BLOCKASTICS

Stochastic models for blockchain analysis

Pierre-O Goffard

updated on June 4, 2024

Contents

Chapter 1

Introduction

A blockchain is a distributed ledger made of a sequence of blocks maintained by achieving consensus among a number of nodes in a Peer-to-Peer network. The blockchain technology has attracted a lot of interest after the advent of the bitcoin cryptocurrency in 2008, see Nakamoto [2008]. Since then, the blockchain concept has been used to develop decentralized systems to store and maintain the integrity of time-stamped transaction data across peer-to-peer networks. Besides the creation of a digital currency, blockchain applications include the sharing of IT resources, the registration of authentication certificate or the implementation of smart contracts.

A blockchain is

- Decentralized as it is maintained by a network. Nodes can be light or full nodes. Light nodes are blockchain users that broadcast transactions, full nodes are in charge of verifying and recording the transactions, see Figure 1.1.

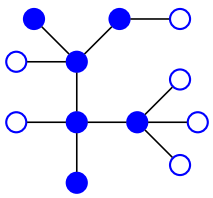


Figure 1.1: A network made of full nodes (blue) and light nodes (white)

  - A local copy is stored by each full node which grants security
  - The governance is not handled by a central authority

- Public or private. In public blockchain anyone can access the data, in private blockchain reading access is restricted.
- permissionned or permissionless. In permissionless blockchain, anyone can join the network as a full node.

- Immutable. Altering the information written in the blockchain is made difficult if not impossible.
- Incentive compatible. The process of reaching consensus is costly to the full nodes who must be compensated for their hard work.

The consensus protocols, at the core of the blockchain technologies, are the focus of these lecture notes. The goal is to evaluate consensus protocol according to three dimensions

1. Efficiency: The amount of data being processed per time unit
2. Decentralization: The fairness of the distribution of the decision power among the nodes
3. Security: The likelihood of a successful attack on the blockchain

Because consensus protocols involve random components, stochastic modelling is required to assess a blockchain system within the Efficiency/Decentralization/Security trilemma in Figure 1.2. As it is hard to improve one dimension without negatively impacting the other two, trade-offs

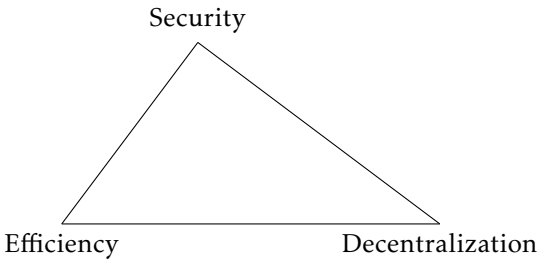


Figure 1.2: The blockchain trilemma

must be made. We will see how to use classical models of applied probability, including urn, epidemic, graph, queue and risk models, to provide numerically tractable indicators to quantify the efficiency, decentralization and security of blockchain systems. These indicators will then allow us to carry out sensitivity analysis with respect to the model parameters to optimize and improve blockchain implementations.

The main application of blockchain systems today is undoubtedly cryptocurrencies, the most well known of which being the bitcoin introduced by Nakamoto [2008]. Public and permissionless blockchain, like the bitcoin one, must be associated to a cryptocurrency. Indeed, to add a block to the bitcoin blockchains the full nodes compete to solve a cryptrographic puzzle using brute force search algorithm. The first node (referred to as a miner) who finds a solution, appends the next block and collects a reward expressed in cryptocurrency. Assuming this reward is worth something, it offsets the operational cost which is essentially the electricity consumed to run the computers 24/7. A cryptocurrency must be equipped with following features

1. No central authority (Decentralized network)

2. Ledger to record all the transactions and coin ownership (the blockchain)
3. A coin generation process (block finding reward)

   ↪ It creates an incentive compatible system to the full nodes
4. Ownership can be proved cryptographically, a wallet is secured with a public/private key system
5. Transactions can be issued by an entity proving ownership of the cryptographic unit through the private key
6. The system cannot process more than one transaction associated to the same cryptographic unit. It must be robust to double spending attack in which a fraudster is issuing two conflicting transactions to recover the funds she already spent

This charcaterization is given by Lansky [2018]. Cryptocurrencies draw their fundamental value from the fact that they

- provide transaction anonymity
- provide a reliable currency in certain regions of the world
- permit money transfer worldwide at low fare
- do not require a thrusted third party

An important implication of this architecture is disintermediation, it creates an environment where multiple parties can interact directly and transparently. Decentralized finance (DeFi) offers a new financial architecture that is non-custodial, permissionless, openly auditable, pseudo-anonymous and with potential new capital efficiencies. It extends the promise of the original bitcoin whitepaper Nakamoto [2008] of non-custodial transaction to more complex financial operations, see the SoK of Werner et al. [2021].

Blockchain is a research topic of interest to many communities. Computing science distributed ledger technologies (synonymous with blockchains) rely on distributed algorithms and enable cooperation within a peer-to-peer network. Linking blocks and checking the authenticity of data uses cryptographic functions which is another field of computer science. The establishment of an incentive system within a network of individuals adopting a strategic behavior naturally leads to problems of game theory similar to those solved by economists. The discussion on the nature of new financial assets such as crypto-currencies, utility tokens and non-fungible tokens, is also at the center of the concerns of researchers in finance and monetary economics.

We focus here on the use of mathematics to optimize blockchain systems which makes our problems very close to those encountered in operations research. These notes are organized as

follows. Chapter 2 presents the various consensus algorithms. Chapter 3 compares traditional and decentralized finance and provide an prominent example of DeFi application with the decentralized Exchange platforms using automated market makers. Chapter 4 focuses on the security aspects. In Chapter 5, we take a look at decentralization. We close on efficiency with Chapter 6.

Chapter 2

Consensus protocol

Transactions flow through the network of full nodes. After reviewing them, the full nodes must agree on the transaction that will be recorded in the next block. To do so, an algorithm must be designed so that consensus is reached. A consensus protocol must be based on one of the scarce resources available to the network peers which include

- bandwidth
- computational power
- storage

The first solution that comes to mind for reaching consensus is a majority vote based on a message exchange system. This solution has been proposed by Lamport et al. [1982] within the famous "Byzantine general problem". A voting system inside a large network involves a colossal number of messages exchanged leading to the consumption of all the bandwidth, the failure of some nodes by denial of service and delays in the synchronization of the network. Practical solution like the celebrated Practical Byzantine Fault Tolerance (PBFT) presented in Castro and Liskov [1999] have been implemented in some blockchain systems. Despite these advances, a change in methods was needed to accommodate a network that could grow indefinitely.

Nakamoto [2008] solved this scaling problem by proposing a system based on the election of a leader. The Proof-of-Work (PoW) protocol appoints a leader based on its computing resources. Each node competes to solve a puzzle with a brute force search algorithm. The first node who is able to propose a solution append the next block. The search for a solution, referred to as mining, is associated with an operational cost borne by the nodes which is compensated by a reward expressed in the native blockchain cryptocurrency. The surge in cryptocurrency prices has led to a rush in block mining, leading to a major spike in the electricity consumption and electronic waste generation of blockchain networks. The blockchain network consumes as much electricity as countries the size of Thailand at the time of the writing. The need for a more environmentally friendly consensus protocol therefore becomes pivotal. Protocol such

as *Proof-of-Capacity* and *Proof-of-Spacetime* use storage. Using storage is seen as a fairer and greener alternative by blockchain enthusiasts due to the general purpose nature of storage and the lower energy cost required by storage. The fact that most storage resources are owned by companies offering cloud storage solution poses a threat to the decentralized nature of the distributed ledger. The Proof-of-Interaction (PoI) protocol, proposed by Abegg et al. [2021], takes as leader the first node that is able to contact and obtain a response from a random sequence of nodes. This is a bandwidth-based alternative that is more scalable than majority voting. Along with bandwidth, computing power, and storage, a new resource has emerged with the advent of cryptocurrencies as a medium of exchange. The Proof-of-Stake protocol, described by Saleh [2020], selects a node with a probability proportional to the number of cryptocoins it holds.

Consensus protocols are applied so that a blocks are appended sequentially and not at the same time. Usually the consensus process is divided into time slots, also called rounds. The block generation time must be higher than the propagation delay in the network. If two blocks are created at the same time then a fork will occur. Two branches of the blockchain co-exists. A fork situation then resolves by applying the *Longest Chain Rule* (LCR).

Definition 1. *The Longest Chain Rule states that if there exist several branches of the blockchain then the longest should be trusted.*

This definition implies that a threshold must be chosen in order to decide when shorter branches of the blockchain should be discarded. For instance, a branch can be considered legitimate if it is $k \in \mathbb{N}$ blocks ahead of its pursuers. For the consensus protocol to be viable, nodes must be incentivized to follow the LCR.

This chapter is organized as follows. Section 2.1 gives a brief description of the voting based ways to get consensus by reviewing the "generals" problem. Section 2.2 goes through the leader based consensus protocols, including PoW in Section 2.2.1, PoSp in Section 2.2.2, PoI in Section 2.2.3, and PoS in Section 2.2.4. For an exhaustive list of the existing protocols the reader is referred to https://tokens-economy.gitbook.io/consensus/.

2.1 Voting system

The problem of reaching consensus in a peer-to-peer network via a majority vote has been abstractedly compared to generals who must agree on a common battle plan. We start from the simple two general case before moving on the the situation of interest with several ones.

2.1.1 Two generals problem

Two generals wish to attack a city but they must agree on timing the attack. If they do not attack at the exact same time then they will fail. They communicate via a messenger who must cross

enemy territory at the risk of being intercepted. The first general G_1 sends a message to the second one G_2 saying

```
"I will attack tomorrow at dawn"
```

For the attack to succeed, both generals must attack at the same time. Because their communication medium is unreliable, then G_1 must await confirmation from G_2 in order to attack. If G_1 does not receive confirmation then she will not attack. G_2 is aware of that and respond

```
"I will follow your lead"
```

G_2 does not know whether the message went through and must wait for confirmation. This creates an infinite loop of messages and response, as on Figure 2.1. The two general problem is

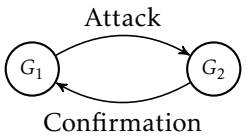


Figure 2.1: Message and confirmation loop

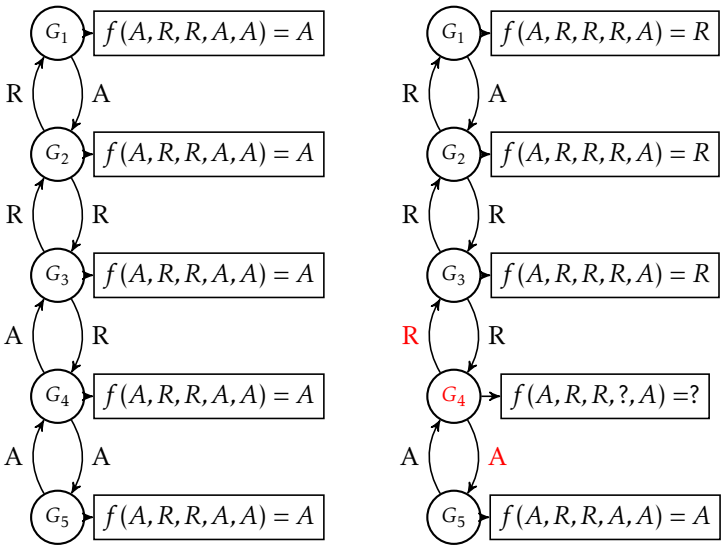
deemed unsolvable from a theoretical point of view and corresponds to a situation where two nodes communicate through an unreliable link. A practical solution for generals is to send many messengers hoping that at least one of them will succeed. This is only a thought experiment leading to the several general problem.

2.1.2 Byzantine General problem

The blockchain network contains more than two nodes, these nodes must agree on the transactions to confirm. In a permissionless blockchain the nodes do not trust each other. The problem of the previous section generalizes to more than two generals, assuming that some generals are traitors which corresponds to faulty nodes in the network. This problem is referred to as The "Byzantine general problem" and was coined by Lamport et al. [1982]. Assume that $n > 2$ generals must agree on a common battle plan for instance "Attack" (A) or "Retreat" (R) and that they can only communicate by two party messages. Denote by $m(i,j)$ the message sent by general i to general j. Each general j receives $n-1$ messages and applies a function f to determine the course of action, for instance

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

If there are no traitors, each general is communicating the same value to all the peers and consensus is reached as in Figure 2.2a. If one general is traitor, then he might not communicate the same value to all the generals and no consensus can be reached. It is the case for G_4 in Figure 2.2b. To handle such a situation, specific roles must be assigned to the generals. One of



(a) No traitor (b) One traitor

Figure 2.2: Majority vote with or without a traitor

them become the leader and the other are the lieutenants. We aim at finding an algorithm such that

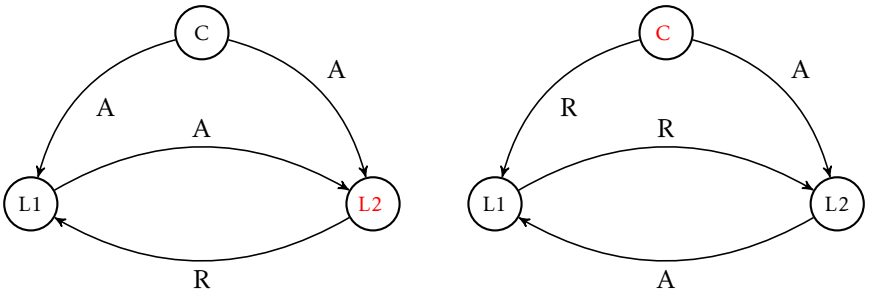
C1 All the loyal lieutenants obey the same order

C2 If the commanding general is loyal, then every loyal lieutenants obey the order he sends

A first result from Lamport et al. [1982] is the following

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

Proof. Consider the situation where $n = 3$ and $m = 1$. The traitor is either the commander or one of the lieutenants as shown in Figure 2.3.



(a) Commander is loyal (b) Commander is a traitor

Figure 2.3: Majority vote with or without a traitor

Unfortunately for Lieutenant 2, there is no way for her to tell apart the situation pictured in Figure 2.3a and Figure 2.3b and therefore no way to ensure both C1 and C2. We prove the

result for $n > 3$ by contradiction. Assume that there is a way to verify both C1 and C2 with $3 < n < 3m + 1$. We then construct a solution with generals by having one general simulate the commander plus at most $m - 1$ generals, and the other two simulating at most m generals. One of the generals gather all the traitors and is therefore a traitor. The other two are loyal generals as they only simulate loyals general. We have built a solution with three generals that we know is impossible. □

Now we need an algorithm that allows $n > 3m + 1$ generals to deal with m traitors. The 'Oral Message' algorithm denoted by OM(m) and summarized in Algorithm 1 can handle m traitors if the number of generals verifies $n > 3m + 1$. Before looking into the theoretical justification of

Algorithm 1 The Oral message algorithm OM(m)

if $m = 0$ **then**;
    for $i = 1 \to n - 1$ **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 $i = 1 \to n - 1$ **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 $i = 1 \to n - 1$ **do**
        Lieutenant i set their value to $f(v_1, \ldots, v_{n-1})$
    end for
end if

OM(m), let us illustrate the algorithm with an example.

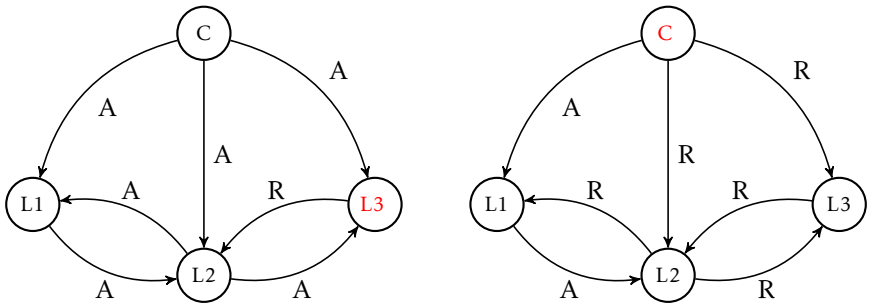
Example 1. *Consider the situation where $n = 4$ and $m = 1$ shown in Figure 2.4. If the commander is loyal then one of the lieutenant is a traitor, see Figure 2.4a. The commander gives the order to attack to all 3 lieutenants. Lieutenant 3 tells the other that she heard retreat from the commander. The loyal lieutenants then apply the map f to agree on their value*

$$f(A, A, R) = A,$$

which coresponds to the order the commander sent, hence IC1 and IC2 are satisfied. If the commander is a traitor as in Figure 2.4b, then he sends conflicting order to the lieutenant but after communicating the value they received to each other finally agree on the following value

$$f(A, R, R) = R,$$

hence IC1 is satisfied and IC2 can be ignored since the commander is a traitor.



(a) Commander is loyal

(b) Commander is a traitor

Figure 2.4: Illustration of the OM(m) algorithm in the case where $n = 4$ and $m = 1$.

Theorem 2. *Algorithm OM(m) satisfies conditions IC1 and IC2 if $n > 3m + 1$.*

Proof. The proof follows from induction.

First assume that the commander is loyal. For $m = 0$, the commanders simply sends the value v to all the lieutenants and IC2 holds. Assume that OM($m-1$) works when the commader is loyal. The commander sends v to all the lieutenants. The lieutenants then applies $OM(m-1)$. Because $n-1 > 2k+m-1$, then it follows from the induction hypothesis that each loyal lieutenants get the value v for each of the loyal lieutenants j. The loyal lieutenants $n-1-m > 2k-1 > m$ outnumber the traitorous lieutenants and therefore set their value to

$$f(v_1,\ldots,v_{n-1}) = v,$$

and both IC1 and IC2 are satisfied.

Let us assume that the commander is a traitor, we only have to worry about IC1 in that case. There are at most m traitors and the commander is one of them. We therefore have $m-1$ traitors among the lieutenants. Since the total number of lieutenants exceeds three times the number of traitors $n-1 > 3m > 3(m-1)$ then by applying OM($m-1$) all the loyal lieutenants receive the same vector of values $v_1,\ldots,v_{n-1}$, agree on the same value

$$f(v_1,\ldots,v_{n-1}) = v,$$

which leads to the verification of IC1. □

The main problem associated to this Oral message algorithm is the number of messages is n^{m+1} which is prohibitive for large values of n and m. A celebrated algorithm, called Practical Byzantine Fault Tolerance (PBFT) has been developped later on by Castro and Liskov [1999] but still not fast enough to enable the infinite growth of the network associated to public and permissionless blockchains.

2.2 Leader system

The scalability issue can be solved by opting for a leader based mechanism instead of a majority vote mechanism. The protocols presented in this section use computational power, storage and bandwidth to elect a leader each time a new block must be appended to the blockchain.

2.2.1 Proof-of-Work

The bitcoin blockchain relies on a consensus protocol based on computational power called Proof-of-Work (PoW), presented in Nakamoto [2008]. 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.

The hash of a block is obtained by concatenating the header and the transactions in a large character string thus forming a "message" denoted by m, to which a hash function h is applied.

Definition 2. *A hash function is a function that can map data of arbitrary size to fixed-sized values,*

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

The hash functions used in blockchain applications must be cryptographic, i.e.

- quick to compute
- one way
- deterministic

Remark 1. *It must be nearly infeasible to generate a message with a given hash value or to find two messages with the same hash value. A small change in the message should change dramatically the hash value so that the new hash value appears to be uncorrelated to the previous hash,*

$$\text{if } m_1 \approx m_2 \text{ then } h(m_1) \neq h(m_2).$$

We will not expand on how to build such a cryptographic hash function, we refer the interested reader to the work of Al-Kuwari et al. [2011].

In the bitcoin blockchain as well as in many other applications, the standard is the SHA-256 function which converts any message into a hash value of 256 bits. The latter is usually translated into a hexadecimal digest, for instance the hash value of the title of the present manuscript reads as

98b1146926548f6b57c4347457713ff2f035beda9c93f12fbc9b202e9c512e80.

Mining a block means finding a block hash value lower than some target which can only be achieved by brute force search thanks to the properties of cryptographic hash functions. In practice, the search for an appropriate hash value, referred to as a solution, is done by appending a nonce to the block message before applying the hash function. A nonce is a 32 bits number, drawn at random by miners until a nonce resulting in a proper block hash value is found. For illustration, consider the block in Figure 2.5.

```
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 2.5: A block that has not been mined yet.

The hash value in decimal notation is $1.43e^{76}$ while the maximum value for a 256 bits number is $2^{256} - 1 \approx 1.16e^{77}$. We refer to the latter as the maximal target and denote it by $T_{\max}$. The Proof-of-Work protocol sets a target $T < T_{\max}$ and ask miners to find a nonce such that the hash value of the block is smaller than T. Practitioners would rather talk about the *difficulty* which is defined as $D = T_{\max}/T$. If the difficulty is one, any hash value is acceptable. Increasing the difficulty reduces the set of allowable hash values, making the problem harder to solve. A hash value is then called *acceptable* if its hexadecimal digest starts with a given number of zeros. If we set the difficulty to 2^4, then the hexadecimal digest of the hash of the block must start with at least 1 leading zero, making the hash value of the block in Figure 2.5 not acceptable. After completing the nonce search we get the block in Figure 2.6. Note that it took 5 attempts to

```
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 2.6: A mined block with a hash value having on leading zero.

find this nonce. The number of needed trials is geometrically distributed with parameter $1/D$,

which means that with a difficulty of $D = 2^4$ it takes on average 16 trials. The protocol adjusts the difficulty automatically every 2,016 block discoveries so as to (globally) maintain one block discovery every 10 minutes on average. The time between two block discoveries depends on the number of hash values computed by the network at a given instant. At the time of writing, the network computes 182.58 Exahashes per second and the difficulty is 27,967,152,532,434.[1] For an exhaustive overview of the mining process in the bitcoin blockchain, we refer the reader to the book of **?**, Chapter 10. As each trial (of the system) for mining a block is independent of the others and leads to a success with very small probability, the overall number of successes is binomially distributed and will be very well approximated by a Poisson random variable. This justifies the Poisson process assumption made in the sequel to model the block arrival and the reward collecting processes. Empirical studies of the block inter-arrival times data tend to confirm this hypothesis, see the work of Bowden et al. Bowden et al. [2020]. The information recorded in a public blockchain may be retrieved by anyone and can be accessed through a blockchain explorer such as blockchain.com, the content of the block of height #724724 may be viewed through the following link block content.

The PoW protocol implies that the nodes are running computations 24/7 therefore consuming humungous quantity of electricity. Bitcoin mining originally started by running computations using the Central Processing Unit (CPU). It turns out that certain kinds of computation are more efficient on Graphics Processing Unit (GPU) than on CPUs. CPU is designed to complete a wide variety of task while computing hashes is very specific. GPU are tailored to run thousands of computation of the same type. Miners then turned to GPUs leading to a shortage of graphics card at the expense of PC gamers around the world! Eventually GPU got replaced by Application Specific Integrated Circuits (ASICs) that are designed to complete very specific task compared to graphic cards. ASICs consumes 10 times more power than graphic cards but compute 10,000 more hashes than a graphic card per time unit. Miners then decided to equip themselves with ASIC chips leading to harmful consequences

- Increase of the network electricity consumption
- Increase in the e-waste generation. ASICs are single purpose and it cannot be repurpose for any other task. When ASICs become obsolete with the arrival of a new generation of chips, they are thrown in the trash.
- The main manufacturer of ASICs is a company called BITMAIN which equips major mining pool such as Antpool and BTC.com. A threat on centralization exists since a company like BITMAIN could take control of the network by owning more than half of the overall hashpower.

A pro-ASIC argument is that it would be impossible for anyone (apart from BITMAIN) to suddenly acquire enough of these chips to have more than half of the world's hash power.

[1] Source: bitcoinblockhalf.com

2.2.2 Proof-of-SpaceTime and Proof-of-Capacity

Consensus protocol based on storage capacities are seen by many as a fairer and greener alternative to PoW. We describe below two such protocols *Proof-of-Capacity* and *Proof-of-Spacetime*.

Proof-of-Capacity

In the *Proof-of-Capacity*, miners compute hashes and cache the result on their hard disk space. Mining then only requires to search through the cache for an admissible solution.

Proof-of-Spacetime

In the *Proof-of-Spacetime*, the nodes store data and produce proofs to show that the data has been stored for a given time period. The probability of a node being chosen is proportional to the amount of data stored. This protocol has been designed for a specific application allowing nodes to provide storage to clients through the Filecoin project.

To some extent the *Proof-of-Capacity* protocol is similar to PoW while the *Proof-of-Spacetime* shares similarities with the *Proof-of-Stake* protocol which is discussed below.

Such protocols do not generate ewaste because disk space can always be used for some other purpose. Storing data is less energy consuming than computing hashes. The problem of hiring external storage capacities from provider remains.

2.2.3 Proof-of-Interaction

The *Proof-of-Interaction* protocol, introduced by Abegg et al. [2021], asks each validating node to get in touch with a sequence of nodes. The number of nodes and the nodes to be contacted are drawn randomly so that the time to complete the task is also varying from one node to another. The block reward is shared by the contacting and responding nodes to create an incentive compatible environment. If we assume that the time required to complete the task is exponentially distributed then the time to generate a new block is the minimum of exponential random variables which is again exponentially distributed. PoI is still in the developping phase and many interesting work must be done to assess the security and viability of such protocol. Some nodes may indeed collude to send replies faster or not to send replies to some node. It is necessary to evaluate the probability and the opportunity for the nodes to collude.

2.2.4 Proof-of-Stake

Besides bandwidth, computing power and storage, one ressource that appears with the advent of cryptocurrencies as medium-of-exchange and store-of-value asset are the cryptocoins. Each time a block must be appended to the blockchain, a coin is drawn at random. The owner of that

coin appends a new block and collect the reward.

Let the network be of size N. We denote by π_i^t the proportion of coins owned by node $i \in \{1,\dots,n\}$ at time $t \in \mathbb{N}$. Note that π_i^t is exactly the probability of node i being elected as leader at time t, we have

$$\begin{cases} \pi_i^t > 0, \\ \sum_{i=1}^N \pi_i^t = 1, \end{cases} \quad \text{for } t > 0.$$

Denote by S_t the total number of coins in circulation at time t and by R_t the size of the reward for appending a new block at time t. Let A_i^t be the event of node i appending a block at time t, the share of coins then evolves as

$$\pi_i^t = \frac{S \cdot \pi_i^0 + \sum_{s=1}^{t-1} \mathbb{I}_{A_i^t} R_s}{S + \sum_{s=1}^{t-1} R_s},$$

where

$$\mathbb{I}_A = \begin{cases} 1 & \text{if A occurs,} \\ 0 & \text{otherwise.} \end{cases}$$

Two potential issues needs to be studied

- The Nothing-at-Stake (NaS) problem: If a a fork is ongoing then each branch will elect a leader who will append a block, collect the reward and perpetuate the disabreement.
- The rich get richer problem: When a node is chosen, it becomes richer which increase its likelihood to be chosen in future rounds.

The "rich get richer" problem will be extensively studied in Chapter 5. Regarding the NaS problem, the nodes when chosen by a branch decide whether they want to add a block, it is an option. The cryptocoin value comes from its use as a medium of exchange. A long lasting disagreement results in a useless cryptocoin with no value.

Let τ be the duration of the fork and let $\delta \in (0,1)$ be a discount rate, then the present value of a coin at τ is $1/(1+\delta)^\tau$. If $\mathbb{P}(\tau = \infty) > 0$ then the coin value is zero when taking the expectation.

The nodes are therefore incentivized to follow the Longest chain rule in order to resolve the fork situation as soon as possible. This is essentially the rationale in Saleh [2020] to show that

- The coin value reaches a maximum if all the nodes follow the longest chain rule
- There exists an equilibrium in which the nodes follow the LCR if

$$\min \pi_i^0 \cdot S \geq \frac{R_0}{\delta(1-\delta)^2},$$

  which corresponds to a minimum stake condition.

- If $\sum_t R_t < \infty$ then there exist no equilibrium for which

$$\mathbb{P}(\tau = \infty) > 0.$$

A modest reward schedule precludes the posssibility of an ever lasting fork.

A practical implementation of the PoS protocol to create a cryptocurrency is PeerCoin, see the white paper by King and Nadal [2012]. The notion of coin age is introduced, the stake is actually defined by the number of coins times the number of time period during which the coins was hold. When a peer finds a block, a *coinstake* transaction is made that transfers the node its own coin to reset the coin age to zero.

Chapter 3

Decentralized Finance and Data Analysis

3.1 Decentralized Finance

Let us first list the distinctions between traditional finance (TradFi) and decentralized finance (DeFi). TradFi is often characterized by high access barriers, requiring specific criteria such as bank accounts, whereas DeFi eliminates these barriers, allowing universal participation. TradFi operates on centralized systems where banks serve as the primary record keepers, exposing them to cyber risks, whereas DeFi utilizes a decentralized ledger system, enhancing security and reducing such vulnerabilities. Moreover, TradFi is plagued by high transaction costs, including fees for account maintenance and wire transfers, and relies on intermediaries for transactions; in contrast, DeFi minimizes these costs by using smart contracts, although users must still handle gas fees. Transactions in TradFi, especially cross-border ones, can be slow, taking days to settle, while DeFi offers near-instant settlement. Furthermore, TradFi lacks transparency and can be difficult to audit, whereas DeFi provides a publicly available ledger and open-source code, ensuring greater transparency and auditability. Additionally, TradFi operations are restricted by geographical and regulatory constraints, and are often limited to business hours, while DeFi offers 24/7 global accessibility without censorship or restrictions by central authorities. TradFi's challenges in providing fractional ownership for assets like real estate and art are addressed in DeFi through the tokenization of real-world assets. Lastly, TradFi often relies on outdated IT solutions, which stifles innovation and interoperability, whereas DeFi fosters rapid innovation and enables seamless interoperability across platforms and projects. These discussion can be found for instance in the bok of Lipton and Treccani [2021].

A fundamental application of DeFi are exchange platforms that allow users to trade the different kind of cryptoassets. We are going to focus on decentralized exchange in the next section.

3.2 Decentralized Exchanges (DEXs) and Automated Market Makers (AMM)

If you were to buy your first units of cryptoasset, you might turn to a centralized exchange platforms for simplicity. Centralized exchange platforms, often referred to as traditional exchanges, are operated by a central authority or company that facilitates trading between users. These exchanges offer convenience and liquidity, as well as features like advanced trading options and customer support. However, they also introduce a single point of failure and require users to trust the exchange operator with their funds and personal information, which can be vulnerable to hacks or regulatory actions. Trading on centralized exchange further also include non transparent and fluctuating transaction cost. On the other hand, decentralized exchange platforms operate on blockchain technology, allowing users to trade directly with each other without the need for an intermediary. This peer-to-peer model eliminates the need to trust a central authority and offers increased security and privacy since users retain control of their funds throughout the trading process. However, decentralized exchanges may suffer from lower liquidity and slower transaction speeds compared to centralized exchanges, and they often lack features like fiat currency support and customer service. Despite these trade-offs, decentralized exchanges are gaining popularity due to their commitment to decentralization and censorship resistance. Censorship resistance means that transactions and trading activities cannot be easily censored, blocked, or controlled by any single party, including governments or regulatory agencies.

Decentralized exchanges (DEXs) Exchange of one cryptocurrency token for another, with one token typically being more volatile or risky while the other aims to maintain stability. This pairing frequently involves a stablecoin, a type of cryptocurrency designed to minimize price volatility by pegging its value to an external reference asset, such as a fiat currency like the US dollar. Stablecoins serve as a reliable medium of exchange and store of value within the cryptocurrency ecosystem, offering traders a way to mitigate the risks associated with price fluctuations while transacting on DEXs.

Stablecoins: Two main types: fiat collateralized and crypto collateralized.

- Fiat collateralized stablecoins are backed by reserves of fiat currency, held in bank accounts or other custodial arrangements, with each stablecoin token representing a claim to a specified amount of the underlying fiat currency. Examples of fiat collateralized stablecoins include USDT (Tether), USDC (USD Coin), and BUSD (Binance USD).
- Crypto collateralized stablecoins are backed by reserves of other cryptocurrencies, typically held in smart contracts or decentralized protocols, with the value of the stablecoin maintained through overcollateralization and algorithmic mechanisms, see Moin et al.

[2020]. An example is provided in Example 2.

Example 2. *One prominent example of a crypto collateralized stablecoin is DAI, which operates on the Ethereum blockchain. DAI is created and managed by the MakerDAO protocol, a decentralized autonomous organization governed by MKR token holders. DAI is collateralized by a pool of other cryptocurrencies, primarily Ethereum (ETH), which users lock into smart contracts known as Collateralized Debt Positions (CDPs) to generate DAI tokens. The value of DAI is maintained close to one US dollar through a combination of market forces and algorithmic mechanisms, including the use of oracles to provide price feeds and the automatic adjustment of interest rates to incentivize or discourage borrowing and lending activities. By leveraging Ethereum's smart contract capabilities and a decentralized governance model, DAI offers users a transparent, trust-minimized stablecoin solution within the decentralized finance (DeFi) ecosystem.*

Centralized exchange use a classical order book mechanism to set the prices while DEXs use Automated Market Makers

Order book-based system[1]**:** buyers and sellers place orders to buy or sell assets at specific prices, creating a dynamic order book that matches buy and sell orders. This system relies on the interaction of buyers and sellers to determine the price and liquidity of assets, with prices fluctuating based on market demand and supply. This require numerous interactions with the protocol and therefore huge amounts of gas fee. The company that manage the centralized platform plays the role of market maker in a non transparent ways. It could probably be automatised within a DeFi application at the cost of writing a complex algorithm in the blockchain programming language (e.g. solidity for the Ethereum blockchain).

Automated Market Makers (AMMs) Algorithms that use liquidity pools to facilitate trading without relying on order books. AMMs determine asset prices algorithmically based on the ratio of assets in liquidity pools, providing liquidity for trades through predefined smart contracts. While order book-based systems offer more flexibility in price setting and execution, they can suffer from issues like low liquidity and price slippage, especially in volatile markets. In contrast, AMMs provide continuous liquidity and can offer lower barriers to entry for traders, but they may suffer from impermanent loss and may not always provide the best prices for large trades. Both mechanisms have their strengths and weaknesses, and their suitability depends on factors such as trading volume, market conditions, and user preferences. To learn more, the reader is referred to the following survey Xu et al. [2023].

In Dexs we have two characters: The Liquidity Providers (LPs) and the exchange users.

Liquidity Providers (LPs) A liquidity provider is an individual or entity that deposits funds into liquidity pools on the DEX, allowing users to trade assets without needing a traditional

[1] https://www.youtube.com/watch?v=K14-VJ2K8Ik

order book or centralized intermediary.

In a decentralized exchange (DEX), liquidity providers play a crucial role in facilitating trading activities and maintaining liquidity within the exchange. A liquidity provider is an individual or entity that deposits funds into liquidity pools on the DEX, allowing users to trade assets without needing a traditional order book or centralized intermediary.

Here's how liquidity providers contribute to a DEX:

1. Providing Liquidity: Liquidity providers deposit pairs of assets into liquidity pools, which are smart contracts that hold reserves of each asset. These assets are used to facilitate trades on the DEX. By contributing liquidity to the pools, providers ensure that there are sufficient funds available for users to trade with.

2. Earning Fees: In return for providing liquidity, liquidity providers earn fees from trading activities that occur within the liquidity pools. When users execute trades on the DEX, they pay a small fee, a portion of which is distributed to liquidity providers as a reward for their contribution. This incentivizes providers to deposit funds into the liquidity pools and helps maintain liquidity on the exchange.

3. Maintaining Price Stability: Liquidity providers help maintain price stability for assets traded on the DEX by ensuring that there are ample funds available for buying and selling. This reduces slippage—the difference between the expected price of a trade and the actual price at which the trade is executed—and improves the overall trading experience for users.

4. Adjusting Positions: Liquidity providers may adjust their positions in the liquidity pools in response to changes in market conditions, asset prices, or trading volume. By rebalancing their holdings or adding/removing liquidity as needed, providers help optimize liquidity provision and maximize their returns.

Exchange User: An exchange user refers to an individual or entity that interacts with an exchange platform to buy, sell, or trade assets. In the context of cryptocurrency exchanges, an exchange user is someone who utilizes the exchange's services to engage in trading activities involving cryptocurrencies or other digital assets.

The impact of trades, and liquidity provision/withdrawal on prices is governed by constant fuction market makers.

Constant Function Market Makers (CFMMS: Mathematical formula or algorithm that determines the pricing and liquidity provision mechanism within the liquidity pools. It is called "constant function" because it maintains a constant product of the quantities of two assets in the liquidity pool.

We focus on a prominent example of such function known as the Constant Product Market Makers Function. Consider an exchange where you can trade token X for token Y and conversely. The exchange starts operating as a first liquidity provider comes in and deposit x of X and y of Y, therefore setting k as

$$x \cdot y = k.$$

The liquidity provided by this LP is measured by $L = \sqrt{k} = \sqrt{x \cdot y}$ (geometric mean).

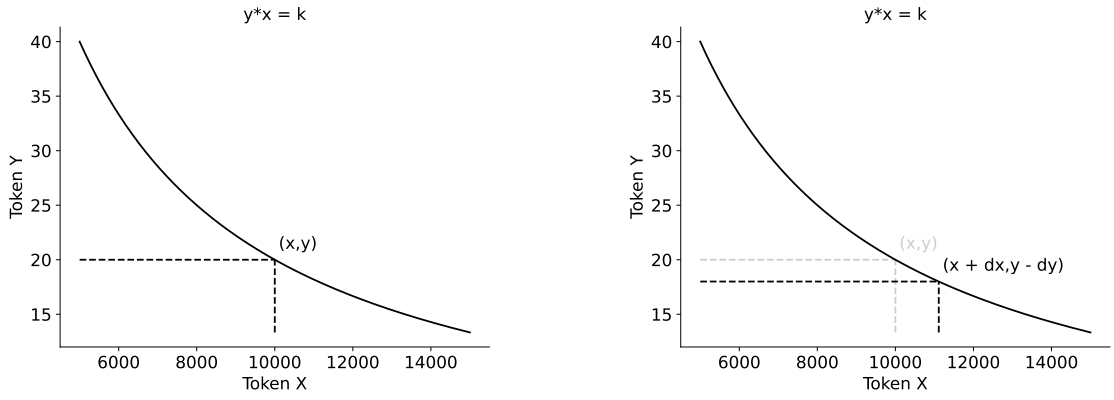
Example 3. *Consider a ETH/DAI Constant Product Market Maker. Assume that the current price for one ETH is* $P = \$500$. *A LP shows up and deposits* 20 *ETH plus* 10,000 *DAI which sets*

$$k = 20,000 \text{ and } L = \sqrt{x \cdot y} \approx 447.$$

Assume that a trader wish to swap X for Y by acquiring $\mathrm{d}y$. He then must deposit in the pool $\mathrm{d}x$ of X that solves

$$(x + dx)(y - dy) = k \Leftrightarrow dx = \frac{x \cdot dy}{y - dy}.$$

Trades (or swap) occur on a curve as pictured on Figure 3.1. Usually, the protocol reward LPs by



(a) Initial state

(b) State after the trade/swap

Figure 3.1: Trade on the $x \cdot y = k$ curve.

including a fee for each trade. The trader then deposits an additional $\alpha \cdot \mathrm{d}x$ worth of fee that will be distributed to the LPs proportionnally to their Liquidity provision to the pool. Let us continue Example 3 to illustrate the swap of an arbitrageur.

Example 4. *Say, an arbitrageur takes* $dy = 2$ *ETH. She then deposits* $dx = 1,111$, *and give out* $0.3 \cdot 1,111 = 333$ *worth of fees, assuming that* $\alpha = 0.3$. *As a consequence, the price of ETH in the pool rises to*

$$\frac{x + dx}{y - dy} = \$617.$$

Another LP can provide dx to the pool and therefore deposits also $dy = \frac{y}{x}dx$ so that the price does not change. The curve is bumped to a new level

$$k' = (x + dx)(y + dy),$$

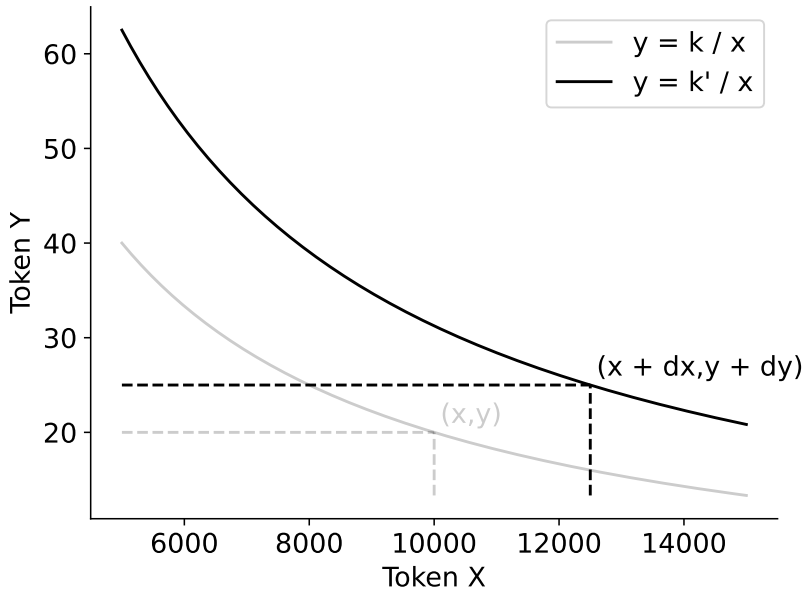


Figure 3.2: Change in the curve after adding liquidity.

as shown on Figure 3.2. The overall liquidity level also rises as

$$L' = \sqrt{x \cdot y} + \sqrt{dx \cdot dy}.$$

The fees are collected by both LPs accodring to the following weights

$$w_1 = \frac{L}{L'}, \text{ and } w_1 = \frac{L' - L}{L'}.$$

We resume Example 4 with the following example

Example 5. *A new LP deposits* $\$5,000$ *worth of tokens to the pool which splits into*

$$dx = 2,500, \textit{ and } dy = 5$$

The new levels of liquidity become

$$k' = 312,500, \textit{ and } L' \approx 559.$$

The weights of both LPs are given by

$$\frac{L}{L'} = 0.8 \textit{ and } \frac{L' - L}{L'} = 0.2.$$

AMM based DEXs commonly suffer from two things: Slippage and divergence (aka impermanent) loss.

Slippage: Slippage refers to the difference between the expected price of a trade and the actual price at which the trade is executed. When a user places a trade on an AMM-based DEX,

the price of the asset may change as a result of the trade itself. This is because the constant product formula used in AMMs adjusts the price of assets based on changes in supply and demand within the liquidity pool. As a result, larger trades or trades in illiquid pools may cause significant price changes, leading to slippage. To mitigate slippage, traders can utilize strategies such as breaking up large orders into smaller ones, selecting pools with higher liquidity, or using advanced trading techniques. Additionally, ongoing improvements in AMM designs and protocols aim to minimize slippage and enhance the trading experience on decentralized exchanges. This behavior may be exploited by validating nodes of the blockachin that are aware of all the pending transaction and may front run well chosen transaction. Such an attack is described in Example 6 and discussed in Park [2023].

Example 6 (Sandwich attack)**.** *The sandwich attack is a form of front-running manipulation that can occur on decentralized exchanges (DEXs) utilizing automated market makers (AMMs). In a sandwich attack, an attacker exploits the predictable price movement resulting from a large trade to profit at the expense of other traders.*

Here's how the sandwich attack typically works:

1. *Identifying the Target Transaction: The attacker monitors the blockchain for pending or recently submitted transactions that involve a large trade on a particular AMM-based DEX. Large trades can create significant price movements due to slippage, as discussed earlier.*

2. *Front-Running the Target Transaction: Before the target transaction is executed, the attacker quickly submits their own transactions to the DEX with the intention of exploiting the anticipated price movement caused by the large trade. The attacker strategically places buy or sell orders in such a way that they benefit from the price movement.*

3. *Executing Trades at Favorable Prices: As the target transaction is processed and causes the price of the asset to move, the attacker's own transactions are executed at more favorable prices due to the price impact of the large trade. This allows the attacker to buy low or sell high, effectively profiting from the price movement caused by the target transaction.*

4. *Exiting the Position: Once the price movement subsides, the attacker may quickly exit their position by selling or buying back the asset at a later time, potentially realizing a profit from the price difference.*

The term "sandwich attack" comes from the idea that the attacker places their own transactions on both sides of the target transaction, effectively sandwiching it between their own trades to exploit the price movement.

The sandwich attack exploits the inherent transparency and predictability of blockchain transactions, as well as the mechanics of AMMs, to profit at the expense of other traders. To mitigate the risk of sandwich attacks, traders should exercise caution when executing large trades on DEXs and consider employing strategies to minimize slippage and prevent front-running. Additionally, ongoing

improvements in blockchain technology and DEX protocols aim to enhance security and protect against such attacks.

Divergence Loss: Also known as impermanent loss, is a phenomenon that occurs when providing liquidity to an automated market maker (AMM) liquidity pool, such as those found in decentralized exchanges (DEXs). It refers to the difference in value between holding assets in the liquidity pool versus simply holding them in one's wallet.

1. Liquidity providers deposit pairs of assets into a liquidity pool on a DEX. For example, they may deposit equal amounts of ETH and DAI into a liquidity pool for the ETH-DAI trading pair.

2. As traders buy and sell assets on the DEX, the prices of the assets within the liquidity pool can change. These price changes can cause the ratio of the deposited assets to deviate from the initial ratio.

3. If the price of one asset in the pool increases relative to the other, liquidity providers will have more of the asset that increased in value and less of the other asset. As a result, when they withdraw their liquidity from the pool, they will receive fewer of the appreciating asset and more of the depreciating asset compared to their initial deposit.

4. The difference in the value of the assets held in the liquidity pool compared to if they were simply held in the wallet is referred to as divergence loss or impermanent loss.

This loss is considered "impermanent" because it may decrease or disappear entirely if the prices of the assets revert to their initial ratio (stong assumption!!!) by the time the liquidity is withdrawn.

Let us consider a pool that contains token X and Y. The price of Y in the pool is given by

$$P = \frac{x}{y},$$

in terms of token X. If the Price of Y becomes $P' > P$ on another trading venue then arbitrageurs wish to find

$$\underset{dy>0}{\operatorname{argmax}}\ dy \cdot P' - dx \cdot (1+\alpha) = \underset{dy}{\operatorname{argmax}}\ dy \cdot P' - \frac{x \cdot dy}{y - dy}(1+\alpha).$$

Solving this optimization problem, we have

$$dy^* = y - \sqrt{\frac{k(1+\alpha)}{P'}}.$$

The arbitrageurs profit is

$$dy^* \cdot P' - dx^*(1+\alpha)$$

which means that the LPs incur a loss of

$$y \cdot P' + x - [(y - dy^*) \cdot P' + x + dx^*(1+\alpha)],$$

if they withdraw. The loss is said impermanent because if the LPs do not withdraw and the price revert to the initial ratio of token in the pool then no incurred loss. Let us build on Example 5.

Example 7. *Recall that the initial price of ETH in the pool was* $P = \$500$. *Assume that the price in another trading venue is* $P' = \$550$ *and* $\alpha = 0$ *then*

- $dy^* = 0.93$
- *The impermanent loss is then* $dy^* \cdot P' - \frac{xdy^*}{y - dy^*} = \$23$

Increasing α *then mitigate the magnitude of impermanent loss. Take* $\alpha = 0.05$ *tehn we have*

- $dy^* = 0.46$
- *The impermanent loss is then* $dy^* \cdot P' - \frac{xdy^*}{y - dy^*} = \$5.81$.

DEXs form a pillar of the DeFi environment but as we have seen there is room for improvement which means many avenues for future research work!

Chapter 4

Security of blockchain systems

The security evaluation of blockchain systems consists of calculating the probability of a successful attack on the blockchain. We will focus on the double spending attack. Section 4.1 provides a brief overview through a simple example. The probability of a successful double spending attack is computed within a random walk model in Section 4.2 and counting processes in Section 4.3.

4.1 Double-spending in PoW

A double spending attack aims to generate a concurrent blockchain to replace the main one. Consider the following scenario:

1. Marie sends BTC10 to John.
2. The transaction from Marie to John is recorded in the blockchain.
3. John is advised to wait for α confirmations, meaning $\alpha - 1$ blocks need to be appended after the block where the Marie to John transaction is recorded.
4. Once α confirmations have been received, John ships the goods.
5. Meanwhile, Marie has started working on her own blockchain version where the Marie to John transaction is replaced by a Marie to Marie transaction.
6. At the shipment date, the main blockchain is ahead by xx blocks.
7. Marie's goal is then to work on her blockchain branch to catch up with the main branch. If she manages to do that, her branch will replace the public branch, and she will recover her bitcoin. She can therefore spend these bitcoins again, hence the name double spending.

The race between the two competing branches of the blockchain is summarized on Figure 4.1.

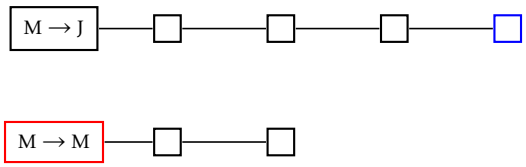


Figure 4.1: Double spending race illustrated, here we have $\alpha = 4$ and $X = 2$.

4.2 Random walk model

We define a discrete-time stochastic process $(X_n)_{n\geq 0}$ as the difference in length between the public and the private branches of the blockchain. At each time step, when a block is found, it belongs to the main branch with probability p and to the other branch with probability $q = 1 - p$. Here, the parameter p represents the proportion of hashpower owned by the honest miners, while q represents that of the attacker. We have

$$X_0 = X, \text{ and } X_n = x + \xi_1 + \ldots + \xi_n.$$

The ξ_i's are i.i.d. random variables such that

$$\mathbb{P}(\xi = 1) = p \in (0,1), \text{ and } \mathbb{P}(\xi = -1) = 1 - p = q,$$

$(X_n)_{n\geq 0}$ is therefore a random walk on $\mathbb{Z}$. We assume that $p > q$ so that the attacker does not hold more than half of the total hashpower. Define the double spending time as

$$\tau_0 = \inf\{n > 0 \; ; \; X_n = 0\}.$$

Our goal is to study the distribution of this stopping time with respect to the filtration

$$\mathcal{F}_n = \sigma(\xi_1, \ldots, \xi_n), \; n \geq 1.$$

An illustration of this first-hitting time problem is provided in Figure 4.2. Let us denote by

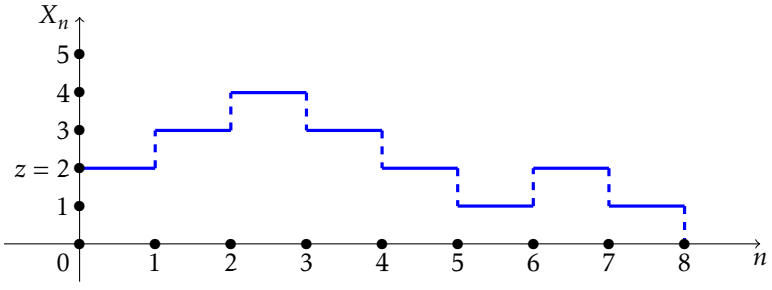


Figure 4.2: Illustration of the first-hitting time problem of a double spending attack.

$$\mathbb{P}_x(\cdot) = \mathbb{P}(\cdot | X_0 = x) \text{ and } \mathbb{E}_x(\cdot) = \mathbb{E}(\cdot | X_0 = x)$$

We are interested for now in the conditional distribution of τ_0 provided that $X_0 = x \geq 0$.

4.2.1 Double spending probability

The double spending probability is defined as $\mathbb{P}_x(\tau_0 < \infty)$. We can compute this probability by solving the so-called gambler's ruin problem. Let $a \geq x$ and define

$$\tau_a = \inf\{n \geq 0 \; ; \; X_n = a\}.$$

Further denote by

$$\phi(x, a) = \mathbb{P}(\tau_0 < \tau_a).$$

Note that

$$\mathbb{P}_x(\tau_0 < \infty) = \lim_{a \to \infty} \phi(x, a).$$

We have the following result

Theorem 3.

$$\phi(x, a) = \begin{cases} \frac{(q/p)^x - (q/p)^a}{1-(q/p)^a} & \text{if } p \neq q, \\ \frac{a-x}{a} & \text{if } p = q. \end{cases} \tag{4.1}$$

We give two proofs for this result, the first one uses simple first step analysis exploiting the Markov property of the random walk. The second one uses Martingale and the optional stopping theorem.

Proof 1:

By the law of total probability, we have

$$\phi(x, a) = p\phi(x, a+1) + (1-p)\phi(x, a-1), \; x \geq 1. \tag{4.2}$$

We also have the boundary conditions

$$\phi(0, a) = 1 \text{ and } \phi(a, a) = 0. \tag{4.3}$$

Equation (4.2) is a linear difference equation of order 2[1] associated to the following characteristic equation

$$px^2 - x + 1 - p = 0 \tag{4.4}$$

1. Assume that $p = 1 - p = 1/2$ then (4.4) has one solution

$$r = 1$$

The solutions of (4.2) are given by

$$\phi(x, a) = A + Bx$$

Using the boudary conditions (4.3), we deduce that

$$\phi(x, a) = \frac{a - x}{a},$$

as announced.

[1] For details about such equation on ecan check for instance https://mjo.osborne.economics.utoronto.ca/index.php/tutorial/index/1/sod/t

2. Assume that $p \neq 1-p$ which has two roots on the real line with

$$r_1 = 1, \text{ and } r_2 = \frac{1-p}{p}.$$

The solution of (4.2) is given by

$$\phi(x,a) = A + B\left(\frac{1-p}{p}\right)^x x,$$

where A and B are constant. Using the boudary conditions (4.3), we deduce that

$$\phi(x,a) = \frac{(q/p)^x - (q/p)^a}{1-(q/p)^a},$$

as announced.

For the second proof we need the notion of martingale

Definition 3. *A stochastic process $(X_n)_{n\geq 0}$, is called a martingale with respect to a filtration $\mathcal{F}_n$, if*

(i) X_n *is* $\mathcal{F}_n$*-adapted*

(ii) $\mathbb{E}(X_n) < \infty$ *for* $n \geq 0$

(iii) $\mathbb{E}(X_n|\mathcal{F}_{n-1}) = X_{n-1}$

and the optional stopping theorem.

Theorem 4. *Let T be a bounded stopping time for the martingale $(X_n)_{n\geq 0}$ then it holds that*

$$\mathbb{E}(X_T) = \mathbb{E}(X_0).$$

Proof 2:

Let $T = \tau_0 \wedge \tau_a$, it is a bounded stopping time.

Assume that $p = 1-p = 1/2$ then $(X_n)_{n\geq 0}$ is a martingale. We apply the optionnal stopping time theorem at T on $(X_n)_{n\geq 0}$. We have

$$\mathbb{E}(X_0) = x$$

on one hand and

$$\begin{aligned}\mathbb{E}(X_T) &= \mathbb{E}(X_{\tau_0}\mathbb{I}_{\tau_0\leq\tau_a} + X_{\tau_a}\mathbb{I}_{\tau_0>\tau_a}) \\ &= a\mathbb{P}(\tau_0 > \tau_a) \\ &= a[1-\phi(x,a)].\end{aligned}$$

From $\mathbb{E}(X_0) = \mathbb{E}(X_T)$, we deduce that

$$\phi(x,a) = \frac{a-x}{a}.$$

Assume that $p \neq 1-p$. Define the process

$$M_n^\theta = \exp\left[\theta X_n - n\kappa_\xi(\theta)\right], \text{ for } n \in \mathbb{N} \text{ and } \theta \in \mathbb{R},$$

where

$$\kappa_\xi(\theta) = \log\Big[\mathbb{E}\Big(e^{\theta\xi}\Big)\Big],$$

is the cumulant generating function of ξ.

Lemma 1. *Take s so that $\kappa_\xi(\theta) < \infty$ then $(M_n^\theta)_{n\geq 0}$ is a $\mathcal{F}_n$-martingale.*

Proof. We have that

$$\begin{aligned}
\mathbb{E}(M_n^\theta|\mathcal{F}_n) &= \mathbb{E}\Big\{\exp\Big[\theta X_n - n\kappa_\xi(\theta)\Big]|\mathcal{F}_{n-1}\Big\} \\
&= \exp\Big[\theta X_{n-1} - n\kappa_\xi(\theta)\Big]\mathbb{E}\left[\exp\left(s\xi_n\right)|\mathcal{F}_{n-1}\right] \\
&= \exp\Big[\theta X_{n-1} - n\kappa_\xi(\theta)\Big]\exp[\kappa_\xi(\theta)] \\
&= M_{n-1}.
\end{aligned}$$

□

The equation $\kappa_\xi(\theta) = 0$ is equivalent to

$$pe^s + qe^{-s} = 1,$$

which admits $\gamma = \log(q/p)$ as only non-zero solution. The process $(\mathrm{e}^{\gamma X_n})_{n\geq 0}$ is a $\mathcal{F}_n$-Martingale. Define $\tau_a = \inf\{n \geq 0\ ;\ X_n = a\}$, for $a > z$. We apply the optionnal stopping time theorem at T on $\left(e^{\gamma X_n}\right)_{n\geq 0}$. We have

$$\mathbb{E}_x(e^{\gamma X_0}) = e^{\gamma x},$$

and

$$\begin{aligned}
\mathbb{E}_x(e^{\gamma X_T}) &= \mathbb{E}_x(e^{\gamma X_{\tau_0}}\mathbb{I}_{\tau_0\leq\tau_a} + e^{\gamma X_{\tau_a}}\mathbb{I}_{\tau_0>\tau_a}) \\
&= \phi(x,a) + e^{\gamma a}(1-\phi(x,a)).
\end{aligned}$$

From $\mathbb{E}_x(X_0) = \mathbb{E}_x(X_T)$, we deduce that

$$\phi(x,a) = \frac{e^{\gamma x} - e^{\gamma a}}{1 - e^{\gamma a}}.$$

which is equivalent to (4.1) (recall that $\gamma = \log(q/p)$).

Corollary 1. *Assume that $p > q$ then the double spending probability is given by*

$$\mathbb{P}_x(\tau_0 < \infty) = \left(\frac{q}{p}\right)^x.$$

Proof. We have

$$\mathbb{P}(\tau_0 < \infty) = \lim_{a\to\infty}\phi(x,a) = \left(\frac{q}{p}\right)^x.$$

□

In practice the number of blocks x is actually random variable

$$X = (\alpha - M)_+,$$

where M corresponds to the number of blocks the attacker managed to mine while the vendor waits for α confirmations. If we assume that a block mined by the honest miners is a success while a block mined by the attacker is a failure then M actually counts the number of failure before α successes. We have that $M \sim \text{Neg-Bin}(\alpha, p)$ where M has a probability mass function (p.m.f.) given by

$$\mathbb{P}(M = m) = \binom{\alpha + m - 1}{m} p^{\alpha} q^{m}.$$

Whenever $X = 0$ then double spending occurs right away as $\phi(0) = 1$. To derive the double spending probability, we condition upon the values of X via the law of total probability as

$$\mathbb{P}(\text{Double Spending}) = \mathbb{P}(M \geq \alpha) + \sum_{m=0}^{\alpha-1} \binom{\alpha + m - 1}{m} q^{\alpha} p^{m}.$$

The analysis conducted here is similar to that of Rosenfeld [2014].

4.2.2 Double spending time

In the block mining world, time is money. Every hour spent computing hashes is costly in terms of energy. It is therefore very interesting to know whether a double-spending attack is meant to last long or not. Intuitively, we can think that if it must occur, it should happen at an earlier stage because, as $p > 1/2$, our random walk $(X_n)_{n \geq 0}$ will eventually drift towards $+\infty$. The following result provides the probability distribution of τ_0 when $X_0 = x$.

Theorem 5. *If $x = 0$ then $\tau_0 = 0$ almost surely. If $x > 0$ then τ_0 admits a p.m.f. given by*

$$\mathbb{P}_x(\tau_0 = n) = \frac{x}{n} \binom{n}{(n-x)/2} p^{(n-x)/2} q^{(n+x)/2} \text{ if } n \geq x \text{ and } n - x \text{ is even},$$

and 0 otherwise.

Proof. We start by showing the following lemma, sometimes referred to as the Markov hitting time theorem.

Lemma 2.

$$\mathbb{P}_x(\tau_0 = n) = \frac{x}{n} \mathbb{P}_x(X_n = 0), \; x \geq 0, \text{ and } n > 0. \tag{4.5}$$

Proof. If $x = 0$ then $\tau_0 = 0$ almost surely and both sides of (4.5) equal to 0. Assume that $x \geq 1$, we have that $\mathbb{P}_x(\tau_0 = n) = 0$ and $\mathbb{P}_x(X_n = 0) = 0$ whenever $n < x$ or $n - x$ is odd. The rest of the proof is by induction on $n \geq 1$, when $n = 1$ we have that

$$\mathbb{P}_x(\tau_0 = 1) = 0 = \frac{x}{1} \mathbb{P}_x(X_1 = 0), \text{ for } x > 1,$$

and

$$\mathbb{P}_1(\tau_0 = 1) = q = \frac{1}{1} \mathbb{P}_1(X_1 = 0), \text{ for } x = 1.$$

The property holds for $n = 1$. Assume that it holds for some $n \geq 1$. The law of total probability yields

$$\begin{aligned}
\mathbb{P}_x(\tau_0 = n+1) &= \sum_{y\in\{-1,1\}} \mathbb{P}_x(\tau_0 = n+1|\xi_1 = y)\mathbb{P}(\xi_1 = y) \\
&= \sum_{y\in\{-1,1\}} \mathbb{P}_{x+y}(\tau_0 = n)\mathbb{P}(\xi_1 = y) \text{ (Strong Markov Property)} \\
&= \sum_{y\in\{-1,1\}} \frac{x+y}{n}\mathbb{P}_{x+y}(X_n = 0)\mathbb{P}(\xi_1 = y).
\end{aligned}$$

Note that for any measurable application g we have

$$\begin{aligned}
\mathbb{E}_x[g(\xi_1)\mathbb{I}_{X_{n+1}=0}] &= \sum_{y\in\{-1,1\}} \mathbb{E}_x[g(\xi_1)\mathbb{I}_{X_{n+1}=0}|\xi_1 = y]\mathbb{P}(\xi_1 = y) \\
&= \sum_{y\in\{-1,1\}} g(y)\mathbb{P}_x(X_{n+1} = 0|\xi_1 = y)\mathbb{P}(\xi_1 = y) \\
&= \sum_{y\in\{-1,1\}} g(y)\mathbb{P}_{x+y}(X_n = 0)\mathbb{P}(\xi_1 = y)
\end{aligned}$$

Take $g(y) = \frac{x+y}{n}$ and further undo the law of total probability,

$$\begin{aligned}
\mathbb{P}_x(\tau_0 = n+1) &= \sum_{y\in\{-1,1\}} \frac{x+y}{n}\mathbb{P}_{x+y}(X_n = 0)\mathbb{P}(\xi_1 = y) \\
&= \sum_{y\in\{-1,1\}} \frac{x+y}{n}\mathbb{P}_x(X_{n+1} = 0|\xi_1 = y)\mathbb{P}(\xi_1 = y) \\
&= \sum_{y\in\{-1,1\}} \frac{x+y}{n}\mathbb{P}_x(\xi_1 = y|X_{n+1} = 0)\mathbb{P}_x(X_{n+1} = 0) \\
&= \frac{\mathbb{P}_x(X_{n+1} = 0)}{n}[x + \mathbb{E}(\xi_1|X_{n+1} = 0)]. \qquad (4.6)
\end{aligned}$$

Since the ξ_i are i.i.d. then it holds that

$$\mathbb{E}(\xi_1|X_{n+1} = 0) = \mathbb{E}(\xi_i|X_{n+1} = 0),\ i = 1,\dots,n+1,$$

and it follows that

$$\mathbb{E}(\xi_1|X_{n+1} = 0) = \frac{1}{n+1}\sum_{i=1}^{n+1}\mathbb{E}(\xi_i|X_{n+1} = 0) = \frac{-x}{n+1}.$$

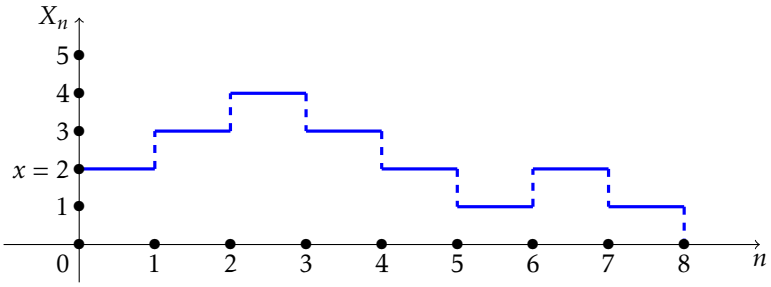
Inserting the above expression in (4.6) yields

$$\mathbb{P}_x(\tau_0 = n+1) = \frac{x}{n+1}\mathbb{P}_x(X_{n+1} = 0).$$

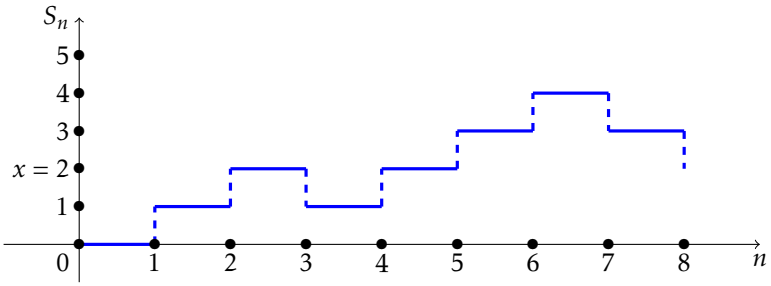
□

Remark 2. *This proof is direct, simple and inspired from van der Hofstad and Keane [2008]. It is possible to make it shorter by taking advantage of the ballot theorem. Indeed, consider again the first hitting problem on Figure 4.2 and reverse the timeline. It corresponds to that of a random walk $(S_n)_{n\geq 0}$ that starts at 0, make upward jumps with probability q, and ends up at the level x at time n without crossing the X axis, see Figure 4.3. We have equivalently*

$$\mathbb{P}_x(\tau_0 = n) = \mathbb{P}(S_k > 0,\ 1 \leq k \leq n|S_n = x, S_0 = 0)\mathbb{P}_0(S_n = x|S_0 = 0),$$



(a) Original first hitting problem



(b) Time reversed first hitting problem

Figure 4.3: Another look at the first hitting time problem.

and

$$\mathbb{P}(S_k > 0,\ 1 \leq k \leq n | S_n = x, S_0 = 0) = \frac{x + (n-x)/2 - (n-x)/2}{n} = \frac{x}{n}.$$

For proof of the ballot theorem, see Renault [2007]. For a general formulation and application to queueing see Takács [1962].

To complete the proof, we just note that

$$\mathbb{P}_x(X_n = 0) = \binom{n}{(n-x)/2} p^{(n-x)/2} q^{(n+x)/2}$$

as it corresponds to a trajectory of $(X_n)_{n\geq 0}$ starting at $X_0 = x$ ending at 0 made of $(n-x)/2$ upward jumps and $(n+x)/2$ downward one. □

Just like in the previous section, the actual double spending time depends on the value of the random variable $Z = (\alpha - M)_+$.

4.3 Counting process model

Our aim is to go from the discrete time framework of the previous section to a continuous time. To do so, we will model the length of the blockchain as counting processes. We will consider renewal processes and more specifically Poisson processes. We start by giving some reminders on the exponential distribution and counting processes before studying the double spending time distribution.

4.3.1 Poisson process, Exponential distributions and friends

Definition 4. *A counting process $(N_t)_{t\geq 0}$ is a continuous time stochastic process that counts the occurence of an event over time such that*

$$N_0 = 0 \text{ and } N_t = \sum_{k=1}^{+\infty} \mathbb{I}_{T_k \leq t}.$$

where $T_1, T_2, T_3, \ldots$ denote the arrival times, with the convention that $T_0 = 0$. Let $\Delta_0^T, \Delta_1^T, \Delta_2^T, \ldots$ be the sequence of inter-arrival times defined as

$$\Delta_k^T = T_{k+1} - T_k,\ k = 0, 1, 2 \ldots.$$

A trajectory of a counting process is given in

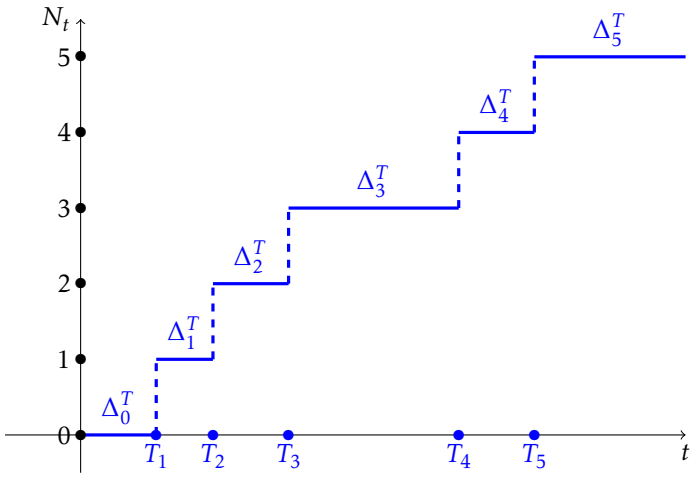


Figure 4.4: Trajectory of the counting process $(N_t)_{t\geq 0}$.

Definition 5. *A Poisson process $(N_t)_{t\geq 0}$ is a counting process whose inter-arrival times are i.i.d. exponential random variables.*

Remark 3. *A Poisson process belongs to the family renewal processes which are counting process with i.i.d. inter-arrival times.*

Definition 6. *A random variable X is exponentially distributed $X \sim \mathsf{Exp}(\lambda)$ if it has p.d.f.*

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x}, & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases}$$

For some reasons, we need to introduce the joint distribution of the order statistics of a uniform random sample.

Proposition 1. *Let $U_1, \ldots, U_n$ be a sample of i.i.d. uniform random variables on (a, b). Denote by*

$$U_{1:n} \leq U_{2:n} \leq \ldots \leq U_{n:n}$$

the order statistics of such a sample. The joint distribution of $(U_{(1)},\dots,U_{(n)})$ *is given by*

$$f_{(U_{1:n},\dots,U_{n:n})}(u_1,\dots,u_n)=\frac{n!}{(b-a)^n}\mathbb{I}_{a<u_1<\dots<u_n<b}(u_1,\dots,u_n).$$

and we denote $(U_{1:n},\dots,U_{n:n})\sim OS_n([a,b])$

Proof. Let $g:\mathbb{R}^n\mapsto\mathbb{R}_+$ be measurable and bounded. We have that

$$\mathbb{E}[g(U_{1:n},\dots,U_{n:n})]=\mathbb{E}\left[\sum_{\sigma\in\mathcal{S}_n}g(U_{\sigma(1)},\dots,U_{\sigma(n)})\mathbb{I}_{U_{\sigma(1)}<\dots<U_{\sigma(n)}}\right]$$

where $\mathcal{S}_n$ the set of all the permutation of $\{1,\dots,n\}$. We note that

$$\begin{aligned}\mathbb{E}\left[g(U_{\sigma(1)},\dots,U_{\sigma(n)})\mathbb{I}_{U_{\sigma(1)}<\dots<U_{\sigma(n)}}\right] &= \mathbb{E}\left[g(U_1,\dots,U_n)\mathbb{I}_{U_1<\dots<U_n}\right]\\ &= \int_{\mathbb{R}^n}g(u_1,\dots,u_n)\mathbb{I}_{u_1<\dots<u_n}\frac{1}{(b-a)^n}\mathrm{d}\lambda_n(u_1,\dots,u_n).\end{aligned}$$

It then follows that

$$\mathbb{E}[g(U_{1:n},\dots,U_{n:n})]=\int_{\mathbb{R}^n}g(u_1,\dots,u_n)\mathbb{I}_{u_1<\dots<u_n}\frac{n!}{(b-a)^n}\mathrm{d}\lambda_n(u_1,\dots,u_n).$$

□

We require some additional result about the gamma distribution.

Proposition 2. *Let* $\Delta_1^T,\dots,\Delta_n^T$ *be i.i.d. exponential* $Exp(\lambda)$ *random variables, define the sequence* $T_k=\sum_{i=1}^k\Delta_i^T, k=1,\dots,n.$

1. *The* T_k*'s are gamma distributed,* $T_k\sim\mathsf{Gamma}(k,\lambda)$ *with p.d.f.*

$$f_{T_k}(t)=\frac{t^{k-1}e^{-\lambda t}\lambda^k}{\Gamma(k)},\ t>0,$$

where $\Gamma(k)=\int_0^\infty \mathrm{e}^{-x}x^{k-1}dx.$

2. *The joint distribution of* $(T_1,\dots,T_n)$ *has p.m.f. given by*

$$f_{(T_1,\dots,T_n)}(t_1,\dots,t_n)=\lambda^n e^{-\lambda t_n}\mathbb{I}_{0<t_1<\dots<t_n}(t_1,\dots,t_n)$$

3. $[(T_1,\dots,T_n)|T_{n+1}=t]\sim OS_n([0,t])$

Proof. 1. We use induction on $k\geq 1$. For $k=1$ we have that $\Delta_1^T=T_1$ so the property holds. Assume that the property hold true for some k and consider $k+1$. We note that $T_{k+1}=T_k+\Delta_{k+1}^T$ then

$$\begin{aligned}f_{T_{k+1}}(t) &= \int_0^t f_{T_k}(x)f_{\Delta_{k+1}^T}(t-x)\mathrm{d}x\\ &= \int_0^t\frac{x^{k-1}e^{-\lambda x}\lambda^k}{(k-1)!}\lambda e^{-\lambda(t-x)}\mathrm{d}x\\ &= \frac{e^{-\lambda t}\lambda^{k+1}}{(k-1)!}\frac{t^k}{k}=\frac{t^k e^{-\lambda t}\lambda^{k+1}}{k!}.\end{aligned}$$

Exercise 1. *Can you propose another way to show this result? Without using induction.*

2. Let $g : \mathbb{R}^n \mapsto \mathbb{R}_+$ be measurable and bounded, we have

$$\begin{aligned}\mathbb{E}[g(T_1,\dots,T_n)] &= \mathbb{E}[g(\Delta_1^T, \Delta_1^T + \Delta_2^T \dots, \Delta_1^T + \dots + \Delta_n^T)] \\ &= \int_{\mathbb{R}^n} g(t_1,\dots,t_1+\dots+t_n) f_{(\Delta_1^T,\dots,\Delta_n^T)}(t_1,\dots,t_n)\mathrm{d}\lambda_n(t_1,\dots,t_n) \\ &= \int_{\mathbb{R}_+^n} g(t_1,\dots,t_1+\dots+t_n)\lambda^n \mathrm{e}^{-\lambda(t_1+\dots+t_n)}\mathrm{d}\lambda_n(t_1,\dots,t_n)\end{aligned}$$

Let us apply the following change of variable

$$\Phi : (u_1,\dots,u_n) \mapsto (u_1, u_2-u_1,\dots,u_n-u_{n-1}) := (t_1,\dots,t_n),$$

minding the change in the integration domain as

$$\Phi(\mathbb{R}_+\times]u_1,\infty[\times\dots\times]u_{n-1},\infty[) = \mathbb{R}_+^n$$

and the Jacobian $\left|\frac{\mathrm{d}\Phi}{\mathrm{d}u}\right| = 1$. It follows that

$$\mathbb{E}[g(T_1,\dots,T_n)] = \int_{\mathbb{R}^n} g(u_1,\dots,u_n)\lambda^n e^{-\lambda u_n}\mathbb{I}_{0<u_1<\dots<u_n}(u_1,\dots,u_n)\mathrm{d}\lambda_n(u_1,\dots,u_n).$$

3. We have that

$$\begin{aligned}f_{T_1,\dots,T_n|T_{n+1}}(t_1,\dots,t_n|t) &= \frac{f_{T_1,\dots,T_n,T_{n+1}}(t_1,\dots,t_n,t)}{f_{T_{n+1}(t)}} \\ &= \frac{n!}{t^n}\mathbb{I}_{0<t_1<\dots<t_n<t}(t_1,\dots,t_n,t).\end{aligned}$$

□

The fact that the Poisson process is a Levy process will be useful later on, so here it is

Proposition 3. *Provided that $\{N_t = n\}$, the jump times $T_1,\dots,T_n$ have the same distribution as the order statistic of an i.i.d. sample of n uniform random variable on $(0,t)$, namely it holds that*

$$[T_1,\dots,T_n|N_t = n] \sim (U_{1:n}(0,t),\dots,U_{n:n}(0,t)).$$

Proof. We have

$$\begin{aligned}&\mathbb{E}[g(T_1,\dots,T_n)|N_t = n] \\ =\ & \frac{\mathbb{E}\left[g(T_1,\dots,T_n)\mathbb{I}_{N_t=n}\right]}{\mathbb{P}(N_t=n)} \\ =\ & \frac{\mathbb{E}\left[g(T_1,\dots,T_n)\mathbb{I}_{T_n\le t}\mathbb{I}_{T_{n+1}>t}\right]}{\mathbb{P}(N_t=n)} \\ =\ & \frac{n!}{e^{-\lambda t}(\lambda t)^n}\int_{\mathbb{R}^{n+1}} g(t_1,\dots,t_n)\mathbb{I}_{t_n\le t<t_{n+1}}(t_n,t_{n+1})f_{T_1,\dots,T_{n+1}}(t_1,\dots,t_{n+1})\mathrm{d}\lambda_{n+1}(t_1,\dots,t_{n+1}) \\ =\ & \frac{n!}{e^{-\lambda t}(\lambda t)^n}\int_{\mathbb{R}^n}\int_t^{+\infty} g(t_1,\dots,t_n)\mathbb{I}_{0<t_1<\dots t_n\le t}(t_1,\dots,t_n)\lambda^{n+1}e^{-\lambda t_{n+1}}\mathrm{d}\lambda_{n+1}(t_1,\dots,t_{n+1}) \\ =\ & \frac{n!}{e^{-\lambda t}(\lambda t)^n}\int_{\mathbb{R}^n} g(t_1,\dots,t_n)\mathbb{I}_{0<t_1<\dots t_n\le t}(t_1,\dots,t_n)\lambda^n\mathrm{d}\lambda_n(t_1,\dots,t_n)\mathrm{e}^{-\lambda t} \\ =\ & \int_{\mathbb{R}^n} g(t_1,\dots,t_n)\frac{n!}{t^n}\mathbb{I}_{0<t_1<\dots t_n\le t}(t_1,\dots,t_n)\mathrm{d}\lambda(t_1,\dots,t_n).\end{aligned}$$

□

The order statistic property of the Poisson process will be useful to solve the first hitting time problem arising later on.

4.3.2 Levy process and continuous time martingale

Levy processes are the continuous time equivalent of random walks. Let $(\Omega, \mathcal{F}, \mathcal{F}_t, \mathbb{P})$ be a filtered probability space and X a $\mathcal{F}_t-$adapted stochastic process.

Definition 7. *X is a Levy process if*

1. $X_0 = 0$
2. $X_t - X_s \overset{\mathcal{D}}{=} X_{t-s}$ *(Stationary increments)*
3. *For all $n \in \mathbb{N}$ and $0 \leq s_1 \leq t_1 \leq s_2 \leq t_2 \leq \ldots \leq s_n \leq t_n < \infty$, the increments*

$$X_{t_i} - X_{s_i},\ i = 1, \ldots, n,$$

   are independent.

4. *The trajectories of X are càdlàg (right-continuous and left-limited)*

Proposition 4. *The following statements are equivalent*

1. $(N_t)_{t\geq 0}$ *is a Poisson process*
2. *The stochastic process $(N_t)_{t\geq 0}$ has*
   - *(i) independent increments, it means that for $0 < t_1 \leq \ldots \leq t_n$, the random variables $N_{t_1}, N_{t_2} - N_{t_1}, \ldots, N_{t_n} - N_{t_{n-1}}$ are independent.*
   - *(ii) stationnary increments in the sense that the event frequency distribution over some time period of duration $s > 0$ only depends on s. Indeed, we have that*

$$N_{t+s} - N_t \sim \mathsf{Poisson}(\lambda s), \text{ for } s, t \geq 0.$$

Poisson processes are prominent examples of Levy processes.

Proof. $\underline{1 \Rightarrow 2}$
Assume that $(N_t)_{t\geq 0}$ is a Poisson process and let $0 < t_1 < \ldots < t_n$ be some times. Consider the folowing probability

$$\mathbb{P}\left(N_{t_1} = j_1, N_{t_2} - N_{t_1} = j_2, \ldots, N_{t_n} - N_{t_{n-1}} = j_n\right)$$

such that $j_1, \ldots, j_n \in \mathbb{N}$. We can rewrite it as

$$\mathbb{P}\left(T_{k_1} \leq t_1 < T_{k_1+1}, T_{k_2} \leq t_2 < T_{k_2+1}, \ldots, T_{k_n} \leq t_n < T_{k_n+1}\right),$$

where $k_i = j_1 + \ldots + j_i, i = 1, \ldots, n$. Conditionning with respect to T_{k_n+1} yields

$$\begin{aligned}
& \mathbb{P}\Big(T_{k_1} \leq t_1 < T_{k_1+1}, T_{k_2} \leq t_2 < T_{k_2+1}, \ldots, T_{k_n} \leq t_n < T_{k_n+1}\Big) \\
= & \int_{t_n}^{+\infty} \mathbb{P}\Big(T_{k_1} < t_1 < T_{k_1+1}, T_{k_2} < t_2 < T_{k_2+1}, \ldots, T_{k_n} < t_n | T_{k_n+1} = t\Big) f_{T_{k_n+1}}(t) \mathrm{d}\lambda(t) \\
= & \int_{t_n}^{+\infty} \binom{k_n}{j_1, \ldots, j_n} \Big(\frac{t_1}{t}\Big)^{j_1} \Big(\frac{t_2 - t_1}{t}\Big)^{j_2} \ldots \Big(\frac{t_n - t_{n-1}}{t}\Big)^{j_n} \frac{e^{-\lambda t} t^{k_n} \lambda^{k_n+1}}{k_n!} \mathrm{d}\lambda(t) \\
= & \frac{e^{-\lambda t_1} (t_1)^{j_1}}{j_1!} \frac{(t_2 - t_1)^{j_2} e^{-\lambda(t_2 - t_1)}}{j_2!} \ldots \frac{(t_n - t_{n-1})^{j_n} e^{-\lambda(t_n - t_{n-1})}}{j_n!}
\end{aligned}$$

From the second to the third equality we simply ask that amomg k_n uniform random variables j_1 fall inside $(0, t_1)$, j_2 fall inside (t_1, t_2), etc...

$\underline{2 \Rightarrow 1}$

We aim at showing that $(T_1, \ldots, T_n)$ has p.d.f. given by

$$f_{T_1, \ldots, T_n}(t_1, \ldots, t_n) = \lambda^n e^{-\lambda t_n} \mathbb{I}_{0 < t_1 < \ldots < t_n}. \tag{4.7}$$

Let $t_1, \ldots, t_n$ and h be nonnegative real numbers such that

$$t_1 < t_1 + h < t_2 < \ldots < t_n < t_n + h,$$

We have

$$\begin{aligned}
& \mathbb{P}(t_1 < T_1 < t_1 + h, \ldots, t_n < T_1 < t_n + h) \\
= & \mathbb{P}(N_{t_1} = 0, N_{t_1+h} - N_{t_1} = 1, \ldots, N_{t_n} - N_{t_{n-1}+h} = 0, N_{t_n+h} - N_{t_n} \geq 1) \\
= & e^{-\lambda t_1} e^{-\lambda h} \lambda h e^{-\lambda[t_2 - (t_1+h)]} e^{-\lambda h} \lambda h \ldots e^{-\lambda[t_n - (t_{n-1}+h)]} [1 - e^{-\lambda h}] \\
= & e^{-\lambda t_n} \lambda^{n-1} h^{n-1} [1 - e^{-\lambda h}]
\end{aligned}$$

Divide by h^n and let h go to 0 to get (4.7). After applying a change of variable (reciprocal of that used in the proof of Proposition 2) to recover the joint distribution of $(\Delta_1^T, \ldots, \Delta_n^T)$, we see that the later is actually that of an i.i.d. sample of size n of exponential random variables. □

Definition 8. *The Laplace exponent of X is given by*

$$\kappa(\theta) = \log \mathbb{E}(\mathrm{e}^{\theta X_1}).$$

Proposition 5. *We have*

$$\log \mathbb{E}(\mathrm{e}^{\theta X_t}) = \kappa(\theta) t, \text{ for } t \geq 0$$

Proof. Let $t \geq 0$ and $n \in \mathbb{N}$, we have

$$X_t = \sum_{k=0}^{n-1} \Big(X_{(k+1)\frac{t}{n}} - X_{k\frac{t}{n}}\Big).$$

It follows that

$$\begin{aligned}\mathbb{E}\left(e^{\theta X_t}\right) &= \mathbb{E}\left(e^{\theta\sum_{k=0}^{n-1}\left(X_{(k+1)\frac{t}{n}}-X_{k\frac{t}{n}}\right)}\right)\\ &= \prod_{k=0}^{n-1}\mathbb{E}\left(e^{\theta\left(X_{(k+1)\frac{t}{n}}-X_{k\frac{t}{n}}\right)}\right)\\ &= \prod_{k=0}^{n-1}\mathbb{E}\left(e^{\theta X_{t/n}}\right)\\ &= \mathbb{E}\left(e^{\theta X_{t/n}}\right)^n\end{aligned}$$

If we denote by

$$\kappa_t(\theta) = \log\mathbb{E}\left(e^{\theta X_t}\right)$$

then it holds that

$$\kappa_t(\theta) = n\kappa_{t/n}(\theta),$$

$\forall t \geq 0$ and $\forall n \in \mathbb{N}$. Furthermore, for $m \in \mathbb{N}$, we have

$$\kappa_m(\theta) = m\kappa(\theta) \text{ and } \kappa_m(\theta) = n\kappa_{m/n}(\theta).$$

Hence

$$\kappa_t(\theta) = t\kappa(\theta) \text{ pour } t \in \mathbb{Q}$$

Consider $(t_n)_{n\geq 0} \in \mathbb{Q}$ (rational numbers) such that $t_n \downarrow t \in \mathbb{R}$, taking the limit yields

$$\kappa_t(\theta) = t\kappa(\theta) \text{ pour } t \in \mathbb{R},$$

thanks to the cadlag nature of the trajectories of X □

Definition 9. *X is a $\mathcal{F}_t$-martingale if it holds that*

- $\mathbb{E}(|X_t|) < \infty$ *for* $t \geq 0$
- $\mathbb{E}(X_t|\mathcal{F}_s) = X_s$ *for every* $s \leq t$.

Theorem 6. *Let X be a martingale and τ be a bounded stopping time, then $(X_{\tau\wedge t})_{t\geq 0}$ is a martingale and we have that*

$$\mathbb{E}(X_\tau) = \mathbb{E}(X_0).$$

Proposition 6 (Wald's exponential martingale)**.** *Let X be a Levy process then the process*

$$M_t = \exp\left[\theta X_t - t\kappa(\theta)\right],\ t \geq 0.$$

is a $\mathcal{F}_t$-martingale.

Proof. Let $s \leq t$, we have

$$\begin{aligned}
\mathbb{E}(M_t|\mathcal{F}_s) &= \mathrm{e}^{-t\kappa(\theta)}\mathbb{E}\left[\mathrm{e}^{\theta X_t}|\mathcal{F}_s\right] \\
&= \mathrm{e}^{-t\kappa(\theta)}\mathbb{E}\left[\mathrm{e}^{\theta(X_t-X_s)+\theta X_s}|\mathcal{F}_s\right] \\
&= \mathrm{e}^{-t\kappa(\theta)}\mathrm{e}^{\theta X_s}\mathbb{E}\left[\mathrm{e}^{\theta(X_t-X_s)}\right] \\
&= \mathrm{e}^{-t\kappa(\theta)}\mathrm{e}^{\theta X_s}\mathrm{e}^{(t-s)\kappa(\theta)} \\
&= M_s
\end{aligned}$$

□

4.3.3 Double spending probability

The *Proof-of-Work* protocol implies a steady block arrival, every 10 minnutes for the bitcoin blockchain. Each trial (of the network) for mining a block is independent of the others and leads to a success with very small probability, the overall number of successes is binomially distributed, very well approximated by a Poisson random variable. This justifies the Poisson process assumption made in the sequel to model the block arrival.

Denote by $(x + N_t)_{t\geq 0}$ and $(M_t)_{t\geq 0}$ the number of blocks found by the honest miners and the attackers respectively. Double spending occurs at time

$$\tau_0 = \inf\{t \geq 0\ ;\ x + N_t = M_t\}.$$

Assume that $(N_t)_{t\geq 0}$ and $(M_t)_{t\geq 0}$ are Poisson processes with intensity λ and μ such that $\lambda > \mu$. Let $(T_i)_{i\geq 0}$ and $(S_i)_{i\geq 0}$ be the arrival times of $(N_t)_{t\geq 0}$ and $(M_t)_{t\geq 0}$ The first-hitting problem along with its notation is illustrated in Figure 4.5. The double spending probability is given by the

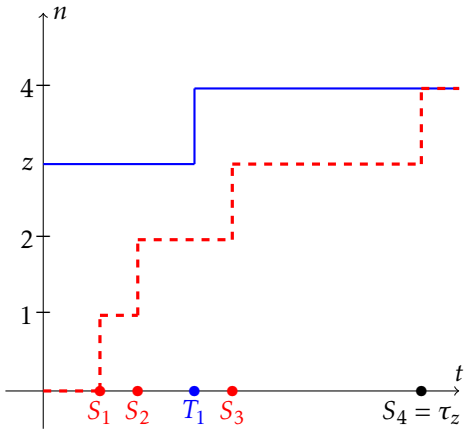


Figure 4.5: Illustration of the double spending problem within a continuous time framework.

following result

Theorem 7. *The double spending probability is given by*

$$\mathbb{P}(\tau_0 < \infty) = \left(\frac{\mu}{\lambda}\right)^x.$$

Proof. Define the processes

$$X_t = x + N_t - M_t$$

and

$$Y_t = x - X_t = M_t - N_t,\ t \geq 0.$$

$(Y_t)_{t\geq 0}$ is a Lévy process such that $Y_t \to -\infty$ because $\lambda > \mu$. Recall that

$$\exp(\theta Y_t - t\kappa(\theta)),\ t \geq 0,$$

is a martingale. We would like to find $\gamma > 0$ such that $\exp(\gamma Y_t)$. Consider the equation

$$\kappa(\theta) = \log \mathbb{E}\left(e^{\theta Y_1}\right) = 0.$$

Tt is equivalent to

$$\mu e^{\theta} + \lambda e^{-\theta} - (\mu + \lambda) = 0,$$

which has two solutions: 0 and

$$\gamma = \log\left(\frac{\lambda}{\mu}\right).$$

It follows from Proposition 6 that $(e^{\gamma Y_t})_{t\geq 0}$ is a martingale. We apply the optional stopping theorem (that's Theorem 6) to the process $e^{\gamma Y_t}$ at $t \wedge \tau_0$. We have

$$\mathbb{E}(e^{\gamma Y_0}) = 1,$$

and

$$\mathbb{E}(e^{\gamma Y_{\tau_0 \wedge t}}) = \mathbb{E}(e^{\gamma Y_{\tau_0}} \mathbb{I}_{\tau_0 < t}) + \mathbb{E}(e^{\gamma Y_t} \mathbb{I}_{\tau_0 \geq t}).$$

We are going to take the limit $t \to \infty$. Note that

$$\tau_0 = \inf\{t \geq 0\ ;\ x + N_t = M_t\} = \inf\{t \geq 0\ ;\ X_t = 0\} = \inf\{t \geq 0\ ;\ Y_t = x\}.$$

We deduce that $Y_{\tau_0} = x$ and that $e^{\gamma Y_t} < e^{\gamma x}$ on $\{\tau_0 \geq t\}$. Applying the dominated convergence theorem yields

$$\mathbb{E}(e^{\gamma Y_{\tau_0 \wedge t}}) \underset{t\to\infty}{\to} e^{\gamma x}\mathbb{P}(\tau_0 < \infty).$$

The identity

$$\mathbb{E}(e^{\gamma Y_0}) = \mathbb{E}(e^{\gamma Y_{\tau_0 \wedge t}})$$

holds for any t including $t \to \infty$ which in turns leads to

$$1 = e^{\gamma x}\mathbb{P}(\tau_0 < \infty) \Leftrightarrow \mathbb{P}(\tau_0 < \infty) = \left(\frac{\mu}{\lambda}\right)^x.$$

□

4.3.4 Double spending time

Just like in Section 4.2.2, we are interested in the time required to complete a double spending attack. Accounting for the cost of electricity, we can approximate the operational cost per time unit by

$$c = \pi_W \cdot W \cdot q,$$

where

- π_W is the electricity cost
- W is the electricity consumed by the network
- q is the attacker's hashpower

The double spending cost reduces to $\tau_z \cdot c$. the following result provides a formula for the p.d.f. of τ_z.

Theorem 8. *If $(N_t)_{t\geq 0}$ is a Poisson process and $(M_t)_{t\geq 0}$ is a renewal process then the **p.d.f.** of τ_z is given by*

$$f_{\tau_0}(t) = \mathbb{E}\left[\frac{x}{x+N_t} f_{S_{N_t+x}}(t)\right], \text{ for } t \geq 0, \tag{4.8}$$

where $S_1, S_2, \ldots$, denotes the arrival times of $(M_t)_{t\geq 0}$.

Proof. The event $\{\tau_0 \in (t, t+\mathrm{d}t)\}$, for $t \geq 0$, corresponds to the exact time at which the double-spending attack is successful as the malicious chain takes over the honest one. At time $t = 0$, the honest chain is ahead by $x \geq 1$ blocks. Assuming that later, at time $t > 0$, the honest miners manage to add $N_t = n \in \mathbb{N}$ blocks to the chain then the malicious chain must be of length $M(t^-) = n + x - 1$ at some time $t^- < t$ and jumps to the level $n + x$ exactly at t. Conditioning over the values of $\{N_t\ ,\ t \geq 0\}$ yields

$$\{\tau_0 \in (t, t+\mathrm{d}t)\} = \bigcup_{n=0}^{+\infty} \{\tau_0 \in (t, t+\mathrm{d}t)\} \cap \{N_t = n\}. \tag{4.9}$$

In the case where $N_t = 0$, the only requirement is that the x^{th} jump of $(M_t)_{t\geq 0}$ occurs at time t. It then follows that

$$\{\tau_0 \in (t, t+\mathrm{d}t)\} \cap \{N_t = 0\} = \{S_x \in (t, t+\mathrm{d}t)\} \cap \{N_t = 0\}, \tag{4.10}$$

and consequently

$$f_{\tau_0|N_t}(t|0) = f_{S_x}(t),\ t \geq 0, \tag{4.11}$$

where $f_{\tau_0|N_t}(t|0)$ denotes the conditional p.d.f. of τ_0 given that $N_t = 0$. On the set $\{N_t \geq 1\}$, one needs to make sure that $\{M_t\ ,\ t \geq 0\}$ behaves properly by constraining its jump times so that it does not reach $N_s + x$ at any time $s < t$ and performs the $(n+x)^{\text{th}}$ jump at t. Hence, it holds that

$$\{\tau_0 \in (t, t+\mathrm{d}t)\} \cap \{N_t \geq 1\} = \bigcup_{n=1}^{+\infty} \bigcap_{k=1}^{n} \{T_k \leq S_{x+k-1}\} \cap \{S_{x+n} \in (t, t+\mathrm{d}t)\} \cap \{N_t = n\}.$$

Applying the law of total probability yields

$$\begin{aligned}
&\mathbb{P}\left(\{\tau_0 \in (t, t+\mathrm{d}t)\} \cap \{N(t) \geq 1\}\right) \\
= &\sum_{n=1}^{+\infty} \mathbb{P}\left[\bigcap_{k=1}^{n} \{T_k \leq S_{x+k-1}\} \cap \{S_{x+n} \in (t, t+\mathrm{d}t)\} \middle| N(t) = n\right] \mathbb{P}[N(t) = n]. \qquad (4.12)
\end{aligned}$$

In virtue of the order statistic property, the successive jump times $(T_1, \ldots, T_n)$ are distributed as the order statistics $(U_{1:n}(0,t), \ldots, U_{n:n}(0,t))$ of a sample of n i.i.d. random variables uniformly distributed on $(0,t)$. The conditional probability in (4.12) may be rewritten as

$$\begin{aligned}
&\mathbb{P}\left[\bigcap_{k=1}^{n} \{U_{k:n}(0,t) \leq S_{x+k-1}\} \cap \{S_{x+n} \in (t, t+\mathrm{d}t)\}\right] \\
= &\mathbb{P}\left[\bigcap_{k=1}^{n} \{U_{k:n}(0,1) \leq S_{x+k-1}/t\} \cap \{S_{x+n} \in (t, t+\mathrm{d}t)\}\right] \\
= &\mathbb{P}\left[\bigcap_{k=1}^{n} \{U_{k:n}(0,1) \leq S_{x+k-1}/t\} \middle| S_{x+n} \in (t, t+\mathrm{d}t)\right] \mathbb{P}[S_{x+n} \in (t, t+\mathrm{d}t)] \\
= &\mathbb{E}\left\{(-1)^n G_n[0 \middle| S_x/t, \ldots, S_{x+n-1}/t] \middle| S_{x+n} \in (t, t+\mathrm{d}t)\right\} \mathbb{P}[S_{x+n} \in (t, t+\mathrm{d}t)], \\
= &(-1/t)^n \mathbb{E}\left\{G_n[0 \middle| S_x, \ldots, S_{x+n-1}] \middle| S_{x+n} \in (t, t+\mathrm{d}t)\right\} \mathbb{P}[S_{x+n} \in (t, t+\mathrm{d}t)],
\end{aligned} \qquad (4.13)$$

where $(G_n(.|.))_{n\geq 0}$ correspond to the sequence of A-G polynomials as defined in Section 4.4. The last equation in (4.13) follows from using the first identity of Proposition 9. Inserting (4.13) into (4.12) and letting $\mathrm{d}t$ be small enough yields

$$\begin{aligned}
f_{\tau_0|N(t)\geq 1}(t) = &\sum_{n=1}^{+\infty} (-1/t)^n \mathbb{E}\left\{G_n[0 \middle| S_x, \ldots, S_{x+n-1}] \middle| S_{x+n} = t\right\} \\
\times &f_{S_{x+n}}(t) \mathbb{P}[N(t) = n]. \qquad (4.14)
\end{aligned}$$

We further work on the AG polynomials to simplify the above expressions. We have that

$$\begin{aligned}
\mathbb{E}\left\{G_n(0 \middle| S_x, \ldots, S_{x+n-1}) | S_{x+n} = t\right\} &= \mathbb{E}\{G_n(-S_x | 0, \ldots, S_{x+n-1} - S_x) | S_{x+n} = t\} \\
&= \mathbb{E}\{\mathbb{E}[G_n(-S_x | 0, \ldots, S_{x+n-1} - S_x) | S_{x+n} - S_x, S_{x+n}] | S_{x+n} = t\} \\
&= \mathbb{E}\left\{(-S_x)(-S_x - S_{n+x} + S_x)^{n-1} | S_{x+n} = t\right\} \\
&= (-1)^n \mathbb{E}\left\{S_x S_{n+x}^{n-1} | S_{x+n} = t\right\} \\
&= (-1)^n t^{n-1} \frac{x}{x+n} t = (-t)^n \frac{x}{x+n} \qquad (4.15)
\end{aligned}$$

Inserting (4.15) into (4.14) yields

$$f_{\tau_0|N_t\geq 1}(t) = \sum_{n=1}^{+\infty} \frac{x}{x+n} f_{S_{x+n}}(t) \mathbb{P}(N_t = n)$$

The final step consists in adding the case $N_t = 0$ to the sum, therefore writing

$$f_{\tau_0}(t) = \sum_{n=0}^{+\infty} \frac{x}{x+n} f_{S_{x+n}}(t) \mathbb{P}(N_t = n)$$

which is equivalent to the announced result (4.8). □

Exercise 2. *Assume that $(M_t)_{t\geq 0}$ is a Poisson process with intensity μ, compute*

$$\mathbb{P}(\tau_0 < \infty) = \int_0^\infty f_{\tau_0}(t)dt = \left(\frac{\mu}{\lambda}\right)^x.$$

See Goffard [2019, Example 1] for the derivation. It is easier to verify the formula using python.

Remark 4. *Just like in the random walk framework of Section 4.2, the number z is actually a random variable defined as*

$$Z = (\alpha - M_{T_\alpha})_+,$$

where T_α is the arrival time of the α^{th} block in the main branch of the blockchain. If $(M_t)_{t\geq 0}$ is a Poisson process with intensity μ then $M_{T_\alpha})$ is mixed Poisson distributed with parameter $\mu \cdot T_\alpha$. We have that

$$\begin{aligned}
\mathbb{P}(M_{T_\alpha} = m) &= \int_0^\infty \frac{e^{-\mu t}(\mu t)^m}{m!}\frac{e^{-\lambda t}t^{\alpha-1}\lambda^\alpha}{(\alpha-1)}dt \\
&= \frac{\mu^m\lambda^\alpha}{m!(\alpha-1)!}\int_0^\infty e^{-t(\mu+\lambda)}t^{m+\alpha-1}dt \\
&= \binom{m+\alpha-1}{m}\left(\frac{\lambda}{\lambda+\mu}\right)^\alpha\left(\frac{\mu}{\lambda+\mu}\right)^m.
\end{aligned}$$

The number of blocks found by the attacker until the vendor's transaction gets α confirmations is governed by a negative binomial distribution.

For further results on the distribution of τ_0 with different set of assumptions, the reader is referred to Goffard [2019].

4.4 Appendix: Appell and Abel-Gontcharov polynomials

Let $U = \{u_i, i \geq 1\}$ be a non-decreasing sequence of real numbers. To U is attached a (unique) family of Appell polynomials of degree n in x, $\{A_n(x|U), n \geq 0\}$ defined as follows.

Definition 10. *Starting with $A_0(x|U) = 1$, the $A_n(x|U)$'s satisfy the differential equations*

$$A_n^{(1)}(x|U) = nA_{n-1}(x|U), \tag{4.16}$$

with the border conditions

$$A_n(u_n|U) = 0, \quad n \geq 1. \tag{4.17}$$

So, each A_n has the integral representation

$$A_n(x|U) = n!\int_{u_n}^x\left[\int_{u_{n-1}}^{y_n} dy_{n-1}\dots\int_{u_1}^{y_1} dy_1\right]dy_n, \quad n \geq 1. \tag{4.18}$$

In parallel, to U is attached a (unique) family of Abel-Gontcharov (A-G) polynomials of degree n in x, $\{G_n(x|U), n \geq 0\}$.

Definition 11. *Starting with $G_0(x|U) = 1$, the $G_n(x|U)$'s satisfy the differential equations*

$$G_n^{(1)}(x|U) = nG_{n-1}(x|\mathcal{E}U), \tag{4.19}$$

where $\mathcal{E}U$ is the shifted family $\{u_{i+1}, i \geq 1\}$, and with the border conditions

$$G_n(u_1|U) = 0, \quad n \geq 1. \tag{4.20}$$

So, each G_n has the integral representation

$$G_n(x|U) = n! \int_{u_1}^{x} \left[\int_{u_2}^{y_1} dy_2 \ldots \int_{u_n}^{y_{n-1}} dy_n \right] dy_1, \quad n \geq 1. \tag{4.21}$$

The Appell and A-G polynomials are closely related through the identity

$$G_n(x|u_1, \ldots, u_n) = A_n(x|u_n, \ldots, u_1), \quad n \geq 1. \tag{4.22}$$

The two families (i.e. for all $n \geq 0$), however, are distinct and enjoy different properties. From (4.18) and (4.21), it is clear that the polynomials A_n and G_n can be interpreted in terms of the joint distribution of the vector $(U_{1:n}, \ldots, U_{n:n})$.

Proposition 7. *For $0 \leq u_1 \leq \ldots \leq u_n \leq x \leq 1$,*

$$P[U_{(1)} > u_1, \ldots, U_{(n)} > u_n \text{ and } U_{(n)} \leq x] = A_n(x|u_1, \ldots, u_n), \quad n \geq 1. \tag{4.23}$$

For $0 \leq x \leq u_1 \leq \ldots \leq u_n \leq 1$,

$$P[U_{(1)} \leq u_1, \ldots, U_{(n)} \leq u_n \text{ and } U_{(1)} > x] = (-1)^n G_n(x|u_1, \ldots, u_n), \quad n \geq 1. \tag{4.24}$$

The representations (4.23) and (4.24) will play a key role for solving first-hitting problem that involve Poisson processes. Numerically, it will be necessary to evaluate some special values of the polynomials. To this end, it is convenient to use the following recusive relations.

Proposition 8. *The Appell polynomials are computed through the expansion*

$$A_n(x|U) = \sum_{k=0}^{n} \binom{n}{k} A_{n-k}(0|U) x^k, \quad n \geq 1, \tag{4.25}$$

where the $A_n(0|U)$'s are obtained recursively from

$$A_n(0|U) = -\sum_{k=1}^{n} \binom{n}{k} A_{n-k}(0|U) u_n^k, \quad n \geq 1. \tag{4.26}$$

The A-G polynomials are computed through the recursion

$$G_n(x|U) = x^n - \sum_{k=0}^{n-1} \binom{n}{k} u_{k+1}^{n-k} G_k(x|U), \quad n \geq 1. \tag{4.27}$$

Proof. The Maclaurin expansion of $A_n(x|U)$ gives (4.25) as

$$A_n(x|U) = \sum_{k=0}^{n} \frac{A_n^{(k)}(0|U)}{k!} x^k = \sum_{k=0}^{n} \binom{n}{k} A_{n-k}(0|U) x^k.$$

Evaluation at $x = u_n$ then provides (4.26). Regarding (4.27), first note that

$$G_n^{(k)}(u_{k+1}|U) = \begin{cases} 1, & \text{if } k = n, \\ 0, & \text{otherwise.} \end{cases}$$

Any polynomials $R(x)$ of degree n can therefore be written as

$$R(x) = \sum_{k=0}^{n} \frac{R^{(k)}(u_{k+1})}{k!} G_k(x|U).$$

By expanding x^n one gets (4.27). □

Hereafter are a couple of useful properties

Proposition 9. *1. For any $a, b \in \mathbb{R}$, it holds that*

$$A_n(x|a+bU) = b^n A_n((x-a)/b|U), \quad n \geq 1, \tag{4.28}$$

with the same identity for G_n.

2. We have

$$A_n(x|1,\ldots,n) = x^{n-1}(x-n), \tag{4.29}$$

$$G_n(x|0,\ldots,n-1) = x(x-n)^{n-1}. \tag{4.30}$$

3. Let $\{X_n\,,\, n \geq 1\}$ be a sequence of i.i.d. nonnegative random variables, of partial sums $S_n = \sum_{k=1}^{n} X_k$ with $S_0 = 0$. Then, for $n \geq 1$,

$$\mathbb{E}[A_n(x|S_1,\ldots,S_n)|S_n] = x^{n-1}(x-S_n), \tag{4.31}$$

$$\mathbb{E}[G_n(x|S_0,\ldots,S_{n-1})|S_n] = x(x-S_n)^{n-1}. \tag{4.32}$$

Proof. 1. Let us use induction on $n \geq 1$. Take $n = 1$, we have

$$A_1(x|a+bu_1) = \int_{a+bu_1}^{x} A_1'(y|a+bu_1)\mathrm{d}y = x - a - bu_1$$

and

$$A_1(x|u_1) = x - u_1.$$

The property holds for $n = 1$. Assume that it holds true for some n and consider $n+1$. We have

$$\begin{aligned}
A_{n+1}(x|a+bU) &= \int_{a+bu_{n+1}}^{x} A_{n+1}'(y|a+bU)\mathrm{d}y \\
&= \int_{a+bu_{n+1}}^{x} nA_n(y|a+bU)\mathrm{d}y \\
&= b^n \int_{a+bu_{n+1}}^{x} nA_n\left(\frac{y-a}{b}\middle|U\right)\mathrm{d}y \\
&= b^{n+1} n \int_{u_{n+1}}^{\frac{x-a}{b}} A_n\left(z\middle|U\right)\mathrm{d}y \\
&= b^{n+1} \int_{u_{n+1}}^{\frac{x-a}{b}} A_{n+1}'\left(z\middle|U\right)\mathrm{d}y \\
&= b^{n+1} A_{n+1}\left(\frac{x-a}{b}\middle|U\right),
\end{aligned}$$

so the property holds for $n+1$. No need to do the job for the $G_n(.|U)$'s thanks to (4.22).

2. Again induction on $n \geq 1$. Take $n = 1$, we have

$$A_1(x|1) = x - 1$$

Assume that the result holds true for some n and consider $n+1$. We have

$$\begin{aligned} A_{n+1}(x|1,\ldots,n,n+1) &= \int_{n+1}^{x} A'_{n+1}(y|1,2,\ldots,n+1)\mathrm{d}y \\ &= (n+1)\int_{n+1}^{x} A_n(y|1,2,\ldots,n)\mathrm{d}y \\ &= (n+1)\int_{n+1}^{x} y^{n-1}(y-n)\mathrm{d}y \\ &= x^n[x-(n+1)] \end{aligned}$$

For identity (4.30), write

$$\begin{aligned} G_n(x|0,\ldots,n-1) &= G_n(x-n|-n,\ldots,-1) \\ &= (-1)^n G_n(n-x|n,\ldots,1) \\ &= (-1)^n A_n(n-x|1,\ldots,n) \\ &= (-1)^n (n-x)^{n-1}(-x) \\ &= x(x-n)^{n-1}. \end{aligned}$$

3. Again induction on $n \geq 1$. Take $n = 1$, we have

$$\mathbb{E}[A_1(x|S_1)|S_1] = x - S_1$$

Assume that the result holds true for some n and consider $n+1$. We have

$$\begin{aligned} \mathbb{E}\left[A_{n+1}(x|S_1,\ldots,S_n,S_{n+1})|S_{n+1}\right] &= \mathbb{E}\left[\int_{S_{n+1}}^{x} A'_{n+1}(y|S_1,\ldots,S_n,S_{n+1})\mathrm{d}y|S_{n+1}\right] \\ &= (n+1)\mathbb{E}\left[\int_{S_{n+1}}^{x} A_n(y|S_1,\ldots,S_n)\mathrm{d}y|S_{n+1}\right] \\ &= (n+1)\int_{S_{n+1}}^{x} \mathbb{E}\left[A_n(y|S_1,\ldots,S_n)|S_{n+1}\right]\mathrm{d}y \text{ (Fubini)} \\ &= (n+1)\int_{S_{n+1}}^{x} \mathbb{E}\left\{\mathbb{E}\left[A_n(y|S_1,\ldots,S_n)|S_n,S_{n+1}\right]|S_{n+1}\right\}\mathrm{d}y \\ &= (n+1)\int_{S_{n+1}}^{x} \mathbb{E}\left\{\mathbb{E}\left[A_n(y|S_1,\ldots,S_n)|S_n\right]|S_{n+1}\right\}\mathrm{d}y \\ &= (n+1)\int_{S_{n+1}}^{x} \mathbb{E}\left[y^{n-1}(y-S_n)|S_{n+1}\right]\mathrm{d}y \\ &= (n+1)\int_{S_{n+1}}^{x} y^n - \frac{n}{n+1}S_{n+1}y^{n-1}\mathrm{d}y \\ &= x^n(x-S_{n+1}) \end{aligned}$$

For identity (4.32), write

$$\begin{aligned}
\mathbb{E}[G_n(x|S_0,\ldots,S_{n-1})|S_n] &= \mathbb{E}[G_n(x-S_n|S_0-S_n,\ldots,S_{n-1}-S_n)|S_n] \\
&= (-1)^n\mathbb{E}[G_n(S_n-x|S_n,\ldots,S_n-S_{n-1})|S_n] \\
&= (-1)^n\mathbb{E}[A_n(S_n-x|S_n-S_{n-1},\ldots,S_n)|S_n] \\
&= (-1)^n(S_n-x-S_n)(S_n-x)^{n-1} \\
&= x(x-S_n)^{n-1}.
\end{aligned}$$

□

Chapter 5

Decentralization of blockchain system

Decentralization represents the fairness of the distribution of the accounting right of the nodes in the blockchain network. The consensus protocol must be designed so that the decision power does not eventually concentrate on a few nodes leading to a centralized system. In leader based consensus protocols, each peer is associated to a probability of being chosen. Measuring decentrality then reduces to computing the entropy of the probability distribution of the random variable equal to the peer selected

5.1 Decentralization in PoS

The *Proof-of-Stake* protocol is a leader based consensus protocol that appoints a block validator depending on how many cryptocoins he owned which corresponds to its stake. In its most basic form a coins is drawn at random, the owner of that coin appends a block and collect a reward. The stake of each peers is governed by stochastic processes with reinforcement similar to that studied in the Polya's urn problem. In Polya's urn, there are balls of various colors. At each time step a ball is drawn, the ball is then replaced in the urn together with a ball of the same color. The coins are the balls and the color is the peer that owns the balls. This analogy has been used to study the decentralization

Let the network be of size p and denote by r the reward collected at each round $n \in \mathbb{N}$ by the lucky node $x \in \{1,\dots,p\} = E$. At time $n = 0$, each peer $x \in E$ has $Z_0^{(x)}$ coins so that the total number of coins is $Z_0 = \sum_{x\in E} Z_0^{(x)}$. The number of coins owned by each peers evolve over time as

$$Z_n^{(x)} = Z_0^{(x)} + r\sum_{k=1}^{n} \mathbb{I}_{A_k^{(x)}} \text{ and } Z_n = \sum_{x\in E} Z_n^{(x)} = Z_0 + nr,$$

where $A_n^{(x)}$ is the event that a coin own by peer $x \in E$ is drawn at time $n \in \mathbb{N}$. Let $(W_n^{(x)})_{n\geq 0}$ be

the proportion of coins owned by peer x at time n, given by

$$W_n^{(x)} = \frac{Z_n^{(x)}}{Z_n}.$$

Let $\mathcal{F}_n = \sigma(\{Z_k^{(x)}\ , x \in E, k \leq n\})$. Note that

$$\mathbb{P}\left(A_n^{(x)} | \mathcal{F}_{n-1}\right) = W_{n-1}^{(x)}.$$

5.2 Average stake owned by each peer

The following result provide the average behaviour of the share of coins owned by each peer.

Proposition 10.

$$\mathbb{E}\left(W_n^{(x)}\right) = \frac{Z_0^{(x)}}{Z_0},\ x \in E\ , n \geq 0.$$

Proof. We show that $(W_n^{(x)})_{n\geq 0}$ is a martingale. We have that

$$\begin{aligned}
\mathbb{E}\left[W_n^{(x)} | \mathcal{F}_{n-1}\right] &= \mathbb{E}\left[\frac{Z_{n-1}^{(x)} + r\mathbb{I}_{A_n^{(x)}}}{Z_0 + rn}\Big| \mathcal{F}_{n-1}\right] \\
&= \frac{Z_{n-1}^{(x)}}{Z_0 + rn} + \frac{rW_{n-1}^{x}}{Z_0 + rn} \\
&= \frac{W_{n-1}^{(x)}[Z_0 + r(n-1)]}{Z_0 + rn} + \frac{rW_{n-1}^{x}}{Z_0 + rn} \\
&= W_{n-1}^{x}.
\end{aligned}$$

It then follows that

$$\mathbb{E}\left(W_n^{(x)}\right) = \frac{Z_0^{(x)}}{Z_0},\ x \in E\ , n \geq 0.$$

□

The long term average of the stake of each peer is stable, we focus on their asymptotic distribution in the following section.

5.3 Asymptotic distribution of the stakes

To go beyond the mean and study the distribution of the stake of the peers, we have to consider the case $r = 1$. We can then show that the joint distribution of $(W_\infty^{(1)}, \ldots, W_\infty^{(p)})$ is the Dirichlet one.

Definition 12. *A random vector $(W_1, \ldots, W_p)$ has a Dirichlet distribution $Dir(\alpha_1, \ldots, \alpha_p)$ if it has a joint p.d.f. given by*

$$f(w_1, \ldots, w_p; \alpha_1, \ldots, \alpha_p) = \frac{1}{B(\alpha)} \prod_{i=1}^{p} w_i^{\alpha_i - 1}, \tag{5.1}$$

for $\alpha_1,\ldots,\alpha_p > 0$, $0 < w_1,\ldots,w_p < 1$ *and* $\sum_{i=1}^p w_i = 1$, *where*

$$B(\alpha) = \frac{\prod_{i=1}^p \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^p \alpha_i)},$$

and $\Gamma(\alpha) = \int_0^\infty \mathrm{e}^{-x} x^{\alpha-1} dx$ *is the gamma function.*

A Dirichlet random vector can be generated by independent Gamma random variables. Recall that $X \sim \mathsf{Gamma}(\alpha,\beta)$ if X has p.d.f.

$$f_X(x) = \begin{cases} \frac{e^{-\beta x} x^{\alpha-1}\beta^\alpha}{\Gamma(\alpha_i)}, & x > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

Proposition 11. *Let* $X_i \sim \mathsf{Gamma}(\alpha_i, 1)$ *for* $i = 1,\ldots,p$ *be independent random variables then*

$$\left(\frac{X_1}{\sum_{i=1}^p X_i},\ldots,\frac{X_p}{\sum_{i=1}^p X_i}\right) \sim Dir(\alpha_1,\ldots,\alpha_p)$$

Proof. Note that because $(w_1,\ldots,w_p)$ belongs to the $p-1$ simplex then the p.d.f. (5.1) may be rewritten as

$$f(w_1,\ldots,w_p;\alpha_1,\ldots,\alpha_p) = \frac{1}{B(\alpha)} \prod_{i=1}^{p-1} w_i^{\alpha_i-1} \left(1 - \sum_{i=1}^{p-1} w_i\right)^{\alpha_p-1},$$

which means that we are only interested in the distribution of the vector $\left(W_1,\ldots,W_{p-1}\right) = \left(X_1/\sum_{i=1}^p X_i,\ldots,X_{p-1}/\sum_{i=1}^p X_i\right)$. Let $g : \mathbb{R}^p \mapsto \mathbb{R}^+$ be measurable and bounded and consider

$$\begin{aligned} &\mathbb{E}\left[g\left(\frac{X_1}{\sum_{i=1}^p X_i},\ldots,\frac{X_{p-1}}{\sum_{i=1}^p X_i}\right)\right] \\ = &\int_{\mathbb{R}_+^l} g\left(\frac{x_1}{\sum_{i=1}^p x_i},\ldots,\frac{x_{p-1}}{\sum_{i=1}^p x_i}\right) \frac{e^{-\sum_i^p x_i} \prod_{i=1}^p x_i^{\alpha_i-1}}{\prod_{i=1}^p \Gamma(\alpha_i)} \mathrm{d}\lambda(x_1,\ldots,x_p) \end{aligned}$$

We use the change of variable

$$\Phi : (w_1,\ldots,w_{p-1},v) \mapsto \left[vw_1,\ldots,vw_{p-1},v\left(1-\sum_{i=1}^{p-1} w_i\right)\right] = \left(x_1,\ldots,x_{p-1},\sum_{i=1}^p x_i\right)$$

minding the change in the integration domain as

$$\Phi(\Delta_{p-1} \times \mathbb{R}_+) = \mathbb{R}_+^p,$$

Δ_{p-1} is the $p-1$ simplex and the Jacobian $\left|\frac{\mathrm{d}\Phi}{\mathrm{d}(w_1,\ldots,w_{p-1},v)}\right| = v^{p-1}$, we get

$$\begin{aligned} &\mathbb{E}\left[g\left(\frac{X_1}{\sum_{i=1}^p X_i},\ldots,\frac{X_{p-1}}{\sum_{i=1}^p X_i}\right)\right] \\ = &\int_{\Delta_{p-1}} \int_{\mathbb{R}_+} g\left(w_1,\ldots,w_{p-1}\right) \frac{e^{-v} \prod_{i=1}^{p-1} w_i^{\alpha_i-1} \left(1-\sum_{i=1}^{p-1} w_i\right)^{\alpha_p-1} v^{\sum_{i=1}^{p-1}\alpha_i-1}}{\prod_{i=1}^p \Gamma(\alpha_i)} \mathrm{d}\lambda(w_1,\ldots,w_{p-1},v) \\ = &\int_{\Delta_{p-1}} g\left(w_1,\ldots,w_{p-1}\right) \frac{\Gamma\left(\sum_{i=1}^p \alpha_i\right)}{\prod_{i=1}^p \Gamma(\alpha_i)} \prod_{i=1}^{p-1} w_i^{\alpha_i-1} \left(1-\sum_{i=1}^{p-1} w_i\right)^{\alpha_p-1} \mathrm{d}\lambda(w_1,\ldots,w_{p-1}). \end{aligned}$$

□

To show that the stochastic process $(W_n^{(1)}, \ldots, W_n^{(p)})$ has a Dirichlet limiting distribution we need to introduce a counting process known as Yule process.

Definition 13. *A Yule process $(Y_t)_{t\geq 0}$ is a pure birth process with linear birth rate given as*

$$\mathbb{P}(Y_{t+h} = y + 1 | Y_t = y) = yh + o(h).$$

The Yule process models the population of some particle over time, assuming that there is one particle at time 0, so $Y_0 = 1$ this particle will split in two after some exponential time and this going on and on, see the illustration on Figure 5.1.

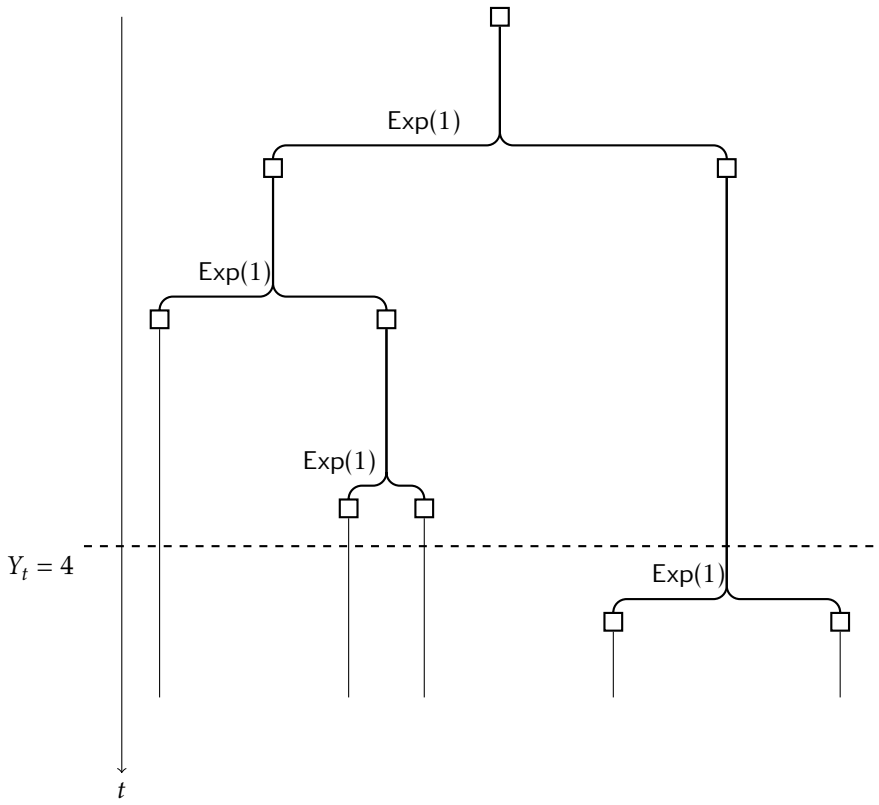


Figure 5.1: Yule tree

Before moving forward, two remarks.

Remark 5. *If we have $Y_0 = y_0$ particles at the initial state, then it is like starting y_0 independent copies of the Yule process with one particle and summing up at time t the number of particles of all the Yule processes. Namely, let $(Y_t)_{t\geq 0}$ be a Yule process such that $Y_0 = y_0$, then*

$$Y_t = \sum_{i=1}^{y_0} Y_t^{(i)},$$

where the $Y_t^{(i)}$'s are independent Yule processes such that $Y_t^{(i)} = 1$ for $i = 1, \ldots, y_0$.

Remark 6. *The Yule process $(Y_t)_{t\geq 0}$ is strong Markov in the sense that for any stopping time τ, the stopped process*

$$\tilde{Y}_t = Y_{\tau+t},\ t \geq 0$$

is again a Yule process such that $\tilde{Y}_0 = Y_\tau$.

Proposition 12. *Let* $(Y_t)_{t\geq 0}$ *be a Yule process such that* $Y_0 = 1$ *then*

$$\mathbb{P}(Y_t = y) = \left(1 - \mathrm{e}^{-t}\right)^{y-1} e^{-t}.$$

Proof. The inter-arrival times $(\Delta_n^T)_{n\geq 1}$ of the Yule process are independent random variable such that $\Delta_n^T = \mathsf{Exp}(n)$. If we have n particles at some time $t \geq 0$, that's n exponential $\mathsf{Exp}(1)$ competing and a new particle appears as soon as one of them ring. We then have $\Delta_n^T = \min(X_1, \ldots, X_n)$, where $X_1, \ldots, X_n \overset{\text{i.i.d.}}{\sim} \mathsf{Exp}(1)$ and so $\Delta_n^T \sim \mathsf{Exp}(n)$. The arrival time of the n^{th} particles is given by

$$T_n = \sum_{k=1}^{n-1} \Delta_k^T, \ n \geq 2.$$

By induction on $n \geq 2$, we can show that

$$\mathbb{P}(T_n \leq t) = \left(1 - e^{-t}\right)^{n-1}.$$

We further deduce that

$$\mathbb{P}(Y_t = y) = \mathbb{P}(Y_t > y) - \mathbb{P}(Y_t > y+1) = \mathbb{P}(T_y \leq t) - \mathbb{P}(T_{y+1} \leq t) = \left(1 - \mathrm{e}^{-t}\right)^{y-1} e^{-t}$$

□

Theorem 9. *We have that*

$$e^{-t} Y_t \xrightarrow{\mathcal{D}} \mathsf{Exp}(1), \text{ as } t \to \infty.$$

Proof. Let us show that $\left(e^{-t} Y_t\right)_{t\geq 0}$ is a martingale. We have that, for $s \leq t$,

$$\mathbb{E}(e^{-t} Y_t | \mathcal{F}_s) = e^{-t} \mathbb{E}(Y_t | \mathcal{F}_s) = e^{-t} Y_s e^{t-s} = e^{-s} Y_s.$$

Because of the martingale convergence theorem, we know that $\left(e^{-t} Y_t\right)_{t\geq 0}$ has a limiting distribution. Consider the Laplace transform

$$\mathbb{E}\left(\mathrm{e}^{-\theta e^{-t} Y_t}\right) = \frac{\mathrm{e}^{-\theta \mathrm{e}^{-t}} \mathrm{e}^{-t}}{1 - \mathrm{e}^{-\theta \mathrm{e}^{-t}}(1 - \mathrm{e}^{-t})} = \frac{1}{e^t\left(\mathrm{e}^{\theta \mathrm{e}^{-1}} - 1\right) + 1} \to \frac{1}{1+\theta}, \text{ as } t \to \infty.$$

which coincides with that of an exponential random variable $\mathsf{Exp}(1)$.

□

We finally link the asymptotic behavior of the Yule processes to our initial question about the asymptotic distributions of the stakes.

Theorem 10. *We have that*

$$\left(W_\infty^{(1)}, \ldots, W_\infty^{(p)}\right) \sim Dir\left(Z_0^{(1)}, \ldots, Z_0^{(p)}\right).$$

Proof. Assume that each coin owned at time $n = 0$ is like the initial particle of a Yule process. That's Z_0 Yule processes $(Y_t^{i,j})_{t\geq 0}$ for $i = 1, \ldots, Z_0^{(j)}$ and $j = 1, \ldots, p$. Each time step n corresponds

to the jump of one of the Yule processes τ_n. The number of coins owned by peer $j = 1,\dots,p$ is then

$$Z_n^{(j)} = \sum_{i=1}^{Z_0^{(j)}} Y_{\tau_n}^{i,j},$$

we have $\tau_n \to \infty$ as $n \to \infty$ and therefore

$$e^{-\tau_n} Z_n^{(j)} = \sum_{i=1}^{Z_0^{(j)}} Y_{\tau_n}^{i,j} e^{-\tau_n} \xrightarrow{\mathcal{D}} \mathsf{Gamma}\left(Z_0^{(j)}, 1\right), \text{ as } n \to \infty$$

Finally

$$\left(W_n^{(1)},\dots,W_n^{(p)}\right) = \left(\frac{e^{-\tau_n} Z_n^{(1)}}{\sum_{j=1}^p e^{-\tau_n} Z_n^{(j)}},\dots,\frac{e^{-\tau_n} Z_n^{(p)}}{\sum_{j=1}^p e^{-\tau_n} Z_n^{(j)}}\right) \xrightarrow{\mathcal{D}} \mathrm{Dir}(Z_0^{(1)},\dots,Z_0^{(p)}), \text{ as } n \to \infty.$$

□

Remark 7. *One can take a shorter road to show the above result. Let $(X_n)_{n\geq 1}$ be the color of the ball drawn during the n^{th} round. We have that*

$$\mathbb{P}(X_1 = x) = \frac{Z_0^{(x)}}{Z_0} \tag{5.3}$$

and

$$\mathbb{P}(X_{n+1} = x) = \frac{Z_0^{(x)} + \sum_{i=1}^n \delta_{X_i}(x)}{Z_0 + n} = \frac{Z_0^{(x)} + \lambda_n(x)}{Z_0 + n} = m_n(x) \tag{5.4}$$

where δ_{X_i} denotes the Dirac measure at X_i. A sequence that satisfies (5.3) and (5.4) is said to be a Polya sequence with parameter N_x, $x \in E$.

Lemma 3. *The following statements are equivalent:*

(i) $X_1, X_2, \dots$, is a Polya sequence

(ii) $\mu^* \sim Dir(N_x, x \in E)$ and $X_1, X_2, \dots$ given μ^* are i.i.d. as μ^*

Consider the event $A_n = \{X_1 = x_1,\dots,X_n = x_n\}$. Induction on n allows us to show that (i) is equivalent to

$$\mathbb{P}(A_n) = \frac{\prod_{x\in E}\left(Z_0^{(x)}\right)^{[\lambda_n(x)]}}{Z_0^{[n]}}, \tag{5.5}$$

where $\lambda_n(x)$ is the number of i's in $1,\dots,n$ for which $x_i = x$ and $a^{[k]} = a(a+1)\dots(a+k-1)$. Now assume that (ii) holds true, then

$$\mathbb{P}(A_n|\mu^*) = \prod_{x\in E} \mu^*(x)^{\lambda_n(x)},$$

recall that μ^* is a random vector, indexed on E, We denote by $\mu^*(x)$ the component associated with $x \in E$. The law of total probability then yields

$$\mathbb{P}(A_n) = \mathbb{E}\left[\prod_{x\in E} \mu^*(x)^{\lambda_n(x)}\right], \tag{5.6}$$

which is the same as (5.5). Applying the lemma together with the law of large number yields

$$n^{-1}\sum_{i=1}^{n}\delta_{X_i}(x)\to\mu^*(x)\text{ as }n\to\infty.$$

and then $m_n(x)\to\mu^*(x)$. This proof is taken from Blackwell and MacQueen [1973].

The asymptotic distribution of the stakes among the peers is a Dirichlet random vector denoted by μ^* which may be considered as a probability distribution over the set of peers. Decentralization is achieved when the weights do not concentrate around a few nodes. The most desirable situation corresponds to all the peers being equally likely to be selected. It would corresponds to a uniform distribution over the set of peers which would maximizes the Shannon entropy. For $\mu^*\sim\text{Dir}(Z_0^{(x)})$, we have

$$H(\mu^*)=-\mathbb{E}\left\{\sum_x\mu^*(x)\ln[\mu^*(x)]\right\}=-\sum_x\frac{Z_0}{Z_0^{(x)}}\left[\psi(Z_0^{(x)}+1)-\psi(Z_0+1)\right],$$

where $\psi(x)=\frac{\mathrm{d}}{\mathrm{d}x}\ln[\Gamma(x)]$ is the digamma function, to be compared to $\ln(p)$.

Chapter 6

Efficiency of blockchain systems

6.1 A queueing model with bulk service

Blockchain users send transactions to the network of validators according to some rate λ. These transactions enter a queue of pending transactions. The validators select a subset of b transactions to be recorded in the next block. The block is built by a leader elected via a consensus protocol. The block is then communicated to the other validators and the b transactions exit the queue. We assume that building a block takes some exponentially distributed time with mean μ. What we just described is exactly a single server with bulk service queueing system, described for instance in Bailey [1954] and Chaudhry and Templeton [1981] with exponential arrival times, that processes k items at a time, with an exponential service time. This a $M/M^b/1$ queue in Kendall's notation summarized in Figure 6.1. One specificity of this queue is that the

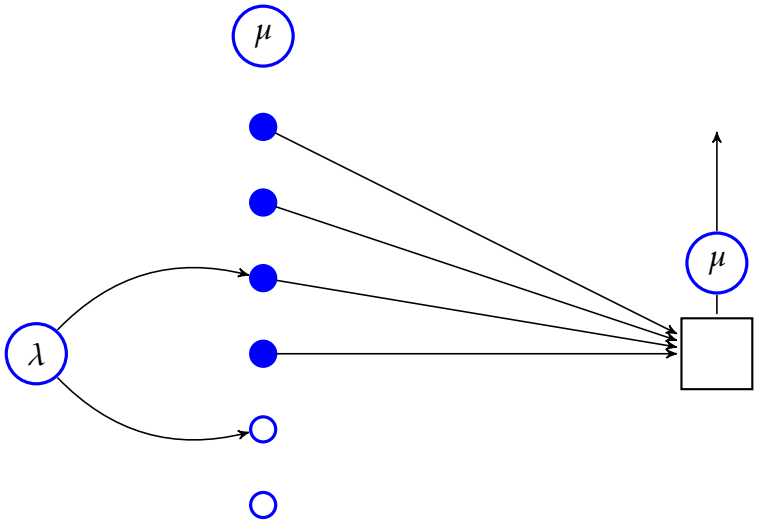


Figure 6.1: Blockchain queue

server is always busy. Our goal is to assess the efficiency which is characterized by

- Throughputs: Number of transaction being processed per time unit

- Latency: Average transaction confirmation time

This can be done by studying the distribution of the number of pending transaction in the queue over the long run. A stationary state can only be reached if

$$\mu \cdot b > \lambda. \tag{6.1}$$

Denote by N^q the length of the queue upon stationarity, The following result holds.

Theorem 11. *Assume that (6.1) holds then N^q is geometrically distributed*

$$\mathbb{P}(N^q = n) = (1-p)\cdot p^n,\ n \geq 0$$

where $p = 1/z^*$ and z^* is the only root of

$$-\frac{\lambda}{\mu} z^{b+1} + z^b \left(\frac{\lambda}{\mu} + 1\right) - 1,$$

such that $|z^*| > 1$.

Proof. Let N_t^q be the number of transactions in the queue at time $t \geq 0$ and X_t the time elapsed since the last block was found. Further define

$$P_n(x,t)\mathrm{d}x = \mathbb{P}[N_t^q = n, X_t \in (x, x+\mathrm{d}x)]$$

If $\lambda < \mu \cdot b$ holds then the process admits a limiting distribution given by

$$\lim_{t\to\infty} P_n(x,t) = P_n(x).$$

Adding the variable X_t is a known trick going back to Cox [1955], it allows us to make the process $(N_t^q)_{t\geq 0}$ Markovian (useful if transaction arrival process or the block arrival process are not Poisson processes) but also to study the process as time goes to infinity while keeping a temporal marker (last arrival time). We aim at finding the distribution of the queue length upon stationarity

$$\mathbb{P}(N^q = n) := \alpha_n = \int_0^\infty P_n(x)\mathrm{d}x,\ n \geq 0. \tag{6.2}$$

Consider the possible transitions over a small time lapse h during which no block is being generated. Over this time interval, either

- no transactions arrives
- one transaction arrives

We have for $n \geq 1$

$$P_n(x+h) = e^{-\mu h}\left[e^{-\lambda h} P_n(x) + \lambda h e^{-\lambda h} P_{n-1}(x)\right].$$

Differentiating with respect to h and letting $h \to 0$ leads to

$$P_n'(x) = -(\lambda+\mu)P_n(x) + \lambda P_{n-1}(x),\ n \geq 1. \tag{6.3}$$

Similarly for $n = 0$, we have

$$P_0'(x) = -(\lambda + \mu)P_0(x). \tag{6.4}$$

We denote by

$$\xi(x)\mathrm{d}x = \mathbb{P}(x \leq X < x + \mathrm{d}x | X \geq x) = \mu \mathrm{d}x,$$

the hazard function of the block arrival time (constant as it is exponentially distributed). The system of differential equations (6.3), (6.4) admits boundary conditions at $x = 0$ with

$$\begin{cases} P_n(0) = \int_0^{+\infty} P_{n+b}(x)\xi(x)\mathrm{d}x = \mu\alpha_{n+b}, & n \geq 1, \\ P_0(0) = \mu \sum_{n=0}^{b} \alpha_n, & n = 0, \ldots, b \end{cases} \tag{6.5}$$

Define the probability generating function of N^q at some elapsed service time $x \geq 0$ as

$$G(z;x) = \sum_{n=0}^{\infty} P_n(x) z^n.$$

By differentiating with respect to x, we get (using (6.3) and (6.4))

$$\frac{\partial}{\partial x} G(z;x) = -\left[\lambda(1-z) + \mu\right] G(z;x),$$

and therefore

$$G(z;x) = G(z;0) \exp\left\{-\left[\lambda(1-z) + \mu\right] x\right\}.$$

We get the probability generating function of N^q by integrating over x as

$$G(z) = \frac{G(z;0)}{\lambda(1-z) + \mu}. \tag{6.6}$$

Using the boundary conditions (6.5), we write

$$\begin{aligned} G(z;0) &= \sum_{n=0}^{\infty} P_n(0) z^n \\ &= P_0(0) + \sum_{n=1}^{+\infty} P_n(0) z^n \\ &= \mu \sum_{n=0}^{b} \alpha_n + \mu \sum_{n=1}^{+\infty} \alpha_{n+b} z^n \\ &= \mu \sum_{n=0}^{b} \alpha_n + \mu z^{-b} \left[G(z) - \sum_{n=0}^{b} \alpha_n z^n \right] \end{aligned} \tag{6.7}$$

Replacing the left hand side of (6.7) by (6.6), multiplying on both side by z^b and rearranging yields

$$\frac{G(z)}{M(z)} [z^b - M(z)] = \sum_{n=0}^{b-1} \alpha_n (z^b - z^n), \tag{6.8}$$

where $M(z) = \mu/(\lambda(1-z) + \mu)$. Using Rouche's theorem, we find that both side of the equation shares b zeros inside the circle $\mathcal{C} = \{z \in \mathbb{C} \,;\, |z| < 1 + \epsilon\}$ for some epsilon.

Lemma 4. *Let $\mathcal{C} \subset \mathbb{C}$ and f and g two holomorphic functions on $\mathcal{C}$. Let $\partial\mathcal{C}$ be the contour of $\mathcal{C}$. If*

$$|f(z) - g(z)| < |g(z)|, \forall z \in \partial\mathcal{C}$$

then $Z_f - P_f = Z_g - P_g$, where Z_f, P_f, Z_g, and P_g are the number of zeros and poles of f and g respectively.

We have $\partial\mathcal{C} = \{z \in \mathbb{C}\,;\, |z| = 1+\epsilon\}$. The left hand side can be rewritten as

$$G(z)\left[-\frac{\lambda}{\mu}z^{b+1} + \left(1+\frac{\lambda}{\mu}\right)z^b - 1\right].$$

Define $f(z) = -\frac{\lambda}{\mu}z^{b+1} + \left(1+\frac{\lambda}{\mu}\right)z^b - 1$ and $g(z) = \left(1+\frac{\lambda}{\mu}\right)z^b$. We have

$$|f(z)-g(z)| = |-\frac{\lambda}{\mu}z^{b+1} - 1| \leq \frac{\lambda}{\mu}(1+\epsilon)^{b+1} + 1 < \left(1+\frac{\lambda}{\mu}\right)(1+\epsilon)^b = |g(z)|, \text{ with } \epsilon \to 0.$$

Regarding the right hand side, define $f(z) = \sum_{n=0}^{b-1}\alpha_n(z^b - z^n)$ and $g(z) = \sum_{n=0}^{b-1}\alpha_n z^b$. We have

$$|f(z)-g(z)| < |\sum_{n=0}^{b-1}\alpha_n z^n| \leq \sum_{n=0}^{b-1}\alpha_n(1+\epsilon)^n < (1+\epsilon)^b\sum_{n=0}^{b-1}\alpha_n = |g(z)|.$$

We deduce from Rouche's theorem that both sides have b share roots inside $\mathcal{C}$. Note that one of them is 1, and we denote by z_k, $k = 1,\ldots,b-1$ the remaining $b-1$ roots. Given the polynomial form of the right hand side of (6.8), the fundamental theorem of algebra indicates that the number of zero is b. Given the left hand side

$$G(z)\left[-\frac{\lambda}{\mu}z^{b+1} + \left(1+\frac{\lambda}{\mu}\right)z^b - 1\right].$$

we deduce that there is one zeros outside $\mathcal{C}$, we can further show that it is a real number z^*. Multiplying both side of (6.8) by $(z-1)\prod_{k=1}^{b-1}(z-z_k)$, and using $G(1) = 1$ yields

$$G(z) = \frac{1-z^*}{z-z^*}.$$

N^q is then a geometric random variable with parameter $p = \frac{1}{z^*}$. □

The result above can be found in Bailey [1954]. The application to blockchain under more general assumptions over the block discovery time is given in Kawase and Kasahara [2017].

6.2 Latency and throughputs computation

The practical computation of latency and throughputs then follow from a standard result in queueing, known as Little's law, see Little [1961].

Theorem 12. *Consider a stationary queueing system and denote by*

- $1/\lambda$ *the mean of the unit inter-arrival times*
- L *be the mean number of units in the system*
- W *be the mean time spent by units in the system*

We have

$$L = \lambda \cdot W$$

- Latency is the confirmation time of a transaction

$$\text{Latency} = W = \frac{\mathbb{E}(N^q)}{\lambda} = \frac{p}{(1-p)\lambda}.$$

- Throughput is the number of transaction confirmed per time unit

$$\text{Throughput} = \mu \mathbb{E}(N^q \mathbb{I}_{N^q \leq b} + b \mathbb{I}_{N^q > b}) = \mu \sum_{n=0}^{b} n(1-p)p^n + bp^{b+1}.$$

Avenue for future research includes

- the inclusion of priority consideration, accounting for the transaction fee, see Kawase et al. [2020]
- Refine the hypothesis of the queueing system to better adapt to the different consensus protocol, see Li et al. [2018] and Li et al. [2019].

Bibliography

Jean-Philippe Abegg, Quentin Bramas, and Thomas Noël. Blockchain using proof-of-interaction. In *Networked Systems*, pages 129–143. Springer International Publishing, 2021. doi: 10.1007/978-3-030-91014-3_9.

Saif Al-Kuwari, James H. Davenport, and Russell J. Bradford. Cryptographic hash functions: Recent design trends and security notions. Cryptology ePrint Archive, Report 2011/565, 2011. https://ia.cr/2011/565.

Norman T. J. Bailey. On queueing processes with bulk service. *Journal of the Royal Statistical Society: Series B (Methodological)*, 16(1):80–87, jan 1954. doi: 10.1111/j.2517-6161.1954.tb00149.x.

David Blackwell and James B. MacQueen. Ferguson distributions via polya urn schemes. *The Annals of Statistics*, 1(2), mar 1973. doi: 10.1214/aos/1176342372.

R. Bowden, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor. Modeling and analysis of block arrival times in the bitcoin blockchain. *Stochastic Models*, 36(4):602–637, July 2020. ISSN 1532-4214. doi: 10.1080/15326349.2020.1786404.

Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, OSDI '99, page 173–186, USA, 1999. USENIX Association. ISBN 1880446391. URL https://dl.acm.org/doi/10.5555/296806.296824.

M.L. Chaudhry and J.G.C. Templeton. The queuing system m/GB/l and its ramifications. *European Journal of Operational Research*, 6(1):56–60, jan 1981. doi: 10.1016/0377-2217(81)90328-3.

D. R. Cox. The analysis of non-markovian stochastic processes by the inclusion of supplementary variables. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):433–441, jul 1955. doi: 10.1017/s0305004100030437.

Pierre-O. Goffard. Fraud risk assessment within blockchain transactions. *Advances in Applied Probability*, 51(2):443–467, jun 2019. doi: 10.1017/apr.2019.18. https://hal.archives-ouvertes.fr/hal-01716687v2.

Yoshiaki Kawase and Shoji Kasahara. Transaction-confirmation time for bitcoin: A queueing analytical approach to blockchain mechanism. In *Queueing Theory and Network Applications*, pages 75–88. Springer International Publishing, 2017. doi: 10.1007/978-3-319-68520-5_5.

Yoshiaki Kawase, , and Shoji Kasahara. Priority queueing analysis of transaction-confirmation time for bitcoin. *Journal of Industrial & Management Optimization*, 16(3):1077–1098, 2020. doi: 10.3934/jimo.2018193.

Sunny King and Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *whitepaper*, 2012.

Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, pages 382–401, July 1982. URL https://www.microsoft.com/en-us/research/publication/byzantine-generals-problem/.

Jan Lansky. Possible state approaches to cryptocurrencies. *Journal of Systems Integration*, 9(1): 19–31, jan 2018. doi: 10.20470/jsi.v9i1.335.

Quan-Lin Li, Jing-Yu Ma, and Yan-Xia Chang. *Blockchain Queue Theory*, pages 25–40. Springer International Publishing, 2018. ISBN 9783030046484. doi: 10.1007/978-3-030-04648-4_3.

Quan-Lin Li, Jing-Yu Ma, Yan-Xia Chang, Fan-Qi Ma, and Hai-Bo Yu. Markov processes in blockchain systems. *Computational Social Networks*, 6(1), jul 2019. doi: 10.1186/s40649-019-0066-1.

Alexander Lipton and Adrien Treccani. *Blockchain and Distributed Ledgers*. WORLD SCIENTIFIC, apr 2021. doi: 10.1142/11857.

John D. C. Little. A proof for the queuing formula:L= λW. *Operations Research*, 9(3):383–387, jun 1961. doi: 10.1287/opre.9.3.383.

Amani Moin, Kevin Sekniqi, and Emin Gun Sirer. *SoK: A Classification Framework for Stablecoin Designs*, pages 174–197. Springer International Publishing, 2020. ISBN 9783030512804. doi: 10.1007/978-3-030-51280-4_11.

S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Available at https://bitcoin.org/bitcoin.pdf, 2008. URL https://bitcoin.org/bitcoin.pdf.

Andreas Park. The conceptual flaws of decentralized automated market making. *Management Science*, 69(11):6731–6751, November 2023. ISSN 1526-5501. doi: 10.1287/mnsc.2021.02802.

Marc Renault. Four proofs of the ballot theorem. *Mathematics Magazine*, 80(5):345–352, dec 2007. doi: 10.1080/0025570x.2007.11953509.

Meni Rosenfeld. Analysis of hashrate-based double spending. *arXiv preprint arXiv:1402.2009*, 2014.

Fahad Saleh. Blockchain without waste: Proof-of-stake. *The Review of Financial Studies*, 34(3): 1156–1190, jul 2020. doi: 10.1093/rfs/hhaa075.

Lajos Takács. A generalization of the ballot problem and its application in the theory of queues. *Journal of the American Statistical Association*, 57(298):327–337, jun 1962. doi: 10.1080/01621459.1962.10480662.

Remco van der Hofstad and Michael Keane. An elementary proof of the hitting time theorem. *The American Mathematical Monthly*, 115(8):753–756, oct 2008. doi: 10.1080/00029890.2008.11920588.

Sam M. Werner, Daniel Perez, Lewis Gudgeon, Ariah Klages-Mundt, Dominik Harz, and William J. Knottenbelt. Sok: Decentralized finance (defi), 2021.

Jiahua Xu, Krzysztof Paruch, Simon Cousaert, and Yebo Feng. Sok: Decentralized exchanges (dex) with automated market maker (amm) protocols. *ACM Computing Surveys*, 55(11):1–50, February 2023. ISSN 1557-7341. doi: 10.1145/3570639.