Stochastic Models for blockchain analysis

Consensus protocols

Pierre-O. Goffard

Institut de Science Financières et d'Assurances
pierre-olivier.goffard@univ-lyon1.fr

30 mars 2022

Consensus protocol

Definition

Algorithm to allows the full nodes to agree on a common data history

It must rely on the scarce resources of the network

- bandwidth
- computational power
- storage (disk space)

Types of consensus protocols

1. Voting based

   L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, pp. 382–401, July 1982.

2. Leader based
   - Proof-of-Work (computational power)
   - Proof-of-Capacity and Proof-of-Spacetime (storage)
   - Proof-of-Interaction (bandwidth)
   - Proof-of-Stake (tokens)

Conflict resolution in blockchain

Fork

A fork arises when there is a disagreement between the nodes resulting in several branches in the blockchain.

LCR

The *Longest Chain Rule* states that if there exist several branches of the blockchain then the longest should be trusted.

In practice

- A branch can be considered legitimate if it is $k \in \mathbb{N}$ blocks ahead of its pursuers.
- Fork can be avoided when

$$\text{block appending time} > \text{propagation delay}$$

Two generals problem

Two nodes who must agree are communicating through an unreliable link.

- Analogy with two generals besieging a city

The generals exchange messages through enemy territory

G1

```
"I will attack tomorrow at dawn, if you confirm"
```

G2

```
"I will follow your lead, if you confirm"
```

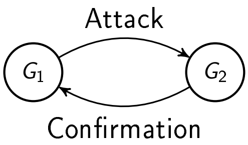


Figure – Message and confirmation loop

Byzantine General problem

n generals must agree on a common battle plan, to either

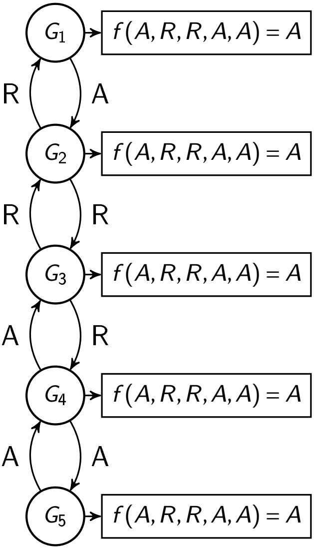
- Attack (A)
- Retreat (R)

Problem

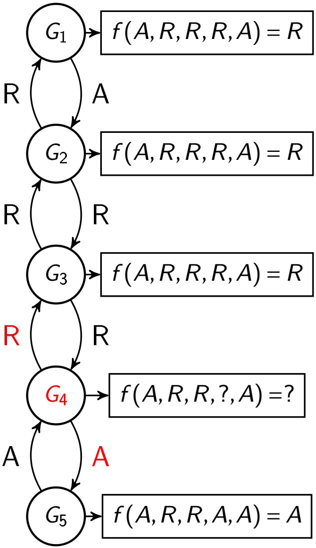
There are $m < n$ traitors among the generals

1. message $m(i,j)$ is sent to general j by general i
2. Consensus is reached as general j applies

$$f(\{m(i,j);\ i = 1,\dots,n\}) = \begin{cases} A, & \text{if } \sum_{i=1}^{n} \mathbb{I}_{m(i,j)=A} > n/2, \\ R, & \text{else.} \end{cases}$$



(a) No traitor



(b) One traitor

Figure – Majority vote with or without a traitor

Commanders and Lieutenants

One general is the commander while the others are the lieutenants

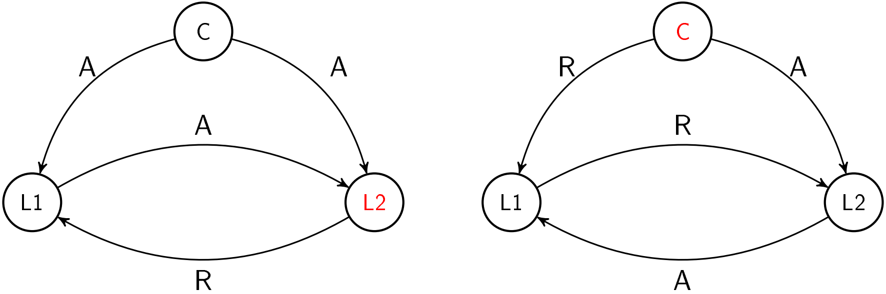
Objective

Design an algorithm so that the following conditions are met :

- C1 All the loyal lieutenants obey the same order
- C2 If the commanding general is loyal, then every loyal lieutenants obey the order he sends

Byzantine Fault Tolerance Theorem (Lamport et al.)

There are no solution to the Byzantine General problem for $n < 3m+1$ generals, where m is the number of traitors.



(a) Commander is loyal

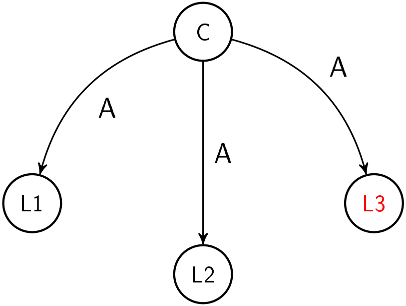
(b) Commander is a traitor

Figure – Majority vote with or without a traitor

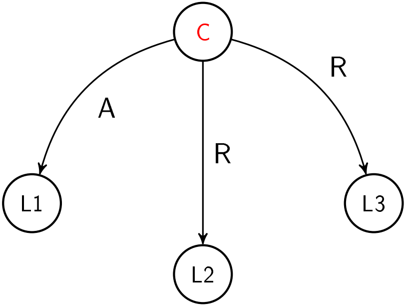
Algorithm The Oral message algorithm OM(m)

```
if m = 0 then ;
    for i = 1 → n−1 do
        Commander sends v_i = v to lieutenant i
        Lieutenant i set their value to v
    end for
end if
if m > 0 then ;
    for i = 1 → n−1 do
        Commander sends v_i to lieutenant i
        Lieutenant i uses OM(m-1) to communicate v_i to the n−2 lieutenants
    end for
    for i = 1 → n−1 do
        Lieutenant i set their value to f(v_1, ..., v_{n−1})
    end for
end if
```

$n = 4$ and $m = 1$: Step 1



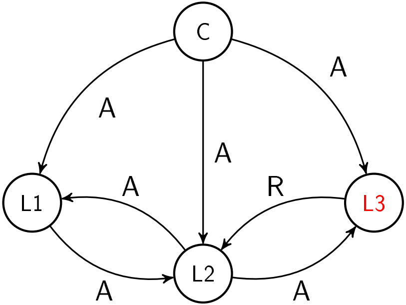
(a) Commander is loyal



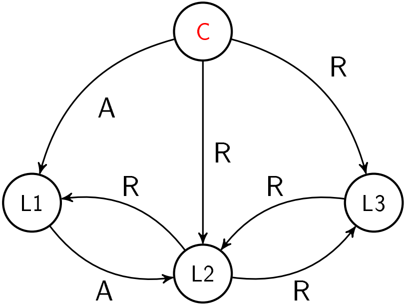
(b) Commander is a traitor

Figure – Illustration of the OM(m) algorithm in the case where $n = 4$ and $m = 1$.

$n = 4$ and $m = 1$: Step 2



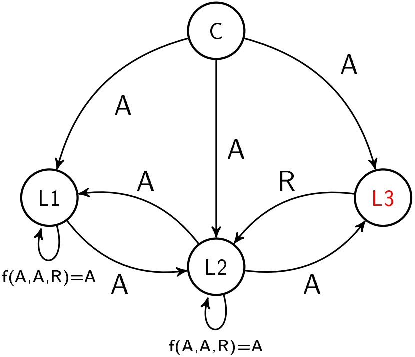
(a) Commander is loyal



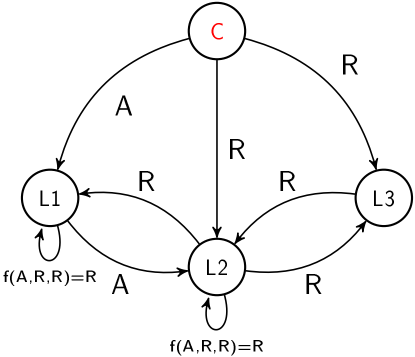
(b) Commander is a traitor

Figure – Illustration of the OM(m) algorithm in the case where $n = 4$ and $m = 1$.

$n = 4$ and $m = 1$: Step 3



(a) Commander is loyal, C1 and C2



(b) Commander is a traitor, C1

Figure – Illustration of the OM(m) algorithm in the case where $n = 4$ and $m = 1$.

The problem with majority vote

The OM algorithm requires to send n^{m+1}

- ⚠ Communication overhead
- ⚠ Denial of service

Solution

Leader based protocols !

Proof-of-Work

Objective

Elect a leader based on computational effort to append the next block.

What's inside a block ?

A block consists of

- a header
- a list of "transactions" that represents the information recorded through the blockchain.

The header usually includes

- the date and time of creation of the block,
- the block height which is the index inside the blockchain,
- the hash of the block
- the hash of the previous block.

Question

What is the hash of a block ?

Cryptographic Hash function

A function that maps data of arbitratry size (message) to a bit array of fixed size (hash value)

$$h : \{0,1\}^* \mapsto \{0,1\}^d.$$

A good hash function is

- deterministic
- quick to compute
- One way
  - $\hookrightarrow$ For a given hash value $\overline{h}$ it is hard to find a message m such that
  $$h(m) = \overline{h}$$
- Colision resistant
  - $\hookrightarrow$ Impossible to find m_1 and m_2 such that
  $$h(m_1) = h(m_2)$$
- Chaotic
  $$m_1 \approx m_2 \Rightarrow h(m_1) \neq h(m_2)$$

SHA-256

The SHA-256 function which converts any message into a hash value of 256 bits.

Example

The hexadecimal digest of the message

`Blockastics is fantastic`

is

`60a147c28568dc925c347bce20c910ef90f3774e2501ac63344f3411b6a6bf79`

Mining a block

```
Block Hash: 1fc23a429aa5aaf04d17e9057e03371f59ac8823b1441798940837fa2e318aaa
Block Height: 0
Time:2022-02-25 12:42:04.560217
Nonce:0
Block data: [{'sender': 'Coinbase', 'recipient': 'Satoshi', 'amount': 100, 'fee': 0}, {'sen
der': 'Satoshi', 'recipient': 'Pierre-O', 'amount': 5, 'fee': 2}]
Previous block hash: 0
Mined: False
--------------
```

Figure – A block that has not been mined yet.

Mining a block

The maximum value for a 256 bits number is

$$T_{\max} = 2^{256} - 1 \approx 1.16e^{77}$$

mining consists in drawing at random a nonce

$$\text{Nonce} \sim \text{Unif}(\{0, \dots, 2^{32} - 1\})$$

until

$$h(\text{Nonce}|\text{Block info}) < T,$$

where T is referred to as the target.

Difficulty of the cryptopuzzle

$$D = \frac{T_{\max}}{T}$$

Mining a block

If we set the difficulty to $D = 2^4$ then the hexadecimal digest must start with at least 1 leading 0

```
Block Hash: 0869032ad6b3e5b86a53f9dded5f7b09ab93b24cd5a79c1d8c81b0b3e748d226
Block Height: 0
Time:2022-02-25 13:41:48.039980
Nonce:2931734429
Block data: [{'sender': 'Coinbase', 'recipient': 'Satoshi', 'amount': 100, 'fee': 0}, {'sen
der': 'Satoshi', 'recipient': 'Pierre-O', 'amount': 5, 'fee': 2}]
Previous block hash: 0
Mined: True
--------------
```

Figure – A mined block with a hash value having on leading zero.

The number of trial is geometrically distributed

- Exponential inter-block times
- Lenght of the blockchain = Poisson process

Bitcoin protocol

- One block every 10 minutes on average
- Depends on the hashrate of the network
- Difficulty adjustment every 2,016 blocks ($\approx$ two weeks)
- Reward halving every 210,000 blocks

Check out `https://www.bitcoinblockhalf.com/`

Mining equipments

How it started

- CPU, GPU

How it is going

- Application Specific Integrated Chip (ASIC)
  - Increase of the network electricity consumption https://digiconomist.net/bitcoin-energy-consumption
  - E-Waste
  - Centralization issue https://www.bitmain.com/
    - Mining pool ranking at https://btc.com/
    - Mining equipment profitability at https://v3.antpool.com/minerIncomeRank

Proof of Stake

PoW is slow and ressource consuming. Let $\{1,\ldots,N\}$ be a set of miners and $\{\pi_1,\ldots,\pi_N\}$ be their share of cryptocoins.

PoS

1. Node $i \in \{1,\ldots,N\}$ is selected with probability π_i to append the next block

Nodes are chosen according to what they own.

- Nothing at stake problem
- Rich gets richer ?

F. Saleh, "Blockchain without waste : Proof-of-stake," *The Review of Financial Studies*, vol. 34, pp. 1156–1190, jul 2020.