus algorithm

- Elects n validators (with PoS for instance)
- The validators use “old-style” Consensus algorithm to agree on the next blocks


onsensus
Ethereum Smart Contracts
IOTA BlockDAG

What is a smart contract ?

What is a smart contract ?

Alice want to sell a singleplayer game

What is a smart contract ?

Alice want to sell a singleplayer game

But users should buy a licence to play it

What is a smart contract ?

Alice want to sell a singleplayer game

But users should buy a licence to play it

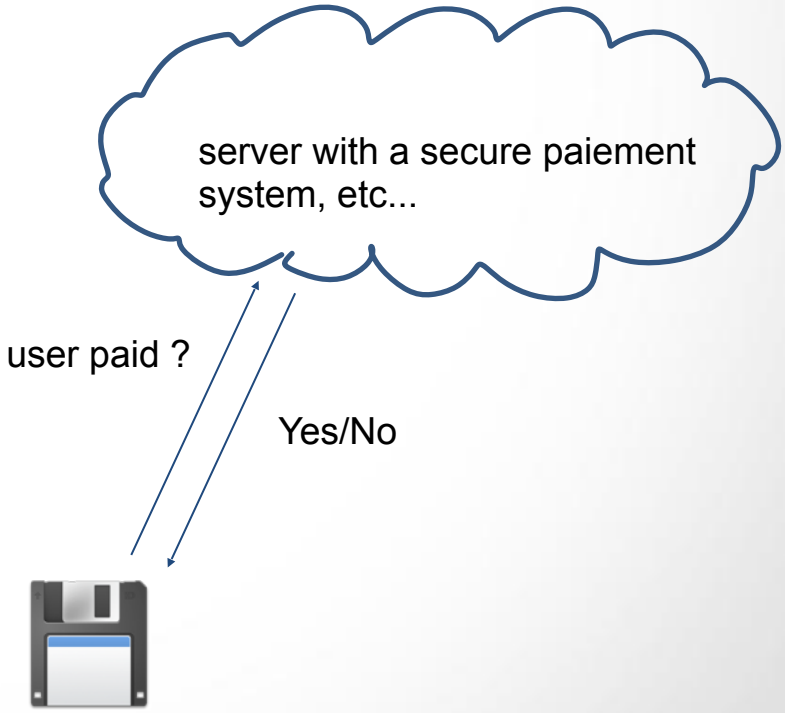
The contract is simple: a user pay 10$ and he can play it

What is a smart contract ?

Alice want to sell a singleplayer game

But users should buy a licence to play it

The contract is simple: a user pay 10$ and he can play it

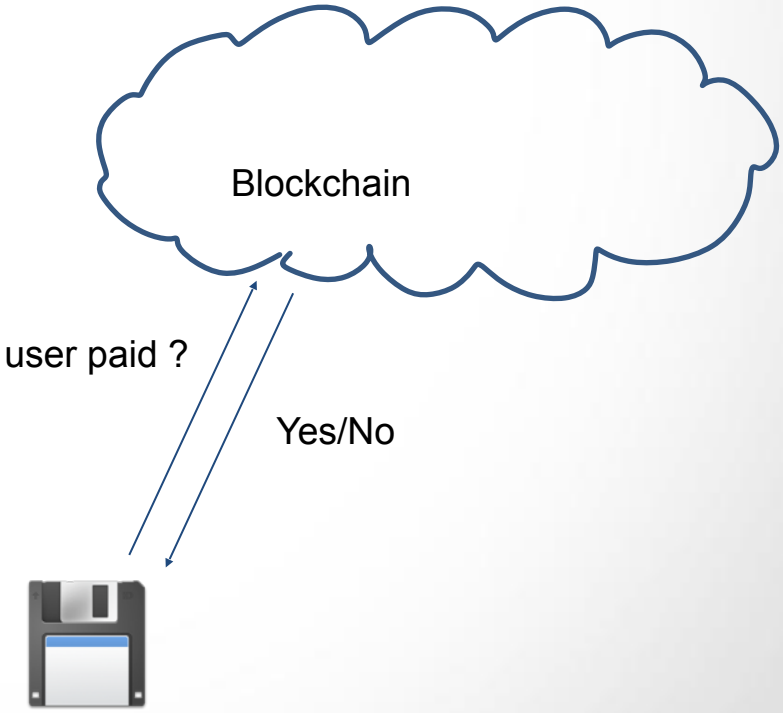


What is a smart contract ?

Alice want to sell a singleplayer game

But users should buy a licence to play it

The contract is simple: a user pay 10$ and he can play it



What is a smart contract ?

What is a smart contract ?

“

var usersWhoOwnTheGame: Set

If an address *adr* sent to Alice 10$ and add “buyTheGame()” in the data of a block **then**
usersWhoOwnTheGame.insert(*adr*)

“

What is a smart contract ?

Alice add this to a block:

“

var usersWhoOwnTheGame: Set

If an address *adr* sent to Alice 10$ and add “buyTheGame()” in the data of a block **then**
usersWhoOwnTheGame.insert(*adr*)

“

What is a smart contract ?

Alice add this to a block:

> “
> var usersWhoOwnTheGame: Set
>
> **If** an address *adr* sent to Alice 10$ and add “buyTheGame()” in the data of a block **then**
> usersWhoOwnTheGame.insert(*adr*)
> “

Then in addition to remember the balance of each address, each nodes remember the value of *usersWhoOwnTheGame* and updates it after each block if the condition is met.

What is a smart contract ?

Should be understood by every nodes in the same way

Alice add this to a block:

“
var usersWhoOwnTheGame: Set

If an address *adr* sent to Alice 10$ and add “buyTheGame()” in the data of a block **then**
usersWhoOwnTheGame.insert(*adr*)
“

Then in addition to remember the balance of each address, each nodes remember the value of *usersWhoOwnTheGame* and updates it after each block if the condition is met.

What is a smart contract ?

Alice add this to a block:

```
pragma solidity ^0.4.18;
contract AliceGame {
  mapping(address => bool) public usersWhoOwnTheGame;

  function buy() public payable {
    if(msg.value >= 10)
    {
      usersWhoOwnTheGame[msg.sender] = true;
    }
  }
}
```

test here: http://remix.ethereum.org/

-> gets translated to EVM instructions


```
PUSH1 0x80
PUSH1 0x40
MSTORE CALLVALUE
DUP1
ISZERO
PUSH2 0x10
JUMPI
PUSH1 0x0
DUP1
REVERT
JUMPDEST
POP
PUSH2 0x166
DUP1
PUSH2 0x20
PUSH1 0x0
CODECOPY
PUSH1 0x0
RETURN
STOP
PUSH1 0x80
PUSH1 0x40
MSTORE
PUSH1 0x4
CALLDATASIZE
LT
```

What is a smart contract ?

What is a smart contract ?

Each and **every** node in the network executes **all** the smart contracts, and keep track of all the values of all the variables.

What is a smart contract ?

Each and **every** node in the network executes **all** the smart contracts, and keep track of all the values of all the variables.

When a transaction arrives with data *@AliceGame.buy()*
the node has to load the context of the smart contract into memory, executes the function, check if everything is ok, performs the corresponding actions.

What is a smart contract ?

Each and **every** node in the network executes **all** the smart contracts, and keep track of all the values of all the variables.

When a transaction arrives with data *@AliceGame.buy()*
the node has to load the context of the smart contract into memory, executes the function, check if everything is ok, performs the corresponding actions.

A client can ask a node “what’s the value of this variable *@AliceGame.usersWhoOwnTheGame[playerAdr]*”

What is a smart contract ?

What is a smart contract ?

Executing a contract has a cost depending on the complexity of the algorithm.

What is a smart contract ?

Executing a contract has a cost depending on the complexity of the algorithm.

In bitcoin, a transaction just consumes space in the block (fees encourage miners to include a transaction)

What is a smart contract ?

Executing a contract has a cost depending on the complexity of the algorithm.

In bitcoin, a transaction just consumes space in the block (fees encourage miners to include a transaction)

In Ethereum, the cost of a calling a contract is calculated in Gas (each EVM instruction has a Gas Cost)

What is a smart contract ?

Executing a contract has a cost depending on the complexity of the algorithm.

In bitcoin, a transaction just consumes space in the block (fees encourage miners to include a transaction)

In Ethereum, the cost of a calling a contract is calculated in Gas (each EVM instruction has a Gas Cost)

When calling a contracts you declare

What is a smart contract ?

Executing a contract has a cost depending on the complexity of the algorithm.

In bitcoin, a transaction just consumes space in the block (fees encourage miners to include a transaction)

In Ethereum, the cost of a calling a contract is calculated in Gas (each EVM instruction has a Gas Cost)

When calling a contracts you declare
- The called function

What is a smart contract ?

Executing a contract has a cost depending on the complexity of the algorithm.

In bitcoin, a transaction just consumes space in the block (fees encourage miners to include a transaction)

In Ethereum, the cost of a calling a contract is calculated in Gas (each EVM instruction has a Gas Cost)

When calling a contracts you declare
- The called function
- How much gas at most the miner will spend executing it

What is a smart contract ?

Executing a contract has a cost depending on the complexity of the algorithm.

In bitcoin, a transaction just consumes space in the block (fees encourage miners to include a transaction)

In Ethereum, the cost of a calling a contract is calculated in Gas (each EVM instruction has a Gas Cost)

When calling a contracts you declare

- The called function
- How much gas at most the miner will spend executing it
- How much Ether you're willing to pay per Gas spent

Fun fact

Fun fact

You can buy this crypto cat for 150 ETH (10000$)

Better Fact

https://www.dash.org/network/#section-governance

So what's the problem?

So what's the problem?

It does not scale

So what's the problem?

It does not scale

Bitcoin : 4 transactions / seconds

So what's the problem?

It does not scale

Bitcoin : 4 transactions / seconds

Visa: 2 000 transactions / seconds

Solutions

Solutions

Sharding (each node own a fraction of the database)

Solutions

Sharding (each node own a fraction of the database)

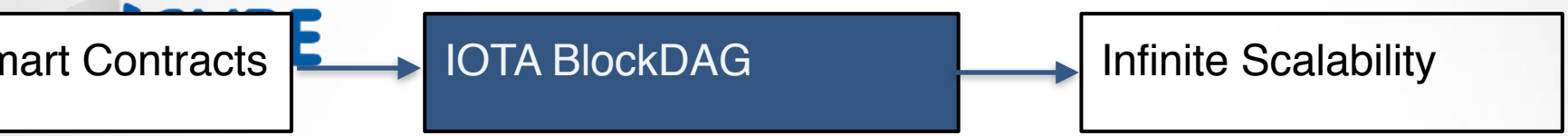
Side-chains

Solutions

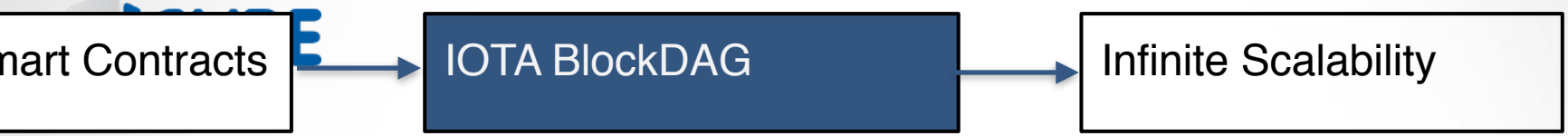
Sharding (each node own a fraction of the database)

Side-chains

DAG instead of Chain



The Tangle (IOTA)



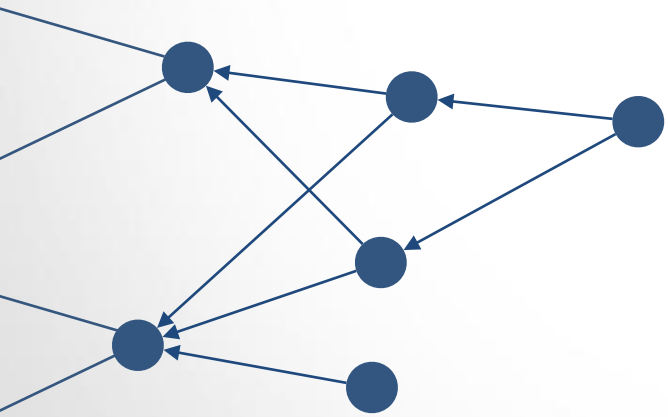
The Tangle (IOTA)

Each transaction is a small block that reference two previous ones

mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

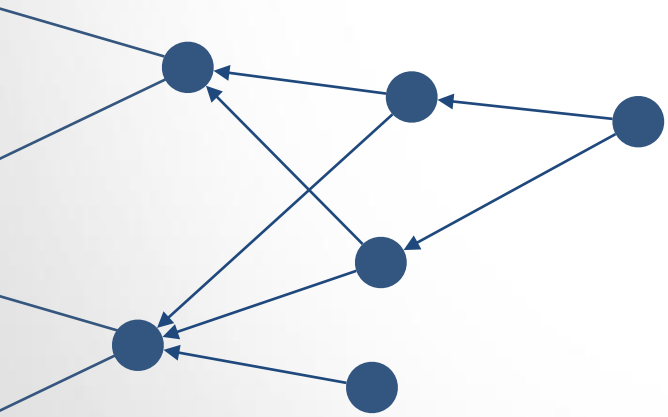
Each transaction is a small block that reference two previous ones



mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

Each transaction is a small block that reference two previous ones

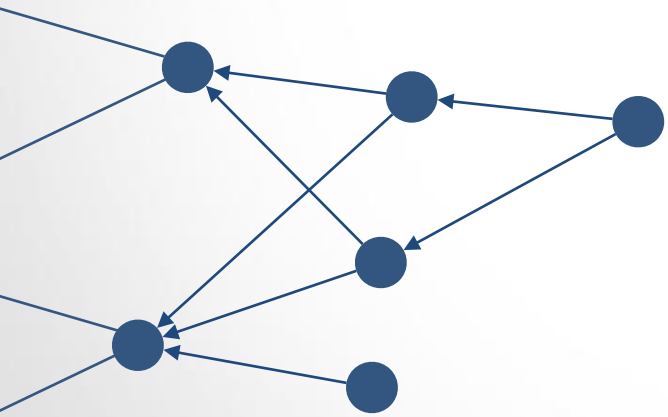


You come up with a DAG
(Directed Acyclic Graph)

mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

Each transaction is a small block that reference two previous ones



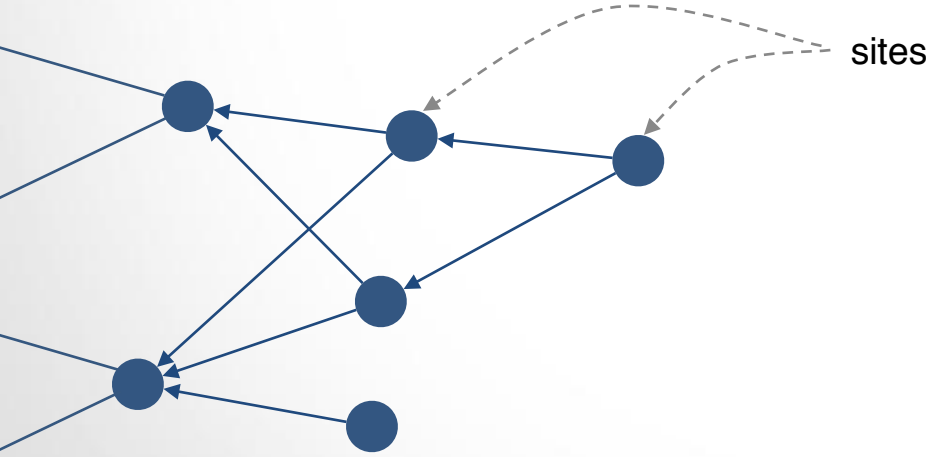
You come up with a DAG
(Directed Acyclic Graph)

You're only limited by bandwidth and storage

mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

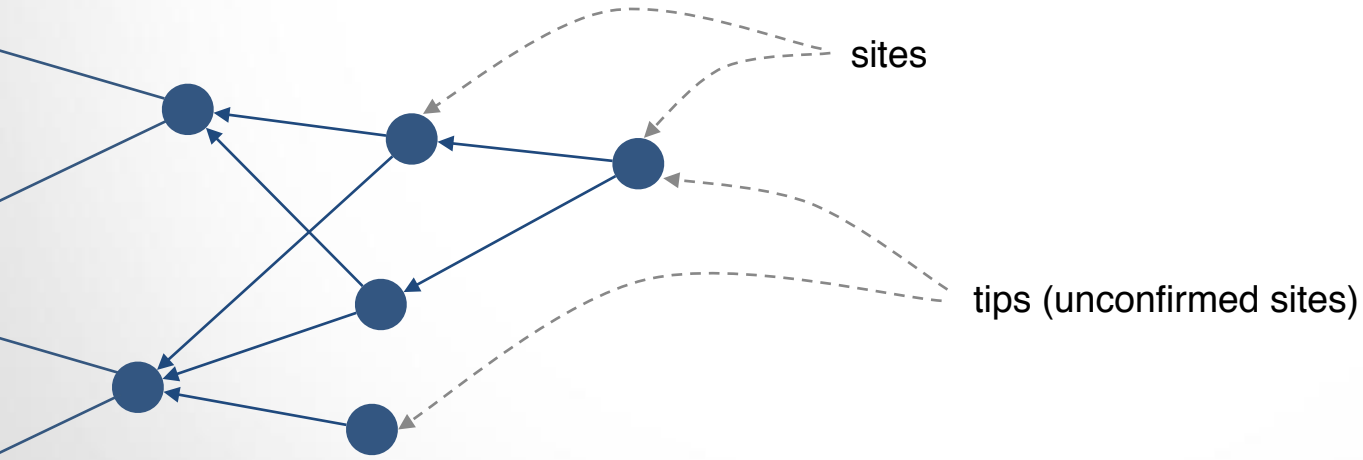
Each transaction is a small block that reference two previous ones



mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

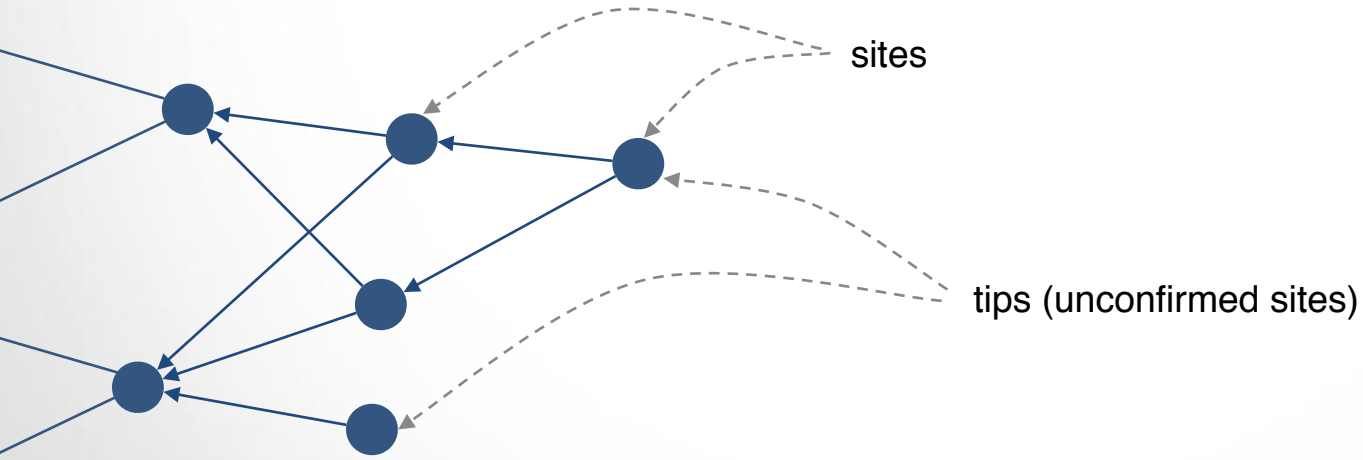
Each transaction is a small block that reference two previous ones



mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

Each transaction is a small block that reference two previous ones

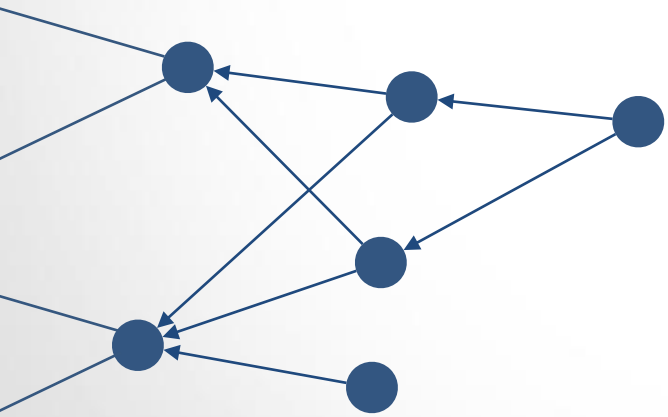


A new site and its parents should not create conflicts.

mart Contracts → IOTA BlockDAG → Infinite Scalability

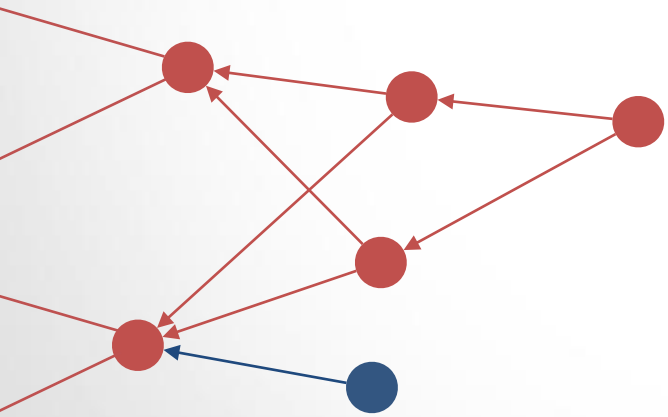
The Tangle (IOTA)

How to read a value?



mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)



How to read a value?

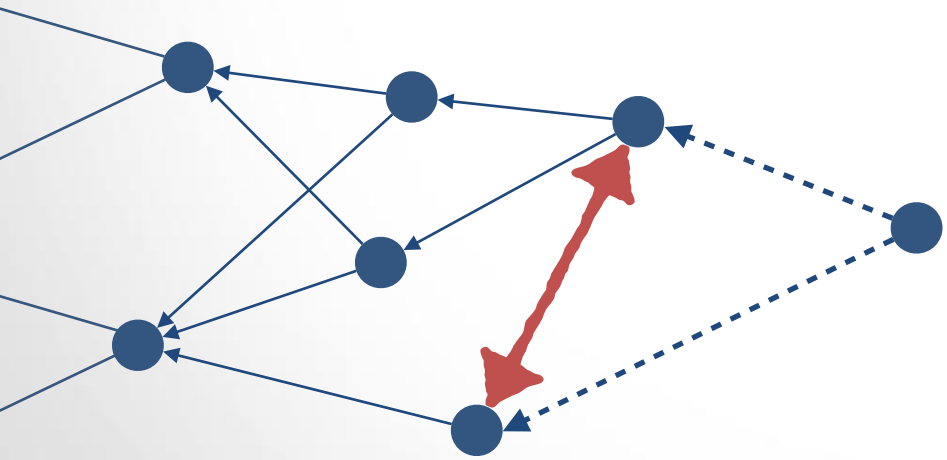
If you take a tip, you can order transactions and do the same as in a blockchain

mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

How to read a value?

What if tips are conflicting?



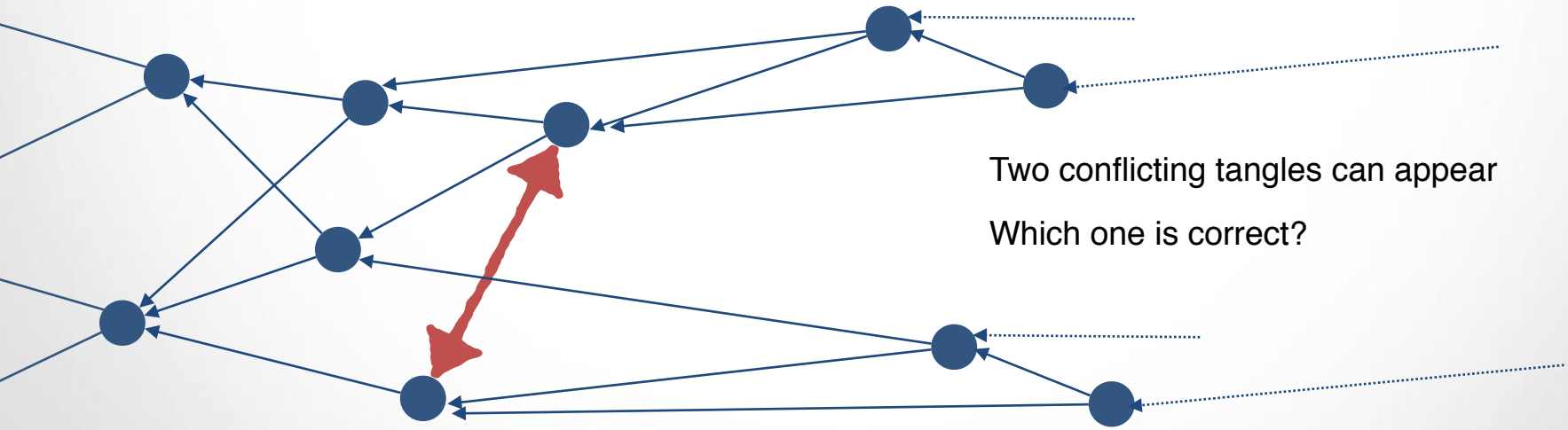
A new site cannot confirm conflicting sites

mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

How to read a value?

What if tips are conflicting?

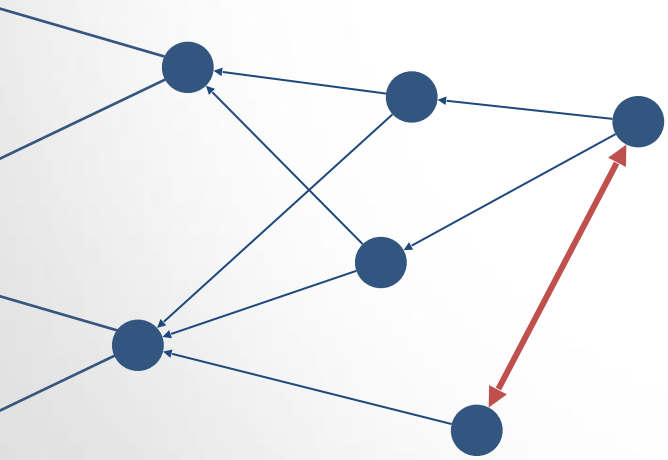


Two conflicting tangles can appear

Which one is correct?

mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

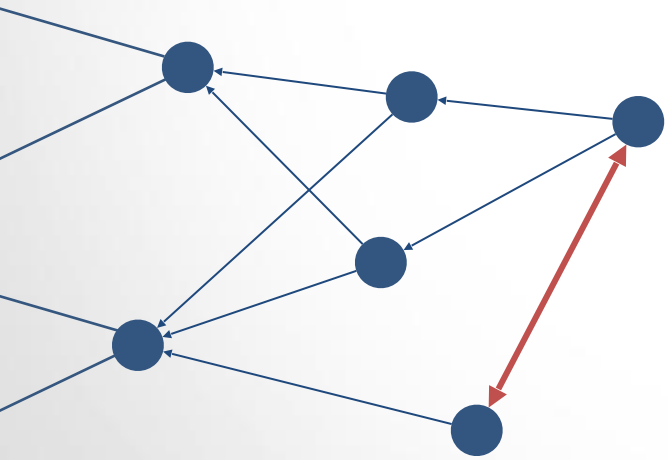


Tip Selection Algorithm (TSA):

- so we know how to read values
- so we know where to extend the Tangle

mart Contracts → IOTA BlockDAG → Infinite Scalability

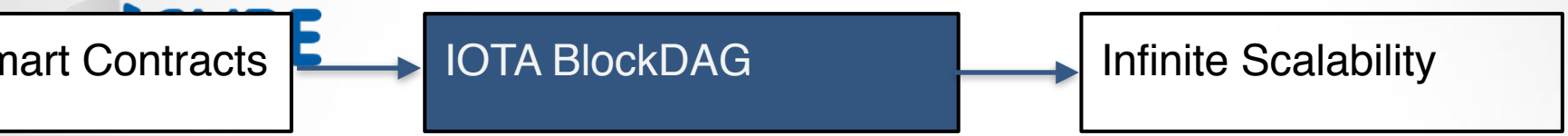
The Tangle (IOTA)



Tip Selection Algorithm (TSA):

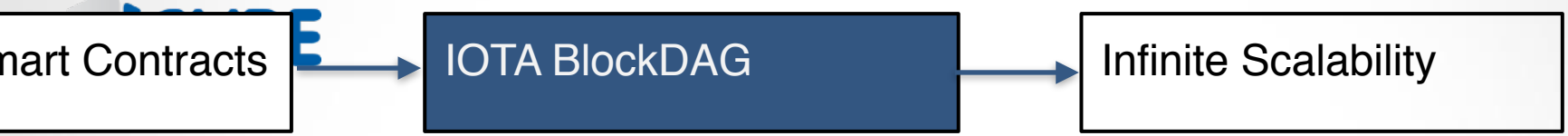
- so we know how to read values
- so we know where to extend the Tangle

In Bitcoin, we read values from, and we try to extend, the longest chain. If you don't follow this, you'll lose money.



The Tangle (IOTA)

In the Tangle, forks are ok if not conflicting



The Tangle (IOTA)

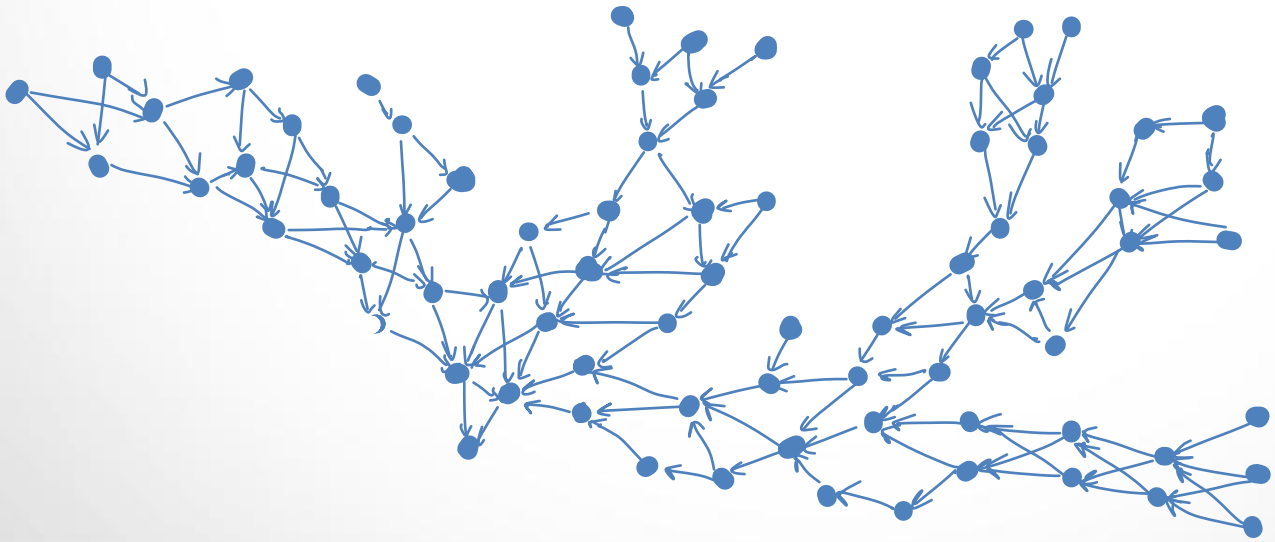
In the Tangle, forks are ok if not conflicting

But conflicting forks are worst in this case

The Tangle (IOTA)

In the Tangle, forks are ok if not conflicting

But conflicting forks are worst in this case

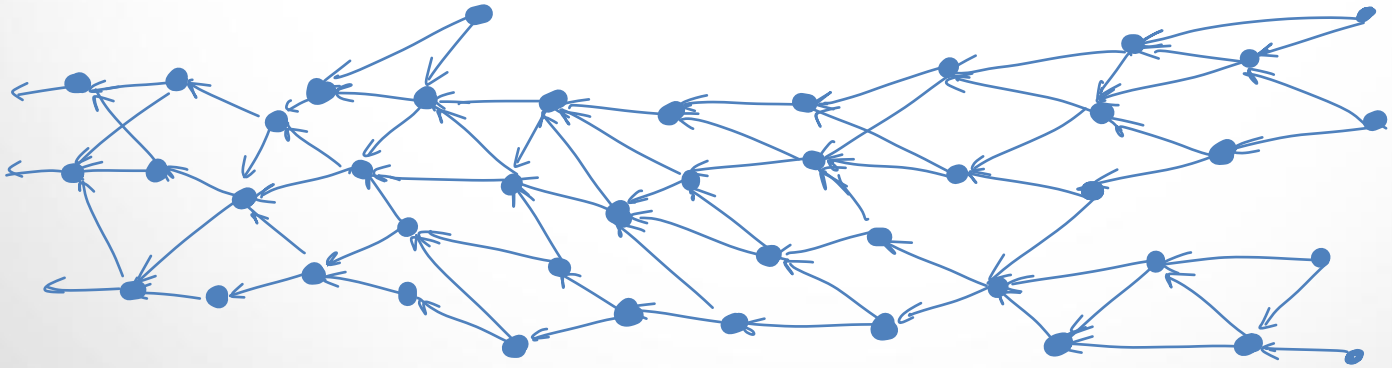


mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

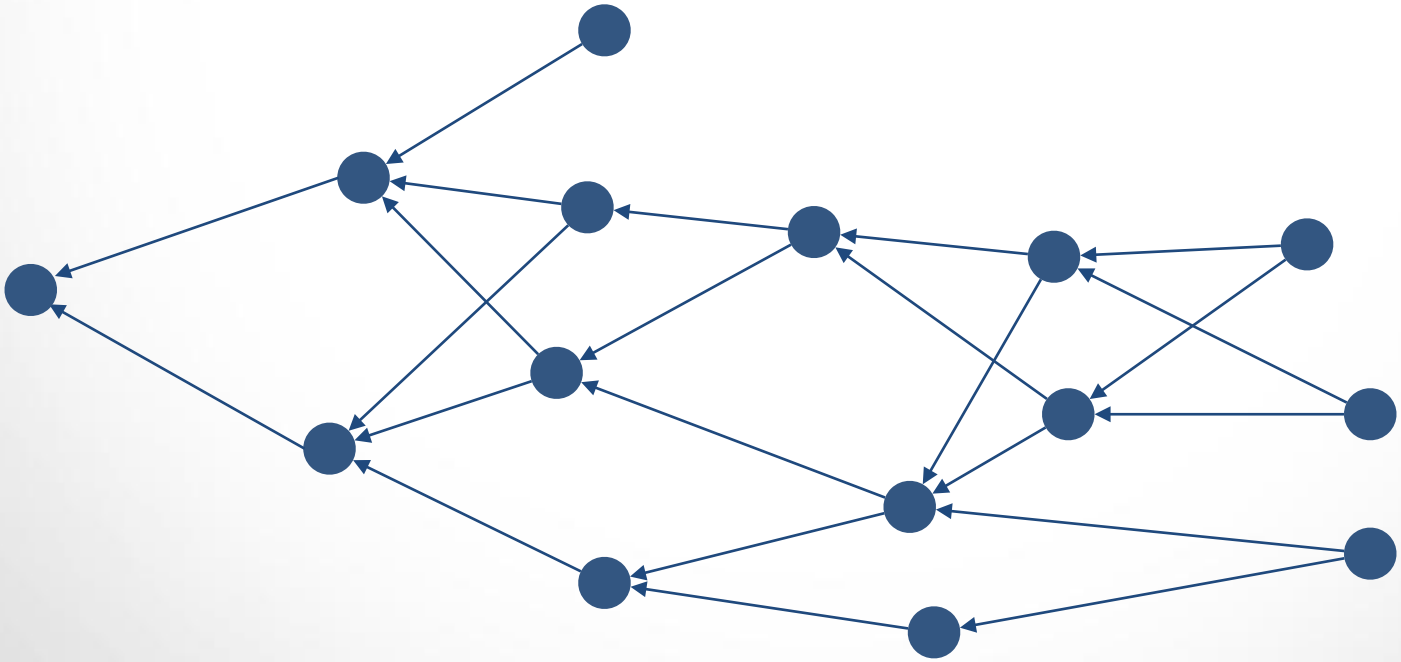
In the Tangle, forks are ok if not conflicting

So its better to have something like this



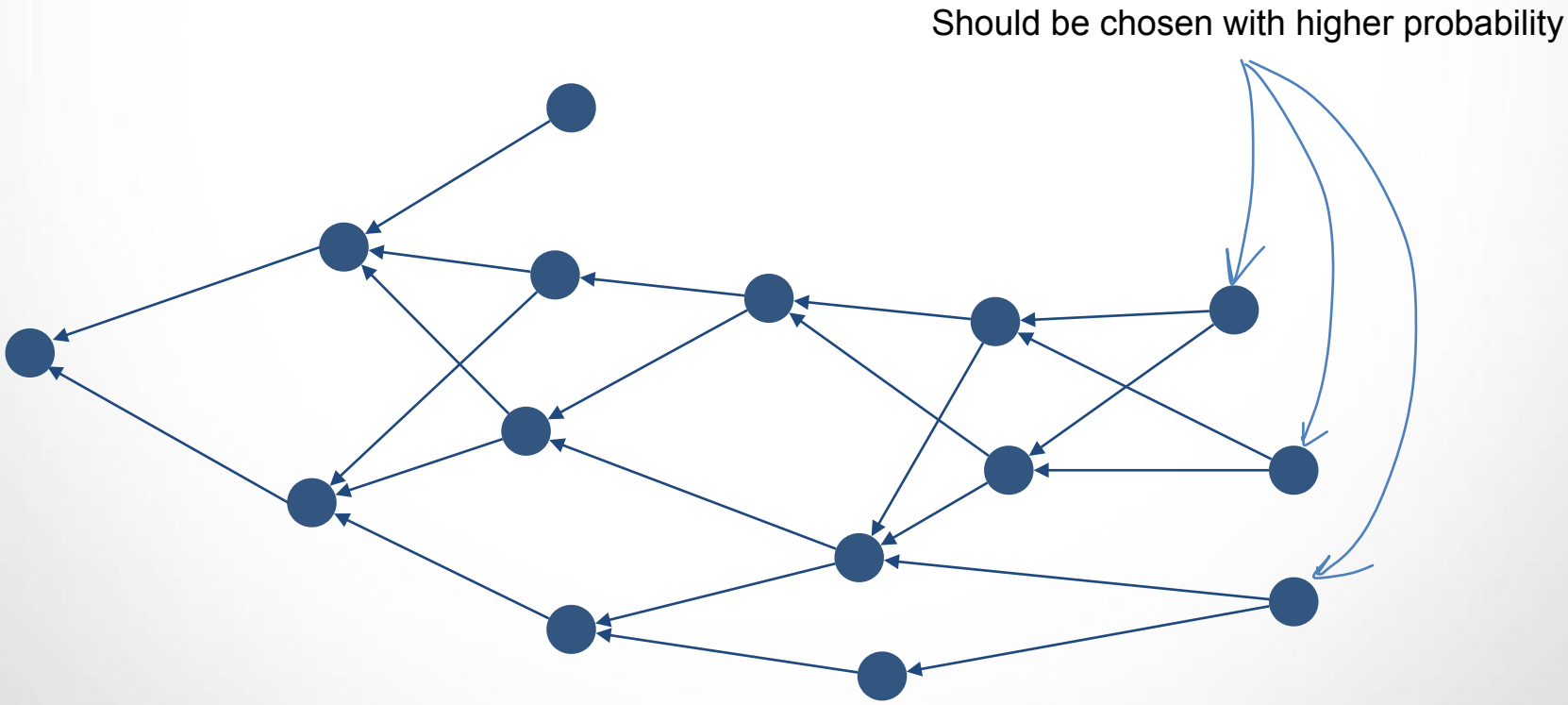
mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)



mart Contracts → IOTA BlockDAG → Infinite Scalability

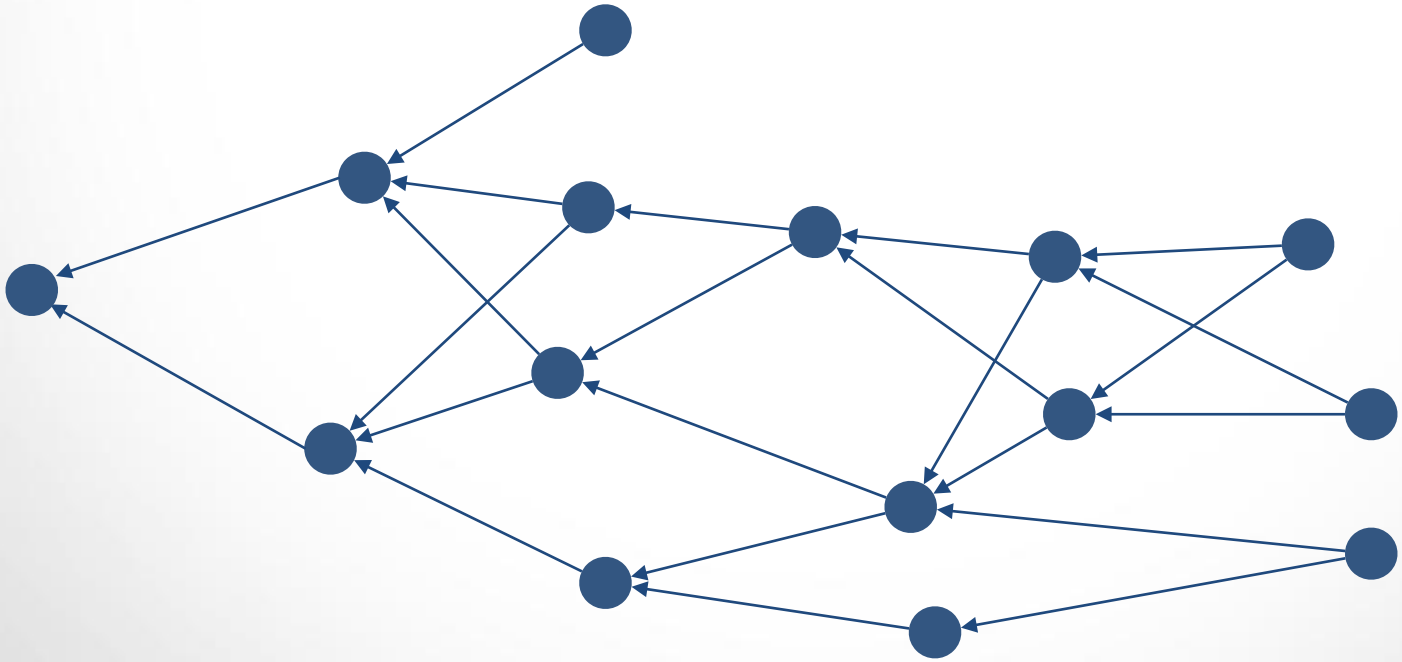
The Tangle (IOTA)



mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

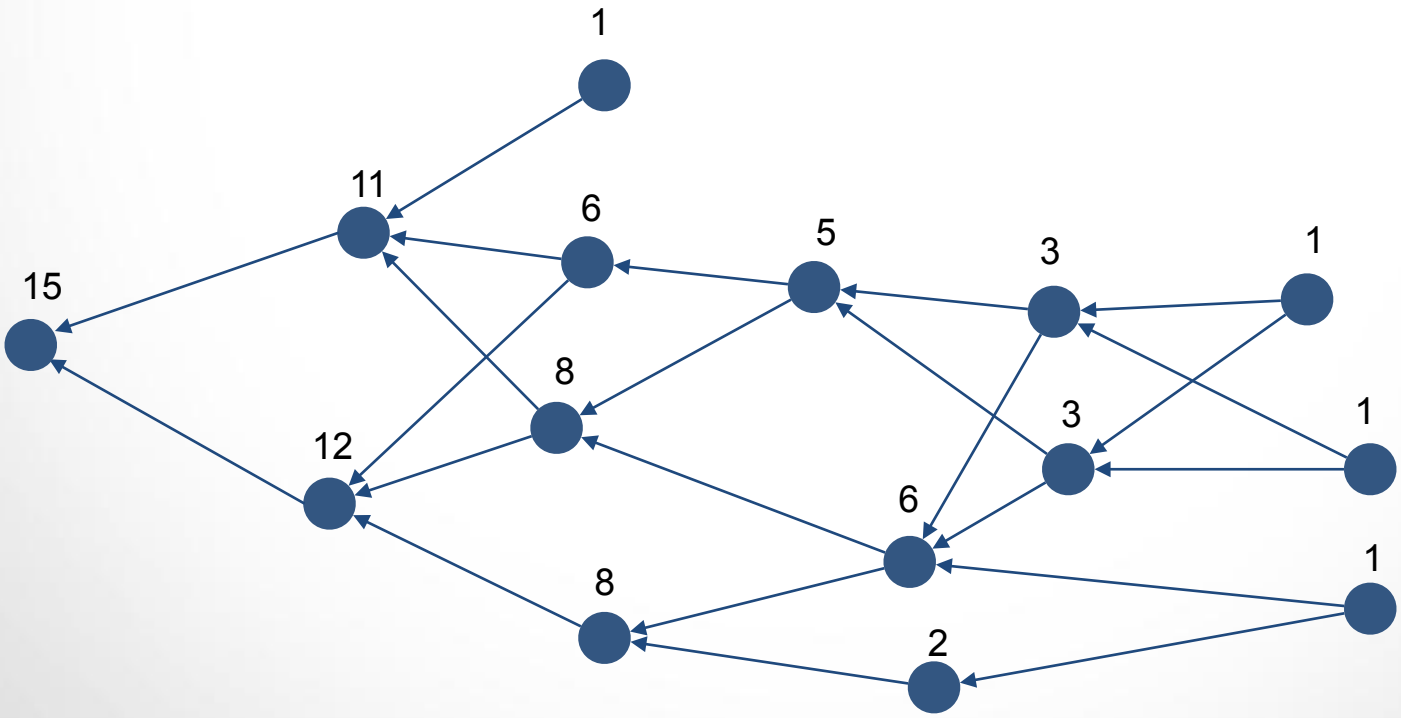
Compute cumulative weight to each site



mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

Compute cumulative weight to each site

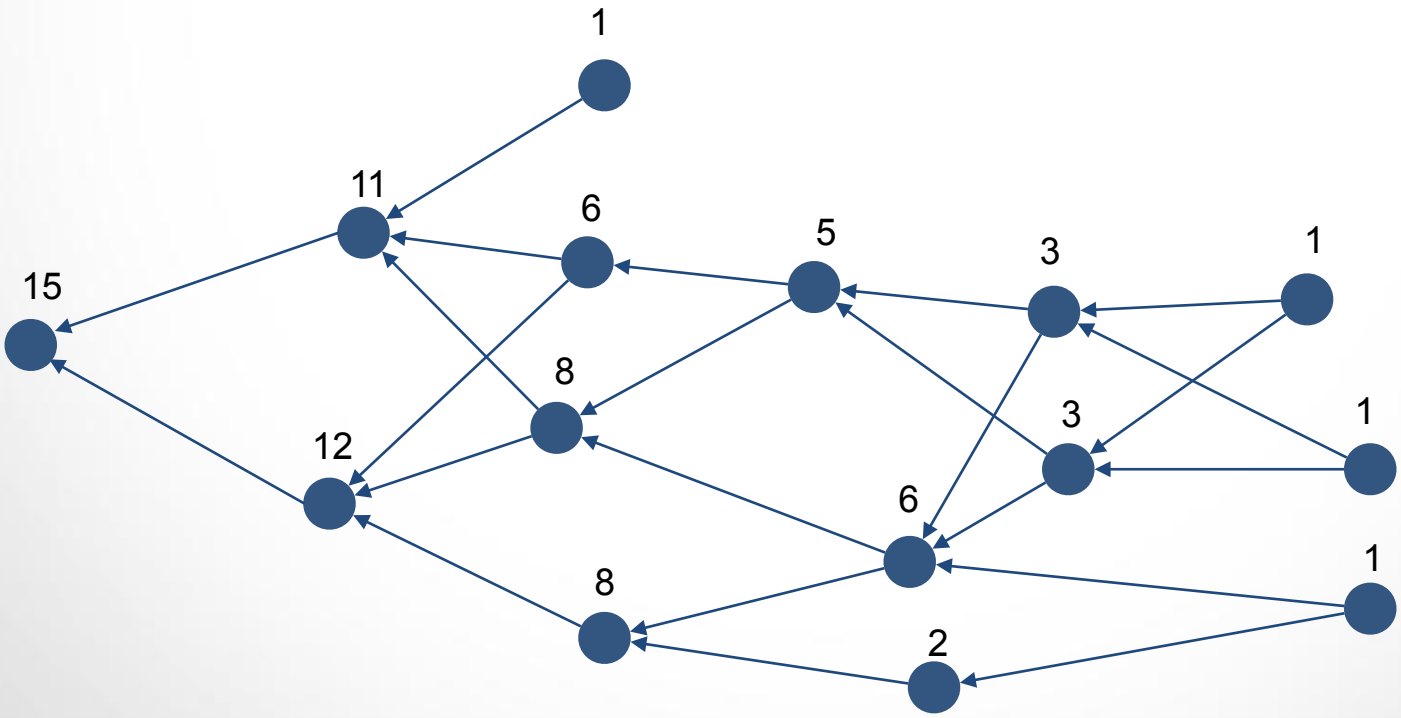


mart Contracts → IOTA BlockDAG → Infinite Scalability

The Tangle (IOTA)

Compute cumulative weight to each site

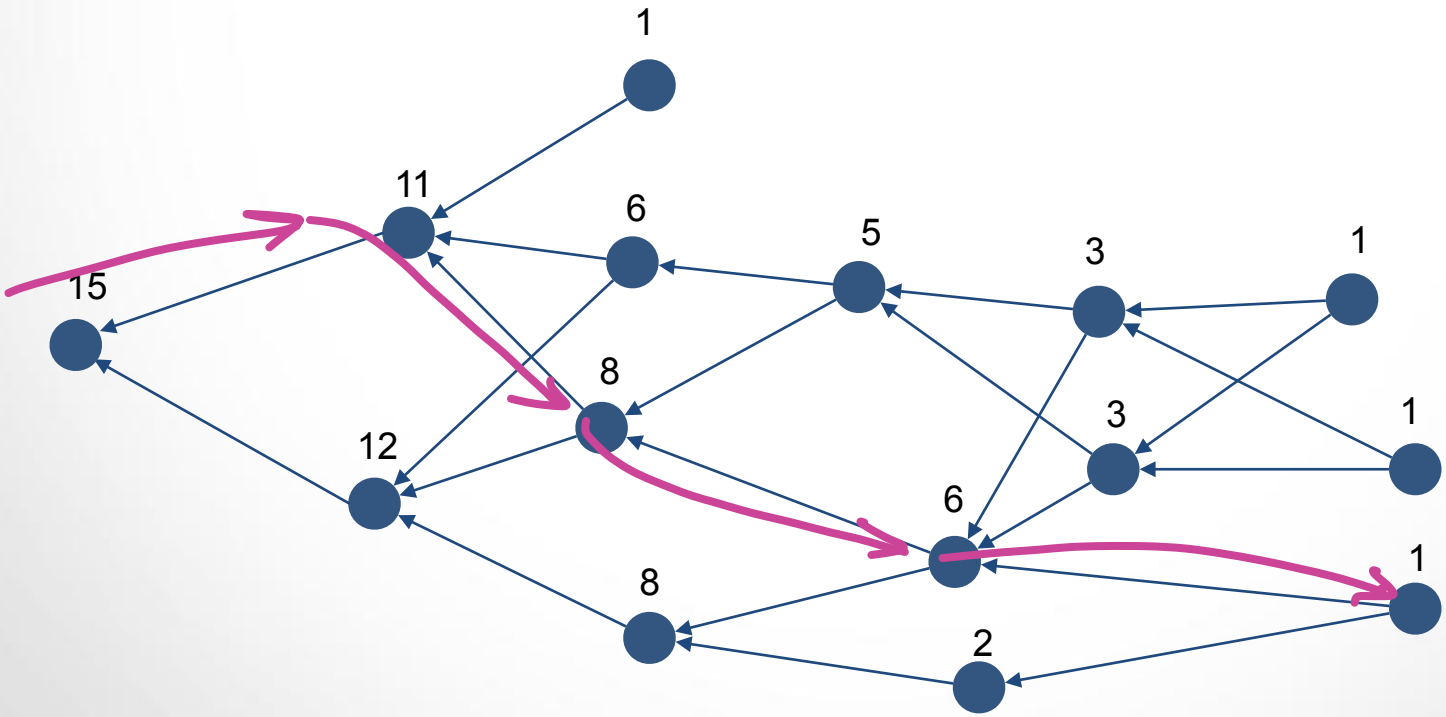
Perform a random walk



The Tangle (IOTA)

Compute cumulative weight to each site

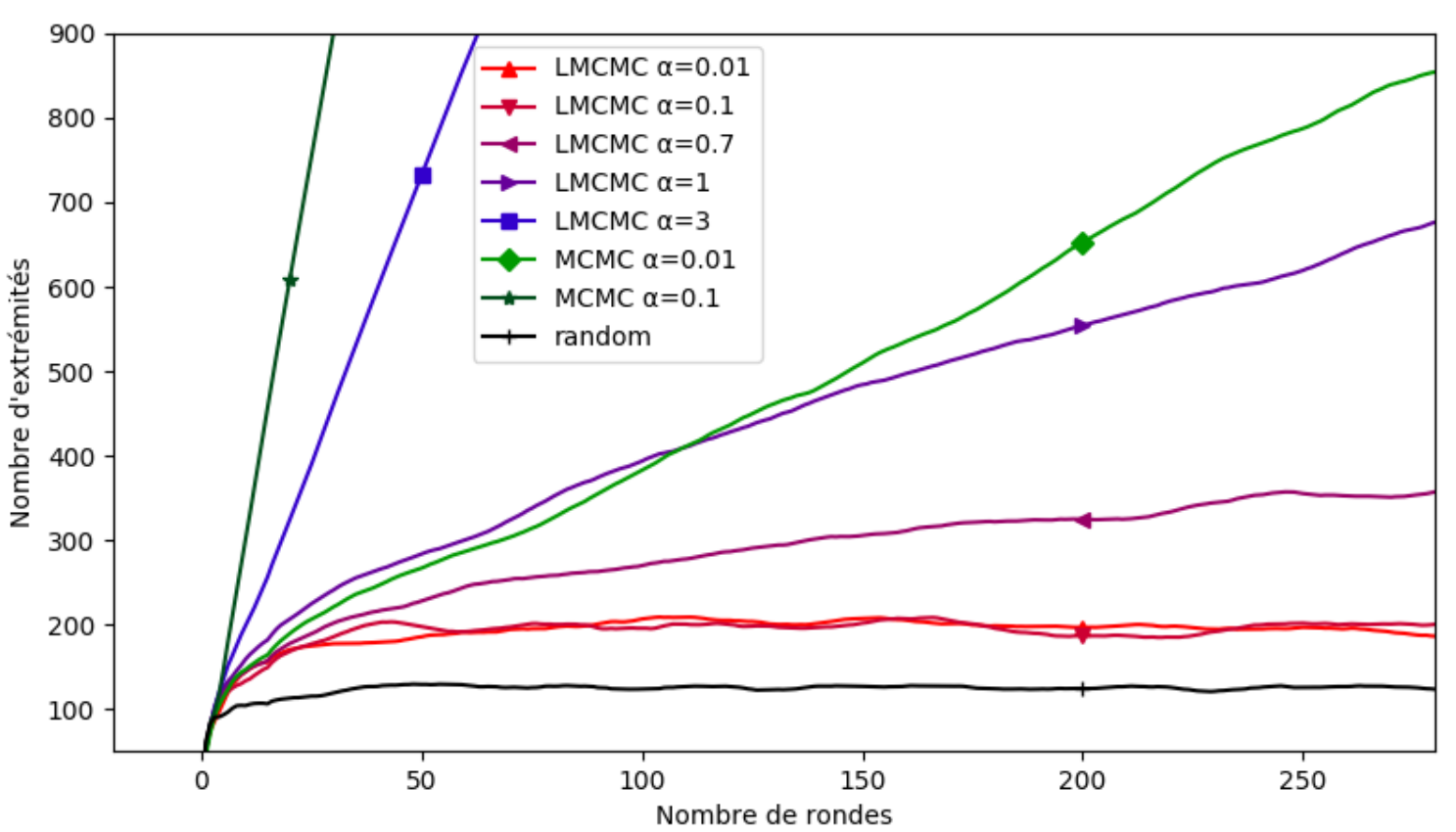
Perform a random walk

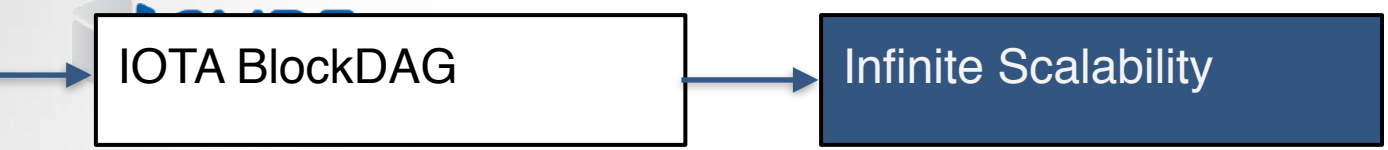


mart Contracts → IOTA BlockDAG → Infinite Scalability

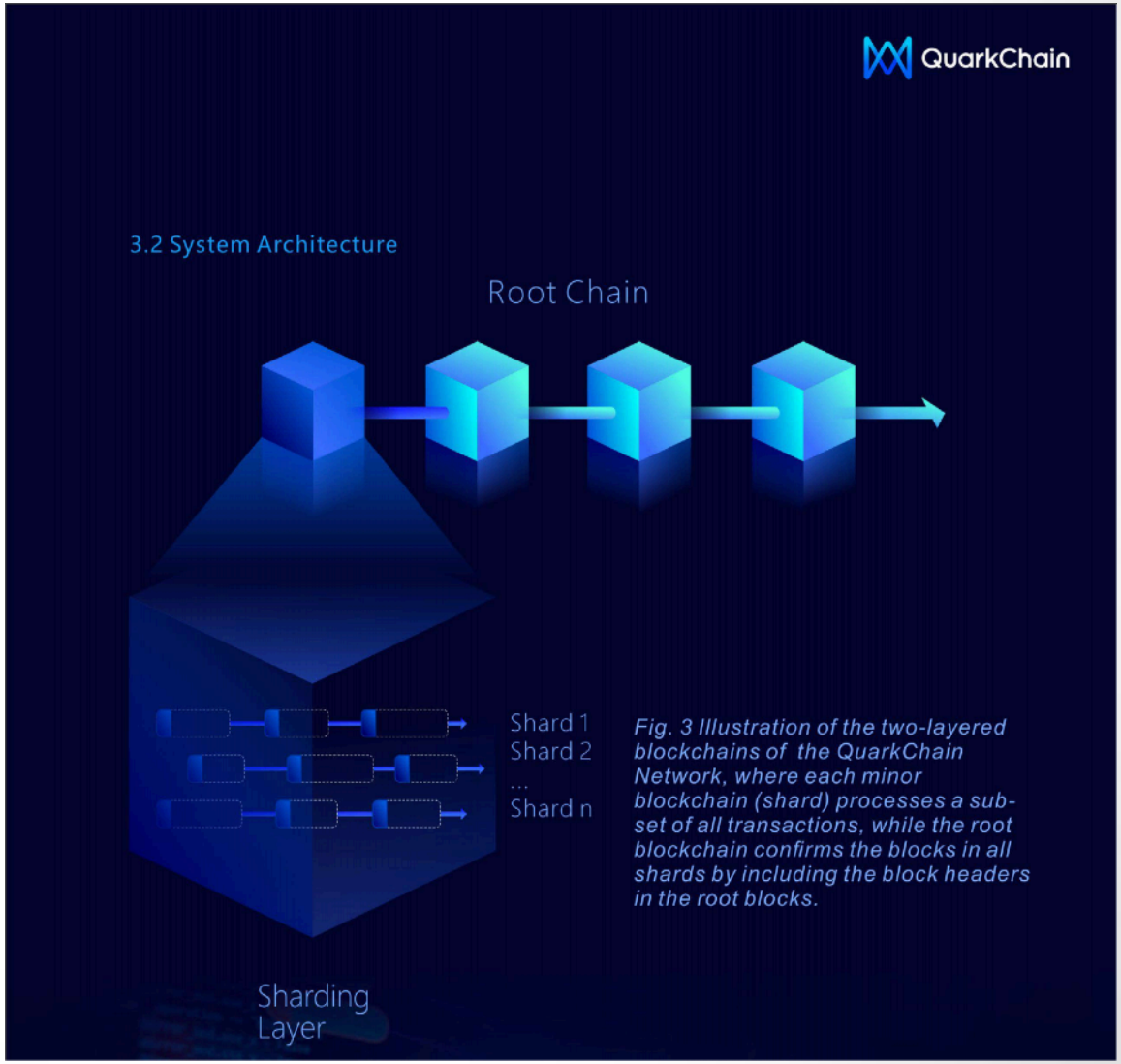
The Tangle (IOTA)

How many tips are left behind ?




IOTA BlockDAG
Infinite Scalability

QuarkChain white paper :

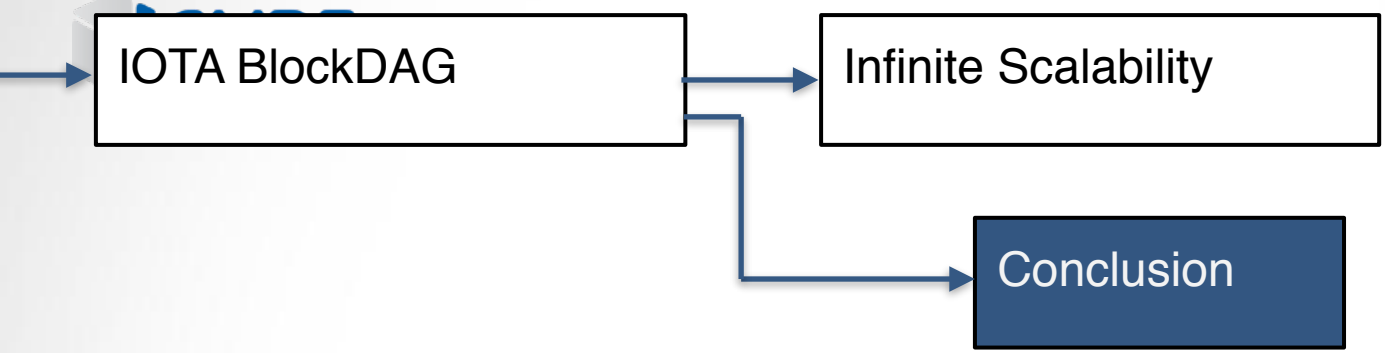


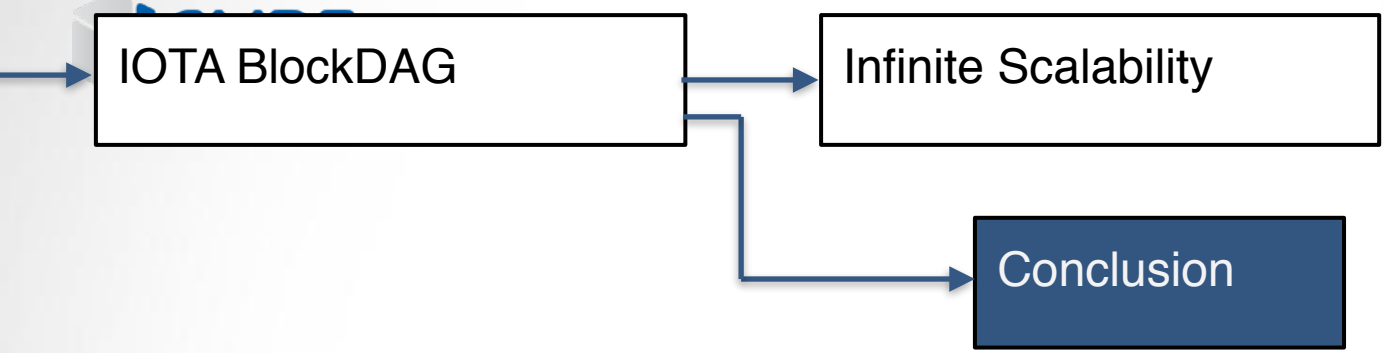
https://quarkchain.io/QUARK%20CHAIN%20Public%20Version%200.3.4.pdf

QuarkChain white paper :

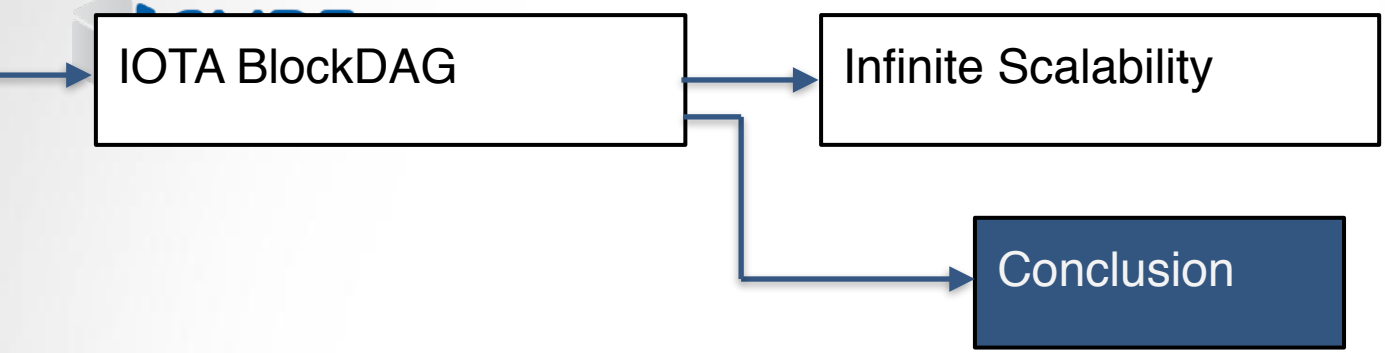
The QuarkChain Network is helping move blockchain into the next generation by increasing the current TPS capacity several-thousand fold of what it is now, to a projected about 100,000 TPS. The network being built is project to be free of congestion, making it affordable for all usage scenarios that demand speed and volume. We envision such a network applied to industries that demand higher TPS. Ultimately, the QuarkChain Network aims to build a high-throughput network to support applications such asdistributed social media, high frequency trading, Internet of Things (IoT), gaming, and payment.

https://quarkchain.io/QUARK%20CHAIN%20Public%20Version%200.3.4.pdf


IOTA BlockDAG
Infinite Scalability
Conclusion

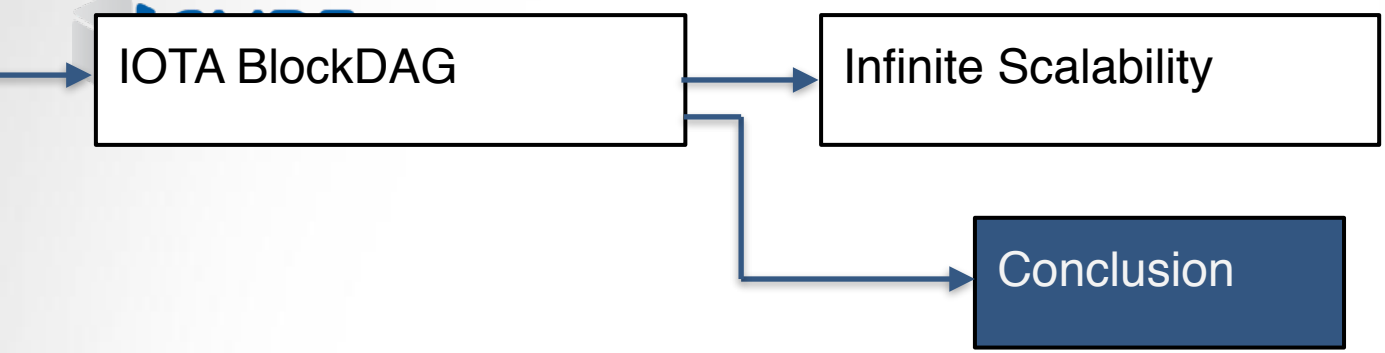


Take Away



Take Away

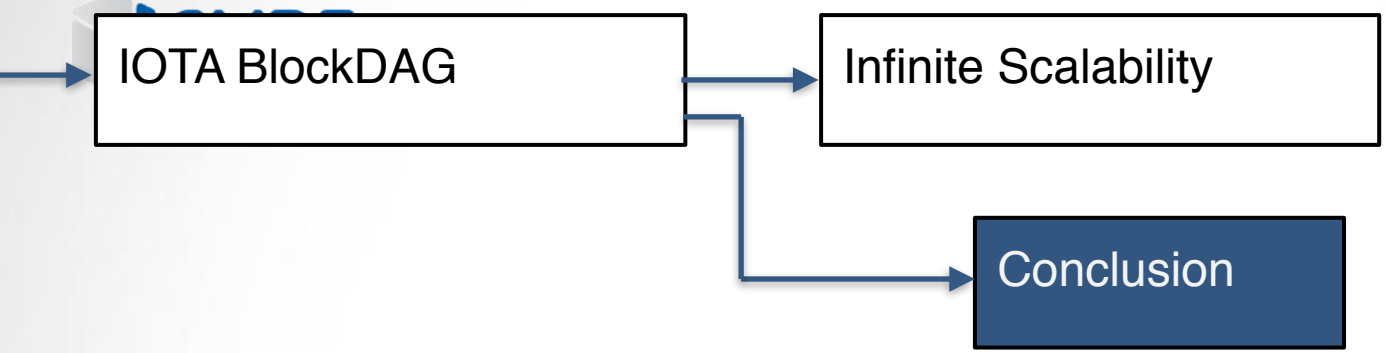
$$\text{quality of a protocol} = \frac{1}{\text{graphic quality of the white paper}}$$



Take Away

$$\text{quality of a protocol} = \frac{1}{\text{graphic quality of the white paper}}$$

Ethereum Smart contracts are smart

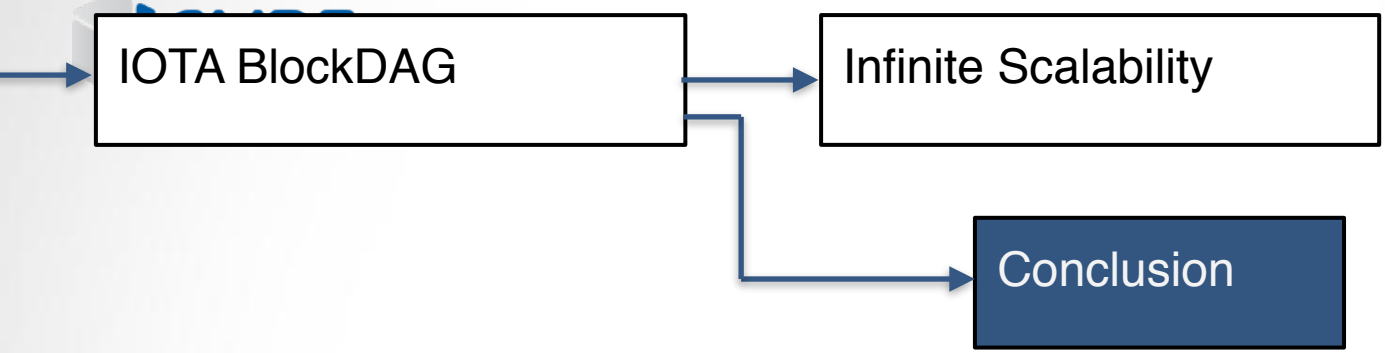


Take Away

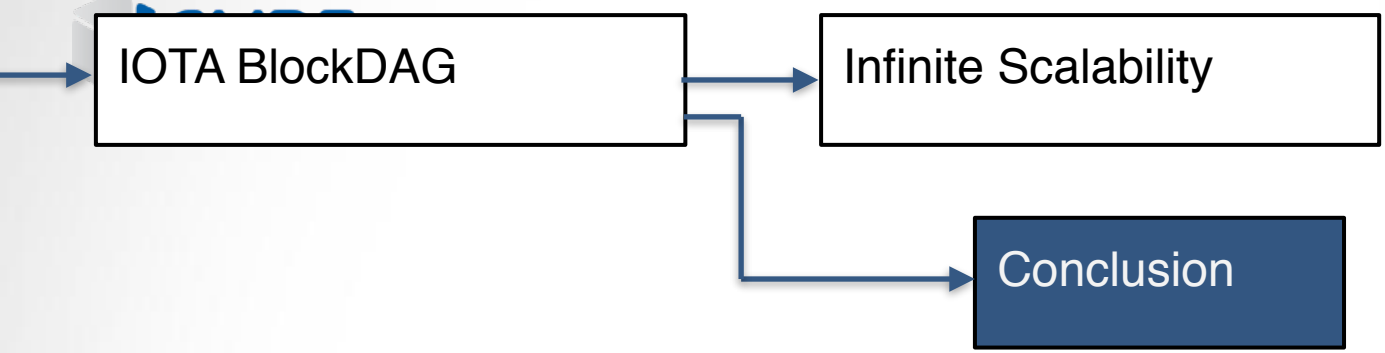
$$\text{quality of a protocol} = \frac{1}{\text{graphic quality of the white paper}}$$

Ethereum Smart contracts are smart

IOTA tangle is tangled

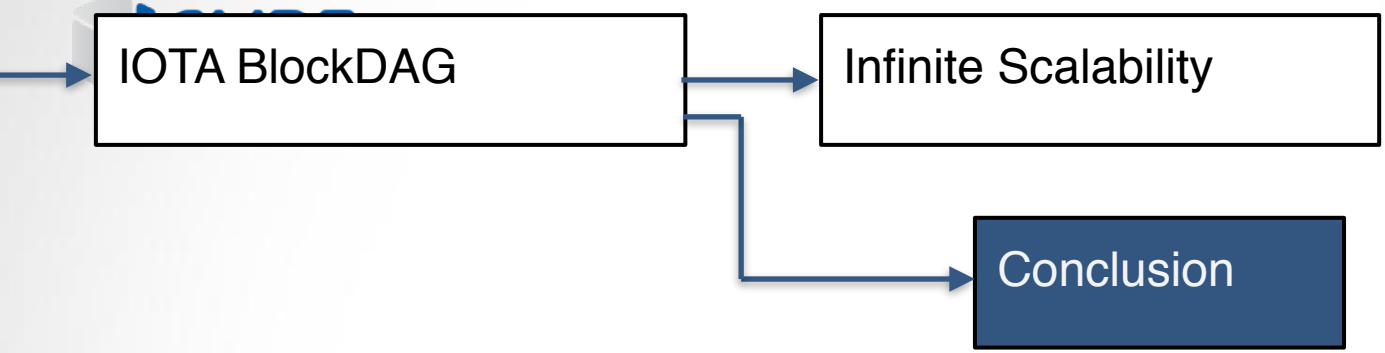


Perspectives



Perspectives

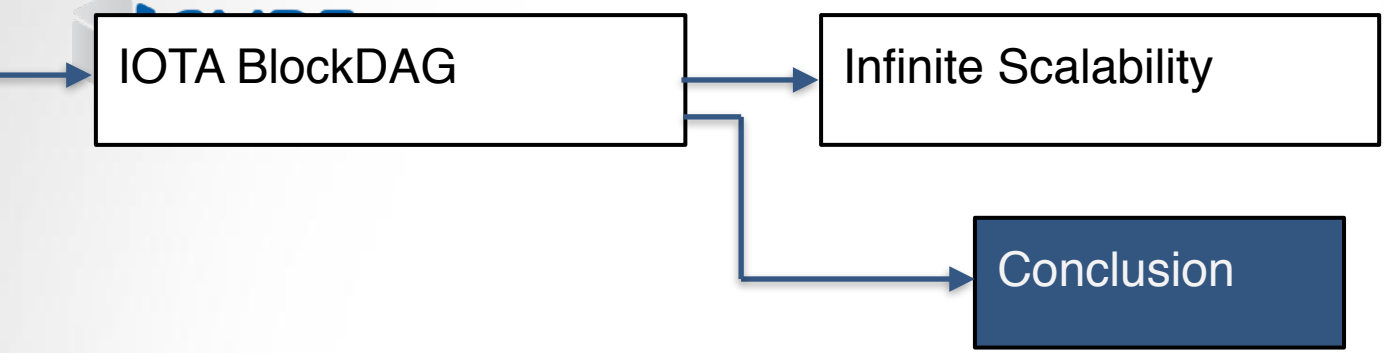
Lots of interesting open problems about :



Perspectives

Lots of interesting open problems about :

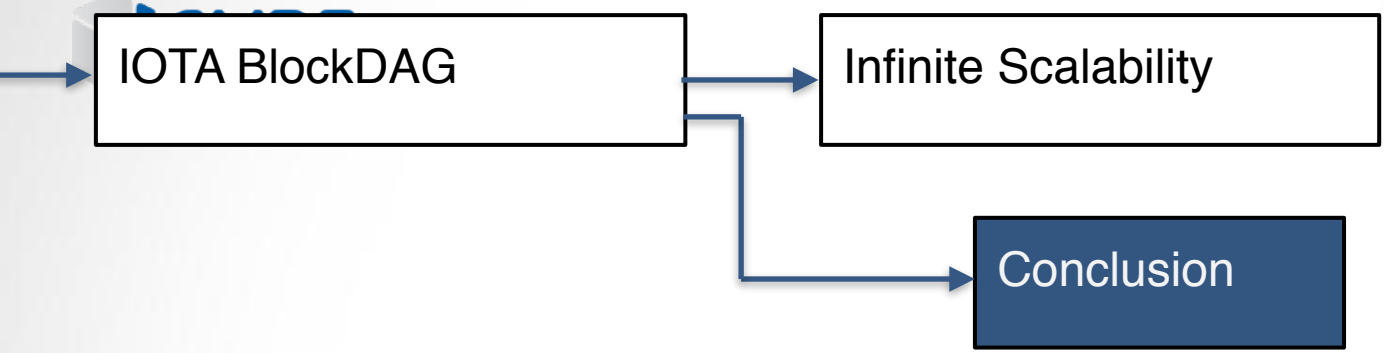
- Network impacts on blockchain security



Perspectives

Lots of interesting open problems about :

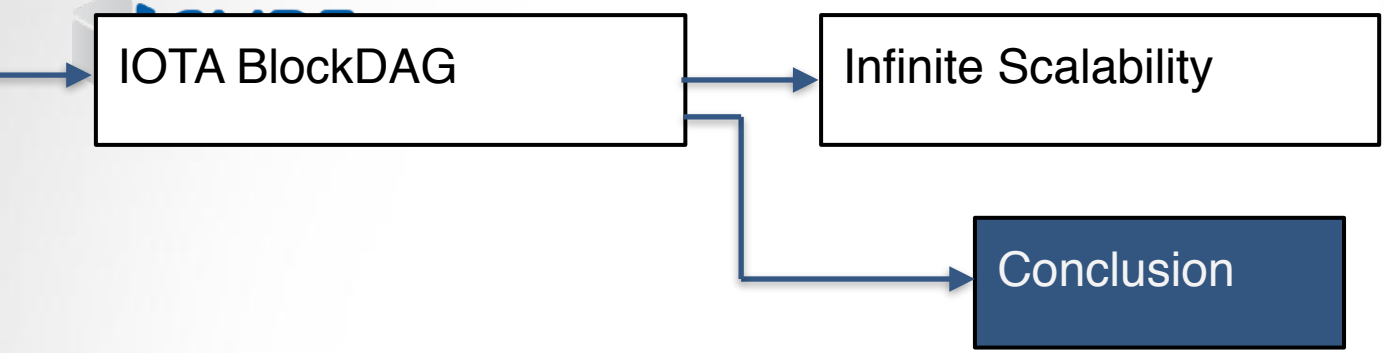
- Network impacts on blockchain security
- Properly defining the attacks



Perspectives

Lots of interesting open problems about :

- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

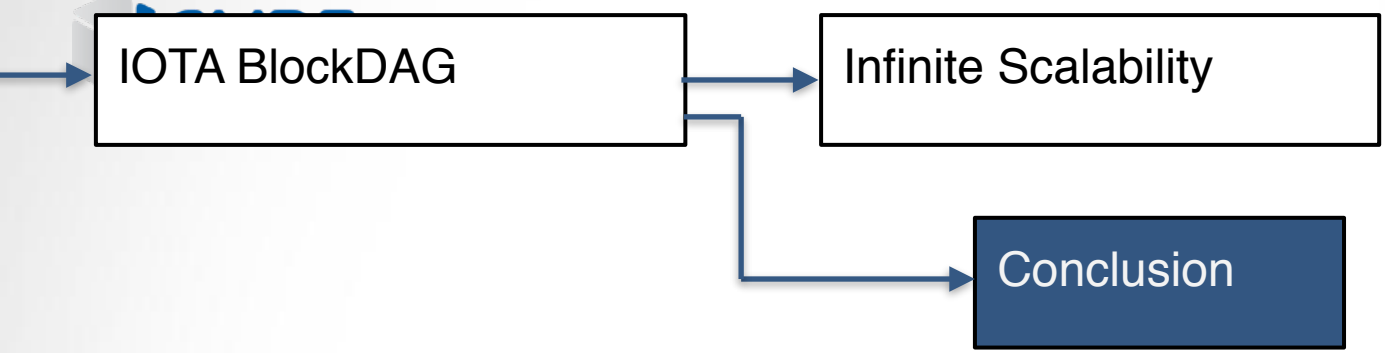


Perspectives

Lots of interesting open problems about :

- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :



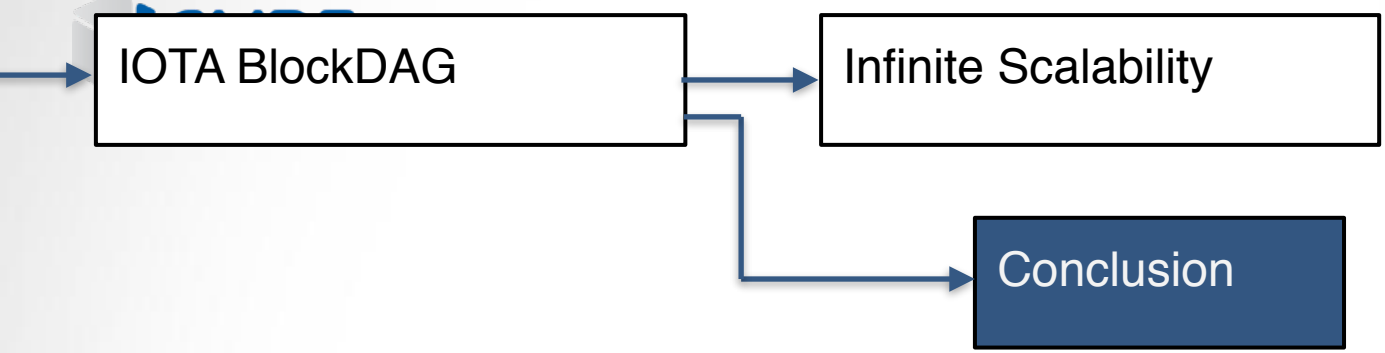
Perspectives

Lots of interesting open problems about :

- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :

- How to store efficiently blockchain data



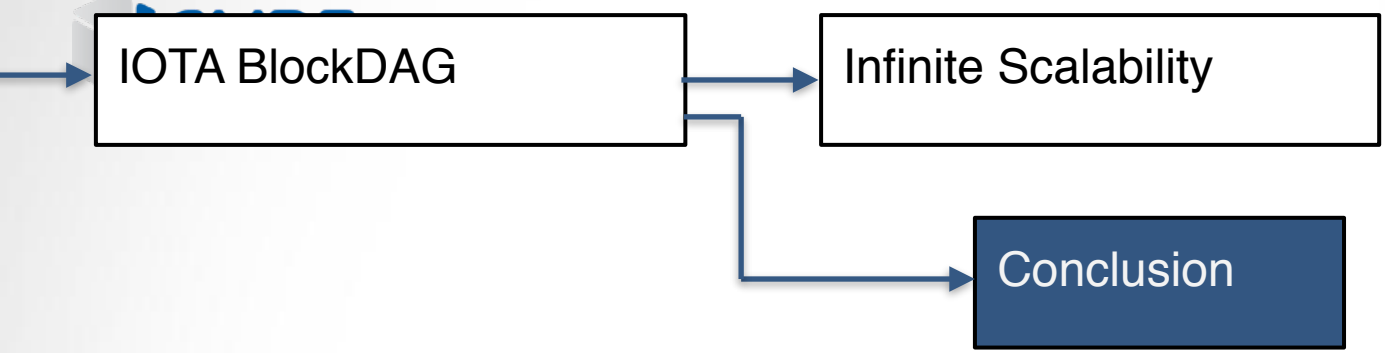
Perspectives

Lots of interesting open problems about :

- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :

- How to store efficiently blockchain data
- Lots of algorithms for quick graph traversal / transactions validation are specific to blockchain and could be optimized



Perspectives

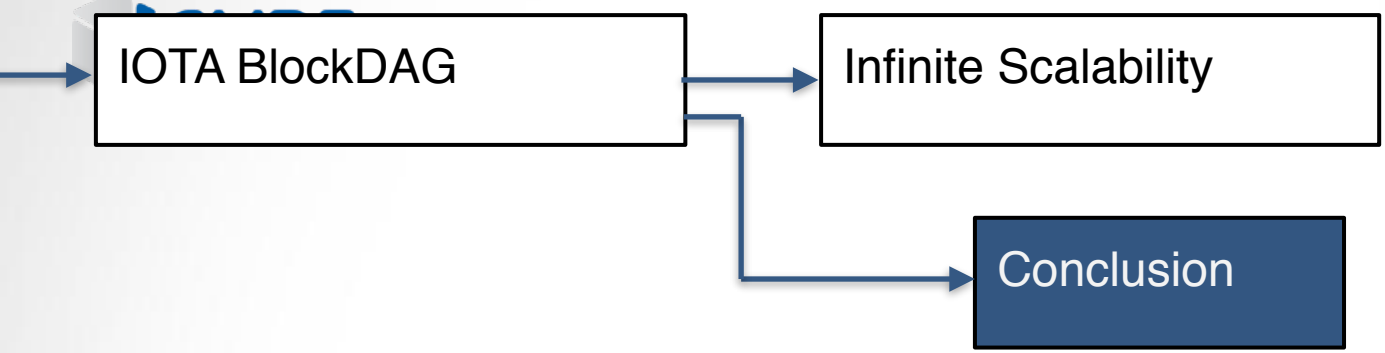
Lots of interesting open problems about :

- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :

- How to store efficiently blockchain data
- Lots of algorithms for quick graph traversal / transactions validation are specific to blockchain and could be optimized

And also :



Perspectives

Lots of interesting open problems about :

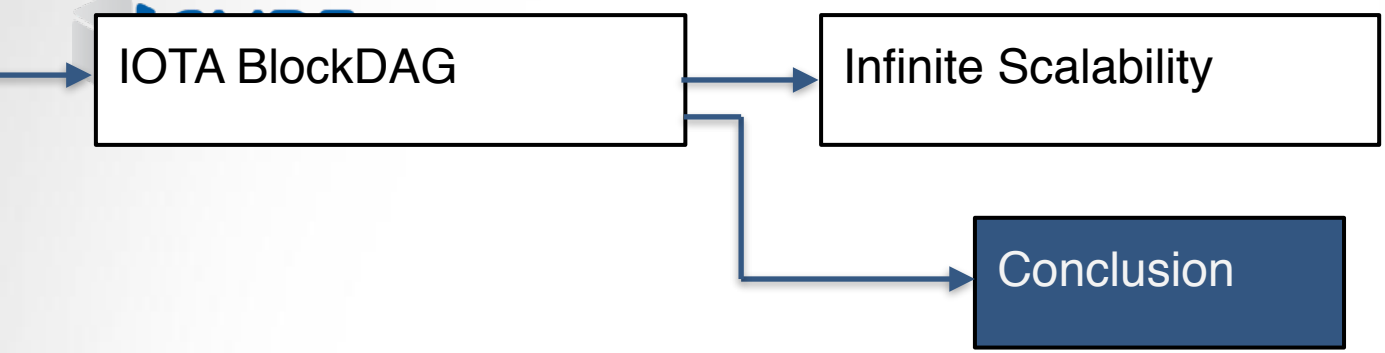
- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :

- How to store efficiently blockchain data
- Lots of algorithms for quick graph traversal / transactions validation are specific to blockchain and could be optimized

And also :

- What programs could run on the blockchain for distributed democracy / authority / management system for IoT devices



Perspectives

Lots of interesting open problems about :

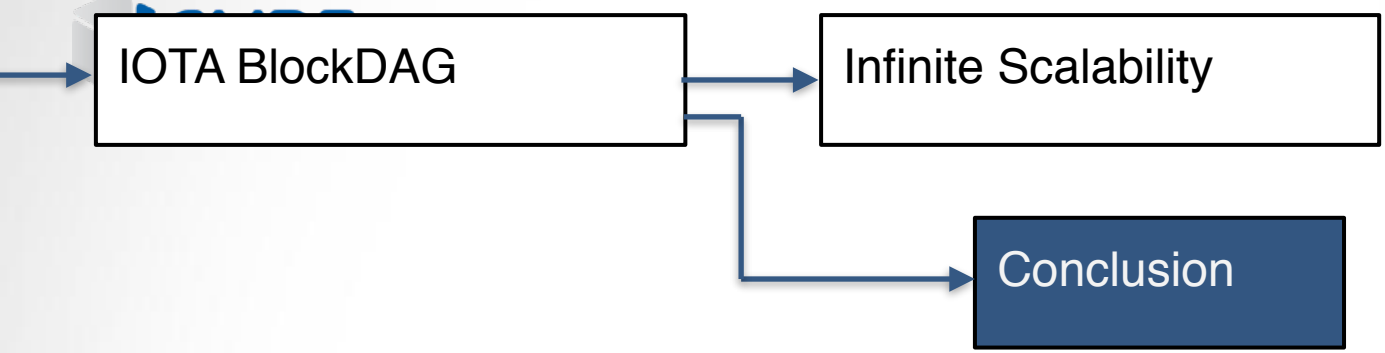
- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :

- How to store efficiently blockchain data
- Lots of algorithms for quick graph traversal / transactions validation are specific to blockchain and could be optimized

And also :

- What programs could run on the blockchain for distributed democracy / authority / management system for IoT devices
- How to analyze blockchain history to predict …



Perspectives

Lots of interesting open problems about :

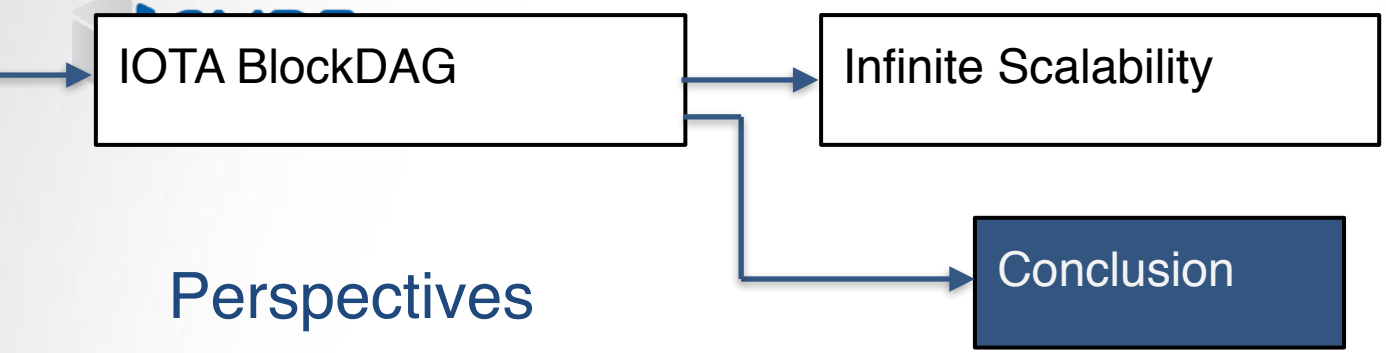
- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :

- How to store efficiently blockchain data
- Lots of algorithms for quick graph traversal / transactions validation are specific to blockchain and could be optimized

And also :

- What programs could run on the blockchain for distributed democracy / authority / management system for IoT devices
- How to analyze blockchain history to predict …



Perspectives

Lots of interesting open problems about :

- Network impacts on blockchain security
- Properly defining the attacks
- How to defend against them

Also :

- How to store efficiently blockchain data
- Lots of algorithms for quick graph traversal / transactions validation are specific to blockchain and could be optimized

And also :

- What programs could run on the blockchain for distributed democracy / authority / management system for IoT devices
- How to analyze blockchain history to predict …

Thank you for your attention !

Quentin Bramas. The Stability and the Security of the Tangle. 2018. 〈hal-01716111v2〉