Julien Riposo

# Some Fundamentals of Mathematics of Blockchain

Springer

# Some Fundamentals of Mathematics of Blockchain

Julien Riposo

# Some Fundamentals of Mathematics of Blockchain

Springer

Julien Riposo
London Stock Exchange Group
Paris, Île-de-France, France

ISBN 978-3-031-31322-6 ISBN 978-3-031-31323-3 (eBook)
https://doi.org/10.1007/978-3-031-31323-3



This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

*To the three women who supported me, while I'm now supporting my dearest son. . .*

# Preface

This book is a possible foundation for the mathematics of blockchain. More specifically, it is a general analysis and synthesis of the current knowledge around the mathematics of blockchain technology. The goal of this study is much more than a review: gathering the main content into one general theoretical framework on blockchain, so that any mathematization of it could start from the content of this book. Specifically, after motivating mathematical aspects, we will set the basis of a blockchain: blocks, transaction contents, block header, nonce, and hashing link. We will connect this with a Graph and Matrix Theories approach, in order to model the peer-to-peer relationship through the Bitcoin Network. We will present a generalization of the used Merkle trees, expose digit patterns from the halving process, and present the Entropy concepts applied to peer-to-peer networks and blockchain. Finally, we will make a connection to some selected research papers.

# Contents

# Chapter 1
# Motivation for Mathematics of Blockchain


First of all, what is a blockchain? It literally is a chain of distinct and consecutive blocks, a technology of data storage, and information transmission without a third party which controls the entire process. It is a distributed database, such that information is sent by users, and internal contents are reviewed on a regular basis under the form of ledge (public for the Bitcoin or Ethereum), thus enhancing a chronological chain (a *time* chain). The whole set is cryptographically secured.

Traditionally, a blockchain is sharing the data it contains, which is the reason why it usually is referred to as a "public ledge." Contrary to the common data-storing technology, blockchain builders have their own copy of the content: the mechanism can assure a unanimous agreement of the correct content of the blockchain, which insures the conformity of the data copies. Thus, any kind of cheat (as double-spending in the context of Bitcoin) is avoided. This allows to a large amount of people, or entities, to agree on a consensus, thus made as incarnating immutable truth.

To properly understand how important the concept of blockchain is, we propose five applications of the blockchain technology below (except Bitcoin which we will see in detail later).

*Immediate insurance*. You are expected to take a plane, but the problem is that it is more than 2 hours delayed. A customer sees an alert on her smartphone, informing she's just earned $100 due to the delay insurance. Such covering is a current way for a refund, but activating it could sometimes be quite complicated (a lot of documents need to be filled, generally before the trip). In our case, the money transfer has automatically been performed, thanks to a blockchain-based application proposed by Axa in 2017 (named Fizzy.Axa). It relied on "smart contracts" (related to the Ethereum blockchain), a computer program which recorded the departure and arrival times. This technology could actually be extended to other insurance products, for instance, to pay farmers who have their harvest destroyed and by means of sensors.

*Tokens*. About 5 years ago, hundreds of companies emitted *security tokens* instead of selling classical equities to investors. Such tokens are a kind of digitalization of equities, and the issue has been performed through *STO* (Security Token



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_1

Offerings). As an example, the European platform ST8 is accompanying start-ups which want to use this process. Contrary to *ICO* (Initial Coin Offerings), which don't give the information of the company's capital but guarantee the access of a future service or to make a plus-value if the token's price increases, STO propose digitized assets which represent a fraction of an underlying financial instrument (e.g., equity, debt, mortgage, ...). For the issuer, the process is less expensive, whereas for the buyer, the execution is instantaneous. Moreover, transactions recorded on the blockchain are unchanged.

*Traced Food.* Since 2019, the French hypermarket Carrefour has allowed its customers to know some products' origins: producer name, transformation place, packing date, etc. The process is quite easy to handle, only by scanning the QR code on the packet with a smartphone. Since then, many other hypermarkets have followed and offer the same service to their customers. The blockchain is used to store the information. Thus, the technology allows to answer the customers' needs to know where all the products come from. This sets confidence in an environment where distrust towards food-processing industry has been omnipresent.

*Art Digital Market.* Becoming the digital landlord of Mona Lisa? This has become possible with the emergence of *NFTs* (Non-Fungible Tokens). In essence, an NFT is a process which pegs a material or immaterial object to a log-note: the object history is entirely traced on the blockchain. It is also possible to "divide" an art object into small-sized tokens. Some artists have already used it, like Mike Winkelmann (Beeple) who sold his digitized art board up to 69.3 million of dollars. In addition, the start-up Arteïa proposes to gather, on its platform, confidential information on artists' work, its values and location, in an immutable and incorruptible fashion. The start-up wishes to compete with traditional auctions, allowing amateurs to sell or buy their works of art on the platform, avoiding the usual fees related to auction sale.

*Video Games.* Usually, features in video games remain in video games. Thus, Fortnite, even if it is free, allowed Ubisoft to make a 2.5 billion of dollars benefit in 2018 thanks to the sale of "features" (dressing-ups, clothes) for the characters. However, none of these objects belong to the gamers. Ubisoft have been working on blockchain (Game Alliance) to remedy this situation: under the form of tokens, all the bought products can have a proper existence outside the game. Gamers can buy, sell, and exchange their tokens. More important, it is planned that the company builds games which offer gamers the freedom to modify the game at their convenience. This might be the case for a next Assassin's Creed.

Many other applications can be found in [23–25].

Thus, blockchain really has the possibility to literally change our lifestyle; hence, it is necessary to model all the possible underlying services. A core modeling, that is, which should support any mathematical approach involving blockchain, has not sufficiently been popularized in the scientific and engineering communities, despite some attempts, however rich the contents [1]. Another recent book has also presented, in a self-consistent way, the Blockchain and its technology, and explains its content also from a quantitative viewpoint [2]. Besides, the content of this book aims at being mathematical, the computer science and practical technological

viewpoints are not engaged, and we hope that it can further assist development and implementation of future services using blockchain.

This book, mainly inspired by Bitcoin (but could be extended provided relevant modifications), is depicted as follows. Chapter 2 deals with the very first modeling of the successive blocks, their content, and can be seen as a posed general modeling for Blockchain. Chapter 3 focuses on the mathematical links between Blockchain and transaction data content, especially digital signatures, that is, it makes the links between Blockchain itself and the classical cryptographic approach to sign a transaction. Chapter 4 extends the Blockchain modeling to the Network, made of miners and users who directly contribute to the blockchain expansion. Thus, the first three sections are proposing a general mathematical framework for the universe of Blockchain. Chapter 5 exposes a theory behind the halving process. Coupled with the digital root function (sum of the digits of a number), a symmetry is revealed, actually broken by the Satoshi precision. Digit patterns are identified and proven. Chapter 6 develops a generalization of the binary Merkle tree, that is, the $n$-tree, where $n$ is a natural integer. This study actually shows that, for a miner, the implementation of $n$-tree is more efficient than Merkle trees. Chapter 7 introduces the Entropy applied to blockchain and peer-to-peer networks, where the concept of Entropy Rate is particularly relevant. The Entropy of Privacy is quantifying the evolution of exchanges of bitcoins through the community and is the witness of a higher privacy protection enhanced through the Particles model, which will also be introduced. Finally, Chapter 8 proposes some applications, especially using the developed theory to apply it to some influential research articles elaborated thus far, and how their contents are related to the approach are explained in this book.

*Note*: At the beginning of the chapters, some mathematical preliminaries are presented. The reader should be familiar with the results cited therein, each of them is, sometimes subtlety, used in the sequence of the section.

# Chapter 2
# Some Fundamentals of Mathematics of Blockchain


This section is introductory. We set the very first mathematics needed for any blockchain modeling, by considering a blockchain itself. The other parts of this chapter are depending on the contents of this section. The reader is supposed to be familiar with the basic mathematics, including probability and the used algebra, as well as the notions of bit, bytes, and hexadecimal in computer science. We re-introduce these notions first of all, for the convenience of the reader.

## 1 Preliminaries on Analysis

We introduce the set of *natural* integer numbers $\mathbb{N} = \{0, 1, 2, \ldots\}$ as well as $\mathbb{N}^* = \mathbb{N}\backslash\{0\} = \{1, 2, \ldots\}$. We also introduce the set of *relative* integer numbers $\mathbb{Z} = \mathbb{N} \cup (-\mathbb{N}) = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$. In the sequence, $\mathbb{N}$ or $\mathbb{Z}$ can be in decimal unit, but we can indicate the size of the considered integer with a subscript when needed. Thus, $\mathbb{N}_4$ is the (finite) set of 4-bytes numbers for hexadecimal or little-endian (byte-reverse hexadecimal) representations. In addition, if $x \in \nleftarrow\mathbb{N}$, then $\overleftarrow{x}$ is the byte reverse order of $x$. For instance, if $x = 01f8 \in \mathbb{N}_2$, then $\overleftarrow{x} = f801 \in \nleftarrow\mathbb{N}_2$.

Let $\mathbb{R}$ be the set of real numbers, and $\mathbb{R}_+ = \{x \in \mathbb{R}, x \geq 0\}$. We note $\lfloor x \rfloor \nleftarrow$the flooring integer part of $x$ and $\lceil x \rceil \nleftarrow$the ceiling integer part of $x$, for any $x \in \nleftarrow\mathbb{R}$. We might as well call E$(\cdot)$ (resp. c$(\cdot)$) the flooring (resp. ceiling) integer part function.

If the set $E$ is finite, we may write $E = \{x_1, \ldots, x_n\}$, where $x_1, \ldots, x_n$ are the distinct elements of $E$, and we set Card$(E) = n$ as the number of elements of $E$.

### *1.1 Basic Functions and Conventions*

Let $E$ and $F$ be two discrete sets. The function $f : E \to F$ is said to be *one-way* (or *onto*, or *injective*) if $f(0) = 0$ and



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_2

$$\forall (x, x') \in E^2 \; f(x) = f(x') \Rightarrow x = x'.$$

It is clear that $f(x) = 0 \Rightarrow x = 0$. This implies that $\mathrm{Card}(E) \leq \mathrm{Card}(F)$. (if this inequality makes sense).

The function $g : E \to F$ is said to be *surjective* (or *into*) if

$$\forall y \in F \quad \exists x \in E \quad y = f(x).$$

This implies that $\mathrm{Card}(E) \geq \mathrm{Card}(F)$ (if this inequality makes sense).

The function $h : E \to F$ is said to be *bijective* if

$$\forall y \in F \quad \exists! x \in E \quad y = f(x).$$

This implies that $\mathrm{Card}(E) = \mathrm{Card}(F)$ (if this equality makes sense), and that $f$ is bijective if and only if it is both injective and surjective.

## 1.2 Hashing Function

Let $E, F \subset \mathbb{N}$ be finite parts of $\mathbb{N}$. A hashing function $H : E \to F$ is expected to have the following three practical properties:

- *Pre-image resistance*: for a given $y \in F$, it is difficult to find $x \in E$ such that $y = H(x)$;
- *Second pre-image resistance*: for a given $x \in E$, it is difficult to find another $x' \neq x$ in $E$ such that $H(x) = H(x')$;
- *Collision resistance*: it is difficult to find two distinct $x$, $x' \in E$ such that $H(x) = H(x')$.

By "it is difficult," we mean that the problem of finding such elements is an NP problem (we do not enter these details in the following). It is worth pointing out that a collision-resistant function is second pre-image resistant (not the converse), but not necessarily pre-image resistant. In addition, the last two points are satisfied if the function $H$ is injective.

It is worth pointing out that a combination of hashing functions is a hashing function. In fact, if $H : E' \to F$ and $H' : E \to E'$ are two hashing functions, for a given $y \in F$, it is difficult to find $x' \in E'$ such that $y = H(x')$, and thus again more difficult to find $x \in E$ such that $y = H \circ H'(x)$. Also, if $\overline{x} \in E'$, it is difficult to find another $\overline{x'} \neq \overline{x} \in E'$, such that $H(\overline{x}) = H\left(\overline{x'}\right)$, thus more difficult to find, for a given $x \in E$, another $x' \neq x$ in $E$ such that $H \circ H'(x) = H \circ H'(x')$. Finally, it is difficult to find two distinct $\overline{x}, \overline{x'} \in E'$ such that $H(\overline{x}) = H\left(\overline{x'}\right)$, and more difficult to find $x, x' \in E$ such that $H \circ H'(x) = H \circ H'(x')$.

In the following, the function $H$ is the *double-SHA256* used for Bitcoin mining. So far, we do not know if this function is injective.

## 1.3 Probability

Let $\Omega$ be the random set, and $\mathcal{T}$ a $\sigma$-algebra of $\Omega$ (which is a part of the set of all parts of $\Omega$). A (real) random variable $X$ is a map $X : (\Omega, \mathcal{T}) \to (\mathbb{R}, \mathcal{B}_{\mathbb{R}})$, where $\mathcal{B}_{\mathbb{R}}$ is the set of Borelians of $\mathbb{R}$ (i.e., sets of $\mathbb{R}$ endowed with intervals of the form $]a, +\infty[$, for any $a \in \mathbb{R}$).

A probability $\mathbb{P}$ is a measure on $\mathbb{R}$. For any $B \in \mathcal{B}_{\mathbb{R}}$, the number $\mathbb{P}(X \in B)$ is in $[0, 1]$.

The cumulative distribution function (CDF) associated with $X$ is $F_X(x) = \mathbb{P}(X \leq x)$. When $F_X$ is absolutely continuous, then there exists a function $f_X$, which is the probability density function (PDF), such that

$$F_X(x) = \int_{-\infty}^{x} f_X(t)\mathrm{d}t.$$

We have

$$f_X(x) = \frac{\mathrm{d}F_X}{\mathrm{d}x}(x).$$

The expected value is given by

$$\mathbb{E}(X) = \int_{-\infty}^{+\infty} t f_X(t)\mathrm{d}t.$$

More generally, the transfer formula gives, for any Borelian function $g$, that

$$\mathbb{E}(g(X)) = \int_{-\infty}^{+\infty} g(t)\, f_X(t)\, \mathrm{d}t.$$

Finally, we recall an important aspect of the *Levy theorem*, stating that if a random variable has its Laplace transform satisfying the one from a random variable with a given law, then the former random variable has the same law as the latter.

## 1.4 Concatenation

Let $x, y \in I \subseteq \mathbb{N}$, where $I$ is finite or not. We introduce the *concatenation operator*, written as $\|$, such that $x\|y \in \mathbb{N}$ is the digit concatenation of the elements $x$ and $y$. Thus, if $x = 7$ and $y = 3$, then $x\|y = 73 \in \mathbb{N}$.

Let $E = \{x_1, \ldots, x_n\}$ be a finite set of $I \subseteq \mathbb{N}$. The number $E^{\|\leftarrow}$ is the concatenation of all its elements: $E^{\|\leftarrow} = x_1 \| \ldots \| x_n$, which we write $E^{\|\leftarrow} = \overset{n}{\underset{i=1}{\|\leftarrow}} x_i$ and $E^{\|} \in \mathbb{N}$ (but perhaps $E \notin I$).

## 1.5 Memoryless Property of Positive Continuous Random Variables

Let $T$ be a positive continuous random variable with the memoryless property, i.e., $\mathbb{P}(T > s + t) = \mathbb{P}(T > s)\,\mathbb{P}(T > t)$, for any $s, t \geq 0$. By using Bayes' Theorem, this is equivalent to

$$\mathbb{P}(T > s + t \,|\, T > s) = \frac{\mathbb{P}(\{T > s + t\} \cap \{T > s\})}{\mathbb{P}(T > s)} = \frac{\mathbb{P}(T > s + t)}{\mathbb{P}(T > s)} = \mathbb{P}(\overleftarrow{T} > \overleftarrow{t}).$$

The memoryless property means that the random variable $T$ *forgets its past*.

We can prove that the random variable $T$ follows an exponential distribution (the converse is straightforward). In fact, by setting $G_T = 1 - F_T$, from the memoryless property, we have

$$G_T(pt) = G_T(t)^p, \quad \forall p \in \mathbb{N}.$$

The case $p = 0$ is clear, since $T \geq 0$ and $F_T(0) = \mathbb{P}(T \leq 0) = 0$.

Similarly, we have

$$G_T\left(\frac{t}{q}\right) = G_T(t)^{\frac{1}{q}}, \quad \forall q \in \mathbb{N}^*.$$

Therefore, we have

$$G_T\left(\frac{p}{q}t\right) = G_T(t)^{\frac{p}{q}}, \quad \forall (p, q) \in \mathbb{N} \times \mathbb{N}^*.$$

Since $G_T$ is continuous and $\mathbb{Q}$ is dense in $\mathbb{R}$, from the following equation, we have

$$G_T(xt) = G_T(t)^x, \quad \forall x \in \mathbb{R}_+.$$

If $t = 1$, we have

$$G_T(x) = G_T(1)^x, \quad \forall x \in \mathbb{R}_+.$$

Set $\lambda = -\ln G_T(1) \geq 0$. We therefore have

$$G_T(x) = \mathrm{e}^{-\lambda x}, \quad \forall x \in \mathbb{R}_+.$$

We conclude.

## 2 Block Modelling

A blockchain is a succession of blocks which are added one by one. A representation is shown in Fig. 2.1.

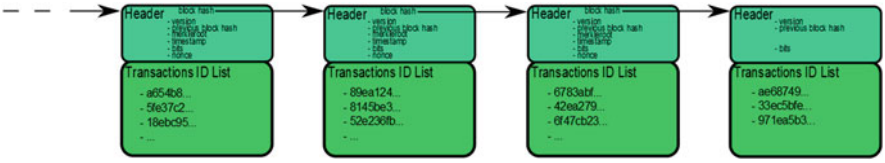


**Fig. 2.1** Representation of a blockchain. (Source: Julien Riposo, 2023)

In the sequence, we mathematically describe the content of each block as well as their interaction.

### *2.1 Mathematics of Blockchain*

More formally, we have the following definition.

***Definition 2.2.1*** *A* block $B_n$*, where* $n \in \mathbb{N}$*, is a pair of families* $B_n = (\mathcal{H}_n, T_n)$*.*
*The* Header $\mathcal{H}_n$ *writes as*

$$\mathcal{H}_n = \left(v_n, H\left(\mathcal{H}_{n-1}^{\|\leftarrow}\right), r(T_n), \mathrm{Stamp}(t), b_n, N_t\right),$$

*where*

- $v_n \in \mathbb{N}_4$ *is the* version *of the block;*
- $H\left(\mathcal{H}_{n-1}^{\|\leftarrow}\right) \in \mathbb{N}_{32}$ *is the hash of the concatenated header (or block hash) of the previous block* $B_{n-1}$;
- $r(T_n) \in \mathbb{N}_{32}$ *is the* Merkle root *number (see Sect. 3);*
- $\mathrm{Stamp}(t) \in \mathbb{N}_4$ *is the* time*, in a readable format, at which the block is being hashed, at time* $t \in \mathbb{R}_+$*;*

– $b_n \in \overleftarrow{\mathbb{N}}_4$ *is the* bits*, shortened version of the* Target*, which is a number representing the difficulty to mine a block (see below);*
– $N_t \in \overleftarrow{\mathbb{N}}_4$ *is the* nonce *found at time t. It represents a number which allows linking the block* $B_n$ *with the previous one (see Definition 2.2.3).*

*The* Transactions List $T_n$ *writes as*

$$T_n = \begin{pmatrix} \mathrm{TXID}_1 \\ \mathrm{TXID}_2 \\ \vdots \\ \mathrm{TXID}_{l_n} \end{pmatrix},$$

*where*

– $l_n \in \overleftarrow{\mathbb{N}}^*$ *is the dimension of* $T_n$*;*
– $\mathrm{TXID}_l \in \mathbb{N}_{32}$, $1 \leq l \leq l_n$, *is the* $l^{\text{th}}$ transaction ID*, being the hash of the* $l^{th}$ *transaction.*

In particular, $\mathrm{TXID}_1$ is the *Coinbase* transaction ID (used to pay the miner who has mined this list of transactions). Note that, when $n = 0$ (*genesis* block), then $H\left(\mathcal{H}_{-1}^{\|}\right)$ is an arbitrary number in $\mathbb{N}_{32}$.

We now introduce the blockchain.

***Definition 2.2.2*** *A (finite) sequence of blocks* $(B_n)_{I \subseteq \mathbb{N}}$ *is called* Blockchain.

The blocks of a blockchain are linked through the proof-of-work, defined now.

***Definition 2.2.3*** *A* Proof-of-Work *is a brute-force iterative trial-and-error process consisting in finding at least one number* $N \in \overleftarrow{\mathbb{N}}_4$ *such that*

$$H\left(v_n \| H\left(\mathcal{H}_{n-1}^{\|\leftarrow}\right) \| r(\tau_n) \| \mathrm{Stamp}(t) \| b_n \| N\right) \leq \mathrm{t}_n,$$

*where*

– $\tau_n$ *is the transaction list still being populated, i.e.,* $\tau_n \subseteq T_n$*;*
– $\mathrm{t}_n \in \overleftarrow{\mathbb{N}}_{32}$ *is the* target*. It is an adjusted number which adjusts the difficulty to have the above inequality satisfied (see below).*

The proof of work is not used for every blockchain (e.g., Ethereum 2.0), but Bitcoin uses it.

The target $\mathrm{t}_n$ and the bits $b_n$ designate the same entity. In fact, $b_n$ is a compact way of writing $\mathrm{t}_n$. Once an adequate nonce is found at time $t$, then the list $\tau_n$ is definitive (i.e., $\tau_n = T_n$) and $N_t = N$. The block $B_n$ is mined, so that $H\left(\mathcal{H}_n^{\|}\right) \leq \mathrm{t}_n$. This inequality is satisfied, for any $n' \leq n$.

It is worth pointing out that the number $\mathrm{t}_n$ (and thus $b_n$) is changed every 2016 blocks (for Bitcoin), and usual standards are that it starts (from the left) with a lot of 0 digits whose number change with time. The higher the number of 0, the lower $\mathrm{t}_n$,

and the more *difficult* it is to find a nonce, thus to mine a block. Hence, the number $t_n$ is also referred to as *difficulty*. This is because hashing a number is giving a "random" number (it is deterministic, but very hard to predict); thus, the probability to find a low hash is smaller than finding a higher one. Hence, it is difficult to solve the inequality if the target $t_n$ is a small number.

## 2.2 Time for Mining Blocks

Let $T$ be the time for mining a block, i.e., for one miner among the pool of miners to find a nonce by the proof-of-work (see Definition 2.2.3).

It is reasonable to assume that the process of finding a nonce is memoryless, since there is an infinite number of possibilities to derive numbers and find one that satisfies the proof-of-work inequality. According to Sect. 1.5, it turns out that $T$ is an exponential random variable.

We will need the following definition.

***Definition 2.2.4*** *Let $(t_i)_{i\in\mathbb{N}^*}$ be a sequence of independent and identically distributed exponential random variables, of parameter $\alpha > 0$. The PDF is given by $f_{t_i}(x) = \alpha e^{-\alpha x}$, for any $x \in \mathbb{R}_+$. Let $\bar{t}_n = \sum_{i=1}^{n} t_i$. We introduce the* Poisson process *of intensity $\alpha$ the process $\mathcal{N}_t$ (continuous random variable) given by*

$$\mathcal{N}_t = \sum_{n\geq 1} 1_{\bar{t}_n \leq t} = \sum_{n\geq 1} n\, 1_{\bar{t}_n \leq t < \bar{t}_{n+1}},$$

*where $1_A = 1$ if the event $A$ is realized, and 0 otherwise.*

We note that $\alpha$ is related to the inverse of the *hash rate* (which is the frequency at which a hash is produced), since the higher the $\alpha$, the higher the expected value: $\mathbb{E}(t_i) = 1/\alpha$, for all $i \geq 1$. Thus, the number $\alpha$ is also related to the difficulty. The random variable $\mathcal{N}_t$ represents the number of points of the sequence $(t_i)_{i\in\mathbb{N}^*}$, which are lower than $t$. It represents the number of mined blocks at time $t$.

Note that $\mathcal{N}_0 = 0$, so we also have

$$\bar{t}_n = \inf\{t \geq 0, \mathcal{N}_t = n\}.$$

This is the random minimum time to which $\mathcal{N}_t = n$.

***Proposition 2.2.5*** *If $(\mathcal{N}_t)_{t\geq 0}$ is a Poisson process of intensity $\alpha$, then for any $t > 0$, the random variable $\mathcal{N}_t$ is a Poisson random variable, of parameter $\alpha t$.*

***Proof***

We can prove that $\bar{t}_n$ is a Gamma random variable of parameter $(n, \alpha)$. In fact, we have

$$f_{\bar{t}_n}(x) = \alpha \mathrm{e}^{-\alpha x} \frac{(\alpha x)^{n-1}}{(n-1)!}, \quad \text{for } x > 0.$$

Indeed, the transfer formula allows to calculate the Laplace transform of $t_i$:

$$\mathbb{E}(\mathrm{e}^{-at_i}) = \frac{\alpha}{\alpha + a}, \quad \text{for any } i \geq 1.$$

Thus, using independence, we have

$$\mathbb{E}\left(\mathrm{e}^{-a\bar{t}_n}\right) = (\mathbb{E}(\mathrm{e}^{-at_i}))^n = \left(\frac{\alpha}{\alpha + a}\right)^n.$$

This is the Laplace transform of a Gamma random variable of parameter $(n, \alpha)$. By the Levy Theorem, we conclude that $\bar{t}_n$ is a Gamma random variable of parameter $(n, \alpha)$.

For any $n \geq 1$, we then have

$$\mathbb{P}(\mathcal{N}_t = n) = \mathbb{P}(\bar{t}_n \leq t) - \mathbb{P}(\bar{t}_{n+1} \leq t) = \int_0^t \alpha \mathrm{e}^{-\alpha x} \frac{(\alpha x)^{n-1}}{(n-1)!} \mathrm{d}x - \int_0^t \alpha \mathrm{e}^{-\alpha x} \frac{(\alpha x)^n}{n!} \mathrm{d}x$$
$$= \frac{(\alpha t)^n}{n!} \mathrm{e}^{-\alpha t}.$$

This concludes the proof. ∎

From the blockchain perspective, the random variables $T$ and the $(t_i)_{i \in \mathbb{N}^*}$ are all independent and identically distributed according to an exponential law of parameter $\alpha = 1/10$ min.

Thus, the number of blocks follow a Poisson process. Conversely, it can be proven that if the number of blocks follow a Poisson process of intensity $\alpha$, then the time to mine $n$ blocks follows a Gamma law of parameters $(n, \alpha)$.

## 3 Merkle Tree

This section explains how the Merkle root number is calculated. The Merkle root is a number that gathers all the transactions information of one block and is calculated from the Transaction List $T_n$. Each $\mathrm{TXID}_l$ in $(\mathrm{TXID}_l)_{1 \leq l \leq l_n}$ is a contributor. In fact, the number $\mathrm{TXID}_l$, $1 \leq l \leq l_n$, is the hash of a *transaction data* (see next section).

Overall, we concatenate the consecutive pairs of TXID in $T_n$, and calculate the hash. We end-up with another floor, basically made of another list of hashes whose number of elements is reduced by half roughly. This process is iteratively performed

until obtaining a unique hash: this is the Merkle root $r(T_n)$. The resulting process is a binary tree. In the following, we enter the mathematical details for constructing a Merkle root.

***Definition 2.3.1*** *A* Merkle Tree *is the resulting binary tree from the ordered (from left to right) leaves, each of them being weighted by a number in $\mathbb{N}_{32}$ (the associated transaction ID). To each parent, node is associated the hash of the concatenated two children numbers. The root of the tree's weight is called* Merkle root.

In the sequence of this section, we make the distinction between complete ($l_n$ is a power of 2) and uncomplete ($l_n$ is not a power of 2, i.e., the general case) Merkle Trees.

## 3.1 Complete Merkle Tree

More specifically, suppose the number of transaction IDs is a non-trivial power of 2, there exists $m \in \mathbb{N}^*$ such that $l_n = 2^m$.

Figure 2.2 shows an example of a complete Merkle Tree (where $m = 4$). From a number $2^m$ of transaction data, the transaction IDs are built, given the list $T_n$, as successive elements on floor $h = 0$ (ground floor). In fact, these are oriented from left (oldest mined transaction) to right (youngest mined transaction).

In the following, we denote by $T_n(h)$ the ordered list from left to right at the $h^{\text{th}}$ floor, and $T_n(h)^{(l)}$ its $l^{\text{th}}$ element, with $1 \leq l \leq l_n/2^h$. In fact, if $0 < h \leq \mathrm{H} = m$ (there are $\mathrm{H} = m$ floors in total), then the $l^{\text{th}}$ element of $T_n(h)^{(l)}$, with $1 \leq l \leq l_n/2^h$, is given by

$$T_n(h)^{(l)} \coloneqq H\left(T_n(h-1)^{(2l-1)} \| T_n(h-1)^{(2l)}\right).$$

***Definition 2.3.2*** *A* Merkle Path *is a list of hashes, such that the first element is in $T_n(0)$, the second element is related to the parent in $T_n(1)$, and so on, until the last element, given by $r(T_n)$.*

Thus, the dimension of a Merkle Path is $m$. An example is shown in Fig. 2.2, green nodes (arrowed path).

***Definition 2.3.3*** *An* Authentication Path *is a list of hashes, such that all the elements are made of the brothers of the ones in the Merkle Path. The Merkle root is not included.*

Thus, the dimension of an Authentication Path is $m - 1$. An example is shown in Fig. 2.2, squared purple nodes.

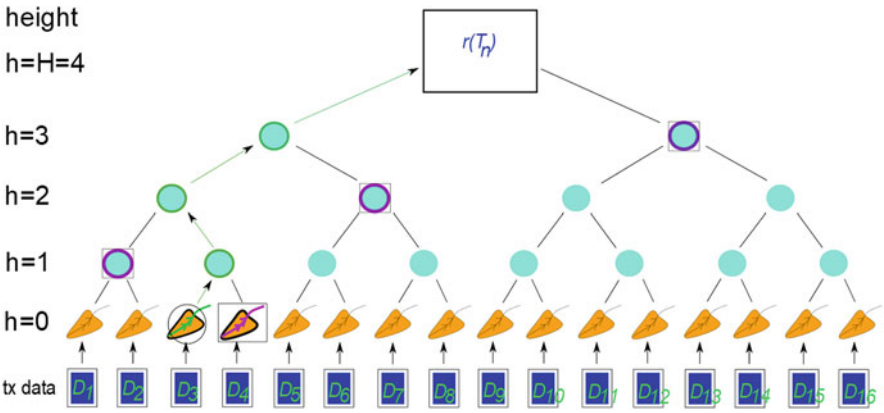


**Fig. 2.2** A complete Merkle Tree. From the bottom to the top, the transaction data (Tx data) are all hashed twice, giving the TXIDs (leaves) (ground floor $h = 0$). The Tx data do not appear in the blockchain. The TXIDs are then paired concatenated and hashed twice (giving nodes at floor $h = 1$), and this process is iteratively repeated until reaching one unique root (top floor), i.e., the Merkle root. Circled nodes do not appear in the blockchain, but the leaves are. We suppose a client's TXID of interest as the green leaf (circled leaf). The *Merkle Path* for this TXID is the unique shortest path up to the root (arrowed green path). The *Authentication Path*, allowing the client to verify if her TXID is mined, corresponds to the squared circled nodes, in purple. The Authentication Path cannot be computed without the full knowledge of the Merkle Tree. (Source: Julien Riposo, 2023)

## 3.2 Uncomplete Merkle Tree

In practice, $l_n$ is not a power of 2. There still is a way to construct a Merkle Tree: by completing the uncomplete tree with the last element of $T_n$. Figure 2.3 is an illustration of an Uncomplete Merkle Tree.

In some contexts (e.g., Sect. 5.2, Chap. 6), it can be interesting to calculate the total number of nodes, leaves included, without including "artificial" additions (see Fig. 2.3). Artificial additions can be removed – thus the total number of vertices should be known.

***Theorem 2.3.4*** *Let $l(h)$ be the dimension of $T_n(h)$ (i.e., number of nodes on floor h). An Uncomplete Merkle Tree with $l(0) = l_n \in \mathbb{N}^*$ leaves has a total number of nodes (leaves included) given by*

$$\sum_{h=0}^{\mathrm{H}} l(h) = \sum_{h=0}^{\left\lceil \frac{\log l_n}{\log 2} \right\rceil} \left\lceil \left\lceil \cdots \frac{\left\lceil \frac{l_n}{2} \right\rceil}{2} \right\rceil /2 \right\rceil \ (h \text{ times}),$$

*where* H *is the height of the tree.*

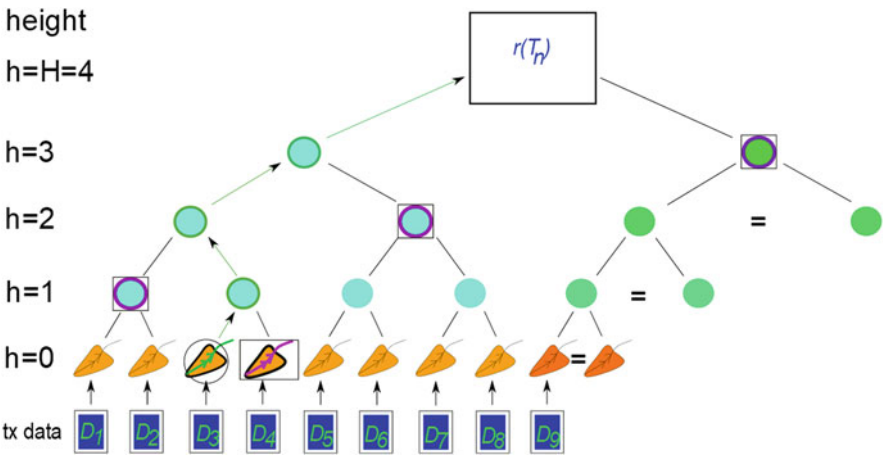


**Fig. 2.3** Example of Uncomplete Merkle Tree. We still can define a Merkle Path and an Authentication Path. The process "=" here means that the number of elements from floors $0 \leq h < \mathrm{H}$ need to be pair ("artificial" addition, which are not real nodes). (Source: Julien Riposo, 2023)

***Proof***

Let H be the height of the tree. Note that, if $l(h)$ is the dimension of $T_n(h)$, for any $0 \leq h < \mathrm{H}$, then the dimension $l(h+1)$ of $T_n(h+1)$ is

$$l(h+1) = \left\lceil \frac{l(h)}{2} \right\rceil^{\leftarrow}.$$

In addition, in fact, for any $l_n \in^{\leftarrow} \mathbb{N}^*$, there exists $m \in^{\leftarrow} \mathbb{N}^*$ such that $2^{m-1} < l_n \leq 2^m$. Indeed, note that this double inequality is equivalent to

$$m - 1 < \frac{\log l_n}{\log 2} \leq m.$$

Therefore, $m = \left\lceil \frac{\log l_n}{\log 2} \right\rceil$. Thus, $\mathrm{H} = m$ is the height of the Merkle Tree. Finally, we obtain the desired result by induction on $h$ with the above iterative equation, that is

$$l(h) = \left\lceil \left\lceil \ldots \frac{\left\lceil \frac{l_n}{2} \right\rceil}{2} \right\rceil /2 \right\rceil \ (h \text{ times}).$$

We have one detail to verify:

$$l(\mathrm{H}) = \left\lceil \left\lceil \ldots \frac{\left\lceil \frac{l_n}{2} \right\rceil}{2} \right\rceil /2 \right\rceil \ (\mathrm{H} \text{ times}) = 1,$$

since the Merkle root is the unique node at floor $h = \mathrm{H}$. However, remember that we have $2^{H-1} < l_n \leq 2^H$. Therefore,

$$\frac{1}{2} < \left\lceil \frac{1}{2} \right\rceil = 1 \leq \left\lceil \left\lceil \ldots \frac{\left\lceil \frac{l_n}{2} \right\rceil}{2} \right\rceil / 2 \right\rceil \text{ (H times)} \leq 1,$$

which finally ends the proof. ∎

Note that this proposition is valid for a complete Merkle Tree. If indeed $l_n = 2^m$, then $l(h) = \left\lceil \left\lceil \ldots \frac{\left\lceil \frac{l_n}{2} \right\rceil}{2} \right\rceil \overleftarrow{/2} \right\rceil$ ($h$ times) $= 2^{m-h}$, for any $0 \leq h \leq m$. Thus,

$$\sum_{h=0}^{m} l(h) = 2^m \, \frac{1 - \left(\frac{1}{2}\right)^{m+1}}{\frac{1}{2}} = 2^{m+1} - 1.$$

In addition, we deduce from the above that Definitions 2.3.2 and 2.3.3 are still valid for an uncomplete Merkle Tree.

In Chap. 6, we will give a generalization of the Merkle Tree, and will present the $n$-Tree (also called *Verkle Tree* by Vitalik Buterin, the inventor of Ethereum). We will show empirical evidence that $n$-Trees are more efficient than Merkle Trees.

## 4 Transaction Data

In this section, we explain in detail the structure of a transaction data, which is a number containing all relevant information needed to fully describe the transaction uniquely. We will need the content of this section to understand the use of Digital Signature, Chap. 3.

Figure 2.4 shows an example of the main components of a transaction data (Tx data).

Before seeing what a Tx data precisely is, we first need the following definitions. We let $n \in \overleftarrow{\mathbb{N}^*}$ be the index of a considered block. We fix $l$ such that $1 \leq l \leq l_n$, where $l_n$ is the dimension of the Transaction List $T_n$ and focus on transaction ID $\mathrm{TXID}^l = T_n(0)^{(l)}$, in accordance with the notations of Sect. 3.2.

***Definition 2.4.1*** *An* Output *related to transaction ID* $\mathrm{TXID}^l$ *is given by the following hash:*

$$\mathrm{Output}_{l,k} = H\left(\overleftarrow{\mathrm{Val}_{l,k}} \| \mathrm{ScripPubKey}_{l,k}\ \mathrm{Size} \| \mathrm{ScriptPubKey}_{l,k}\right),$$

*where*

– $k \in \overleftarrow{\{1, \ldots, K'\}}$ *is the index of the output and* $K' \in \overleftarrow{\mathbb{N}}$ *is the number of outputs for transaction ID* $\mathrm{TXID}^l$*;*

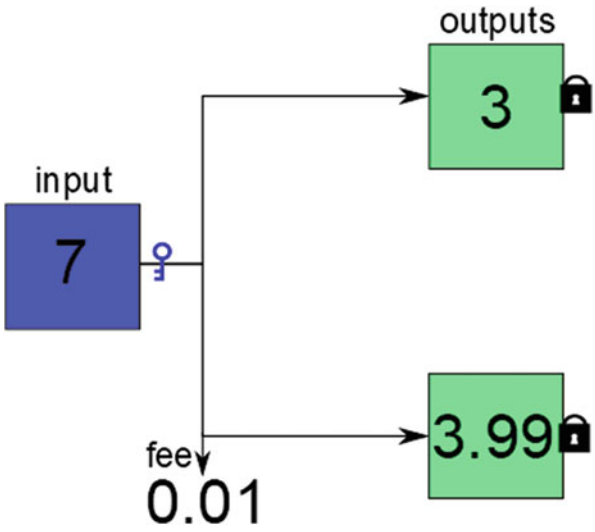


**Fig. 2.4** A transaction data (Tx data) is a concatenation of inputs and outputs. The text explains the precise constitution of a Tx data. (Source: Julien Riposo, 2023)

- $\text{Val}_{l,\,k} \in \mathbb{N}_8$ *is the value of bitcoins (in satoshis) in the output;*
- $\text{ScripPubKey}_{l,\,k}$ Size $\in \mathbb{N}$ *is the size of* $\text{ScriptPubKey}_{l,\,k}$*;*
- $\text{ScriptPubKey}_{l,\,k}$ *is the hash of the public key. This is represented by the lock in Fig. 2.4 (see also Definition 3.4.1).*

In practice, an output is a "house" where there are bitcoins. The entrance door of the house is unlocked with the private key (in practice, any randomly generated number preciously conserved for future transactions, see Section III.4) giving the public key $\text{ScriptPubKey}_l$. The format of the public key depends on the blockchain. For Bitcoin, $\text{ScriptPubKey}_l$ is the hash of the public key (the used *script pattern* – i.e., the format – is *Pay To Pubkey Hash*, or P2PKH).

***Definition 2.4.2*** *An* Input *(*or *Address) is given by the following hash:*

$$\text{Input}_{l,k} = H\left(\overleftarrow{\text{TXID}_{l'}} \| \overleftarrow{k'} \| \text{ScriptSig}_{l,k}\ \text{Size} \| \text{ScriptSig}_{l,k} \| \overleftarrow{\text{Seq}_{l,k}}\right),$$

*where*

- $k \in \{1, \ldots, K\}$ *is the index of the input and* $K \in \mathbb{N}$ *is the number of inputs for transaction ID* $\text{TXID}_l$*;*
- $\text{TXID}_{l'}$ *is a TXID belonging to* $T_{n'}$*, where either* $n' < n$ *and* $1 \le l' \le l_{n'}$*, or if* $n' = n$*, then* $l' < l$*;*
- $k'$ *is one* output index *(to designate the output) of the transaction related to* $\text{TXID}_{l'}$*, with* $1 \le k' \le K'$*,* $K' \in \mathbb{N}$ *being the number of outputs for the transaction* $\text{TXID}_{l'}$*;*
- $\text{ScriptSig}_{l,\,k}$ Size $\in \mathbb{N}$ *is the size of* $\text{ScriptSig}_{l,\,k}$*;*

– $\mathrm{ScriptSig}_{l,\,k}$ *is a script which unlocks the input. This is represented by the blue key in Fig. 2.4. This contains the digital signature (see Sects. 4 and 5, Chap. 3).*
– $\mathrm{Seq}_{l,\,k} \in \overleftarrow{\mathbb{N}}_4$ *expresses the signal size. Default is the max size as* $0 \times \mathit{ffffffff}$.

An input is a *spent output*, thus is not spendable and belongs to the past of the blockchain.

***Definition 2.4.3*** *A* Transaction data (Tx data)*, written as* $\mathrm{Tx}_l$*, is the concatenation of several numbers. With notations of Definitions 2.4.1 and 2.4.2, we have*

$$\mathrm{Tx}_l = \overleftarrow{v_n} \| K \underset{k=1}{\overset{K}{\| \leftarrow \mathrm{Input}_{l,k}}} \| \overleftarrow{K'} \underset{k'=1}{\overset{K'\leftarrow}{\| \leftarrow \mathrm{Output}_{l,k'}}} \| \overleftarrow{\mathrm{LockTime}_l},$$

*where* $\mathrm{LockTime}_l$ *is the earliest time (in Unix measure) a transaction can be mined in to a block.*

The locktime indicates when a transaction could be relayed on the network.

***Definition 2.4.4*** *A* Transaction ID (TXID) *is the hash of a Tx data, that is*

$$\mathrm{TXID}_l = H(\mathrm{Tx}_l).$$

We keep in mind that the element $\mathrm{ScriptSig}_l$ is containing the *digital signature* allowing to unlock the output and make the money transfer from outputs to others.

# Chapter 3
# Digital Signature


The authentication of documents is based on commercial exchange systems (e.g., contracts, receipts, notary documents). Diffie and Hellman were the first ones, in 1976, to develop a numerically authentication process offering the same warranties as the manuscript signatures. These *digital signatures* need to ensure the genuineness and integrity of the signed message (i.e., they need to ensure that the data have not been modified through an unauthorized process and that they well come from the indicated source) as well as non-renunciation (i.e., they need to ensure that an entity cannot deny engagements already done). This last property is generally assured by a public verification, but the first one implies that *one* entity needs to produce a couple "message/signature" accepted by this process. In particular, it must be impossible to deduce the way to sign from the verification process, and consequently this type of protocol needs to be *asymmetric*.

The *Digital Signature Algorithm* is a particular asymmetric process, which is applied to the blockchain environment, to sign a transaction. Thus, this section is the straight continuation of the previous Sect. 4, Chap. 2 and explains the mechanism for digital signatures. In particular, the aim is to outline the purpose of unlocking an output for transacting. The content of this section has strongly been inspired by [3].

## 1 Preliminaries on Needed Algebra

### *1.1 Group*

A group is a pair $(G, \cdot)$, where $G$ is a set of elements and $\cdot$ is an internal operation law, which are such that

- $\forall (x, y, z) \in \not{G} \quad (x \cdot y) \cdot z = x \cdot (y \cdot z)$ (the operation $\cdot$ is associative);
- there exists $e \in \not{G}$ such that $\forall x \in G \; x \cdot e = e \cdot x = x$ ($e$ is the (unique) neutral element);



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_3

– $\forall x \in \not{G} \quad \exists x^{-1} \in G \quad x \cdot x^{-1} = x^{-1} \cdot x = e$ ($x^{-1}$ is the symmetrical (or the inverse) of $x$).

When $\forall (x, y) \in G$, $x \cdot y = y \cdot x$, we say that $(G, \cdot)$ is an Abelian (or commutative) group. We will refer $G$ to be the group instead of the pair $(G, \cdot)$ when there is no ambiguity. When Card $G < +\infty$, we say that $G$ is a *finite* group.

## 1.2 Field

A field is a triplet $(F, +, \cdot)$, where $F$ is a set of elements and $+$ and $\cdot$ are internal operation laws, which are such that

– the pair $(F, +)$ is an Abelian group;
– the pair $(F^*, \cdot)$ is a group;
– $\forall (x, y, z) \in \not{F}$, $x \cdot (y + z) = x \cdot y + x \cdot z$, and $(y + z) \cdot x = y \cdot x + z \cdot x$ (the law $\cdot$ is distributive with respect to the law $+$).

Generally, the pair $(F^*, \cdot)$ is supposed to be an Abelian group. Also, when there is no systematic inverse for the elements of the group $(F^*, \cdot)$, we say that $(F, +, \cdot)$ is a *ring*, thus a field is a particular case of ring. A ring which has the division property but which is not commutative is named *division ring*. Finally, a *F-vector space* $(V, +, \cdot)$ is a space stable through addition $+$ and such that, for any $\lambda \in \not{F}$ and $x \in \not{V}$, then $\lambda \cdot x \in \not{V}$ (here $\cdot$ is an associative and distributed external law). If $F$ is a (division) ring, we call $(V, +, \cdot)$ an *F-module*.

We will refer $F$ to be the field instead of the triplet $(F, +, \cdot)$ when there is no ambiguity. When Card $F < +\infty$, we say that $F$ is a *finite* field.

The sets $G$ and $F$ are thus endowed with some *algebraic structures*.

Finally, if $F$ is a field, the set $\overline{F} \supset F$ is called an *algebraic closure* of $F$ if: any polynomial of degree higher or equal than 1, with coefficients in $F$, admits at least a root in $\overline{F}$ (thus all the roots are in $\overline{F}$). When $F$ is a finite field, we call a *generator* of $F$ an element $g \in \not{F}$, which generates **all** the elements of $F$. It turns out that, in this case, we have $F^* = \{1, g, g^2, \ldots, g^{q-1}\}$, where 1 is the neutral element of the group $F^*$ and $q = \text{Card } F$ is the *order* of the group $F^*$. If $h \in \not{F^*}$, we call $p$ the *order* of $h$, the smallest natural integer such that $h^p = 1$. It turns out that $p \leq q$. Also, the Lagrange Theorem states that $p$ divides $q$. These notions also apply to any finite group.

A particular field is the finite field $(\mathbb{Z}/q\mathbb{Z}, +, \times)$, where $q$ is a **prime** number, its order. We have

$$\mathbb{Z}/q\mathbb{Z} = \left\{\overline{0}, \overline{1}, \ldots, \overline{q-1}\right\},$$

where for any $k \in \{0, 1, \ldots, q-1\}$, the set $\overline{k}$ is the class, whose $k$ is a representant, in the following sense:

$$\forall l \in \overline{k} \quad \exists m \in \mathbb{Z} \quad l - k = mq.$$

We write the above equation $l \equiv k\ [q]$. It turns out that $l$ is another representant of $\overline{k}$ and $\overline{l} = \overline{k}$. When there won't be any ambiguity, $\overline{k}$ will also designate the number $k$.

## 1.3 Morphism

A morphism is a map that *conserves the algebraic structure* from a set to another one.

Thus, if $(G, \cdot)$ is a group and $G'$ a set, then a group morphism $f : G \to G'$ is such that we can endow $G'$ with an operator $\times$ such that

$$\forall (x, y) \in G \quad f(x \cdot y) = f(x) \times f(y), \quad \text{and } f(e) \text{ is the neutral of } G' \text{ when } e \text{ is the one of } G,$$

thus making $(G', \times)$ being a group. Conversely, if $(G, \cdot)$ and $(G', \times)$ are groups, then any map verifying the above equation is a group morphism. The same reasoning can be performed for a ring, and a field.

Finally, a bijective morphism is called an *isomorphism*, and two sets are isomorphic if there exists an isomorphism mapping one to another.

## 1.4 Characteristic

If $A$ is a ring with neutral $0_A$ for + and neutral $1_A$ for $\cdot$, we introduce the following (unitary, i.e., preserving the identity) ring morphism:

$$f : \left| \begin{array}{c} \mathbb{Z} \longrightarrow \mathrm{A} \\ n \longmapsto 1_A + \cdots + 1_A \ (n \text{ times}) \end{array} \right.$$

The characteristic of a ring $A$ is the smallest $n \in \mathbb{N}$ such that $f(n) = 0_A$ if such $n \in \mathbb{N}$ exists, otherwise it is given by $+\infty$. In the case where $A$ is a field, and such $n$ exists, then $n$ is a prime number, and $A$ and the quotient $\mathbb{Z}/n\mathbb{Z}$ are isomorphic (if such $n$ does not exist, then $A$ and $\mathbb{Z}$ are isomorphic). We write Char($A$) the number $n$.

Thus, when $A$ is a field and Char($A$) $= n$, then $n\,x = 0$, for any $x \in A$.

# 2 Elliptic Curves

## 2.1 Weierstrass Equation

Every elliptic curve can be written as a pair of a locus on the plane $\mathbb{R}^2$, given by a cubic equation, the *Weierstrass* Equation, and a point which is located at infinity, i.e., which belongs to the vertical line, at infinity.

***Definition 3.2.1*** *A* Weierstrass Equation *is an equation of the form*

$$W: \quad Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3,$$

*where* $(X, Y, Z) \in \mathbb{R}^3$ *are coordinates on the space* $\mathbb{R}^3$, $(a_1, a_2, a_3, a_4, a_6) \in \overline{K}^5$ *for a given field K.*

The point at infinity is the point written as $O(0, 1, 0)$. We immediately have the following simplification, by using non-homogeneous coordinates $x = X/Z$ and $y = Y/Z$.

***Definition 3.2.2*** *A* homogeneous Weierstrass Equation *is an equation of the form*

$$W_{\mathrm{H}}: \quad y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6,$$

*where* $(x, y) \in \mathbb{R}^2$ *are coordinates on the map* $\mathbb{R}^2$, $(a_1, a_2, a_3, a_4, a_6) \in \overline{K}^5$ *for a given field K.*

In the sequence, we will write $W$ for $W_{\mathrm{H}}$. The following theorem is important for the sequence.

***Theorem 3.2.3*** *If* $\mathrm{Char}(K)$ *is neither* 2 *nor* 3 *for a given field K, then*

$$W: \quad y^2 = x^3 + Ax + B,$$

*where* $(x, y) \in \mathbb{R}^2$ *are coordinates on the map* $\mathbb{R}^2$, $(A, B) \in \overline{K}$.

***Proof***
Let

$$W: \quad y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

with $(x, y) \in \mathbb{R}^2$ are coordinates on the map $\mathbb{R}^2$, $(a_1, a_2, a_3, a_4, a_6) \in \overline{K}^5$ for a given field $K$. If $\mathrm{Char}(K) \neq 2$, we perform the following transformation:

$$y \mapsto \frac{1}{2}\,(y - a_1x - a_3).$$

We also set the following parameters:

$$b_8 = a_1^2 a_6 + \cancel{4} a_2 a_6 - a_1 a_3 a_4 + \cancel{a_2} a_3^2 - a_4^2,$$

$$c_4 = b_2^2 - 24\, b_4,$$

$$c_6 = -b_2^3 + \cancel{36}\, b_2 b_4 - 216\, b_6,$$

$$\Delta = -b_2^2 b_8 - 8\, b_4^3 - 27\, b_6^2 + \cancel{9} b_2 b_4 b_6.$$

We obtain

$$W: \quad y^2 = 4x^3 + \cancel{b_2} x^2 + \cancel{2} b_4 x + \cancel{b_6}.$$

Now, suppose Char$(K) \neq 2, 3$. We perform the following transformation:

$$(x, y) \mapsto \left( \frac{x - 3\, b_2}{36}, \frac{y}{108} \right).$$

This eliminates the $x^2$ term, and we finally have

$$W: \quad y^2 = x^3 - 27 c_4 x - 54 c_6.$$

We set $A = -27 c_4$ and $B = -54 c_6$, which ends the proof. ∎

Figure 3.1 is showing three examples of elliptic curves.

***Definition 3.2.4*** *The number* $\Delta$ *(see the above proof) is called the* discriminant *of the Weierstrass Equation.*

According to the last theorem, this number reduces to

$$\Delta = -16 \left( 4 A^3 + \cancel{27} B^2 \right).$$

## 2.2 Non-singularity

Let $(x, y)$ be a point satisfying $f(x, y) = 0$, where

$$f(x, y) = y^2 - x^3 - Ax - B.$$

In other words, $(x, y)$ are the coordinates of a point on the elliptic curve defined by the Weierstrass Equation of Theorem 3.2.3.

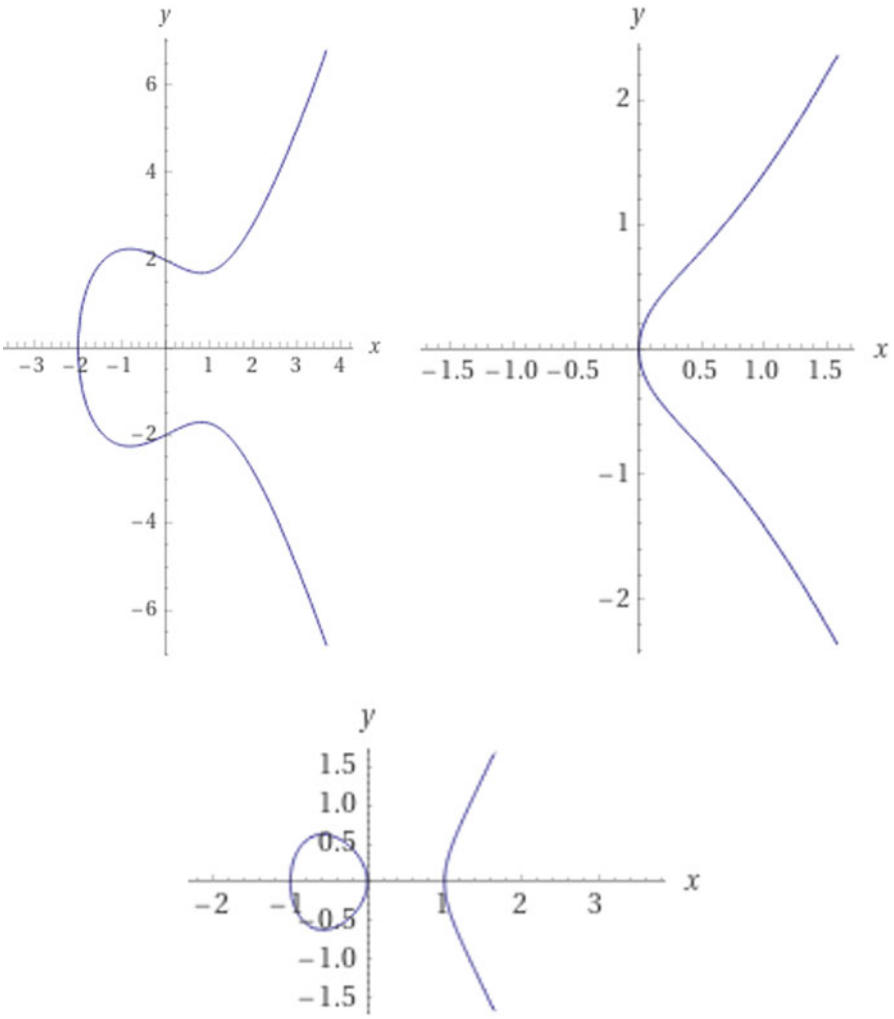


**Fig. 3.1** Elliptic curves with respective Weierstrass equations $y^2 = x^3 - 2x + 4$ (top left), $y^2 = x^3 + x$ (top right), and $y^2 = x^3 - x$ (bottom). (Source: Julien Riposo, 2023)

***Definition 3.2.5*** *The point on the curve of coordinates* $(x, y)$ *is said to be* singular *if*

$$\frac{\partial f}{\partial x}(x, y) = \frac{\partial f}{\partial y}(x, y) = 0.$$

This is to roughly link to the fact that, if $f$ is a $C^1$-class function (which is the case here since $f$ is polynomial), its differential d$f$ is zero at $(x, y)$, therefore is not invertible (i.e., it is singular) at this point. We say that the elliptic curve (or the Weierstrass Equation) is non-singular if there is no singular point. We have the important following theorem.

***Theorem 3.2.6*** *The curve given by the Weierstrass Equation is non-singular if and only if* $\Delta \neq 0$.

***Proof When Char(K) Is Not 2 or 3***
Suppose Char($K$) is neither 2 nor 3 for a given field $K$, and

$$W: \quad y^2 = x^3 + \cancel{A}x + \cancel{B},$$

where $(x, y) \in \mathbb{R}^2$ are coordinates on the map $\mathbb{R}^2$, $(A, B) \in \overline{K}^2$, and set

$$f(x, y) = y^2 - x^3 - Ax - B.$$

Suppose $W$ is singular at $(x_0, y_0) \in \mathbb{R}$. Therefore,

$$2y_0 = 3x_0^2 + \cancel{A} = 0.$$

If $A > 0$, the above equation is impossible, hence $W$ is non-singular and $\Delta > 0$. Thus, suppose that $A \leq 0$ from now. It turns out that, since $f(x_0, y_0) = 0$, then

$$\frac{(-A)^{\frac{3}{2}}}{3^{\frac{3}{2}}} - \frac{(-A)^{\frac{3}{2}}}{\sqrt{3}} + \cancel{B} = 0 \iff \Delta = 0.$$

Conversely, if $\Delta = 0$, then we set $x_0 \in \mathbb{R}$ such that $3x_0^2 + A = 0$. Then the point $(x_0, 0)$ is a singular point of $W$.

There remains to see if the point at infinity $O$ is singular. We need to come back to the most generic form, i.e.,

$$W: \quad Y^2Z + \cancel{a_1}XYZ + \cancel{a_3}YZ^2 = X^3 + \cancel{a_2}X^2Z + \cancel{a_4}XZ^2 + \cancel{a_6}Z^3,$$

with

$$f(X, Y, Z) = Y^2Z + a_1XYZ + \cancel{a_3}YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a_6Z^3.$$

We have

$$\frac{\partial f}{\partial Z}(0, 1, 0) = 1 \neq 0,$$

therefore $O$ is a non-singular point. ∎

For the Bitcoin's elliptic curve, $A = 0$ and $B = 7$, so that $\Delta = -21168 \neq 0$. Thus, this curve is non-singular.

## 3 The Underlying Abelian Group

The elliptic curve $(W, O)$ can be endowed with an internal operation law $\bigoplus$. Suppose $L \subset \mathbb{R}^2$ be a line that intersects the elliptic curve. There are at most *three* distinct points of intersection, because $W$ is of degree 3.

***Definition 3.3.1*** *Let $P, Q \in \not{W}$ be two points of the elliptic curve, and let $L \subset \mathbb{R}^2$ be the line defined by $P$ and $Q$ (if $P = Q$, as $W$ is smooth, $L$ is the tangent line). Let $R$ be the third point of intersection – if it exists, otherwise set $R = O$ – and finally, let $L'$ be the line through $R$ and $O$ (if $R = O$, then $L'$ is the vertical line through $O$). Then $L' \cap W = \{R, O, S\}$, where the third point is written $S = P \bigoplus Q$.*

Figure 3.2 is showing illustrations of this rule thus defined.

***Theorem 3.3.2*** *The rule given by Definition 3.3.1 makes the pair $((W, O), \bigoplus)$ being an Abelian group, with neutral element $O$.*

***Proof***
Let $L$ be a line that intersects $W$ such that $L \cap \not{W} = \{P, Q, R\}$, where the points $P$, $Q$, $R$ are not necessarily distinct. Our first observation consists in notifying that

$$\left(P \bigoplus Q\right) \bigoplus R = O. \qquad (*) \leftarrow$$

Indeed, the points $R$ and $P \bigoplus Q$ are vertically aligned, which therefore shows that the third point of the vertical line is $O$. We obviously have $P \bigoplus Q = Q \bigoplus P$, since considering Definition 3.3.1, we have $L' \cap \not{W} = \{R, O, P \bigoplus Q\} = \{R, O, Q \bigoplus P\}$ (change order of $P$ and $Q$), hence the equality.

We check every three properties for a group to satisfy.

– *Neutral element*

Let $P \in \not{W}$. From Definition 3.3.1, we set $Q = O$. Thus, the line $L$ necessarily is vertical. Therefore, $L = L'$, and since $L \cap W = \{R, O, P\}$ and $L' \cap \not{W} = \{R, O, P \bigoplus O\}$, we conclude that $P \bigoplus O = P$. We identically have $O \bigoplus P = P$.

– *Inverse*

If we use $(*)$ and that $P \bigoplus O = P$, then we have

$$\left(P \bigoplus O\right) \bigoplus R = P \bigoplus R = O.$$

Now using commutativity, we have $P \bigoplus R = R \bigoplus P = O$, which proves that $R$ is the inverse of $P$. We write $-P$ such inverse.

– *Associativity*

This point is by far the toughest. We skip this for the moment. ∎

In Fig. 3.2, we have plotted on the right-hand side the third point as being $P \bigoplus P$. This and the previous theorem motivate the following definition.

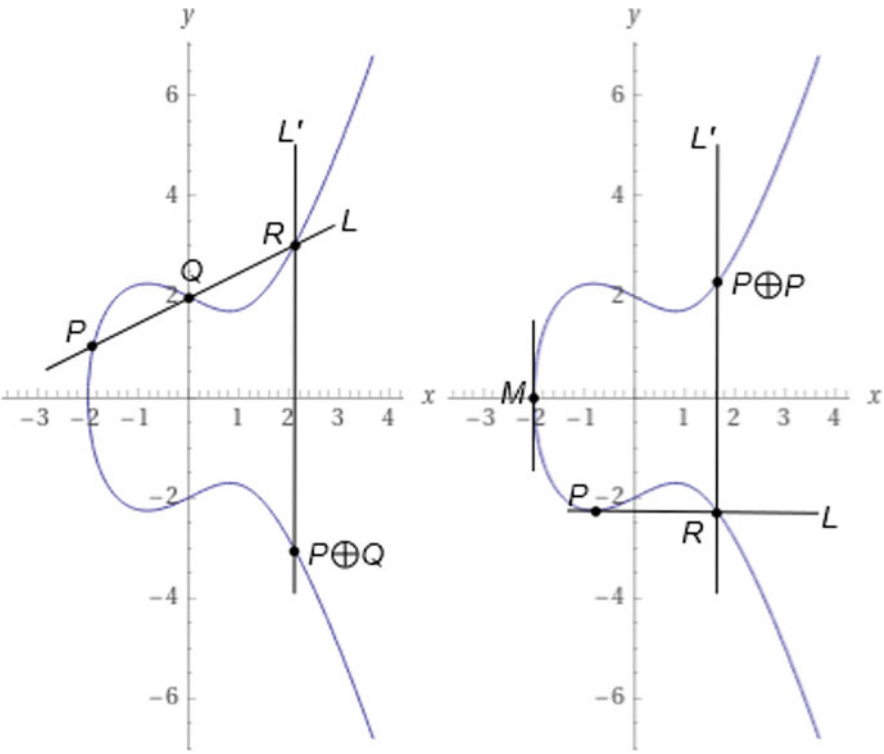


**Fig. 3.2** Example of the rule defined in Definition 3.3.1, on the elliptic curve of equation $y^2 = x^3 - 2x + 4$. The point $M$ is the only one such that $P = Q = R$. (Source: Julien Riposo, 2023)

***Definition 3.3.3*** *Let $P \in \mathcal{W}$ be a point of the elliptic curve. We define the* scalar multiplication *of $P$ with the* natural *integer $n \in \mathbb{N}^*$ as $n\,P = P \bigoplus \cdots \bigoplus P$, $n$ times. If $n = 0$, we set $0\,P = O$. If $n \in (-\mathbb{N})^*$, we set $n\,P = (-P) \bigoplus \cdots \bigoplus (-P)$, $-n$ times.*

We establish analytical rules for this law thus defined in terms of coordinates on the curve.

***Proposition 3.3.4*** *Let*

$$W: \quad y^2 = x^3 + \mathcal{A}x + \mathcal{B}$$

*and $P, Q \in \mathcal{W}$ with coordinates $(x_0, y_0)$ and $\left(x_0^{\prime\leftarrow}, y_0^{\prime\leftarrow}\right)$, respectively. Then:*

(a) *$-P$ has coordinates $(x_0, -y_0)$;*
(b) *If $x_0 = x_0^{\prime\leftarrow}$ and $y_0 = -y_0^{\prime\leftarrow}$, then $P + Q = O$. Otherwise set*

$$s = \begin{cases} \dfrac{y_0^{\prime\leftarrow} - y_0}{x_0^{\prime\leftarrow} - x_0} & \text{if } x_0 \neq x_0^{\prime\leftarrow} \\ \dfrac{3x_0^2 + \mathcal{A}}{2y_0} & \text{if } x_0 = x_0^{\prime\leftarrow} \end{cases}$$

and

$$l = \begin{cases} \dfrac{y_0 x_0' - y_0' x_0}{x_0' - x_0} & \text{if } x_0 \neq x_0' \\ \dfrac{-x_0^3 + Ax_0 + B}{2y_0} & \text{if } x_0 = x_0' \end{cases}$$

*Then $P + Q$ has coordinates $(x_0'', y_0'') = (s^2 - x_0 - x_0', -s\,x_0'' - l)$.*

***Proof***

(a) From Definition 3.3.1, we take the vertical line through $P$ and $O$, which is the line given by $x = x_0$. Thus, $-P$ has coordinates $(x_0, y)$, with $y$ to be determined. Actually, we just note that $(x_0, y_0)$ is solution of $y^2 - (x^3 + Ax + B) = 0$ if and only if $(x_0, -y_0)$ also is solution. Since the polynomial function $y \mapsto y^2 - (x^3 + Ax + B)$ is of degree 2 (in $y$), we conclude that $-P$ has coordinates $(x_0, -y_0)$ (also note that $y \mapsto y^2 - (x^3 + Ax + B)$ is a symmetric function).

(b) *Suppose $x_0 = x_0'$ and $y_0 = -y_0'$. Then we come back to point (a), so that $Q = -P$ and $P + Q = O$. Suppose $x_0 \neq x_0'$ or $y_0 \neq -y_0'$. The line $L$ through $P$ and $Q$ has a parametric equation of the form $y = s\,x + l$.*

*We note that the polynomial $x \mapsto (s\,x + l)^2 - (x^3 + Ax + B)$ is of degree 3, therefore the three solutions are $x_0$, $x_0'$, and $x_0''$. It is worth pointing out that*

$$(s\,x + l)^2 - (x^3 + Ax + B) = -(x - x_0)(x - x_0')(x - x_0'').$$

*This allows to write*

$$x_0'' = s^2 - x_0 - x_0'$$

*Therefore, using point (a), we deduce that $y_0'' = -(s\,x_0'' + l)$, and we obtain the desired results.*

*Now, the expression for $s$ and $l$ are straightforward when $x_0 \neq x_0'$.*

*Suppose $x_0 = x_0'$. The case $y_0 \neq -y_0'$ is the only to check.*

*Thus, $y_0 = y_0'$ (since $P, Q \in W$), and the slope $s$ is given by the tangent at point $(x_0, y_0)$.* ∎

***Sketch of Proof for the Associativity***

Set $P_1$, $P_2$, $P_3 \in W$, with coordinates $(x_1, y_1)$, $(x_2, y_2)$, and $(x_3, y_3)$, respectively, and supposed all distinct. We follow the proof in [4] (presented differently though), which uses the software *Mathematica* to perform complicated analytical calculations.

The idea is to calculate the coordinates of the point $(P_1 \bigoplus P_2) \bigoplus P_3 = (C, D)$ on the one hand and $P_1 \bigoplus (P_2 \bigoplus P_3) = (F, G)$ on the other hand, and compare.

We are going to abundantly use Proposition 3.3.4.b. We more specifically set $s(P, Q)$ and $l(P, Q)$, the slope and intercept of the line defined by points $P$, $Q$.

According to Proposition 3.3.4.b, we have

$$C = \left(\frac{y_3 - l(P_1,P_2)}{x_3 - s(P_1,P_2)}\right)^2 - (\overleftarrow{s}(P_1,P_2) + x_3) \quad \leftarrow \qquad \leftarrow$$

$$D = -\left(\frac{y_3 - l(P_1,P_2)}{x_3 - s(P_1,P_2)}\right)^3 + \cancel{\left(\frac{y_3 - l(P_1,P_2)}{x_3 - s(P_1,P_2)}\right)}(s(\overleftarrow{P_1,P_2}) + x_3) \leftarrow \leftarrow$$
$$+ \frac{s(P_1,P_2)y_3 - l(P_1,P_2)x_3}{x_3 - s(P_1,P_2)}. \leftarrow$$

Identically, we have

$$F = \left(\frac{y_1 - l(P_2,P_3)}{x_1 - s(P_2,P_3)}\right)^2 - (\overleftarrow{s}(P_2,P_3) + x_1) \quad \leftarrow \qquad \leftarrow$$

$$G = -\left(\frac{y_1 - l(P_2,P_3)}{x_1 - s(P_2,P_3)}\right)^3 + \cancel{\left(\frac{y_1 - l(P_2,P_3)}{x_1 - s(P_2,P_3)}\right)}(s(\overleftarrow{P_2,P_3}) + x_1) \leftarrow \leftarrow$$
$$+ \frac{s(P_2,P_3)y_1 - l(P_2,P_3)x_1}{x_1 - s(P_2,P_3)}. \leftarrow$$

So far so good, but here is the trick: $C - F$ and $D - G$ have the common polynomial factor $\mathcal{F}$ given by

$$\mathcal{F} = (x_2 - x_3)y_1^2 + \cancel{(x_3 - x_1)y_2^2} + \cancel{(x_1 - x_2)y_3^2} + \cancel{(x_1 - x_2)(x_2 - x_3)(x_3 - x_1)} \times (x_1 + \cancel{x_2} + \cancel{x_3}).$$

It is therefore sufficient to prove that $\mathcal{F} = 0$.
Here is the other trick: to focus on the matrix determinant $d$ given by

$$d = \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1^2 & y_2^2 & y_3^2 \end{vmatrix}.$$

On the one hand, note that

$$\begin{aligned} d &= x_2y_3^2 + \cancel{x_3}y_1^2 + \cancel{x_1}y_2^2 - x_2y_1^2 - x_1y_3^2 - x_3y_2^2 \\ &= -\left[(x_2 - x_3)y_1^2 + \cancel{(x_3 - x_1)}y_2^2 + (x_1 - x_2)y_3^2\right]. \end{aligned}$$

On the other hand, using that

$$\begin{cases} y_1^2 = x_1^3 + \cancel{A}x_1 + \cancel{B} \\ y_2^2 = x_2^3 + \cancel{A}x_2 + \cancel{B} \\ y_3^2 = x_3^3 + \cancel{A}x_3 + \cancel{B} \end{cases},$$

we have, by replacing

$$d = \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ x_1^3 + Ax_1 + \cancel{B} & x_2^3 + \cancel{A}x_2 + \cancel{B} & x_3^3 + \cancel{A}x_3 + \cancel{B} \end{vmatrix},$$

and, by multi-linearity of the determinant map, we have

$$d = \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ x_1^3 & x_2^3 & x_3^3 \end{vmatrix} + \cancel{A} \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ x_1 & x_2 & x_3 \end{vmatrix} + B \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ 1 & 1 & 1 \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ x_1^3 & x_2^3 & x_3^3 \end{vmatrix} + \cancel{0} + \cancel{0}$$
$$= \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ x_1^3 & x_2^3 & x_3^3 \end{vmatrix}.$$

Now, by taking the differences of columns 2 and 3 with 1, we have

$$d = \begin{vmatrix} 1 & 0 & 0 \\ x_1 & x_2 - x_1 & x_3 - x_1 \\ x_1^3 & x_2^3 - x_1^3 & x_3^3 - x_1^3 \end{vmatrix}.$$

Next, we note that $x_2^3 - x_1^3 = (x_2 - x_1)\left(x_2^2 + \cancel{x_2}x_1 + x_1^2\right)$ and $x_3^3 - x_1^3 = (x_3 - x_1)\left(x_3^2 + \cancel{x_3}x_1 + \cancel{x_1^2}\right)$; thus, by factorizing (i.e., again using multi-linearity):

$$d = (x_2 - x_1)(x_3 - \cancel{x_1}) \begin{vmatrix} 1 & 0 & 0 \\ x_1 & 1 \leftarrow & 1 \\ x_1^3 & x_2^2 + \cancel{x_2}x_1 + x_1^2 & x_3^2 + x_3x_1 + \cancel{x_1^2} \end{vmatrix}.$$

Now take the difference of the last column with the second:

$$d = (x_2 - x_1)(x_3 - \cancel{x_1}) \begin{vmatrix} 1 & 0 & 0 \\ x_1 & 1 \leftarrow & 0 \\ x_1^3 & x_2^2 + \cancel{x_2}x_1 + x_1^2 & (x_3 - x_2)(x_3 + \cancel{x_2} + \cancel{x_1}) \end{vmatrix},$$

to finally obtain

$$d = (x_2 - x_1)\cancel{(}(x_3 - x_1\cancel{)}\cancel{(}x_3 - x_2\cancel{)}\cancel{(}x_3 + \cancel{x_2} + \cancel{x_1}\cancel{)}.$$

It appears that $\mathcal{F} = d - d$, hence $\mathcal{F} = 0$, which, up to some skipped details, concludes the proof. Finally, the pair $((W, O), \bigoplus)$ *is an Abelian group*. ∎

***Corollary 3.3.5*** *Let K be a field. The pair* $((W_K, O), \bigoplus)$*, where*

$$W_K : y^2 = x^3 + \cancel{A}x + \cancel{B}$$

*with* $(x, y) \in K^2$ *is a subgroup of* $((W, O), \bigoplus)$*, where* $\mathbb{R}$ *is the considered field for W.*

***Proof***
Suppose $P, Q \in \cancel{W_K}$. Then equation of $L$ has coefficients in $K$. Bearing in mind that $L \cap W_K = \{P, Q, R\}$, it turns out that the coordinates of $R$ are rational function of elements in $K$ (i.e., the coordinates of $P$ and $Q$). In short, the points $P$, $Q$ and $R$ have their coordinates in $K$, which implies that $(W_K, O)$ is stable by $\bigoplus$, hence $((W_K, O), \bigoplus)$ is a subgroup of $((W, O), \bigoplus)$. ∎

When $F = \mathbb{Z}/q\mathbb{Z}$, with $q = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$ (prime number), $A = 0$, and $B = 7$, the subgroup $((W_{\mathbb{Z}/q\mathbb{Z}}, O), \bigoplus)$ is named secp256k1 (used for Bitcoin).

## 4 Elliptic Curve Digital Signature Algorithm (ECDSA) and Schnorr Signature

We are equipped to explain the ECDSA for Bitcoin. Generally speaking, a digital signature is a proof of knowledge of a private key, without revealing it to the public. In essence, the ECDSA is a particular Digital Signature Algorithm (DSA), but the considered group is $((W_{\mathbb{Z}/q\mathbb{Z}}, O), \bigoplus)$, where

$$W_{\mathrm{B}} : \quad y^2 \equiv x^3 + \cancel{7}[q].$$

This curve is non-singular, as we have seen, since the discriminant is such that $\Delta \neq 0$. We will need the following definitions.

***Definition 3.4.1*** *Let* $x \in \cancel{\mathbb{Z}}/q\mathbb{Z}$ *where q is a prime number. Let G be a generator of the group* $((W_{\mathbb{Z}/q\mathbb{Z}}, O), \bigoplus)$*. The* public key *associated with x, thus named* private key *in this context, is given by*

$$y = x\,G.$$

The element "$x\,G$" is the scalar multiplication of $x$ with $G$ (see Definition 3.3.3). It is worth pointing out that the private key generates the public key used for the Script Pub Key to form the output, see Definition 2.4.1. Finding $x$, given $y$ and $G$, is a discrete logarithm problem (which is an NP problem).

In the context of Bitcoin, the SEC recommendations for the choice of $G$ is

$$G = \begin{pmatrix} 55066263022277343669578718895168534326 \\ 250603453777594175500187360389116729240, \\ 32670510020758816978083085130507043184 \\ 471273380659243275938904335757337482424 \end{pmatrix}.$$

The order of $G$ is

$$p = 115792089237316195423570985008687907852$$
$$837564279074904382605163141518161494337.$$

As written above, the number $q$ is given by:

$$q = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1.$$

The following protocol gives the ECDSA.

***Protocol 3.4.2 (ECDSA)*** *For a given private key $x \in \mathbb{Z}/q\mathbb{Z}$, the public key $y = x\,G$ is generated from the secp256k1 group. Given a message $m \in \mathbb{N}$, the ECDSA is performed as follows:*

1. *Generate the random number $r \in (\mathbb{Z}/q\mathbb{Z})^*$ and calculate $P = r\,G = (p_X, p_Y)$.*
2. *Calculate the hash of $m$, giving $z = H(m)$, and the number $\alpha \equiv r^{-1}\,(H(m) + p_X x)\,[q]$. If $\alpha > q/2$, then make the transformation $\alpha \mapsto q - \alpha$ (low $\alpha$ value). Note that, using the little Fermat theorem $r^{q-1} \equiv 1\,[q]$, we have $r^{-1} \equiv r^{q-2}\,[q]$. This is a practical way to calculate the inverse on $(\mathbb{Z}/q\mathbb{Z})^*$. The pair $(r, \alpha)$ thus formed is called* signature.
3. *Verification: the signature is verified if the x-axis coordinate $e$ of the point $z\,G + P_X\,y$ is equal to the one of $\alpha\,P$.*

Figure 3.3 is an illustration of the process.

We justify the verification step 3 above.

$$r\,(z\,G + p_X\,y) = z\,r\,G + p_X\,x\,r\,G = (z + p_X\,x)\,P, \qquad \leftarrow$$

hence

$$z\,G + p_X\,y = \frac{z + p_X\,x}{r}\,P = r^{-1}\,(H(m) + p_X x)\,P,$$

thus

$$e \equiv \alpha\,p_X\,[q],$$

which terminates the verification.

We are now giving another protocol, which has the virtue to be faster and quantum resistant.

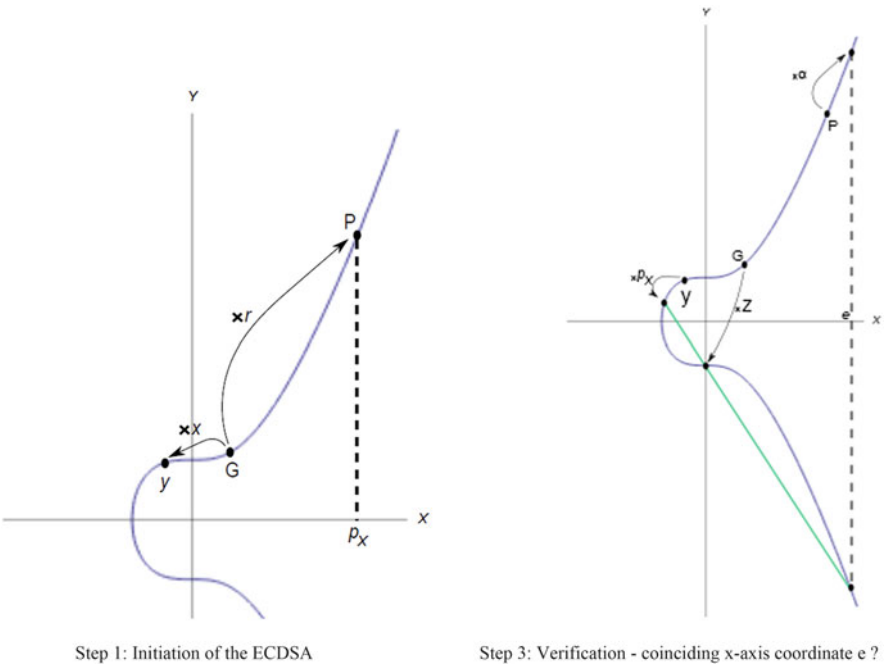


**Fig. 3.3** Illustration of the ECDSA. (Source: Julien Riposo, 2023)

***Protocol 3.4.3 (Schnorr Signature)*** *For a given private key* $x \in \mathbb{Z}/q\mathbb{Z}$*, the public key* $y = x\,G = (y_X, y_Y)$ *is generated from the secp256k1 group. Given a message* $m \in \mathbb{N}$*, the Schnorr Signature is performed as follows:*

1. *Generate the random number* $L \in (\mathbb{Z}/q\mathbb{Z})^*$ *and calculate* $Q = L\,G = (Q_X, Q_Y)$.
2. *Calculate the hash of concatenated information, including the message m, as follows:*

$$h = H\left(y_X \middle\| y_Y \middle\| Q_X \middle\| Q_Y \middle\| m\right).$$

*Then calculate*

$$\beta \equiv (L + hx)[q].$$

*The pair* $(L, \beta)$ *thus formed is called* Schnorr Signature.

3. *Verification: the signature is verified if* $Q + hy$ *is equal to* $\beta G$.

We justify the verification step 3 above.

$$Q + hy = LG + hxG = (L + hx)G = \beta G,$$

which terminates the verification.

One interesting application of the Schnorr protocol is multi-signature: it is possible to enhance one signature process for several private keys. This is very practical if a customer of a crypto-exchange has several associated private keys. In particular, the following protocol is a multi-signature protocol quite efficient against attacks.

***Protocol 3.4.4 (Schnorr MuSig Signature)*** *For given private keys $x_1, \ldots, x_l \in \mathbb{Z}/q\mathbb{Z}$, the public keys $y_i = x_i G$ are generated from the secp256k1 group, $i \in \{1, \ldots, l\}$. Given a message $m \in \mathbb{N}$, the Schnorr MuSig Signature is performed as follows:*

1. *Generate the random variables $L_i \in (\mathbb{Z}/q\mathbb{Z})^*$ and calculate $Q_i = L_i G$.*
2. *Calculate the sum*

$$Q = \sum_{i=1}^{l} Q_i,$$

*and the aggregated public key*

$$y = \sum_{i=1}^{l} \alpha_i y_i, \quad \alpha_i = H(m \| y_i), \quad i \in \{1, \ldots, l\}.$$

3. *Calculate the hash of concatenated information, including the message m, as follows:*

$$h = H\left(y_X \middle\| y_Y \middle\| Q_X \middle\| Q_Y \middle\| m\right).$$

*Then calculate, for all $i \in \{1, \ldots, l\}$*

$$\beta_i \equiv L_i + h\alpha_i x_i [q].$$

*Setting*

$$\beta = \sum_{i=1}^{l} \beta_i,$$

*the pair $(L, \beta)$ thus formed is called* Schnorr MuSig Signature.
4. *Verification: the signature is verified if $Q + hy$ is equal to $\beta G$.*

We justify the verification step 4 above.

$$Q + hy = \sum_{i=1}^{l} (Q_i + h\alpha_i y_i) = \sum_{i=1}^{l} (L_i + h\alpha_i x_i) G = \left( \sum_{i=1}^{l} (L_i + h\alpha_i x_i) \right) G = \beta G,$$

which terminates the verification.

It is worth stressing that a multi-signature is impossible for the ECDSA, because it is not a *linear* protocol, precisely because of the factor $r^{-1}$ in the calculation of $\alpha$. The justification above for the MuSig verification uses the linearity of the Schnorr protocol. For this reason, also the Schnorr protocol shows more practicality than the ECDSA. In addition, it is known to be particularly fast when aggregating, and can increase scalability.

## 5 Back to the Blockchain

According to Definition 2.4.2, an input is given by the following hash:

$$\text{Input}_{l,k} = H\left(\overleftarrow{\text{TXID}_{l'}} \middle\| \overleftarrow{k'} \middle\| \text{ScriptSig}_{l,k}\ \text{Size} \middle\| \text{ScriptSig}_{l,k} \middle\| \overleftarrow{\text{Seq}_{l,k}}\right),$$

where, in particular, $k \in \{1, \ldots, K\}$ for $K \geq 1$ inputs and $l \in \{1, \ldots, l_n\}$ with $l_n \geq 1$ while we are focusing on $\text{TXID}_l$, i.e., the $l^{\text{th}}$ element of the transaction list $T_n$ for block $n$.

The following process needs to be repeated for all $k \in \{1, \ldots, K\}$.

Before the signature, the numbers $\text{ScriptSig}_{l,\,k}$ Size and $\text{ScriptSig}_{l,\,k}$ are empty and the goal is to have an adequate value for them. If $\text{ScriptSig}_{l,\,k}$ is not empty, then $\text{ScriptSig}_{l,\,k}$ Size is deduced automatically. Thus, our attention focuses on $\text{ScriptSig}_{l,\,k}$.

A standard approach consists as follows. By bearing in mind that the goal is to perform a transaction, i.e., construct a hash as depicted in Definition 2.4.4, it is possible to obtain a *temporary* value for $\text{ScriptSig}_{l,\,k}$, which is given by:

$$\text{ScriptSig}_{l,k}^{\text{Temp}} = H\left(\mathop{\Big\|}_{k'=1}^{K'\leftarrow} \text{ScriptPubKey}_{l,\,k'}\right),$$

that is the hash of the concatenation of all the outputs' public keys. The number $\text{ScriptSig}_{l,k}^{\text{Temp}}$ Size is deduced.

Then, we set the message as

$$m_{l,k} = \overleftarrow{\text{TXID}_{l'}} \| \overleftarrow{k'} \| \text{ScriptSig}_{l,k}^{\text{Temp}}\ \text{Size} \| \text{ScriptSig}_{l,k}^{\text{Temp}} \| \overleftarrow{\text{Seq}_{l,k}},$$

and its hash as

$$z_{l,k} = H(m_{l,k}) = H\left(\overleftarrow{\text{TXID}_{l'}} \| \overleftarrow{k'} \| \text{ScriptSig}_{l,k}^{\text{Temp}}\ \text{Size} \| \text{ScriptSig}_{l,k}^{\text{Temp}} \| \overleftarrow{\text{Seq}_{l,k}}\right).$$

Protocol 3.4.2 is applied (bearing in mind all the notations therein), leading to the signature $(r_{l,\,k}, \alpha_{l,\,k})$ from the public key $y_{l,k} = \left(y_{l,k_X}, y_{l,k_Y}\right)$ generated by the private key of the considered input. After conversion of this signature by the *DER encoding* (basically obtaining the numbers $r_{l,\,k}$, $\alpha_{l,\,k}$ into hexadecimal format), we finally set $\mathrm{Len}_{l,\,k}$ as the length of $r_{l,\,k} \| \alpha_{l,\,k}$, also $\mathrm{Len}^{y}_{l,k}$ as the length of $y_{l,k_X} \| y_{l,k_Y}$, and last but not least:

$$\mathrm{ScriptSig}_{l,k} = \mathrm{Len}_{l,k} \| r_{l,k} \| \alpha_{l,k} \| \mathrm{Len}^{y}_{l,k} \| y_{l,k_X} \| y_{l,k_Y},$$

which also allows to get the permanent $\mathrm{ScriptSig}_{l,\,k}$ Size.

It turns out that the $k^{\mathrm{th}}$ input is fully constructed, as well as the required transaction script.

# Chapter 4
# Blockchain Contributors: The Network of Users


In this chapter, we focus on the communication between users and nodes through the Network of Users. Each user has connections with some others, and this allows exchange of information. A particular type of information concerns transaction, thus exchange of cash, or transaction lists between all the miners. These exchanges of information can be modeled by means of Graph and Matrix Theories. Papers [26, 27] are giving applications of the below theorization.

## 1 Preliminaries on Graph and Matrix Theories

### *1.1 Notations and Facts*

A graph $G = (V, E)$ is a pair of two sets. The first set $V$ is a countable set, named the set of *vertices*. We write $V = \{1, 2, \ldots, n\} \subset \mathbb{N}^*$, for some $n \in \mathbb{N}^*$. The second set $E$ is a countable set of pairs of elements in $V$, which we name set of *edges* (a pair of vertices is an edge).

From this graph, we can derive a square matrix $M = (m_{ij})_{1 \leq i,\, j \leq n}$, named *adjacency matrix* associated with graph $G$, and such that

$$m_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

We could generalize by *weighting* the edges, i.e., $m_{ij}$ can take any value, and is therefore named *weight*.

The *degree* of node $i \in V$ is given by



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_4

$$d_i = \text{Card}(j \in V, (i,j) \in E) = \sum_{j=1}^{n} m_{ij}.$$

The sum of all the elements of $M$ is the sum of all the connections, that is

$$\sum_{i,j=1}^{n} m_{ij} = \text{Card}(E).$$

When the graph $G$ is undirected, then $(i,j) \in E$ if and only if $(j,i) \in E$ (in particular $m_{ij} = m_{ji}$), thus $(i,j)$ is not counted twice in $E$. Thus,

$$\sum_{i,j=1}^{n} m_{ij} = 2\ \text{Card}(E).$$

An *eigenvalue* of $M$ (and thus of $G$) is a number $\lambda \in \mathbb{C}$ such that the matrix $M - \lambda\,\mathbb{I}$ is not injective ($\mathbb{I}$ is the identity matrix). Thus, there exists $v \in \mathbb{C}^n \backslash \{0_{\mathbb{C}^n}\}$ such that $Mv = \lambda v$. The vector $v$ is an *eigenvector* associated with the eigenvalue $\lambda$. There are $n$ eigenvalues (identical or not) and we call $\text{Sp}(M) = \{\lambda_1, \ldots, \lambda_n\}$ the spectrum of $M$. The eigenvalues are the solutions of the *characteristic polynomial* of $M$, given by

$$\chi_M(X) = \det(M - X\,\mathbb{I}),$$

where det is the *determinant* map. The *Cayley-Hamilton* theorem states that, when $\chi_M$ is also a matrix map, then $\chi_M(M) = 0$.

It is worth pointing out that if the matrix $M$ is real symmetric (i.e., $m_{ij} = m_{ji} \in \mathbb{R}$ for any $i,j \in \{1,\ldots,n\}$), then the eigenvalues are all real (we thus set $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$) and $M$ is $\mathbb{R}$-diagonalizable. If $v_i$ is the eigenvector associated with the eigenvalue $\lambda_i$, the *spectral decomposition* of $M$ is given

$$M = \sum_{i=1}^{n} \lambda_i\ v_i\ v_i^{\mathrm{T}},$$

where $v_i^{\mathrm{T}}$ is the transpose of $v_i$. In addition, the trace of $M$ is given by

$$\text{Tr}(M) \stackrel{\text{def}}{=} \sum_{i=1}^{n} m_{ii} = \sum_{i=1}^{n} \lambda_i$$

and the determinant by

$$\det(M) = \prod_{i=1}^{n} \lambda_i.$$

Another property of the trace is given by the equation

$$\mathrm{Tr}\left(M^k\right) = \sum_{i=1}^{n} \lambda_i^k, \quad \forall k \in \mathbb{N}.$$

Because $M$ is symmetric, we have the *variational theorem*, giving

$$\lambda_1 = \max_{v^{\mathrm{T}} v = 1} \left(v^{\mathrm{T}} M v\right).$$

Finally, if $M$ is positive semi-definite (i.e., for all $x \in \mathbb{R}^n \backslash \{0_{\mathbb{R}^n}\}$, $x^{\mathrm{T}} M x \geq 0$), then all the eigenvalues are non-negative (and conversely) and we thus set $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$.

Suppose now that there are $p \leq n$ distinct eigenvalues. The *Jordan Normal Form theorem* states that there exists $(P, J) \in \mathcal{GL}_n(\mathbb{C}) \times \mathcal{M}_n(\mathbb{C})$ such that $M = PJP^{-1}$, where

$$J = \begin{pmatrix}
\lambda_1 & 1 & 0 & \cdots & 0 & & & & & & & & & & \\
0 & \lambda_1 & 1 & \cdots & 0 & & & & & & & & & & \\
0 & 0 & \lambda_1 & \cdots & 0 & & & 0 & & & \cdots & & & 0 & \\
\vdots & \vdots & \vdots & \ddots & \vdots & & & & & & & & & & \\
0 & 0 & 0 & \cdots & \lambda_1 & & & & & & & & & & \\
 & & & & & \lambda_2 & 1 & 0 & \cdots & 0 & & & & & \\
 & & & & & 0 & \lambda_2 & 1 & \cdots & 0 & & & & & \\
 & & 0 & & & 0 & 0 & \lambda_2 & \cdots & 0 & \cdots & & & 0 & \\
 & & & & & \vdots & \vdots & \vdots & \ddots & \vdots & & & & & \\
 & & & & & 0 & 0 & 0 & \cdots & \lambda_2 & & & & & \\
 & & \vdots & & & & & \vdots & & & \ddots & & \vdots & & \\
 & & & & & & & & & & & \lambda_p & 1 & 0 & \cdots & 0 \\
 & & & & & & & & & & & 0 & \lambda_p & 1 & \cdots & 0 \\
 & & 0 & & & & & 0 & & & \cdots & 0 & 0 & \lambda_p & \cdots & 0 \\
 & & & & & & & & & & & \vdots & \vdots & \vdots & \ddots & \vdots \\
 & & & & & & & & & & & 0 & 0 & 0 & \cdots & \lambda_p
\end{pmatrix}$$

A *formal series* is of the form $\sum_{i=1}^{+\infty} a_i X^i$, where $(a_i)_{i \in \mathbb{N}}$ is a sequence of $\mathbb{R}$ or $\mathbb{C}$. Without considering convergence in details, the *Cauchy product* of two formal series $\sum_{i=1}^{+\infty} a_i X^i$ and $\sum_{i=1}^{+\infty} b_i X^i$ is given by

$$\left(\sum_{i=1}^{+\infty} a_i X^i\right) \times \left(\sum_{i=1}^{+\infty} b_i X^i\right) = \sum_{i=1}^{+\infty} \left(\sum_{k=1}^{i} a_k \, b_{i-k}\right) X^i.$$

Finally, the *Banach fixed point theorem* states that, on a non-empty complete metric space (metric $d$), then the sequence given by the iteration

$$x_{n+1} = f(x_n)$$

converges to the unique fixed point of $f$ if $f$ is a contraction, i.e., if $d(f(x),f(y)) \leq kd(x,y)$ with $k < 1$.

## 1.2 Equivalence Between the Set of Graphs and the Set of Matrices

We establish an isomorphism of $\mathbb{K}$-modules (see Sect. 1.2, Chap. 3 for definition of a $\mathbb{K}$-module) between the set of graphs with weights in $\mathbb{K}$ and the set of matrices with elements in $\mathbb{K}$, where $\mathbb{K}$ is a division ring. This proves that treating the problem through graphs is the same as treating the problem through matrices; thus, for one derived property in matrices, the same property appears into the equivalent graph language, and conversely.

Let $\mathbb{K}$ be a division ring and $n \in N^*$. We denote by $\mathcal{G}_n(\mathbb{K})$ by the set of (directed) graphs whose weights are elements in $\mathbb{K}$, and $\mathcal{M}_n(\mathbb{K})$ the set of square matrices of size $n$ whose elements are in $\mathbb{K}$. We want to prove that there exists an isomorphism of $\mathbb{K}$-module between both sets. We suppose known the fact that $\mathcal{M}_n(\mathbb{K})$, endowed with the usual addition and Cauchy product, is a $\mathbb{K}$-module. We first construct a specific addition and Cauchy product to endow $\mathcal{G}_n(\mathbb{K})$ with, before constructing an isomorphism of $\mathbb{K}$-module.

***Definition 4.1.1 (Addition of Graphs)*** *Let $G = (V,E) \in \mathcal{G}_n(\mathbb{K})$, with $V = \{u_1, \ldots, u_n\}$ the set of vertices of G, and E the set of pairs of vertices, or edges. Let $G' = (V,E') \in \mathcal{G}_n(\mathbb{K})$, with $E'$ another (or not) set or edged. We write "+" the addition of G and $G'$ as*

$$G + G' = (V, E \cup E').$$

*Furthermore, if $(u_i,u_j) \in E \cap E'$, then the final weight is the sum of the weight in graph G and the weight in graph $G'$.*

As an example, let $V = \{u_1,u_2,u_3\}$, $E = \{(u_1,u_2),(u_1,u_3)\}$ with weights $m_{12}$ and $m_{13}$, respectively, and $E' = \{(u_1,u_2),(u_2,u_3)\}$ with weights $m'_{12}$ and $m'_{23}$, respectively. Then $G + G' = \{(u_1,u_2),(u_2,u_3),(u_1,u_3)\}$ with weights $m_{12} + m'_{12}$, $m'_{23}$, and $m_{13}$, respectively.

***Definition 4.1.2 (Cauchy Product of Graphs)*** *Let $G = (V,E) \in \mathcal{G}_n(\mathbb{K})$, with $V = \{u_1, \ldots, u_n\}$ the set of vertices of G, and $E = \{(u_i,u_j)_{(i,j) \in I \times J}\}$ a set of pairs of vertices, with $I, J \subset \{1, \ldots, n\}$. Let $G = (V,E') \in \mathcal{G}_n(\mathbb{K})$, with*

$E' = \left\{ \left(u_i, u_j\right)_{(i,j)\in I' \times J'} \right\}$ *another (or not) set of pairs of vertices, with* $I', J' \subset \{1, \ldots, n\}$. *We write "*$*$*" the Cauchy product of G and* $G'$ *as*

$$G * G' = (V, E * E'),$$

*where* $E * E' = \emptyset$ *if* $I \cap J' = \emptyset$*; otherwise for all* $k \in J \cap I'$ *such that* $(u_i, u_k) \in E$ *and* $(u_k, u_j) \in E'$, *then* $(u_i, u_j) \in E \cap E'$ *for any* $i, j \in I \times J'$, *with weight* $\sum_{k\in J\cap I'} m_{ik} m'_{kj}$.

As an example, we consider the graphs of the previous illustration. Then $E * E' = \{(u_1, u_3)\}$ with weight $m_{12}m_{23}$.

***Proposition 4.1.3*** *Let* $M \in \mathcal{M}_n(\mathbb{K})$ *and* $M' \in \mathcal{M}_n(\mathbb{K})$ *be the adjacency matrices of the graphs G and* $G'$, *respectively. Then the adjacency matrix of* $G + G'$ *is* $M + M'$, *and the adjacency matrix of* $G * G'$ *is* $MM'$.

***Proof***
This immediately follows from the definition above. In particular, regarding the graph Cauchy product, we have

$$(MM')_{ij} = \sum_{k=1}^{n} m_{ik} m'_{kj} = \sum_{k\in J\cap I'} m_{ik} m'_{kj},$$

since $m_{ik} = 0$ if $k \notin J$ and $m'_{kj} = 0$ if $k \notin I'$. For $\lambda \in \mathbb{K}$, we define the external lax $\cdot$ as $\lambda \cdot G$, which is the same graph as $G$ but the weights all multiplied by $\lambda$, $\forall G \in \mathcal{G}_n(\mathbb{K})$. ∎

***Proposition 4.1.4*** *Let* $\mathbb{K}$ *be a division ring. The quadruplet* $(\mathcal{G}_n(\mathbb{K}), +, *, \cdot)$ *is a* $\mathbb{K}$ *-algebra.*

In other words, the triple $(\mathcal{G}_n(\mathbb{K}), +, *)$ is a ring, while the triple $(\mathcal{G}_n(\mathbb{K}), +, \cdot)$ is a $\mathbb{K}$-module.

***Proof***
Let $\lambda, \mu \in \mathbb{K}$ and $G, G' \in \mathcal{G}_n(\mathbb{K})$, with a fixed set of vertices $V$. The +-neutral graph clearly is $0_{\mathcal{G}_n(\mathbb{K})} = (V, \emptyset)$ (graph with no edge), so that $G + 0_{\mathcal{G}_n(\mathbb{K})} = 0_{\mathcal{G}_n(\mathbb{K})} + G = G$. In addition, we have $G + (-1) \cdot G = (-1) \cdot G + G = 0_{\mathcal{G}_n(\mathbb{K})}$, so that the pair $(\mathcal{G}_n(\mathbb{K}), +)$ is an Abelian group. We now have

1. $\lambda \cdot (G + G') = \lambda \cdot G + \lambda \cdot G'$;
2. $(\lambda + \mu) \cdot G = \lambda \cdot G + \mu \cdot G$;
3. $(\lambda\mu) \cdot G = \lambda \cdot (\mu \cdot G)$;
4. $1 \cdot G = G$,

so that the triplet $(\mathcal{G}_n(\mathbb{K}), +, \cdot)$ is a $\mathbb{K}$-module. Finally, the operator $*$ is associative since the Cauchy product is also associative. The $*$-neutral graph is $1_{\mathcal{G}_n(\mathbb{K})} = \left(V, \left\{ (u_i, u_i)_{i\in\{1,\ldots,n\}} \right\}\right)$ with weight 1 (one-loops only with weights 1), so

that $1_{\mathcal{G}_n(\mathbb{K})} * G = G * 1_{\mathcal{G}_n(\mathbb{K})} = G$. Thus, the law $*$ is distributive so that the triplet $(\mathcal{G}_n(\mathbb{K}), +, *)$ is a ring. ∎

From the above, the *canonical basis* for $\mathcal{G}_n(\mathbb{K})$ is $\{(V, \{(u_i, u_j)\})_{1 \leq i,j \leq n}\}$, so that the dimension notion on graphs make sense, and $\dim_{\mathbb{K}} \mathcal{G}_n(\mathbb{K}) = n^2$.

We can now "gather" graphs and matrices within the same class of objects (i.e., the same category). We remind the reader that, for any given graph, its adjacency matrix is unique up to permutations of vertices, which is equivalent to conjugate its adjacency matrix with the relevant permutation matrix.

***Theorem 4.1.5*** *We consider the following map:*

$$\Phi : \begin{cases} \mathcal{G}_n(\mathbb{K}) \longrightarrow \mathcal{M}_n(\mathbb{K}) \\ G \longmapsto M \end{cases}$$

*where $\Phi(G) = M$ is the adjacency matrix of the graph G. Then, up to permutations, the $\Phi$ is a $\mathbb{K}$-algebra isomorphism.*

This means that $\Phi$ is a $\mathbb{K}$-module isomorphism and ring isomorphism.

***Proof***

Let $\lambda, \mu \in \mathbb{K}$ and $G, G' \in \mathcal{G}_n(\mathbb{K})$. Note that $\Phi\left(0_{\mathcal{G}_n(\mathbb{K})}\right) = 0_{\mathcal{M}_n(\mathbb{K})}$ and $\Phi\left(1_{\mathcal{G}_n(\mathbb{K})}\right) = 1_{\mathcal{M}_n(\mathbb{K})}$. In addition, by construction of $\cdot$, $\Phi(\lambda \cdot G) = \lambda\, M = \lambda\, \Phi(G)$, but also $\Phi(G + G') = M + M' = \Phi(G) + \Phi(G')$ (see Proposition 4.1.3), which proves that $\Phi$ is a $\mathbb{K}$-module morphism. Next, by construction of $*$ and from Proposition 4.1.3, we immediately have $\Phi(G * G') = MM' = \Phi(G)\Phi(G')$, which proves that $\Phi$ is a ring morphism.

Now consider the map

$$\Psi : \begin{cases} \mathcal{M}_n(\mathbb{K}) \longrightarrow \mathcal{G}_n(\mathbb{K}) \\ N \longmapsto H \end{cases}$$

as follows: let $V = \{1, \ldots, n\}$. We build the set of all the pairs $(i, j)$ such that $n_{ij} \neq 0$, for any $i, j \in \{1, \ldots, n\}$. If $E$ designates this set, then $H = (V, E)$ with weights $(n_{ij})_{1 \leq i,j \leq n}$. It is clear that any permutation conjugation of $N$ will permute $V$ and $E$ accordingly, so that $H$ is unique up to permutations, and $\Psi$ is well defined. Bearing this in mind, we see that $\Phi(H) = N$ and $N$ is the unique adjacency matrix of $H$, which henceforth means that $\Psi \circ \Phi = \mathrm{Id}_{\mathcal{G}_n(\mathbb{K})}$. Namely, we have $\Phi \circ \Psi = \mathrm{Id}_{\mathcal{M}_n(\mathbb{K})}$. In other words, $\Psi = \Phi^{-1}$ up to permutations, i.e., $\Phi$ is bijective up to permutations.

We have proven that $\Phi$ is a $\mathbb{K}$-algebra isomorphism. ∎

In particular, we could construct a Random Matrix Theory from the Random Graph Theory using the $\mathbb{K}$-algebra isomorphism of the proof of Theorem 4.1.5. For instance, random graphs constructed in an Erdos-Reyni style should reflect an infinite-side random matrix construction. Thus, adding rows and columns to a finite-sized matrix now makes sense. Or, solving an infinite-sized Hamiltonian system in a graph language now makes sense as well.

## 2 Bitcoin Network

The purpose of this section is to motivate the needs for a modeling of the full network of users.

The blockchain is constructed by *miners*, which each are individuals who build the content of the block and assembles the full chain. Figure 4.1 is an illustration of the relationship between users, miners, and the blockchain.

More specifically, at least two users are making an agreement, resulting in a flow of cash from one to another, the transaction is "encrypted" as explained in Definitions 2.4.3 and 2.4.4. The transaction is sent to the *memory* (or *mining*) *pool* of a *miner* (yellow node on the graph), which can be seen as a set of processes launched by a computer to mine the blocks. The memory pool is then *diffused* throughout the network to the other miners so that all of them are updated each time a new transaction needs to be mined.

For each miner, the transactions in the mining pool are one by one stored in a *candidate* block, or set of blocks. Each miner is then performing a *proof-of-work*, consisting in solving the problem exposed in Definition 2.2.3, i.e., finding a nonce $N$ such that $H\left(\mathcal{H}_n^{\|}\right) \leq \mathrm{t}_n$ for block $n$. Figure 4.2 is an illustration for this process.

Once a miner has its blocks in the blockchain, it is immediately rewarded thanks to its "wallet coordinates" left as the Coinbase transaction ID $\mathrm{TXID}_1$ on the list of transactions $T_n$ of the block $n$ just mined.

In case two blocks have found a nonce at the same time, the wining miner is the one that has the largest number of candidate blocks: this rule is named *consensus*.

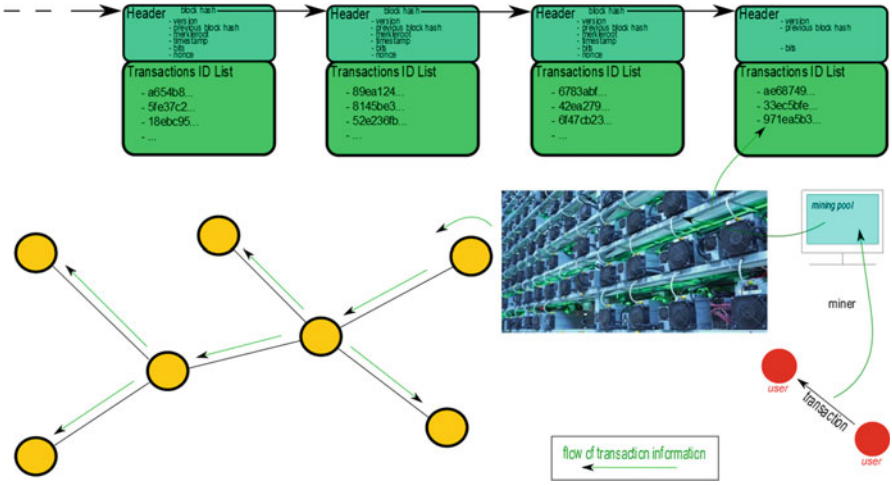


**Fig. 4.1** Network of users and relationship with the blockchain. (Source: Julien Riposo, 2023)

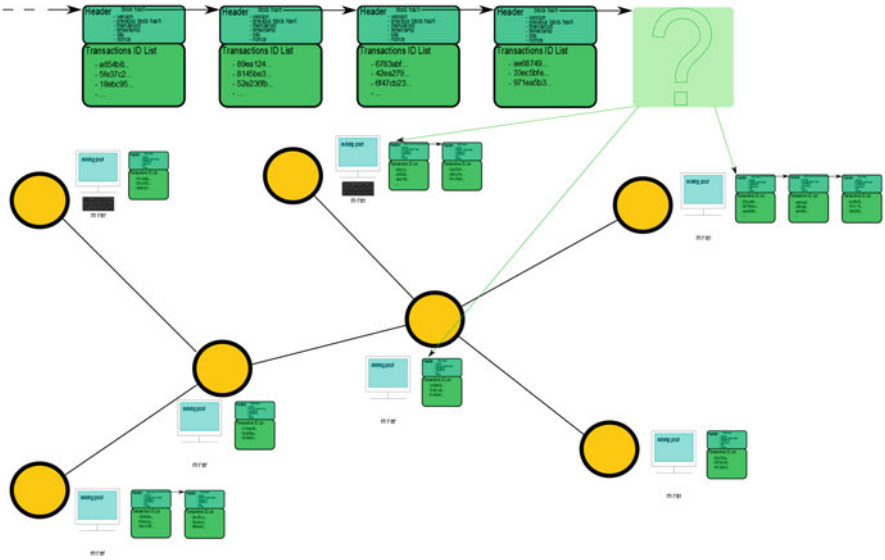


**Fig. 4.2** Mining competition. Each miner is having its own candidate block, or set of blocks. The first miner to solve the proof-of-work problem has its candidate blocks on the blockchain. (Source: Julien Riposo, 2023)

The other candidate blocks are immediately thrown, unless a losing miner believes he can construct a longer chain than that already on the blockchain. This could result in a *51% attack*: a miner can change the past of part of the blockchain, provided it has more blocks than the ones on the blockchain from a certain moment in time. This means that the transactions that are not in the new coming blocks are directly cancelled, as changing a block into another changes the history of transactions. However, a unique miner is very likely to spend a lot of time and energy for a 51% attack, and the longer the process, the more energy does it needs to set its blocks.

Once a block or set of blocks is mined, then the miner diffuses the information to the rest of the network, and gets rewarded by (1) the *transaction fees* for each transaction related to the mined block (see Fig. 2.4) and (2) the *block reward*. Figure 4.3 is an illustration for this process.

The time to add a block is modeled in Sect. 2.2, Chap. 2.

From the above, it is worth stressing that a certain number of types of information is diffusing/propagating through the whole network. Thus, questions like first passage time, shortest paths, and usual optimality procedures are of relevant matter. The paper [26] studies the distribution of degrees among several cryptocurrency blockchains.

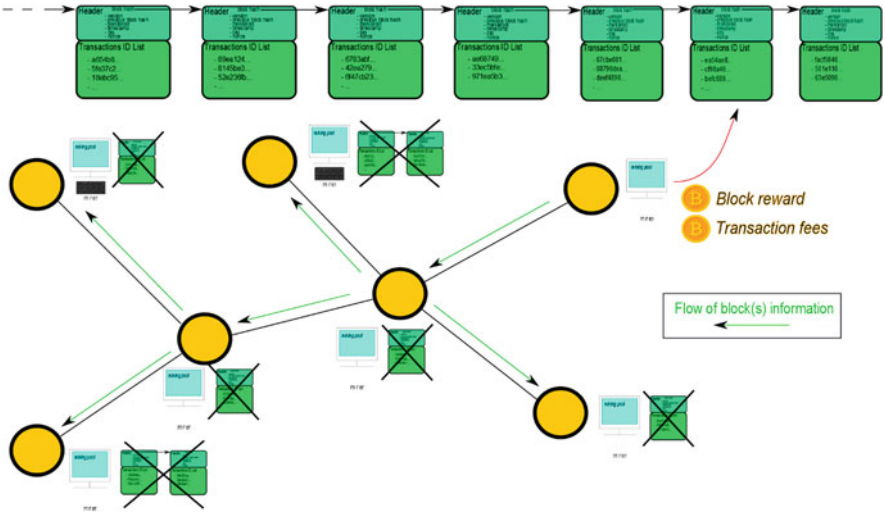


**Fig. 4.3** The wining miner is the one that solves the proof-of-work problem. The other candidate blocks are generally cancelled; the wining miner earns the block reward and the transaction fees. (Source: Julien Riposo, 2023)

# 3 Some Modeling for the Network of Users

This is where modeling gets involved in: theoretically, the network can be represented as a graph, whose vertices are miners and/or users. Some important models are proposed in [5], which inspired the below content of this section.

In particular, an important tool for making operations (typically addition and multiplication) on graphs is through the matrix universe.

## *3.1 Perron-Frobenius Theorem*

The Perron-Frobenius Theorem should be the fundamental known property for any graph and adjacency matrix considerations, as the *principal component(s)* contain important information on the system. We shall start with this theorem (focusing on *positive* (i.e., positive elements) symmetric matrices), then we focus on more general considerations around graphs and matrices.

***Theorem 4.3.1 (Perron-Frobenius)*** *Let $M$ be a symmetric positive matrix. Then*

(i) $\lambda_1 > |\lambda_i|$ *for all* $i \in \{2, \ldots, n\}$ *(nondegeneracy and dominancy of the highest eigenvalue),*
(ii) *The components of* $v_1$ *(eigenvector associated with* $\lambda_1$*) are non-negative.*

***Proof***
We write

$$M\ v_1 = \lambda_1\ v_1.$$

On the one hand, note that $\lambda_1 > 0$ necessarily, since otherwise we could not have $\mathrm{Tr}(M) > 0$, which is true since $M$ is a positive matrix. On the other hand, we have

$$\lambda_1 = v_1^{\mathrm{T}} M v_1 = \left| \sum_{i,j=1}^{n} m_{ij} v_1^{(i)} v_1^{(j)} \right| \leq \sum_{i,j=1}^{n} m_{ij} \left| v_1^{(i)} \right| \left| v_1^{(j)} \right|.$$

Because of the variational theorem, we have

$$\sum_{i,j=1}^{n} m_{ij} \left| v_1^{(i)} \right| \left| v_1^{(j)} \right| \leq \lambda_1,$$

allowing to conclude equality. Therefore, the vector $w = \left( \left| v_1^{(1)} \right|, \left| v_1^{(2)} \right|, \ldots, \left| v_1^{(n)} \right| \right)^{\mathrm{T}}$ is an eigenvector of $M$ associated with the eigenvalue $\lambda_1$, i.e., $Mw = \lambda_1 w$.

Now, suppose there exists $i \in \{1, \ldots, n\}$ such that $\left| v_1^{(i)} \right| = 0$, aka $v_1^{(i)} = 0$. Then

$$Mw = \lambda_1 w \Longrightarrow \sum_{j=1}^{n} m_{ij} \left| v_1^{(j)} \right| = 0.$$

Since $m_{ij} > 0$ for all $i, j \in \{1, \ldots, n\}$, then $\left| v_1^{(j)} \right| = 0$ for all $j \in \{1, \ldots, n\}$. This is impossible. Therefore, $\left| v_1^{(i)} \right| > 0$ for all $i \in \{1, \ldots, n\}$, thus $v_1^{(i)} \neq 0$ for all $i \in \{1, \ldots, n\}$. Suppose there exists $i \in \{1, \ldots, n\}$ such that $v_1^{(i)} < 0$. As we have seen,

$$\begin{cases} \displaystyle\sum_{j=1}^{n} m_{ij} v_1^{(j)} = \lambda_1 v_1^{(i)}, \\ \displaystyle\sum_{j=1}^{n} m_{ij} \left| v_1^{(j)} \right| = \lambda_1 \left| v_1^{(i)} \right|. \end{cases}$$

Then, by summing, we have

$$\lambda_1 \left( v_1^{(i)} + \left| v_1^{(i)} \right| \right) = 0 = \sum_{j=1}^{n} m_{ij} \left( v_1^{(j)} + \left| v_1^{(j)} \right| \right).$$

Since $v_1^{(j)} + \left|v_1^{(j)}\right| \geq 0$, it turns out that $v_1^{(j)} + \left|v_1^{(j)}\right| = 0$, for all $j \in \{1, \ldots, n\}$. Therefore, $v_1^{(j)} < 0$ for all $j \in \{1, \ldots, n\}$.

Suppose there exists another eigenvector $u \notin \{v_1, w\}$ associated with the eigenvalue $\lambda_1$. Then the same applies to $u$. Since $M$ is a real symmetric matrix, then necessarily $u$ and $v_1$ are orthogonal. However, from the above, we have

$$\sum_{j=1}^{n} v_1^{(j)} u^{(j)} = \sum_{j=1}^{n} \left|v_1^{(j)} u^{(j)}\right| > 0.$$

This is impossible. We conclude that

- $v_1 = w$, and
- There is no other eigenvector associated with the eigenvalue $\lambda_1$.

Thus, the highest eigenvalue is nondegenerate, and $v_1$ has positive components.

Finally, let $\overline{v}$ be an eigenvector associated with the eigenvalue $\lambda_i < \lambda_1$, for any $i \in \{2, \ldots, n\}$. We therefore have

$$\lambda_1 > \sum_{i,j=1}^{n} m_{ij} \left|\overline{v}_1^{(i)}\right| \left|\overline{v}_1^{(j)}\right| \geq \left|\sum_{i,j=1}^{n} m_{ij} \overline{v}_1^{(i)} \overline{v}_1^{(j)}\right| = |\lambda_i|.$$

Therefore, the highest eigenvalue is dominant. This concludes the proof. ∎

This theorem still applies when the matrix $M$ has its components non-negative and is *irreducible*, i.e., there is no any (permutation) matrix $P$ such that $P^{-1}MP$ is block upper triangular.

***Definition 4.3.2*** *A matrix $M = (m_{ij})_{1 \leq i,\, j \leq n}$ possesses the* Perron-Frobenius property *if $\lambda_1 > |\lambda_i|$ for all $i \in \{2, \ldots, n\}$ and the component of $v_1$ are all non-negative.*

***Definition 4.3.3*** *A matrix $M = (m_{ij})_{1 \leq i,\, j \leq n}$ is said to be* eventually positive *if there exists $k_0 \in \mathbb{R}$ such that $M^k$ has its elements all positive, for any $k \geq k_0$.*

***Theorem 4.3.4*** *For a symmetric matrix $=(m_{ij})_{1 \leq i,\, j \leq n}$, the following properties are equivalent:*

(i) $M$ possesses the Perron-Frobenius property;
(ii) $M$ is eventually positive.

***Proof*** (see [6])

Suppose that

$$\lambda_1 > |\lambda_2| \geq \ldots \geq |\lambda_n|.$$

Let $v_i$ be the eigenvector associated with the eigenvalue $\lambda_i$, for all $i \in \{1, \ldots, n\}$. Since $M$ is symmetrical, then they form an orthonormal basis of $\mathbb{R}^n$. Let

$$\mathcal{V} = \left\{ v \in \mathbb{R}^n \backslash \{0_{\mathbb{R}^n}\} \text{ with non} - \text{negative components, and } v^{\mathrm{T}} v = 1 \right\}.$$

For $v \in \mathcal{V}$, let $\alpha_i = v_i^{\mathrm{T}} v$ (inner product on $\mathbb{R}^n$) so that $v = \sum_{i=1}^{n} \alpha_i \, v_i$. Note that $\alpha_1 > 0$.

Because $M$ is symmetric, so as $M^k$ for any $k \in \mathbb{N}^*$ and, using the spectral decomposition of $M$, we have

$$M^k = \sum_{i=1}^{n} \lambda_i^k \, v_i \, v_i^{\mathrm{T}},$$

thus $M^k v \sim \lambda_1^k \, \alpha_1 \, v_1$ when $k \longrightarrow +\infty$. Thus, there exists $m \in \mathbb{N}^*$ such that $M^k v$ has its components non-negative, for any $k \geq m$.

The set $\mathcal{V}$ is compact. Indeed, $\mathcal{V} \subset \{v \in \mathbb{R}^n \backslash \{0_{\mathbb{R}^n}\}, v^{\mathrm{T}} v = 1\}$, which is compact. Moreover, if $(u_n)_{n \in \mathbb{N}}$ is a sequence of $\mathcal{V}$ with limit $u$, then $u \in \{v \in \mathbb{R}^n \backslash \{0_{\mathbb{R}^n}\}, v^{\mathrm{T}} v = 1\}$ (it is compact therefore closed) and the components of $u$ are non-negative since $\mathbb{R}_+$ is closed in $\mathbb{R}$. Thus, $\mathcal{V}$ is closed, therefore it is compact.

Now set

$$k_0 = \max_{v \in \mathcal{V}} (m, M^k v \text{ has its components non} - \text{negative, for any } k \geq m),$$

which exists since $\mathcal{V}$ is compact. Therefore, for all $v \in \mathcal{V}$, then for all $k \geq k_0$, all the components of $M^k v$ are positive. By choosing $v = (0, \ldots, 0, 1, 0, \ldots 0)^{\mathrm{T}}$ ($j^{\text{th}}$ position for any $j \in \{1, \ldots, n\}$), then element $(i, j)$ of $M^k$ is positive, for all $i \in \{1, \ldots, n\}$. This demonstrates that $M$ is eventually positive.

Conversely, if $k_0 \in \mathbb{N}$ is such that $M^k$ has positive elements for all $k \geq k_0$, picking a particular such $k$, and noticing that $M^k$ is a symmetric positive matrix, Theorem 4.3.1 applies to $M^k$. The highest eigenvalue of $M^k$ is positive and dominant, and the unique eigenvector associated with this eigenvalue has positive components. It is worth noticing that the highest eigenvalue of $M$ is the $k^{\text{th}}$ root of the one of $M^k$ with the same eigenvector. Finally, the proof is achieved. ∎

## 3.2 *Invariants of an Adjacency Matrix*

Invariants of matrices are useful to characterize the network, e.g., to calculate the probability that a random walker revisits a node in a given number of steps. Here, "invariants" refer to the fact that any map from $V$ to $V$ that conserves adjacency also keeps these "invariant" values. We will need further definitions to calculate these probabilities. It is worth pointing out that adjacency matrices are invariant on such maps. Sections 3.2, 3.3 and 3.4 are dealing with these calculations.

***Definition 4.3.5*** *The* principal invariant *of the graph's $n \times n$ adjacency matrix $M$ are the coefficients of its characteristic polynomial, that is $(I_k(M))_{k\in\{0,1,\ldots,n\}}$, such that*

$$\chi_M(\lambda) = \det(M - \lambda\,\mathbb{I}) = \sum_{k=0}^{n} I_k(M)(-\lambda)^{n-k} = 0.$$

***Proposition 4.3.6*** *We have the* Newton Identity:

$$\mathcal{I}_k(M) = \frac{(-1)^{k-1}}{k} \sum_{l=0}^{k-1} (-1)^l\, \mathcal{I}_l(M)\, \mathrm{Tr}\big(M^{k-l}\big), \quad \forall k \in \{1,\ldots,n\}.$$

***Proof***
We use the identification through formal series. The characteristic polynomial has $n$ roots (the eigenvalues $\lambda_1, \ldots, \lambda_n$) on the algebraically closed field $\mathbb{C}$, which is a separable field, thus we can write

$$\chi_M(X) = (-1)^n \prod_{i=1}^{n} (X - \lambda_i),$$

hence

$$\prod_{i=1}^{n} (X - \lambda_i) = \sum_{k=0}^{n} (-1)^k\, \mathcal{I}_k(M)\, X^{n-k}.$$

Multiply both sides by $X^{-n}$ and set $Y = X^{-1}$, so that we have

$$\prod_{i=1}^{n} (1 - \lambda_i Y) = \sum_{k=0}^{n} (-1)^k\, \mathcal{I}_k(M)\, Y^k.$$

We now calculate the differential polynomial (with respect to $Y$) of the right-hand side, which we multiply by $Y$, simply being $\sum_{k=1}^{n} (-1)^k\, k\, \mathcal{I}_k(M)\, Y^k$, while for the left-hand side and assuming that $|Y|$ is sufficiently small, we have

$$\begin{aligned} Y \sum_{i=1}^{n} \left( (-\lambda_i) \prod_{j=1, j\neq i}^{n} (1 - \lambda_j Y) \right) &= -\left( \sum_{i=1}^{n} \frac{\lambda_i Y}{1 - \lambda_i Y} \right) \prod_{j=1}^{n} (1 - \lambda_j Y) \\ &= -\left( \sum_{i=1}^{n} \sum_{l=1}^{+\infty} (\lambda_i Y)^l \right) \left( \sum_{l'=0}^{n} (-1)^{l'}\, I_{l'}(M)\, Y^{l'} \right) \end{aligned}$$

$$= \left( \sum_{l=1}^{+\infty} \sum_{i=1}^{n} (\lambda_i Y)^l \right) \left( \sum_{l'=0}^{n} (-1)^{l'-1} \, I_{l'}(M) \, Y^{l'} \right)$$
$$= \left( \sum_{l=1}^{+\infty} \mathrm{Tr}\big(M^l\big) \, Y^l \right) \left( \sum_{l'=0}^{n} (-1)^{l'-1} \, I_{l'}(M) \, Y^{l'} \right).$$

We have indeed assumed that $|Y|$ was sufficiently small, so that the double sum in the first factor could be inverted (because there is uniform convergence), and we used the fact that $\sum_{i=1}^{n} \lambda_i^l = \mathrm{Tr}\big(M^l\big)$. Therefore,

$$\sum_{k=1}^{n} (-1)^k \, k \, \mathcal{I}_k(M) \, Y^k = \left( \sum_{l=1}^{+\infty} \mathrm{Tr}\big(M^l\big) \, Y^l \right) \left( \sum_{l'=0}^{n} (-1)^{l'-1} \, \mathcal{I}_{l'}(M) \, Y^{l'} \right).$$

Now we want to use the Cauchy product of two formal series. We set $\mathcal{I}_k(M) = 0$ if $k > n$, and

$$a_l = \begin{cases} 0 & \text{if } i = 0 \\ \mathrm{Tr}(M^l) & \text{if } i \geq 1 \end{cases} \qquad \text{and} \qquad b_k = \begin{cases} (-1)^{k-1} \, \mathcal{I}_k(M) & \text{if } k \leq n \\ 0 & \text{if } k > n \end{cases}$$

so that

$$\sum_{k=1}^{n} (-1)^k \, k \, \mathcal{I}_k(M) \, Y^k = \sum_{k=0}^{+\infty} \left( \sum_{l=1}^{k} (-1)^{k-l-1} \, \mathcal{I}_{k-l}(M) \, \mathrm{Tr}\big(M^l\big) \right) Y^k,$$

We therefore identify the coefficients of the same degree, and we have, for any $k \in \{1, 2, \ldots, n\}$,

$$(-1)^k \, k \, I_k(M) = \sum_{l=1}^{k} (-1)^{k-l-1} \, I_{k-l}(M) \, \mathrm{Tr}\big(M^l\big) = (-1)^{k-1} \sum_{l=1}^{k} (-1)^l \, I_{k-l}(M) \, \mathrm{Tr}\big(M^l\big),$$

which gives the required expression by a straightforward index change. ∎

Thus, when $k = 1$, we have

$$I_1(M) = \mathrm{Tr}(M).$$

When $k = 2$, we have

$$\mathcal{I}_2(M) = \frac{1}{2} \, \mathcal{I}_1(M) \, \mathrm{Tr}(M) - \frac{1}{2} \, \mathcal{I}_0(M) \, \mathrm{Tr}\big(M^2\big) = \frac{1}{2} \, \Big( \mathrm{Tr}(M)^2 - \mathrm{Tr}\big(M^2\big) \Big).$$

When $k = 3$, we have

$$\begin{aligned}\mathcal{I}_3(M) &= \frac{1}{3}\left(\mathcal{I}_2(M)\,\mathrm{Tr}(M) - \mathcal{I}_1(M)\,\mathrm{Tr}\left(M^2\right) + \mathcal{I}_0(M)\,\mathrm{Tr}\left(M^3\right)\right)\\ &= \frac{1}{6}\mathrm{Tr}(M)^3 - \frac{1}{2}\mathrm{Tr}\left(M^2\right)\mathrm{Tr}(M) + \frac{1}{3}\mathrm{Tr}\left(M^3\right).\end{aligned}$$

As a remark, $\mathcal{I}_k(M)$ writes as

$$\mathcal{I}_k(M) = \sum_{l=0}^{k-1}\left(-\mathrm{Tr}\left(\frac{(-M)^{k-l}}{k}\right)\right)\mathcal{I}_l(M),$$

hence $\mathcal{I}_k(M)$ is a weighted average of the $\mathcal{I}_{k-1}(M), \ldots, \mathcal{I}_0(M)$, and the weights are $\left(-\mathrm{Tr}\left(\frac{(-M)^{l+1}}{k}\right)\right)_{l\in\{0,\ldots,k-1\}}$.

A very important note is that the $\mathcal{I}_k(M)$'s write with respect to *structural characteristics* of the graph, such as the number $l$-loops $N_l^G$, for $l \in \{0, \ldots, k-1\}$. Indeed, we have the following proposition.

***Proposition 4.3.7*** *We have*

$$N_k^G = \frac{1}{k!}\,\mathrm{Tr}\left(M^k\right), \quad \text{for all } k \in \mathbb{N}^*.$$

***Proof***
When $k = 1$, $\mathrm{Tr}(M)$ is the sum of the diagonal elements of $M$, i.e., the sum of self-loops. If $k > 1$, recall about the meaning of the $(i, j)$ element of $M^k$: this is the number of paths connecting nodes $i$ and $j$. If $(M^k)_{i,j}$ is this element, we have

$$\left(M^k\right)_{i,i} = \sum_{j=1}^{n}\left(M^{k-1}\right)_{i,j} m_{j,i}.$$

Therefore, $(M^k)_{i,i}$ is the number of all paths of length $k$ from $i$ to itself. Summing over $i$, we obtain the number of all paths of length $k$ from $i$ to itself, for all $i$, but counted $k!$ too much (there are $k!$ such permutations of loops in the graph). ∎

The following theorem is the main result of this section.

***Theorem 4.3.8*** *We have*

$$\mathcal{I}_k(M) = \sum_{\substack{\sum_{i=1}^{k} i\, l_i = k\\ \forall i\ \ l_i \geq 0}} \prod_{j=1}^{k} (-1)^{l_j} \frac{\left(\mathrm{Tr}(M^j)\right)^{l_j}}{l_j! j^{l_j}}, \quad \forall k \in \mathbb{N}^*.$$

Note that $\sum_{\substack{\sum_{i=1}^{k} i\, l_i = k \\ \forall i \;\; l_i \geq 0}}$ is

$$\sum_{\substack{\sum_{i=1}^{k} i\, l_i = k \\ \forall i \;\; l_i \geq 0}} = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_k}$$

such that $i_1\, l_1 + \ldots + i_k l_k = k$, where $l_1, \ldots, l_k \geq 0$. This equation is the *natural integer partition* of the number $k$.

***Proof***
We now introduce the two generating functions:

$$\begin{cases} \Phi_{\mathcal{I}}(z) = \sum_{k=0}^{+\infty} \mathcal{I}_k(M) z^k, \\ \Phi_{\mathrm{T}}(z) = \sum_{k=0}^{+\infty} \mathrm{Tr}\Big( (-M)^k \Big) z^k. \end{cases}$$

Both of them converge for $|z|$ sufficiently small (we invite the reader to prove this assertion). From Proposition 4.3.6, we have that, by applying the Cauchy product in the other way:

$$\frac{\mathrm{d}}{\mathrm{d}z}\, \Phi_{\mathcal{I}}(z) = \sum_{k=1}^{+\infty} k\, \mathcal{I}_k(M)\, z^{k-1} = -\sum_{k=1}^{+\infty} \sum_{l=0}^{k-1} \mathrm{Tr}\Big( (-M)^{k-l} \Big)\, \mathcal{I}_l(M)\, z^{k-1}$$

$$= -\left( \sum_{k=0}^{+\infty} \mathrm{Tr}\Big( (-M)^k \Big) z^k \right) \left( \sum_{k=0}^{+\infty} \mathcal{I}_k(M) z^k \right) = -\Phi_{\mathcal{I}}(z) \Phi_{\mathrm{T}}(z).$$

Thus, given that $\Phi_{\mathcal{I}}(0) = 1$, we solve this differential equation to obtain

$$\Phi_{\mathcal{I}}(z) = \exp\left( -\int_0^z \Phi_{\mathrm{T}}(z')\mathrm{d}z' \right) = \exp\left( \sum_{k=1}^{+\infty} \frac{(-1)^{k+1}}{k} \mathrm{Tr}\Big( (-M)^k \Big) z^k \right)$$

$$= \prod_{k=1}^{+\infty} \exp\left( \frac{(-1)^{k+1}}{k} \mathrm{Tr}\Big( (-M)^k \Big) z^k \right) = \prod_{k=1}^{+\infty} \exp\left( -\frac{\mathrm{Tr}(M^k)}{k} z^k \right).$$

Using the fact that

$$\mathrm{e}^z = \sum_{k=0}^{+\infty} \frac{z^k}{k!}, \quad \forall z \in \mathbb{C},$$

we therefore deduce that

$$\Phi_{\mathcal{I}}(z) = \sum_{k=0}^{+\infty} \left( \sum_{\substack{\sum_{i=1}^{k} i\, l_i = k \\ \forall i \;\; l_i \geq 0}} \prod_{j=1}^{k} (-1)^{l_j} \frac{(\mathrm{Tr}(M^j))^{l_j}}{l_j! j^{l_j}} \right) z^k.$$

The uniqueness of the coefficients of $\Phi_{\mathcal{I}}$ allows to achieve the proof. ∎

## 3.3 Random Walk and Principal Invariants

To calculate the probabilities, we need to introduce random walks. In Sect. 2.2, Chap. 7, we reintroduce random walks and Markov Chain in a more formal way.

***Definition 4.3.9*** *Let $G = (V, E)$ be a graph. A* random walk *of size $N \in \mathbb{N}$ on $G$ is a sequence $(X_k)_{k \in \{0,1,\ldots,N\}}$ such that $X_k \in V$ for any $k \in \{0, 1, \ldots, N\}$, and $(X_k, X_{k+1}) \in E$ for any $k \in \{0, 1, \ldots, N-1\}$.*

***Definition 4.3.10*** *A random walk $(X_k)_{k \in \{0,1,\ldots,N\}}$ with $N \in \mathbb{N}$ is a* Markov Chain *if*

$$\mathbb{P}(X_n = i | X_{n-1} = j, X_{n-2} = i_{n-2}, \ldots, X_0 = i_0) = \mathbb{P}(X_n = i | X_{n-1} = j),$$

for any $(n, i, j, i_{n-2} \ldots, i_0) \in \{1, \ldots, N\} \times V^{n+1}$. We say that the Markov Chain *is* homogeneous if $\mathbb{P}(X_n = i | X_{n-1} = j)$ does not depend on $n$.

***Definition 4.3.11*** *A* Transition Matrix *$\mathcal{T}$ is a matrix whose element $(i, j)$ is the probability that the homogeneous Markov-Chain random walker gets from state $i \in V$ to state $j \in V$. This is a stochastic matrix, that is*

$$\sum_{j=1}^{n} \mathcal{T}_{ij} = 1, \quad \forall i \in \{1, \ldots, n\}.$$

Theorem 4.3.8 applies to $\mathcal{T}$, that is

$$\mathcal{I}_k(\mathcal{T}) = \sum_{\substack{\sum_{i=1}^{k} i\, l_i = k \\ \forall i \;\; l_i \geq 0}} \prod_{j=1}^{k} (-1)^{l_j} \frac{\left(\mathrm{Tr}\left(\mathcal{T}^j\right)\right)^{l_j}}{l_j! j^{l_j}}, \quad \forall k \in \mathbb{N}^*.$$

Thus, we particularly have

- $\mathcal{I}_1(\mathcal{T}) = \mathrm{Tr}(\mathcal{T})$ is the probability that a random walker stays at a node in one time step;
- $\mathcal{I}_n(\mathcal{T}) = \det(\mathcal{T})$ is the probability that a random walker revisits an initial node in $n$ steps.

## 3.4 Symmetric Group and Principal Invariants

In this section, we give a simple combinatorics interpretation of Theorem 4.3.8. The *symmetric group* $o_n$ is the group of permutations from $\{1, \ldots, n\}$ to $\{1, \ldots, n\}$. There are $n!$ such permutations.

***Definition 4.3.12*** *The* cycle index polynomial *of the symmetric group $o_n$ is given by*

$$Z_n(x_1, \ldots, x_n) = \frac{1}{n!} \sum_{\sigma \in o_n} \prod_{j=1}^{n} x_j^{l_j(\sigma)},$$

*where $l_j(\sigma)$ is the number of (disjoint) cycles of length $j$ in the permutation $\sigma$.*

The cycle index reveals compactly some information about how the symmetric group $o_n$ acts on the set $\{1, \ldots, n\}$.

***Theorem 4.3.13*** *We have*

$$Z_n(x_1, \ldots, x_n) = \sum_{\substack{\sum_{i=1}^{n} i\, l_i = n \\ \forall i \quad l_i \geq 0}} \prod_{j=1}^{n} \frac{x_j^{l_j}}{l_j! j^{l_j}}.$$

***Proof***

The quantity of interest to calculate is $\sum_{\sigma \in o_n} \prod_{j=1}^{n} x_j^{l_j(\sigma)}$. To transform it, we partition the integer $n$ as follows:

$$n = \sum_{i=1}^{n} i\, l_i, \quad \text{where} \quad l_i \in \mathbb{N}, \forall i \in \{1, \ldots, n\}.$$

This is a compact way to write $n = \alpha_1 + \ldots + \alpha_N$, for some $N \in \mathbb{N}$, and where $\alpha_i \in \mathbb{N}$. Some $\alpha_i$s are the same, hence the above formulation.

We need to calculate the total number of *equivalent* permutations containing $l_k$ cycles of length $k$ between the vertices $\{1, \ldots, n\}$, for all $k \in \{1, \ldots, n\}$. There are $n!$ such permutations in total.

However, for all $k \in \{1, \ldots, n\}$, the $l_k$ cycles of length $k$ could be written in any order (e.g., $(14)(23) = (23)(14)$), and there are $l_k!$ ways to write these unique permutations.

In addition, for all $k \in \{1, \ldots, n\}$, a cycle of length $k$ is the same by circular permutation (e.g., $(123) = (231)$). There are $k$ ways to write a cycle of length $k$; therefore, there are $k^{l_k}$ ways to write $l_k$ cycles of length $k$.

Finally, if we consider one family of cycles of length $k$ and another one of length $k' \neq k$, the total number of configurations is the multiplication of the ones for each family, as the classes of distinct-length cycle are disjoint.

Bearing all above in mind, we conclude that

$$\sum_{\sigma \in O_n} \prod_{j=1}^{n} x_j^{l_j(\sigma)} = \sum_{\substack{\sum_{i=1}^{n} i\, l_i = n \\ \forall i \;\; l_i \geq 0}} \frac{n!}{\left(\prod_{k=1}^{n} l_k!\right)\left(\prod_{k=1}^{n} k^{l_k}\right)} x_1^{l_1} \ldots x_n^{l_n},$$

hence the result. ∎

***Corollary 4.3.14*** *We have*

$$\sum_{\substack{\sum_{i=1}^{n} i\, l_i = n \\ \forall i \;\; l_i \geq 0}} \prod_{j=1}^{n} \frac{1}{l_j! j^{l_j}} = 1.$$

***Proof***

This comes from the identity of the expressions of $Z_n$ in Definition 4.3.12 and Theorem 4.3.13, and by setting $x_1 = \ldots = x_n = 1$. ∎

From this equality, we have

$$n! = \sum_{\substack{\sum_{i=1}^{n} i\, l_i = n \\ \forall i \;\; l_i \geq 0}} \prod_{j=1}^{n} \frac{n!}{l_j! j^{l_j}},$$

therefore, the term $\prod_{j=1}^{n} \frac{n!}{l_j! j^{l_j}}$ is the total number of equivalent permutations containing $l_k$ disjoint cycles of length $k$, for all $k \in \{1, \ldots, n\}$.

We now come back to Theorem 4.3.8. Especially, we have

$$\mathcal{I}_k(M) = Z_k\big(-\mathrm{Tr}(M), -\mathrm{Tr}\big(M^2\big), \ldots, -\mathrm{Tr}\big(M^k\big)\big), \quad \forall k \in \mathbb{N}^*.$$

Therefore, $\mathcal{I}_k(M)$ is nothing else but the cycle index polynomial of the symmetric group $o_k$, valued at point $(-\mathrm{Tr}(M), -\mathrm{Tr}(M^2), \ldots, -\mathrm{Tr}(M^k))$.

This proves that the invariants contain information on the act of the symmetric group to the network.

## 3.5 First Hitting Times and Absorption Probability

This section is dealing with the time needed for a random walker to first hit a vertex on average, as well as the mean time to reach a state to which it stays – this state is called *absorbing*. This could deal with an information shared by a miner to another one, or a transaction information entering the network. Thus, absorption of information is an important matter for understanding the Network of Users.

We consider a graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ and $G$ is *connected*, which means that for any two nodes $i, j \in V$, there exists a *path* $((x_k, x_{k+1}))_{k \in \{0, \ldots, N-1\}}$ for a given $N \in \mathbb{N}^*$ and $x_0, \ldots, x_N \in V$, such that $(x_k, x_{k+1}) \in E$ for all $k \in \{0, \ldots, N-1\}$. Figure 4.4 shows an example of connected graph.

We consider a homogeneous Markov chain (see Definition 4.3.10) $(X_k)_{k \in \mathbb{N}}$ on this graph, whose transition matrix (see Definition 4.3.11) is given by $\mathcal{T}$. For example, the transition matrix for the graph shown in Fig. 4.4 is given by

$$\mathcal{T} = \begin{pmatrix} 0 & 0.5 & 0 & 0.5 \\ 0.5 & 0 & 0.25 & 0.25 \\ 0 & 0.4 & 0.4 & 0.2 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

***Definition 4.3.15*** *A state $i \in V$ is called* absorbing state *if it is impossible to leave this state on the graph, that is if $\mathcal{T}_{ii} = 1$, implying that $\mathcal{T}_{ij} = 0$ for any $j \in V \backslash \{i\}$.*

***Theorem 4.3.16 (First Hitting Times)*** *Let $l \in V$. If $\mu_i$ is the expected time to reach state $l$ from state $i \in V \backslash \{l\}$, then*

$$\mu_i = 1 + \sum_{j=1}^{n} \mathcal{T}_{ij}\, \mu_j.$$

***Proof***

Let $T_l = \min_{t \geq 0} (X_t = l)$ *be the* First Hitting Time *to reach state $l$.* Let $l \in V$ and $t \in \mathbb{N}^*$. By definition, we have $\mu_i = \mathbb{E}(T_l | X_0 = i)$.

From the *law of total expectation*, given by

$$\mathbb{E}(T_l | X_t = i) = \sum_{j=1}^{n} \mathbb{E}(T_l | X_{t-1} = j)\ \mathbb{P}(X_t = i | X_{t-1} = j) = \sum_{j=1}^{n} \mathbb{E}(T_l | X_{t-1} = j)\ \mathcal{T}_{ij}$$

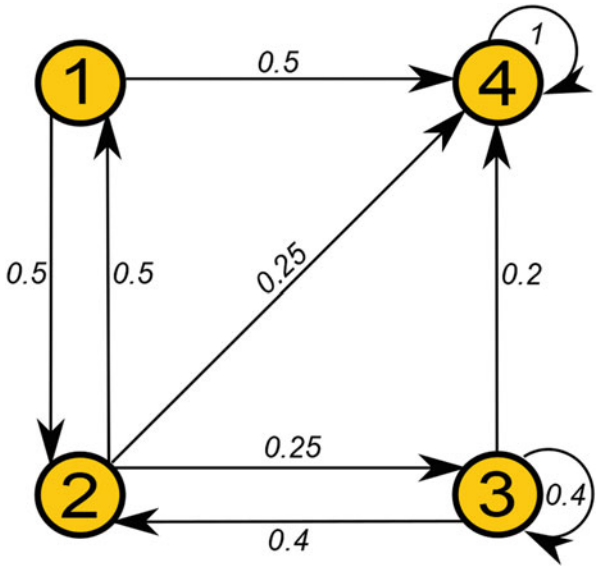


**Fig. 4.4** Graph $G = (V, E)$ with $V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (1, 4), (2, 1), (2, 3), (2, 4), (3, 3), (3, 4), (4, 4)\}$. The transition matrix is also represented by the weight on each arrow. (Source: Julien Riposo, 2023)

Noticing that $\mathbb{E}(T_l|X_{t-1} = j) = \mathbb{E}(T_l|X_t = j) + 1$ and that, because the Markov chain is homogeneous, $\mathbb{E}(T_l|X_t = i) = \mu_i$, we then achieve the proof. ∎

Regarding the graph shown in Fig. 4.4, if we set $l = 4$ (absorbing state), then

$$\begin{cases} \mu_1 = 1 + \dfrac{1}{2}\,(\mu_2 + \mu_4) \\ \mu_2 = 1 + \dfrac{1}{2}\,\mu_1 + \dfrac{1}{4}\,(\mu_3 + \mu_4) \\ \mu_3 = 1 + \dfrac{2}{5}\,(\mu_2 + \mu_3) + \dfrac{1}{5}\,\mu_4 \\ \mu_4 = 0 \end{cases}$$

Hence

$$\begin{cases} \mu_1 = \dfrac{37}{4} \\ \mu_2 = \dfrac{23}{7} \\ \mu_3 = \dfrac{27}{7} \\ \mu_4 = 0 \end{cases}$$

***Theorem 4.3.17 (Absorbing Probability)*** *Let $l \in V$ be an absorbing state. We consider $A_l = \{\text{absorption at state } l\}$ and $a_i = \mathbb{P}(A_l | X_0 = i)$, for any $i \in V$. Then*

$$a_i = \sum_{j=1}^{n} \mathcal{T}_{ij}\, a_j, \quad \forall i \in V \backslash \{l\}.$$

***Proof***
Let $l \in V$ and $t \in \mathbb{N}^*$. By definition, we have $a_l = 1$. For any other absorbing state $l' \neq l$, if any, then $a_{l'} = 0$. We use the *law of total probability*, given by

$$\mathbb{P}(A_l | X_t = i) = \sum_{j=1}^{n} \mathbb{P}(A_l | X_{t-1} = j)\ \mathbb{P}(X_t = i | X_{t-1} = j) = \sum_{j=1}^{n} \mathbb{P}(A_l | X_{t-1} = j)\ \mathcal{T}_{ij}.$$

Since the Markov chain is homogeneous, we have $\mathbb{P}(A_l | X_t = i) = \mathbb{P}(A_l | X_{t-1} = i) = \mathbb{P}(A_l | X_0 = i)$, hence the result. ∎

Regarding the graph shown in Fig. 4.4, if we set $l = 4$ (absorbing state), then

$$\begin{cases} a_1 = \dfrac{1}{2}\ (a_2 + a_4) \\ a_2 = \dfrac{1}{2}\ a_1 + \dfrac{1}{4}\ (a_3 + a_4) \\ a_3 = \dfrac{2}{5}\ (a_2 + a_3) + \dfrac{1}{5}\ a_4 \\ a_4 = 1 \end{cases}$$

Hence $a_1 = a_2 = a_3 = a_4 = 1$. Without surprise, the walker will be absorbed by the only absorbing state 4.

## 3.6 A Quick Word on Diffusion and Propagation of Information

The Diffusion Theory on a graph definitely is a topic that applies here. If $f$ materializes the information rate on a graph, that is $f$ is a two-variable function defined on $\mathbb{R} \times V$, which is such that $t \mapsto f(t, x)$ is a $\mathcal{C}^1$ class function for any $x \in V$, then for an appropriate defined Laplacian operator $\Delta$, we have the following *Diffusion Equation*:

$$\frac{\partial f}{\partial t}(t, x) = -D\, \Delta f(t, x), \quad \forall (t, x) \in U \subseteq \mathbb{R} \times V,$$

where $U$ is an open set of $\mathbb{R} \times V$, and $D \in \mathbb{R}_+^*$ (notice the minus sign on the right-hand-side).

A fitting for the actual network could be performed, and information about the diffusion power of the network could be obtained. It should be interesting to derive the analytical solutions for the above equation, when the graph is undirected, but also when it is directed, so that a proper fit could be performed. For more details, the reader is invited to focus on the article [12], which construct from scratch a Peer-to-Peer network model describing the diffusion of information through it. The main advantage of this model is to be able to fit the network for forecasting purposes.

# 4 Diffusion Models on a Graph

In this section, we are reviewing famous diffusion models. There are needs for understanding network influence behaviors. Specific decisions can be taken from a part of the network, while the other part might be influenced and decide either the same, or differently. The latter decision might influence the former and a game theory starts within the network. There is a diffusion of information, while some agents learn from others.

The two main aspects we would like to emphasize here are

- the *diffusion* aspect, or how the network does impact behaviors, and
- the *learning* aspect, or how the opinion might change from information reception.

The diffusion aspect can be modeled by the Bass Model (1969), while the learning aspect can be modeled by the DeGrout Model (1974). Both models are connected to the important "S" shape. Further considerations are explicitly shown in the detailed book [13].

## *4.1 "S" Shapes*

The "S" Shape is a network characteristic that appears almost systematically in every diffusion on a graph. It can model some news propagating throughout the internet network, or a virus reaching several people.

Figure 4.5 is explaining the characteristic through an example of virus outbreak. We may explain the concept through the example of a piece of news reaching a group of persons, or a list of transactions reaching all the Bitcoin miners. Thus, the "S" Shape can be divided into three consecutive time regimes.

- *Initiation*: A miner is receiving a new transaction to mine. It is (1) starting to hash and (2) diffusing the new piece of information to its direct connected miners. The latter are receiving the information, and this repeats again. At earlier stages, the percentage of people concerned with this new piece of information remains very low, and the diffusion *rate* is low as well, but increasing, because there are more and more miners getting the information.

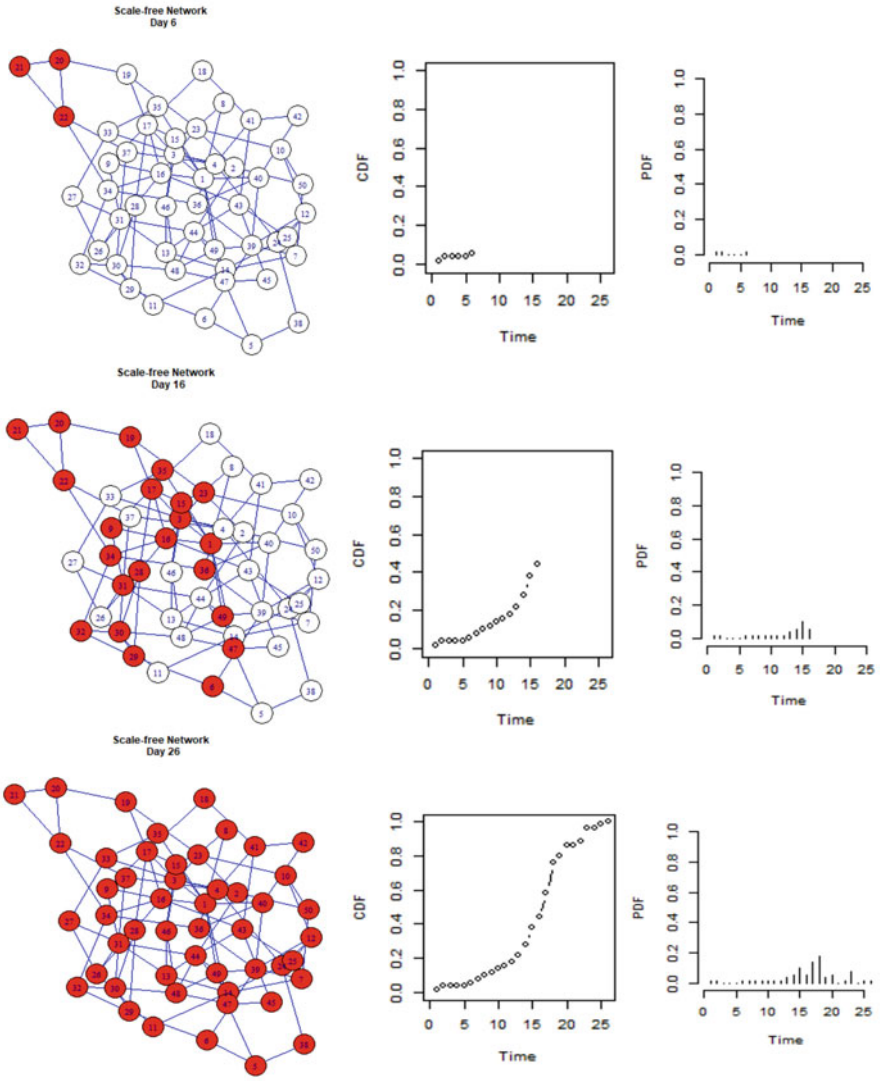


**Fig. 4.5** "S" Shape illustration. Here, a contagious virus is spreading throughout a community of people. First (top panel), a very few number of people are contaminated. Second (middle panel), the rate of contagions is exploding. Last (bottom panel), the rate decreases, since everyone has been contaminated. On the right-hand side are shown the empirical PDF and corresponding CDF, which shows an "S" Shape once the virus has reached a sufficiently high number of people. (Source: Julien Riposo, 2023)

– *Transition*: The diffusion rate is exploding. The number of miners who have received and diffused the information is now sufficiently high to quickly reach most of the community.
– *Termination*: A very few number of persons without the information are left, and it remains less likely to reach them, because not everyone in the community knows them, but only a very few.

In Fig. 4.5, for each panel, we have represented the PDF (bottom figure) and the CDF (top figure), as a function of time of the contagion. On the bottom panel, the CDF has the "S" Shape.

## 4.2 Bass Model

This model aims at reproducing the "S" Shape. The network is not supposed to have any particular structure, except it needs to be connected (i.e., for any two vertices, there exists one path in the graph connecting them). To each vertex is associated two states, either state 0 or 1 (e.g., the information is received or not; the decision is made or not). We introduce the function $F$ of time, representing the fraction of population who have adopted action 1 at time $t$. We wish to know the evolution of $F$.

Let $p$ be the rate of spontaneous adoption, i.e., adoption *without* particular influence from the others in the community. Let $q$ be the rate of imitation adoption.

Consider the time $t \in \mathbb{R}_+$, and consider the increment $\delta t > 0$. What is $F(t + \delta t)$?

First, $F(t + \delta t)$ is expressed as the rate $F(t)$ at time $t$ if no adoptions have been made from times $t$ to $t + \delta t$. Second, some spontaneous adoptions could have been made, with probability $p$ times the number of non-adoptions. Third, some imitation adoption could also have been made, with probability $q$, which we have to multiply by two other factors: the number of non-adoptions (since they are going to adopt) times the number of adoptions (since they are the influencers). To sum up, we have the discrete Bass Model equation.

***Proposition 4.4.1 (Discrete Bass Model Equation)*** *The evolution of the factor of population who adopted action* 1 *from times t to t + δt is given by the equation*

$$F(t + \delta t) = F(t) + p\delta t(1 - F(t)) + q\delta t(1 - F(t))F(t).$$

This equation can be rewritten as

$$F(t + \delta t) - F(t) = (p + qF(t))(1 - F(t))\delta t.$$

Because $F$ is a continuous function of time, we then have

$$\frac{\mathrm{d}F(t)}{\mathrm{d}t}\delta t = (p + qF(t))(1 - F(t))\delta t.$$

We deduce the following proposition.

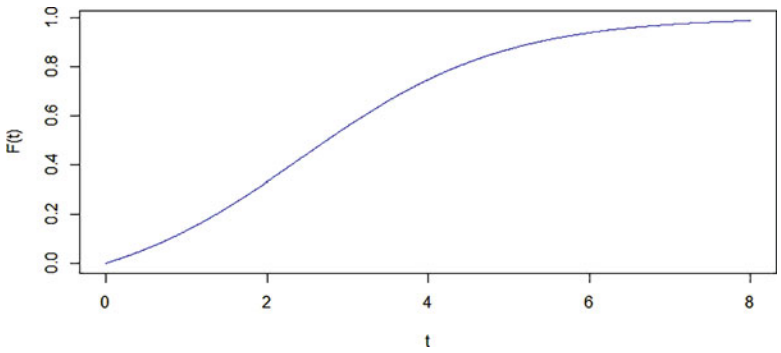


**Fig. 4.6** Proportion of adoptions given by the Bass Model. Here $p = 1/10$ and $q = 1/2$. (Source: Julien Riposo, 2023)

***Proposition 4.4.2 (Continuous Bass Model)*** *The evolution of the factor of population who adopted action* 1 *at time t is given by the following differential equation*

$$\frac{\mathrm{d}F(t)}{\mathrm{d}t} = (p + qF(t))(1 - F(t)).$$

*The solution to this differential equation, with* $F(0) = 0$*, is given by*

$$F(t) = \frac{1 - \mathrm{e}^{-(p+q)t}}{1 + \frac{q}{p}\mathrm{e}^{-(p+q)t}}.$$

*In addition, this function presents an "S" Shape when* $p < q$.

See Fig. 4.6 for an illustration. The "S" Shape appears when the agents are more influenced than making their own decision. This behavior might imply a domino effect, faster than without influence, in the decision process.

***Proof***
For any $t \in \mathbb{R}_+^*$, the function $F$ is lower and upper bounded: $0 \leq F(t) < 1$. The following problem

$$\begin{cases} \dfrac{\mathrm{d}F(t)}{\mathrm{d}t} = (p + qF(t))(1 - F(t)) \\ F(0) = 0 \end{cases}$$

is defining a Cauchy problem, which admits a unique solution.

From the differential equation, we have

$$\int_0^{F(t)} \frac{\mathrm{d}F(t')}{(p+qF(t'))(1-F(t'))} = \int_0^t \mathrm{d}t' = t.$$

The left-hand side writes as

$$\int_0^{F(t)} \left(\frac{q}{p+qF(t')} + \frac{1}{1-F(t')}\right)\mathrm{d}F(t') = \ln\left(\frac{p+qF(t)}{1-F(t)}\right) - \ln(p),$$

and after some quick calculation, we deduce the solution. We note that this solution is well defined on $\mathbb{R}_+^*$ and is therefore maximal on $\mathbb{R}_+^*$, which means it is the solution satisfying the differential equation on $\mathbb{R}_+^*$.

It is clear that

- the function $F$ is strictly increasing from $\mathbb{R}_+^*$ to $[0, 1[$, therefore it is bijective;
- $\lim_{t\to 0^+} F(t) = F(0) = 0$; $\lim_{t\to+\infty} F(t) = 1$;
- $\lim_{t\to 0^+} F'(t) = p$; $\lim_{t\to+\infty} F'(t) = 0$;

Therefore, proving that this function admits an "S" Shape is the same as proving that there is a unique inflection point. The function $F$ is twice differentiable on $\mathbb{R}_+^*$. We have

$$F''(t) = (q - p - 2qF(t))F'(t).$$

Thus, $F''(t) = 0$ if and only if

$$F(t) = \frac{1}{2}\left(1 - \frac{p}{q}\right).$$

Hence, the point $t = F^{-1}\left(\frac{1}{2}\left(1 - \frac{p}{q}\right)\right)$ is an inflection point if and only if $q > p$, and it is the unique inflection point. This ends the proof. ∎

## 4.3 DeGroot Model

The model is now a description of the decision process through how the agents treat the received information by *learning* from their neighbors' decision process. Besides, the DeGroot Model has the assumption that vertices are influenced by their neighbors in their belief, resonating with the Bass Model for a nonzero rate $q$. Thus, we consider a graph with nodes and connections, as defined in Sect. 1.

At each time *step* $t$ (now $t \in \mathbb{N}$), vertex $i$ updates her belief $b_i(t) \in [0, 1]$ ($b_i(t) = 0$ if there is no belief at all, while $b_i(t) = 1$ if there is a belief that cannot be stronger). At each time step and for each node, the information about the belief of the closest neighbors is received *once*.

We introduce the *probability transition matrix* $P = (p_{ij})_{1 \leq i,\, j \leq n}$ as a *stochastic matrix*, that is

$$\sum_{j=1}^{n} p_{ij} = 1, \quad \forall i \in \{1, \ldots, n\}.$$

$p_{ij}$ represents the probability of information transition from vertex $i$ to vertex $j$ (see Sect. 2, Chap. 7 for a more formal definition). Each node $i \in \{1, \ldots, n\}$ starts with the belief $b_i(0) \in [0, 1]$, and the system evolves according to a Markov chain process: for any $t \in \mathbb{N}^*$ and vertex $i$, the *belief update* is given by

$$b_i(t) = \sum_{j=1}^{n} p_{ji} b_j(t-1).$$

If we now set as $B(t) = (b_i(t))_{1 \leq i \leq n}$ the vector of beliefs, we thus have

$$B(t) = P' B(t-1),$$

or

$$B(t) = (P')^t B(0).$$

The vector of beliefs then evolves as explained in the previous sections.

In particular, the question about reaching equilibrium (i.e., does $\lim_{t \to +\infty} (P')^t$ make sense?) is an important aspect of the model.

We give one condition below for which this equilibrium exists.

***Definition 4.4.3 (Primitive Matrix)*** *Let $P$ be a stochastic matrix. It is* primitive *if there exists $k \in \mathbb{N}$ such that $P^k$ has positive elements.*

***Theorem 4.4.4 (Perron)*** *Let $P$ be an $n \times n$ primitive stochastic matrix. The sequence of matrices $(P^k)_{k \in \mathbb{N}}$ converges and its limit $Q$ is a stochastic matrix of rank* 1.

***Proof*** (see [17])

We are endowing the space $\mathbb{R}^n$ with the norm defined by $\|X\| = \sum_{i=1}^{n} | x_i |$, for $X = (x_i)_{1 \leq i \leq n}$. We set $f(X) = \sum_{i=1}^{n} x_i$. We also write

$$\Sigma = \{X \in \mathbb{R}^n, \quad f(X) = 1\}.$$

$\Sigma$ is a convex and compact set of $\mathbb{R}^n$. Let $m \in \mathbb{N}^*$ be such that the elements of $P^m = \left(p_{ij}^m\right)_{1 \le i,j \le n}$ are all positive. We go through the proof step by step.

1. *For any $X \in \mathbb{R}^n$, we have $|\|P'X\|| < |\|X\||$ and $f(P'X) = f(X)$.*

   Indeed, we set $Y = PX$, so that

$$y_i = \sum_{j=1}^{n} p_{ji} x_j, \quad \forall i \in \{1, \ldots, n\}.$$

   Hence,

$$\|Y\| = \sum_{i=1}^{n} |y_i| \le \sum_{j=1}^{n} p_{ji} |x_i| \le \sum_{j=1}^{n} |x_i| = \|X\|.$$

   In addition, we have

$$f(Y) = \sum_{i=1}^{n} y_i = \sum_{i,j=1}^{n} p_{ji} x_j = \sum_{j=1}^{n} x_j = f(X).$$

2. *For any $X \in \mathbb{R}_+^n$, the elements of $P'X$ are non-negative, and the ones of $P^mX$ are higher than $\left(\min_{i,j \in \{1,\ldots,n\}} p_{ji}^m\right)\|X\|$.*

   We set $Y = P'X$ as in the previous step 1. Since $y_i = \sum_{j=1}^{n} p_{ji} x_j, \quad \forall i \in \{1, \ldots, n\}$, we see that $y_i \ge 0, \quad \forall\, i \in \{1, \ldots, n\}$. Setting $Z = (P')^m X = (z_i)_{1 \le i \le n}$, we have

$$z_i = \sum_{j=1}^{n} p_{ji}^m x_j \ge \left(\min_{i,j \in \{1,\ldots,n\}} p_{ji}^m\right) \sum_{j=1}^{n} x_j = \left(\min_{i,j \in \{1,\ldots,n\}} p_{ji}^m\right)\|X\|.$$

3. *If $X \in \mathbb{R}^n$ such that $f(X) = 0$, then $\|(P')^m X\| \le \left(1 - n\left(\min_{i,j \in \{1,\ldots,n\}} p_{ji}^m\right)\right)\|X\|$.*

   Let $Z = (P')^m X$ as in step 2. We have

$$z_i = \sum_{j=1}^{n} \left(p_{ji}^m - \left(\min_{i',j' \in \{1,\ldots,n\}} p_{j'i'}^m\right)\right) x_j,$$

so that

$$|z_i| \le \sum_{j=1}^{n} \left(p_{ji}^m - \left(\min_{i',j' \in \{1,\ldots,n\}} p_{j'i'}^m\right)\right)|x_j| = \sum_{j=1}^{n} p_{ji}^m |x_j| - \left(\min_{i',j' \in \{1,\ldots,n\}} p_{j'i'}^m\right)\|X\|.$$

By summing over $i$, we deduce that

$$\|Z\| \leq \|X\| - n\left(\min_{i,j\in\{1,\ldots,n\}} p_{ji}^m\right)\|X\|,$$

concluding.

4. *The set* $\Sigma$ *is stable by the map* $X \mapsto (P')^m X$ *and this map is contractant with respect to the norm* $\|\cdot\|$.

The stability of $\Sigma$ comes from point 1 above. Let $X_1, X_2$ be elements of $\Sigma$. We set $Y_1 = (P')^m X_1$ and $Y_2 = (P')^m X_2$. Then $f(X_2 - X_1) = 0$, $Y_2 - Y_1 = (P')^m(X_2 - X_1)$, and, from point 3, we have

$$\|Y_2 - Y_1\| \leq \left(1 - n\left(\min_{i,j\in\{1,\ldots,n\}} p_{ji}^m\right)\right)\|X_2 - X_1\|.$$

We conclude since $\left(1 - n\left(\min_{i,j\in\{1,\ldots,n\}} p_{ji}^m\right)\right) < 1$.

5. *The sequence of matrices* $((P^m)^k)_{k\in\mathbb{N}}$ *converges; its limit is a stochastic matrix* $Q$ *of rank* 1.

The Banach fixed point theorem allows to assert that the map $X \mapsto (P')^m X$ has a unique fixed point $K$ in $\Sigma$, and, for any vector $X \in \Sigma$, the sequence $(((P')^m)^k X)_{k\in\mathbb{N}}$ converges to $K \in \Sigma$ (since $\Sigma$ is compact, then closed). If now $(e_1, \ldots, e_n)$ is the canonical basis of $\mathbb{R}^n$, then since $e_i \in \Sigma$ for all $i \in \{1, \ldots, n\}$, then $(((P')^m)^k e_i)_{k\in\mathbb{N}}$ converges to $K$, which proves that the $i^{\text{th}}$ row of the sequence $(((P')^m)^k)_{k\in\mathbb{N}}$ converges to $K$. Hence $(((P')^m)^k)_{k\in\mathbb{N}}$, and $((P^m)^k)_{k\in\mathbb{N}}$ converges to the matrix $Q$ where all the rows are equal to $K$.

6. *The sequence of matrices* $(P^k)_{k\in\mathbb{N}}$ *converges to* $Q$.

The Euclidean division of $k$ by $m$ gives $k = mq + r$, $q \in \mathbb{N}$ and $0 \leq r < m$, thus $(P')^k = (P')^r((P')^m)^q$. Set $k \equiv r[m]$ (see Sect. 1, Chap. 5) and $k$ sufficiently high. We have that $(P')^k$ tends to $(P')^r Q$, and since all the rows of $(P')^r$ are elements of $\Sigma$ (by point 1) and the rows of $Q$ are the fixed point on $\Sigma$ for the map $X \mapsto P^m X$, we deduce that $(P')^r Q' = Q'$. It implies that all the subsequences (with $k$ being element of all the equivalence classes mod $m$) have the same limit, that is $Q'$, i.e., the sequence of matrices $((P')^k)_{k\in\mathbb{N}}$ converges to $Q'$, therefore $(P^k)_{k\in\mathbb{N}}$ converges to $Q$. ∎

Coming back to the equation $B(t) = (P')^t B(0)$, if $P$ is a primitive matrix, then equilibrium makes sense and coincides with the steady state given by $Q = \lim_{t\to+\infty} P^t$, that is $\lim_{t\to+\infty} B(t) = Q'B(0)$. In addition, if $B(0)$ is a stochastic matrix, it turns out from the proof above that $B(0) \in \Sigma$, which means that $\lim_{t\to+\infty} B(t)$ does not depend on the initial state $B(0)$. We say that $\lim_{t\to+\infty} B(t)$ is the *unique* stationary law of the associated Markov Chain, and rather write it as $(\mu_i)_{1\leq i\leq n}$.

## 4.4 Diffusion on the Peer-to-Peer Network

In this section, we sum up the important findings elaborated in the paper [12]. In this paper, we construct a diffusion theory that mathematically describes the diffusion of any kind of information through a peer-to-peer network. As specified in this article, an application is to construct a trading engine, since, from the law of diffusion, we can forecast flow of information. More specifically, we construct a system that allows the derivation of a differential equation – the diffusion equation – and we also derive the analytical solutions, in case the peer-to-peer network is undirected, but also, as a brand-new result, when the peer-to-peer network is directed. The latter case is more realistic, since if a miner possesses a piece of information, e.g., a new transaction, then she must share it as specified in the protocol, but the other miner who just received this piece of information from her does not need to send this back to her. In addition, adding delays and boundaries achieves an exhaustive construction, which it is possible to fit to the blockchain. Below, we present the main steps and results. The main difficulty was to elaborate a definition of a reasonable Laplacian operator on a graph.

Let $G = (V, E)$ be a graph (i.e., the peer-to-peer network) and $A$ its adjacency matrix. Let $f\colon \mathbb{R}_+ \times V \to \mathbb{R}$ be an information function (e.g., representing some cash flow proportion). Let $t \in \mathbb{R}_+$ be the time index.

***Definition 4.4.5*** *The* Directional Derivative *from $z \in V$ to $y \in V$ is given by*

$$\partial_z f(t,y) = (f(t,y) - f(t,z))A_{yz}.$$

It is just the finite difference between two connected vertices.

***Definition 4.4.6*** *The* Laplacian *at $x \in V$ is given by*

$$Lf(t,x) = \sum_{z \in V} \partial_z \partial_z f(t,x).$$

Hence, we prove the following theorem:

***Theorem 4.4.7*** *The* Diffusion Equation *on $G$ is given by*

$$\frac{\partial f}{\partial x}(t,x) = -DLf(t,x) + \Gamma(t,x),$$

*where*

- *$D \in \mathbb{R}_+$ is the diffusion coefficien;*
- *$\Gamma$ is the information creation/destruction rate at vertex $x \in V$ at time $t \in \mathbb{R}_+$.*

It is worth stressing that the negative sign in front at the first term on the right-hand side is necessary in our context (otherwise the solution diverges).

We also derive the analytical solution.

***Theorem 4.4.8 (Analytical Solution of the Diffusion Equation on Undirected Graph)*** *Let $(\mu_i)_{i \in V}$ and $(v_i)_{i \in V}$ be the eigenvalues and eigenvectors of L. The solution of the diffusion equation is given by*

$$f(t,x) = \sum_{i \in V} \left( C_i \, \mathrm{e}^{-D\mu_i t} + \int_0^t \mathrm{e}^{-D\mu_i (t-\tau)} \sum_{j \in V} \Gamma(\tau, j) v_i(j) \, \mathrm{d}\tau \right) v_i(x),$$

*where $(C_i)_{i \in V}$ is a family of real numbers.*

Without any loss of generality, we write $V = \{1, \ldots, n\}$ with $n \in \mathbb{N}^*$.

***Theorem 4.4.9 (Analytical Solution of the Diffusion Equation on Directed Graph)*** *Let $L = u \mathcal{S} v'$ be the Singular Value Decomposition of L, and let $(u_i)_{i \in V}$ and $(v_i)_{i \in V}$ be the columns of the matrices u and v, respectively. The solution for the directed diffusion is given by*

$$f(t,x) = (\alpha^u(t))' u(x) = (\alpha^v(t))' v(x),$$

*where*

$$\begin{pmatrix} \alpha^u(t) \\ \alpha^v(t) \end{pmatrix} = \begin{pmatrix} \mathrm{ch}(D v' u \mathcal{S} t) & -\mathrm{sh}(D v' u \mathcal{S} t) v' u \\ -\mathrm{sh}(D \mathcal{S} v' u t) u' v & \mathrm{ch}(D \mathcal{S} v' u t) \end{pmatrix} \cdot \begin{pmatrix} \alpha^u(0) \\ \alpha^v(0) \end{pmatrix} + \int_0^t \begin{pmatrix} \mathrm{ch}(D v' u \mathcal{S}(t-\tau)) & -\mathrm{sh}(D v' u \mathcal{S}(t-\tau)) v' u \\ -\mathrm{sh}(D \mathcal{S} v' u(t-\tau)) u' v & \mathrm{ch}(D \mathcal{S} v' u(t-\tau)) \end{pmatrix} \cdot U \mathrm{d}\tau,$$

*and*

$$U = \begin{pmatrix} \sum_{j \in V} \Gamma(\tau, j) u_1(j) \\ \vdots \\ \sum_{j \in V} \Gamma(\tau, j) u_n(j) \\ \sum_{j \in V} \Gamma(\tau, j) v_1(j) \\ \vdots \\ \sum_{j \in V} \Gamma(\tau, j) v_n(j) \end{pmatrix}.$$

One very exciting analogy is the expression for the propagation of a quantum particle throughout a network, see [22] and compare the equation expressions.

Figure 4.7 shows an important example for the implementation.

Adding now boundary conditions, we have the following theorem, expressed in an intuitive way.

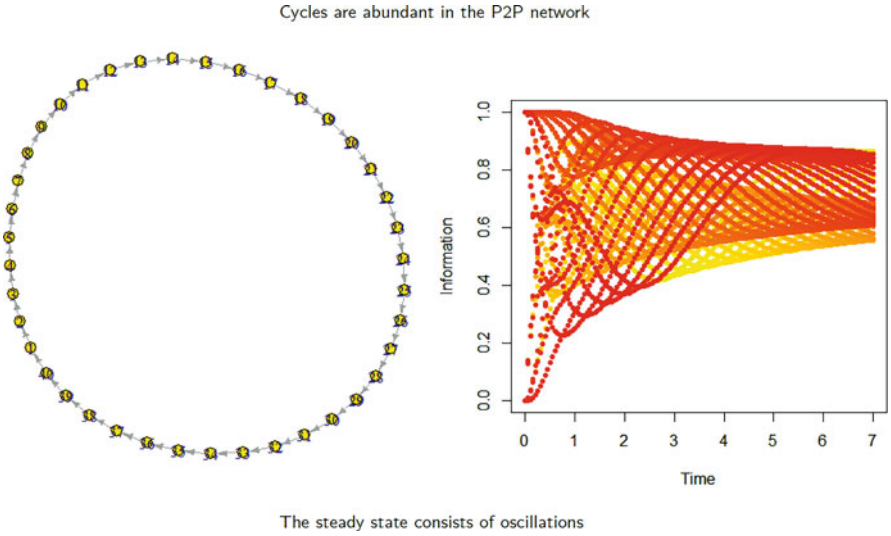


**Fig. 4.7** A clockwise-diffusion on a ring graph. Initially, some vertices have the full amount of information, while the others have none. From time $t = 0$, the diffusion is occurring: the information propagates clockwise and diffuses as a spread throughout the ring. On the right panel, one graph corresponds to one node. We observe a steady state with oscillations. This reminds of the behavior, e.g., RLC circuits in Physics. (Source: [12], 2022)

***Theorem 4.4.10 (Boundary Conditions)*** *Adding boundaries to a graph is the same as adding vertices and imposing an information creation/destruction rate on them. Specifically, if* $G = S \cup \overline{S}$ *where* $f(t,x) = B(t,x)$ *(fixed) for all* $(t,x) \in \mathbb{R}_+ \times \overline{S}$ *(boundary), then the creation/destruction rate becomes*

$$\tilde{\Gamma}(t,x) = \Gamma(t,x) + D \sum_{y \in \overline{S}} B(t,y)\, A_{yx}.$$

Adding now delays, giving a realistic model (a miner will not immediately treat the piece of information she receives, as she will need time to be aware), we have the following theorem.

***Theorem 4.4.11 (Delay)*** *The Laplacian operator L (and the adjacency matrix A) has its elements being Heaviside functions of time (*0 *below a threshold, and* 1 *above).*

Bearing all above in mind, the total information of the blockchain (see Sect. 1, Chap. 7 for the equivalence between the blockchain and the peer-to-peer network) is contained in one differential equation. A fit on the blockchain becomes relevant, at any instant, and a forecasting of flows is possible.

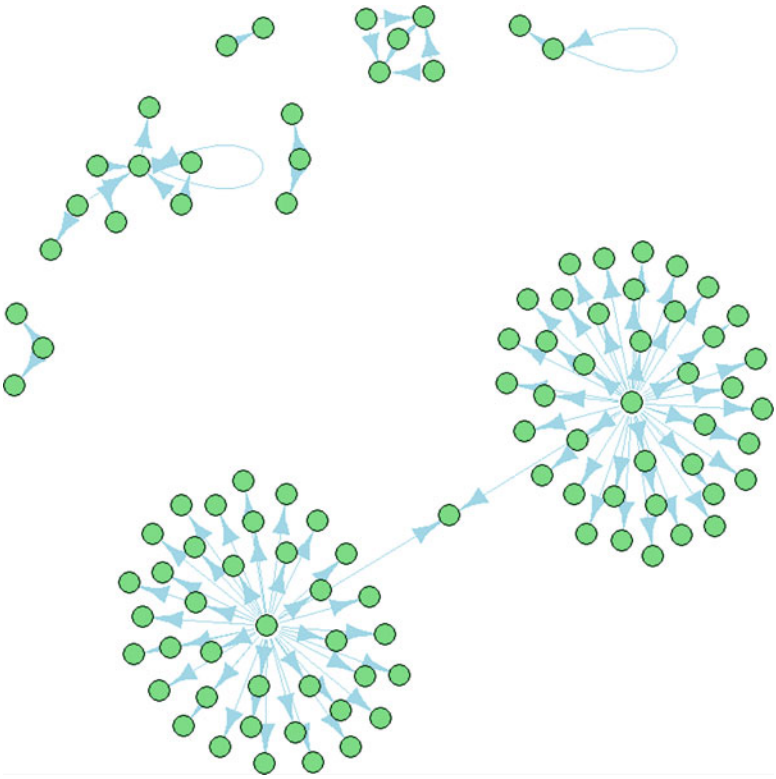

**Fig. 4.8** Outflows of bitcoins (real data) on the P2P network. This is issued from the list of outputs, 5100 to 5200, for block 680,000 of the BTC blockchain.

Thus, we end this chapter by an illustration of a tiny part of the Bitcoin Network (see Fig. 4.8), representing the outflows of bitcoins on real blockchain data.

In Fig. 4.8, two crypto-exchanges are represented (center of the star subgraphs) with buy-orders of bitcoins to several addresses (i.e., client accounts). Note that one client has one account at the two exchanges. We also observe additional transactions within users without proper intermediate institution.

# Chapter 5
# Halving and Cycles Theorem


This chapter is studying a pattern that has been identified in the Bitcoin halving process. As spurious as it might appear, the division by 2 of the block reward actually reveals some interesting characteristics in terms of available bitcoins. Collecting the 21 million bitcoins might appear to be impossible due to the protocol imposing the irreducible value of bitcoin, i.e., the Satoshi precision. The question of a change in the protocol through a soft fork might appear legitimate in order to allow the obtention of all the bitcoins. The reason is that minting new bitcoins is going to be less and less likely; this possibly could imply an inflation effect, thus motivating crypto-exchanges to possess more bitcoins than allowed by the current protocol.

I am especially signing this chapter for Bitstocks Ltd, where the CEO had the intuition of the digit pattern (Theorem 5.3.1), while I provided the proof.

## 1 Preliminaries on Modular Arithmetic

We recall that $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ is the set of relative integer numbers, while $\mathbb{N} = \{0, 1, 2, \ldots\}$ is the set of natural integer numbers, with $\mathbb{N}^* = \mathbb{N}\backslash\{0\}$. We call $\mathbb{D} = \left\{ \sum_{i=0}^{m} 10^{-i} a_i, m \in \mathbb{N}, a_i \in \mathbb{Z} \ \forall i \right\}$ the set of decimal numbers.

For any $n \in \mathbb{N}$, the set $\mathbb{Z}/n\mathbb{Z} = \left\{\overline{0}, \overline{1}, \ldots, \overline{n-1}\right\}$ is such that each element is a set, and, for instance, $\overline{2}$ is the *equivalence class* for the equivalent relationship $x - 2 = p \times 9$, for some $x, p \in \mathbb{Z}$. We then write $x \equiv 2[9]$ or $x \in \overline{2}$. More generally, if $t \in \{0, 1, \ldots, 8\}$ and if $x = t + p \times 9$ for some $p \in \mathbb{Z}$, we write $x \in \overline{t}$ or $x$ is a representative for the class $\overline{t}$. As an example, we have $10 \equiv 1\ [9]$ since $10 - 1 = 1 \times 9$ (here $p = 1$), and we write $10 \in \overline{1}$ and 10 is a representative for the class $\overline{1}$ in $\mathbb{Z}/9\mathbb{Z}$. Another example: $31 \equiv 4\ [9]$ since $31 - 4 = 3 \times 9$ (here $p = 3$), and we write $31 \in \overline{4}$ and 31 is a representative for the class $\overline{4}$ in $\mathbb{Z}/9\mathbb{Z}$.



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_5

It is important to stress that the set $\mathbb{Z}/9\mathbb{Z}$ endowed with the standard additive operation + is a group, and in particular, if $a \equiv b[9]$ and $a^{'} \equiv b^{'}[9]$, then $a + a^{'} \equiv b + b^{'}[9]$. We have the same property for the standard product $\times$. Thus, the triplet $(\mathbb{Z}/9\mathbb{Z}, +, \times)$ forms a ring.

Bearing all above in mind, for any $i \in \mathbb{N}$, $i[9]$ will designate the smallest number in the set $\{0, 1, 2, \ldots, 9\}$ in the sequence of this chapter, e.g., $1[9] = 1$, or $11[9] = 2$. By "modular arithmetic," we mean arithmetic on a set $\mathbb{Z}/n\mathbb{Z}$.

***Definition 5.1.1 (Digital Root)*** *The digital root of a decimal number is the sum of its digits iteratively proceeding until reaching a number in the set* $\{0, 1, 2, \ldots, 9\}$. *We call D the digital root function. We extend the definition set of D to decimal numbers. We endow D with the property of being an additive (semi-)group homomorphism, that is* $D(0) = 0$, *and*

$$D(a + b) = D(D(a) + D(b)), \quad \forall a, b \in \mathbb{D}.$$

For example, $D(1) = 1$, $D(11) = 2$, $D(139) = 4$, $D(0.5) = 5$, or $D(0.25) = 7$. This implies that, e.g., $D(0.5) = D(5) = D(50) = 5$. This generalizes straight forwardly. In addition, $D(11) = 2 = D(D(1) + D(10))$, or $D(58) = 4 = D(6 + 7) = D(D(24) + D(34)) = D(D(50) + D(8))$.

***Proposition 5.1.2*** *We have the correspondence between D and modular arithmetic*

$$D(n) = 1 + (n - 1)[9], \quad \forall n \in \mathbb{N}^*.$$

***Proof***

We proceed by mathematical induction on $n \in \mathbb{N}^*$. If $n = 1$, then $D(1) = 1 = 1 + 0$ [9]. Suppose that $D(n) = 1 + (n - 1)[9]$ for some $n \in \mathbb{N}^*$. We have

$$D(n + 1) = D(D(n) + D(1)) = D(D(n) + 1) = D(1 + (n - 1)[9] + 1[9]).$$

We used $1 = 1[9]$. Thus,

$$D(n + 1) = D(1 + n[9]) = 1 + n[9].$$

This completes the proof. ∎

***Proposition 5.1.3*** *We have*

$$D(ab) = D(D(a)D(b)), \quad \forall a, b \in \mathbb{D},$$

*which makes D being a (semi-)ring homomorphism.*

***Proof***

We use Proposition 5.1.2 three times. For any $a, b \in \mathbb{D}$, and bearing in mind that $(c[9])[9] = c[9]$ for any $c \in \mathbb{D}$, we have

$$D(D(a)D(b)) = 1 + \{(1 + (a-1)[9])(1 + (b-1)[9]) - 1\}[9] = 1$$
$$+ \{1 + (a - 1 + b - 1 + (a-1)(b-1))[9] - 1\}[9] = 1 + (ab-1)[9] = D(ab).$$

Finally, note that the equation works for $a = b = 1$: $D(1) = 1 = D(D(1)D(1))$. Thus, $D$ is a (semi-)ring homomorphism. This completes the proof. ∎

Finally, we call E(·) the integer part function, and for any integer numbers $a, b \in \mathbb{Z}$, gcm$(a, b)$ is the *greatest common divisor* of $a$ and $b$.

## 2 Bitcoin Halving

***Definition 5.2.1 (Halving)*** *Let $n \in \mathbb{N}$. The* Block reward *(see Fig. 4.2) is equal to $50/2^N$ bitcoins if and only if focusing on block $B_n$ (see Definition 2.2.1), $210{,}000\ N \leq n < 210{,}000\ (N + 1)$, for some $N \in \mathbb{N}$.*

Thus, for the first 210,000 blocks, the block reward is 50 bitcoins, while for the second 210,000 blocks, the block reward is 25 bitcoins, and so on.

From Definition 5.2.1, we can calculate the theoretical total number of bitcoins. In fact, the total number of minted bitcoins from block 210,000 $N$ to block 210,000 $(N + 1) - 1$ is $210{,}000 \times 50/2^N$, for any $N \in \mathbb{N}$. Hence, the theoretical total number of bitcoins is

$$\sum_{N=0}^{+\infty} 210{,}000 \frac{50}{2^N} = 21{,}000{,}000 \text{ bitcoins}.$$

## 3 The Digit Pattern

The digit pattern states that the digital root of the block reward is a 6-periodic function of $N$, while the digital root of the number of bitcoins within each series of 210,000 blocks is a 2-periodic function of $N$.

We are rigorously stating these two assertions as follows, and explain them before giving the proof.

***Theorem 5.3.1 (Halving Digit Pattern)*** *For any $(i, N) \in \{1, 2, 3, 4, 5, 6\} \times \mathbb{N}$, we have*

$$D\left(5^{i+6N}\right) = f(i),$$

*where*

$$f(i) = \begin{cases} 5 & \text{if } i = 1 \\ 7 & \text{if } i = 2 \\ 8 & \text{if } i = 3 \\ 4 & \text{if } i = 4 \\ 2 & \text{if } i = 5 \\ 1 & \text{if } i = 6 \end{cases}$$

*This equation reflects the digital root evolution of the block reward.*
*In addition, we have*

$$D\left(5^{i+6N} \times 21\right) = g(i),$$

*where*

$$f(i) = \begin{cases} 3 & \text{if } i \in \{2, 4, 6\} \\ 6 & \text{if } i \in \{1, 3, 5\} \end{cases}$$

*This equation reflects the digital root evolution of the number of minted bitcoins.*

The first equation in Theorem 5.3.1 is the translation for the following six equations (for any $N \in \mathbb{N}$):

$$D(50) = D\left(\frac{50}{2^{6N}}\right) = 5,$$

$$D(25) = D\left(\frac{25}{2^{6N}}\right) = 7,$$

$$D(12.5) = D\left(\frac{12.5}{2^{6N}}\right) = 8,$$

$$D(6.25) = D\left(\frac{6.25}{2^{6N}}\right) = 4,$$

$$D(3.125) = D\left(\frac{3.125}{2^{6N}}\right) = 2,$$

$$D(1.5625) = D\left(\frac{1.5625}{2^{6N}}\right) = 1.$$

Figure 5.1 gives an illustration of this result.

The second equation in Theorem 5.3.1 is the translation for the following six equations (for any $N \in \mathbb{N}$):

$$D(50 \times 210{,}000) = D\left(\frac{50}{2^{6N}} \times 210{,}000\right) = 6,$$

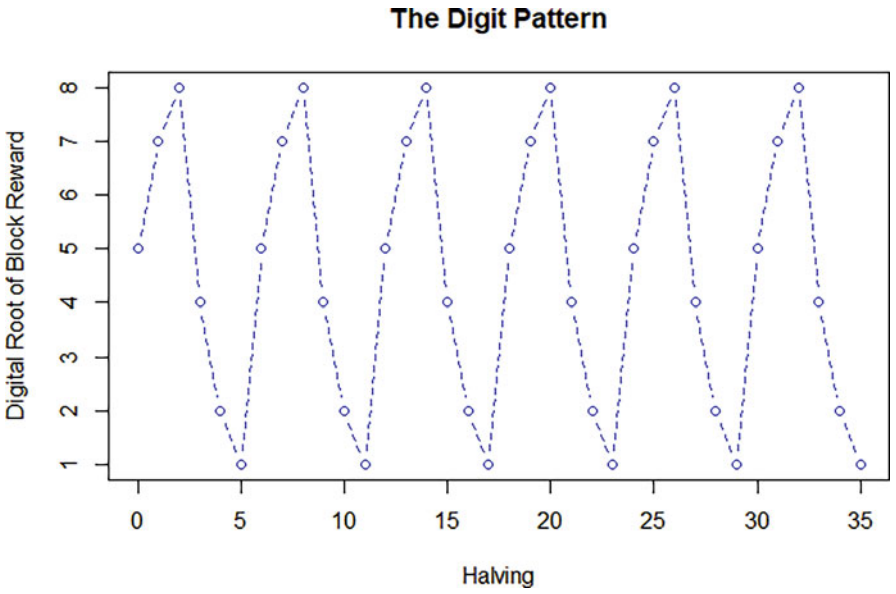


**Fig. 5.1** Digit Pattern: the digital root of the block reward is a 6-periodic function of time. For instance, for the first 210,000 blocks, the block reward is 50, whose digital root is 5 (extreme left blue point). (Source: Julien Riposo, 2023)

$$D(25 \times 210{,}000) = D\left(\frac{25}{2^{6N}} \times 210{,}000\right) = 3,$$

$$D(12.5 \times 210{,}000) = D\left(\frac{12.5}{2^{6N}} \times 210{,}000\right) = 6,$$

$$D(6.25 \times 210{,}000) = D\left(\frac{6.25}{2^{6N}} \times 210{,}000\right) = 3,$$

$$D(3.125 \times 210{,}000) = D\left(\frac{3.125}{2^{6N}} \times 210{,}000\right) = 6,$$

$$D(1.5625 \times 210{,}000) = D\left(\frac{1.5625}{2^{6N}} \times 210{,}000\right) = 3.$$

Note that the number 9 does not belong to the image by the digital root function.

***Proof***

We prove the first equation. Fix $i \in \{1, 2, 3, 4, 5, 6\}$. We have

$$D\left(\frac{50}{2^{i-1}}\right) = D\left(D(50)D\left(\frac{1}{2^{i-1}}\right)\right) = D\big(D(5)D\big(0.5^{i-1}\big)\big).$$

When $i = 1$ (resp. 2, etc.), this is related to the first term of the first (resp. second, etc.) formula among the six above. We have, if $i \neq 1$:

$$D\left(0.5^{i-1}\right) = D(D(0.5) \times \ldots \times D(0.5))$$

with a number of factors of $i - 1$ (the case $i = 1$ is trivial). Thus,

$$D\left(0.5^{i-1}\right) = D(D(5) \times \ldots \times D(5)) = D\left(5^{i-1}\right).$$

Therefore,

$$D\left(\frac{50}{2^{i-1}}\right) = D\left(5^{i}\right).$$

Given that we also have ($N \in \mathbb{N}$, and replace $i - 1$ with $6N$ above)

$$D\left(\frac{1}{2^{6N}}\right) = D\left(5^{6N}\right),$$

we then deduce that

$$D\left(\left(\frac{50}{2^{i-1}}\right) \times \left(\frac{1}{2^{6N}}\right)\right) = D\left(5^{i+6N}\right), \quad \forall N \in \mathbb{N}.$$

This proves that the first equation reflects the digital root evolution of the block reward. Now, we may prove the first equation enunciated in the theorem by mathematical induction on $N \in \mathbb{N}$, for all $i \in \{1, 2, 3, 4, 5, 6\}$. The equation is straightforward to prove for $N = 0$ (left to the reader).

Suppose this first equation is true for a given $N \in \mathbb{N}$, and fix $i \in \{1, 2, 3, 4, 5, 6\}$. Since $D(5^6) = 1$, we have

$$\begin{aligned} D\left(5^{i+6(N+1)}\right) &= D\left(5^{i+6N} \times 5^6\right) = D\left(D\left(5^{i+6N}\right)D\left(5^6\right)\right) = D\left(f(i)D\left(5^6\right)\right) \\ &= D(f(i) \times 1) = f(i) \end{aligned}$$

since $f(i) \in \{1, 2, 4, 5, 7, 8\} \subset \{0, 1, \ldots, 9\}$. The equation is true for all $n \in \mathbb{N}$, which achieves the proof for the first equation.

We may directly prove the second equation (proving that it is reflecting the digital root of the number of bitcoins could be done the same way as previously).

Fix $i \in \{1, 2, 3, 4, 5, 6\}$. We have

$$D\left(5^{i+6N} \times 21\right) = D\left(D\left(5^{i+6N}\right)D(21)\right) = D(f(i) \times 3) = D(3f(i)).$$

We derive case by case, i.e., for the six values of $i$, and we find that

$$D(3f(i)) = \begin{cases} 3 & \text{if } i \in \{2,4,6\} \\ 6 & \text{if } i \in \{1,3,5\} \end{cases}.$$

This finally achieves the proof. ∎

The fact that the digital root is a periodic function of time reveals a symmetry in the halving process. In reality, this symmetry is actually broken from the $10^{\text{th}}$ halving. This comes from the fact that the Bitcoin protocol imposes an irreducible value for bitcoin, which is 1 Satoshi Unit:

$$1\text{S} = 10^{-8} \text{ bitcoin}.$$

On the blockchain, there is no amount whose precision is below the Satoshi limit. From the $10^{\text{th}}$ halving, the block reward should be $50/2^{10} = 0.048828125$, but actually is 0.04882812. Therefore, from the $33^{\text{rd}}$ halving, the block reward will be 0. Figure 5.2 shows an illustration for the first block reward values.

In order to consider the Satoshi Unit, one way to proceed is to truncate the digital root function up to $10^{-8}$. We thus introduce the following definition.

***Definition 5.3.2 (Truncated Digital Root)*** *The truncated digital root of a number is the sum of its digits iteratively proceeding until reaching a number in the set* $\{0, 1, 2, \ldots, 9\}$*, but by only considering its digits up to the* $10^{-8}$ *precision included. We call* $\tilde{D}$ *the truncated digital root function.*

The function $\tilde{D}$ is still a (semi-)ring homomorphism, that is it satisfies the equation in Definition 5.1.1 and Proposition 5.1.2. We actually have the following proposition.

***Proposition 5.3.3*** *The truncated digital root writes as the digital root as*

$$\tilde{D}(a) = D\big(\mathrm{E}\big(a\ 10^8\big)\big), \qquad \forall a \in \mathbb{D}.$$

***Proof***

Indeed, the number $\mathrm{E}(b\ 10^8)/10^8$ is exactly $b$ up to precision of $10^{-8}$, for any $b \in \mathbb{D}$. Thus,

$$\tilde{D}(b) = D\left(\frac{\mathrm{E}\big(b\ 10^8\big)}{10^8}\right)$$

by definition. We conclude by the Proposition 5.1.3. ∎

| Halving | Block Reward (bitcoin) | Satoshi Numbers |
|---|---|---|
| 0 | 50 | |
| 1 | 25 | |
| 2 | 12,5 | |
| 3 | 6,25 | |
| 4 | 3,125 | |
| 5 | 1,5625 | |
| 6 | 0,78125 | |
| 7 | 0,390625 | |
| 8 | 0,1953125 | |
| 9 | 0,09765625 | |
| 10 | 0,048828125 | 5 |
| 11 | 0,0244140625 | 25 |
| 12 | 0,01220703125 | 125 |
| 13 | 0,006103515625 | 5625 |
| 14 | 0,0030517578125 | 78125 |
| 15 | 0,00152587890625 | 890625 |
| 16 | 0,000762939453125 | 9453125 |
| 17 | 0,0003814697265625 | 97265625 |
| 18 | 0,00019073486328125 | 486328125 |
| 19 | 0,000095367431640625 | 7431640625 |
| 20 | 0,0000476837158203125 | 37158203125 |

**Fig. 5.2** Digits of the block rewards and digits below the Satoshi precision, giving the Satoshi Numbers. (Source: Julien Riposo, 2023)

With this expression of the truncated digital root, we could plot the evolution of the values for $\tilde{D}\left(50/2^N\right)$ with respect to $N$, representing the $N^{\text{th}}$ halving. Figure 5.3 shows the result.

The Effective total number of bitcoins is therefore

$$\sum_{N=0}^{+\infty} 210{,}000\,\frac{\mathrm{E}\left(\frac{50}{2^N}10^8\right)}{10^8} = \sum_{N=0}^{32} 210{,}000\,\frac{\mathrm{E}\left(\frac{50}{2^N}10^8\right)}{10^8} < \sum_{N=0}^{+\infty} 210{,}000\,\frac{50}{2^N}$$

$$= 21{,}000{,}000.$$

Although it is very close to 21,000,000, the estimation of this sum is left to the reader.

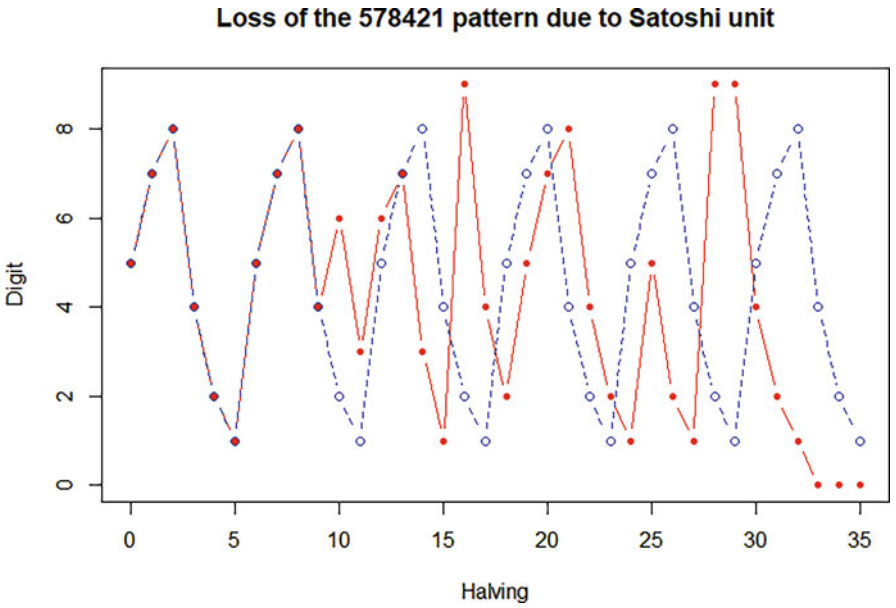


**Fig. 5.3** Real Digit Pattern: from the $10^{\text{th}}$ halving, the Satoshi precision in the block reward is reached. The periodicity is broken. (Source: Julien Riposo, 2023)

## 4 The Cycles Theorem

The object of this section is to reveal an interesting trend in the block reward digits by removing the Satoshi precision. In fact, any last digits repeat infinitely in cycles, allowing to dress a binary tree where the nodes are the last digits of the block rewards.

More specifically, for any integer $N \geq 2$, the last $N$ digits of the halvings are cycling with period $2^{N-2}$. We start this section with some observations, and then prove this statement.

### *4.1 Observation of Some Cycles*

We observe that, for any $N \in \mathbb{N}$, we have

$$\frac{50}{2^N} = \frac{5^{N+1}}{10^{N-1}}.$$

Hence, intuitively, focusing on the last $N$ digits (here $N \in \mathbb{N}^*$) means focusing on the number $10^{N-1}\frac{50}{2^N} = 5^{N+1}$ modulo $10^N$.

First, we have

$$5^1 \equiv 5[10],$$
$$5^2 = 25 \equiv 5[10].$$

More generally, we can easily, by mathematical induction, show that

$$5^N \equiv 5[10], \quad \forall N \in \mathbb{N}^*.$$

This proves that the last digit of the block rewards is always 5, which is a quite intuitive statement (since halving means multiplying by 5 another number whose last digit is always 5).

We now focus on the last two digits. We have

$$5^2 \equiv 25[100],$$
$$5^3 = 125 \equiv 25[100],$$
$$5^4 = 625 \equiv 25[100].$$

More generally, we can easily, by mathematical induction, show that

$$5^N \equiv 25[100], \quad \forall N \in \mathbb{N}^* \backslash \{1\}.$$

This proves that the last two digits of the block rewards are always 2 followed by 5, i.e., the period of this cycle is $2^{2-2} = 1$, i.e., the last two digits 25 repeat all the time (see Fig. 5.2).

We now focus on the last three digits. We have

$$5^3 \equiv \mathbf{125}[1000],$$
$$5^4 \equiv 625[1000],$$
$$5^5 = 3125 \equiv \mathbf{125}[1000],$$
$$5^6 = 15625 \equiv 625[1000].$$

More generally, we can show, by mathematical induction, that

$$5^{2l} \equiv 625[1000], \quad \forall l \in \mathbb{N}^* \backslash \{1\},$$
$$5^{2l+1} \equiv 125[1000], \quad \forall l \in \mathbb{N}^*.$$

This proves that the last digits of the block rewards are always either 125 or 625, starting from the second halving. The period here is $2^{3-2} = 2$. In other words, the last digits 125 and 625 systematically alternate.

We finish this section by focusing on the last four digits. We have

$$5^4 \equiv \mathbf{625}[10000],$$
$$5^5 \equiv 3125[10000],$$
$$5^6 = 15625 \equiv 5652[10000],$$
$$5^7 = 78125 \equiv 8125[10000],$$
$$5^8 = 390625 \equiv \mathbf{625}[10000],$$
$$5^9 = 1953125 \equiv 3125[10000],$$
$$5^{10} = 9765625 \equiv 5625[10000],$$
$$5^{11} = 48828125 \equiv 8125[10000].$$

We see that the digits 625, 3125, 5625, 8125 repeat in order for a period of $2^{4-2} = 4$. In the next section, we are going to prove the general statement, which we name Cycles Theorem.

## 4.2 The Cycles Theorem

***Theorem 5.4.1 (The Cycles Theorem)*** *The last digit of the block rewards is always 5. In addition, for the $N \geq 2$ last digits, we have*

$$5^{l+2^{N-2}} \equiv 5^l \left[10^N\right], \quad \forall N \in \mathbb{N}^* \backslash \{1\}, \quad \forall l \in \mathbb{N} \backslash \{0, \ldots, N-1\}.$$

This means that, for $N \geq 2$, the last $N$ digits of the halvings are cycling with period $2^{N-2}$ from the $(N-1)^{\text{th}}$ halving. In other words, this means that the $N$ final digits of the halvings determine a *cycle* of period $2^{N-2}$. Again said otherwise, $N$ last given digits will exactly reappear $2^{N-2}$ halvings later.

We will use the following lemma.

***Lemma 5.4.2*** *Suppose $l \in \mathbb{N}$. Then $5^{2^l} - 1$ is divisible by $2^{l+2}$, but not by $2^{l+p}$, $p \geq 3$.*

***Proof***

We can write, by mathematical induction on $l \in \mathbb{N}$, that

$$5^{2^l} - 1 = 4(5+1)\left(5^2+1\right)\ldots\left(5^{2^{l-1}}+1\right).$$

In fact, if $l = 0$, $5^1 - 1 = 4$ and for some $l \in \mathbb{N}$, we have

$$5^{2^{l+1}} - 1 = \left(5^{2^l}\right)^2 - 1 = \left(5^{2^l} - 1\right)\left(5^{2^l} + 1\right) = 4(5+1)\left(5^2+1\right)\ldots\left(5^{2^l}+1\right).$$

We see that 4 divides $5^{2^l} - 1$. We are going to see that 2 but not 4 divides each factor $5^{2^k} + 1$, for all $k \in \{0, \ldots, l-1\}$. First note that $5^{2^k}$ is odd, and therefore $5^{2^k} + 1$ is even, hence divisible by 2. In addition, it is worth pointing out that

$$5^{2^k} \equiv 25[100] \Rightarrow 5^{2^k} + 1 \equiv 26[100].$$

Therefore, the last two digits form a number that is not divisible by 4, and $5^{2^k} + 1$ is not divisible by 4 (another way to show it is to prove that $* \, 5^{2^k} + 1$ is not 0 modulo 4 – otherwise, simply note that 26 is not divisible by 4). Bearing all above in mind, we proved that $5^{2^l} - 1$ is divisible by $2^{l+2}$, but not by $2^{l+3}$. We conclude the proof. ∎

***Proof of Theorem 5.4.1*** Let $N \geq 2$ and $T = 2^{N-2}$. The idea is first to demonstrate the following points, which straightforwardly imply the result:

- $5^{l+T} \equiv 5^l[2^N], \quad \forall\, l \in \mathbb{N}$;
- $5^{l+T} \equiv 5^l[2^N] \Leftrightarrow 5^{l+T} \equiv 5^l[10^N], \quad \forall\, l \in \mathbb{N}\backslash\{0, \ldots, N-1\}$.

Regarding the first point, we introduce the Euler's totient function $\phi$ defined as

$$\phi(N) = \text{Card}\{k \in \{1, \ldots, N\}, \text{gcm}(k, N) = 1\},$$

that is, $\phi(N)$ is the cardinal of the set of numbers lower than $N$ and co-prime with $N$. The *Euler's theorem* states that

$$5^{\phi\left(2^N\right)} \equiv 1\left[2^N\right].$$

We have

$$\phi\left(2^N\right) = 2^N \times \left(1 - \frac{1}{2}\right) = 2^{N-1}.$$

The order of 5 (see Sect. 1.2, Chap. 3) modulo $2^N$ therefore divides $\phi(2^N) = 2^{N-1}$, hence it is a power of 2. The order is $2^\alpha$, where $\alpha \in \{0, \ldots, N-1\}$ and we have $5^{2^\alpha} \equiv 1\left[2^N\right]$ by definition of the order of 5 modulo $2^N$.

From Lemma 5.4.2, setting $l = N - 2$, we have

$$5^{2^{N-2}} \equiv 1\left[2^N\right], \qquad (*)$$

Thus, $\alpha < N - 1$. In addition, $\alpha \notin \{0, \ldots, N-3\}$ since, otherwise $\left(5^{2^{N-1}} - 1\right)/2^{\alpha}$ would not be divisible by 2 (because it would be 1), which is impossible according to Equation $(*)$. Another equivalent way to see it is that, according to Lemma 5.4.2, $5^{2^{\alpha}} - 1$ is *not* divisible by $2^{\alpha+p}$, $p \geq 3$, which contradicts Equation $(*)$ if $\alpha \in \{0, \ldots, N-3\}$.

We conclude that $\alpha = N - 2$, and the order of 5 modulo $2^N$ is $T$. Therefore,

$$5^{l+T} = 5^l 5^T \equiv 5^l \left[2^N\right], \quad \forall l \in \mathbb{N},$$

which is exactly the first point in the theorem.

Regarding the second point, suppose $5^{l+T} \equiv 5^l[2^N]$ for some $l \in \mathbb{N}$. This means that there exists $k \in \mathbb{Z}$ such that $5^{l+T} - 5^l = 2^N k$, hence $5^l(5^T - 1) = 2^N k$. According to Lemma 5.4.2, $2^N$ divides $5^T - 1$. It turns out that $5^l$ divides $k$. In particular, let us choose $l \geq N$. In this case, $5^N$ divides $k$, i.e., there exists $k' \in \mathbb{Z}$ such that $k = 5^N k'$.

Therefore, we have

$$5^l\left(5^T - 1\right) = 2^N 5^N k' = 10^N k'.$$

Hence $5^{l+T} \equiv 5^l[10^N]$, for any $l \geq N$.

Conversely, if $5^{l+T} \equiv 5^l[10^N]$, then there exists $k \in \mathbb{Z}$ such that $5^{l+T} - 5^l = 10^N k$, i.e., $5^{l+T} - 5^l = 2^N(5^N k)$. Since $5^N k \in \mathbb{Z}$, we conclude that $5^{l+T} \equiv 5^l[2^N]$. This proves the second point, which concludes the proof. ∎

## 4.3 Binary Tree Representation of Halving

Figure 5.4 shows the binary tree illustration of the last digits breakdown.

The order of appearance of the last five digits are 5, 25, 125, 625, **3125**, 15625, 78125, 90625, 53125, 65625, 28125, 40625, **03125**, and so on. For a given level (a given modulo, i.e., a given column in Fig. 5.4), we change branch in the tree: if the number is in the upper branch, then we are sure the next one is in the lower branch, then we come back to the upper branch. Every branch will be first visited exactly once, and then they are revisited a second time in the same order. This pattern is intrinsically the Cycles Theorem.

The binary tree reveals an important behavior of the proportion of appearances of these numbers. We take an example with the last digits 03125. These last digits appear at the fourth halving for the first time. On average these digits appear 1/8 times in the whole halving process, because the period of the related cycle is 8. In fact, the proportion of appearances is the frequency of appearance, which is the inverse of the period. As another example, 5 and 25 appear all the time.

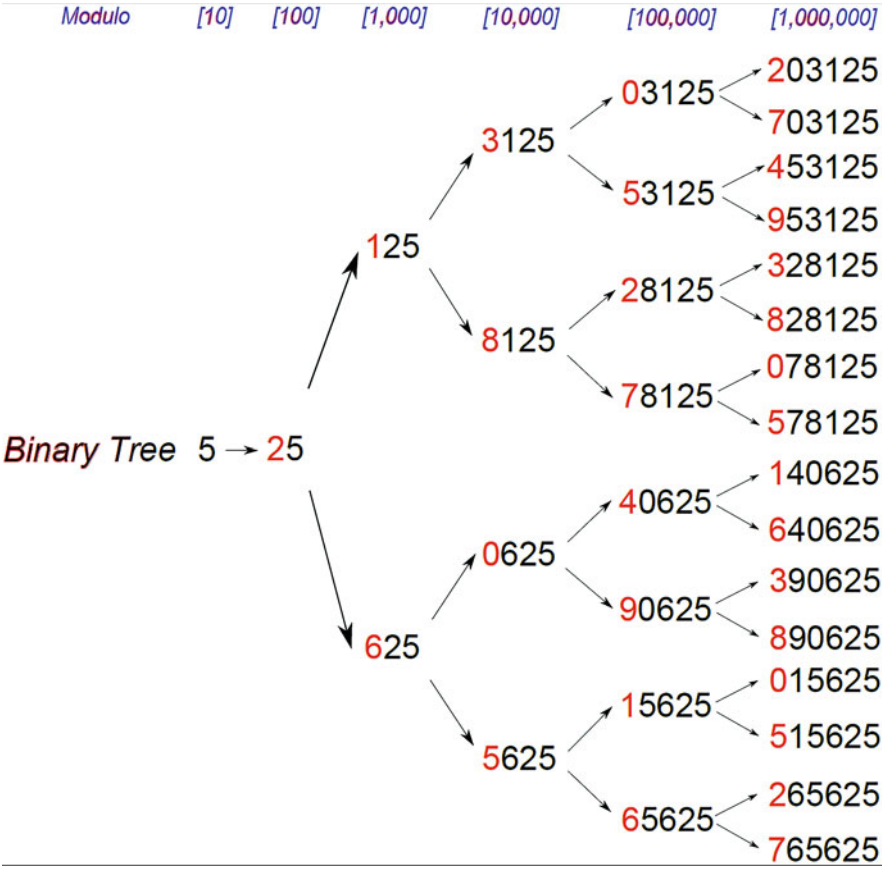


**Fig. 5.4** The last digits behave as a *binary tree*, or a *tournament*. Each time there is a new non-zero digit appearing, this new digit has only two possible values. This configuration allows to calculate the overall proportion last digits appear in the halving (see text). (Source: Julien Riposo, 2023)

It turns out that the binary tree is an implication of the Cycles Theorem: every number appearing at the $n^{\text{th}}$ column, giving the $N$ last digits, appears with a frequency of $1/2^{N-2}$ (provided $N \geq 2$) in the whole halving process.

## *4.4 On an Algorithm for Predicting the Satoshi Digits*

From one level to another in the binary tree (see Fig. 5.4) predicting the red digits would allow travelling in this tree. The development of a path from little-amount Satoshis to bigger-amount Satoshis, and conversely, would use this algorithm to reveal further causalities between the coins.

We will need the following theorems, which we also prove in the following.

***Theorem 5.4.3*** *The sum of all digits at position N from the right (with $N > 2$), of the numbers defining one $2^{N-2}$-length cycle, is congruent to 7 modulo 9.*

For example, modulo 1000 ($N = 4$), the four possible numbers are 3125, 8125, 0625, and 5625. Thus, 3 + 8 + 0 + 5 ≡ 7[9] (see Fig. 5.4).

***Proof***
First, we note that, for any $N > 1$ and $l \in \{N, \ldots, N-1+2^{N-2}\}$ (the length here is one period $2^{N-2}$), we have

$$\exists m \in \mathbb{N} \quad 5^l \equiv 5^N(4m+1)\left[10^N\right]. \qquad (**)$$

Indeed, on the one hand, 4 divides $2^N$, and also $5^{l-N}-1$, which is obvious when $l = N$, and when $l > N$, we have (see also Lemma 5.4.2)

$$5^{l-N} - 1 = 4\prod_{k=1}^{l-N}\left(5^k+1\right).$$

Bearing this in mind, there exists $m \in \mathbb{N}$ and $\alpha \in \mathbb{Z}$ such that $4m = (5^{l-N}-1) - 2^N\alpha$, which is equivalent to

$$5^l - 5^N(4m+1) = \alpha 5^N 2^N = \alpha 10^N,$$

hence Equation (∗∗). It is worth pointing out that one value of $l$ gives one value of $m$, i.e., the map $l \mapsto m$ is injective.

Next, for any $N > 1$ and $l \in \{N, \ldots, N-1+2^{N-1}\}$, note that the number $5^l - 10^N \,\mathrm{E}\left(\dfrac{5^l}{10^N}\right)$ is the last $N$ digits of the number $5^l$ (it is $5^l[10^N]$). Then let

$$\Gamma_N = \bigcup_{l=N}^{N-1+2^{N-1}} \left\{5^l - 10^N \,\mathrm{E}\left(\frac{5^l}{10^N}\right)\right\}.$$

For example, $\Gamma_2 = \{25\}$, $\Gamma_3 = \{125,625\}$, etc. (see Fig. 5.4). From Equation (∗∗) and the injectivity of $l \mapsto m$, we can write that

$$\Gamma_N = \bigcup_{m=0}^{2^{N-1}-1} \left\{5^N(4m+1)\right\}.$$

Let $\delta_N = \sum_{x\in\Gamma_N} x$ and $\gamma_N = \sum_{x\in\Gamma_N} \mathrm{E}\left(\frac{x}{10^{N-1}}\right)$. The number $\delta_n$ is the sum of all the numbers in $\Gamma_N$, while the number $\gamma_N$ is the sum of the $N^{\text{th}}$ digits (from the right) of all the numbers in $\Gamma_N$.

We want to prove that $\gamma_n \equiv 7[9]$. We now set $N > 2$.

First, note that the last $N$ digits repeat once when the last $N-1$ digits repeat twice (consequence of Theorem 5.4.1), so one contribution to $\delta_N$ is $2\delta_{N-1}$. Second, another contribution to $\delta_N$ is the sum of the $N^{\text{th}}$ digits (from the right), that is $10^{N-1}\gamma_N$. In fact, we have

$$\delta_N = 10^{N-1}\gamma_N + 2\delta_{N-1}.$$

Since $b \equiv 10^a b[9]$ for any $a \in \mathbb{N}$ and $b \in \mathbb{Z}$ (as $10^a - 1$ is divisible by 9), we have $\gamma_N \equiv 10^{N-1}\gamma_N[9]$, thus the previous equation becomes

$$\gamma_N \equiv \delta_N - 2\delta_{N-1}[9].$$

We first calculate $\delta_N$ using the second expression for $\Gamma_N$, and we then deduce $\gamma_N$. We have

$$\begin{aligned}\delta_N &= \sum_{m=0}^{2^{N-2}-1} 5^N(4m+1) = 5^N\left(4\frac{2^{N-2}\left(2^{N-2}-1\right)}{2} + 2^{N-2}\right) = \left(5^2 \times 5^{N-2}\right)\\ &\times 2^{N-2}\left(2\left(2^{N-2}-1\right)+1\right) = 25 \times 10^{N-2}\left(2\left(2^{N-2}-1\right)+1\right)\\ &\equiv 25\left(2^{N-1}-1\right)[9] \equiv 7\left(2^{N-1}-1\right)[9],\end{aligned}$$

since $25 \equiv 7[9]$. Replacing in the previous equation, we therefore have

$$\gamma_N \equiv \left(7\left(2^{N-1}-1\right) - 2 \times 7\left(2^{N-2}-1\right)\right)[9] = 7 \times (0+1)[9],$$

and finally

$$\gamma_N \equiv 7[9],$$

which is what we wanted to prove. ∎

***Theorem 5.4.4*** *For given last $N-1$ digits (with $N > 2$), the two numbers that appear at position $N$ from the right is congruent to 5 modulo 10.*

For example, modulo 10000, the numbers 8125 and 3125 are the only possibilities for ended digits 125. We have $8 - 3 = 5$.

***Proof***

Let $N > 2$ and $l' > l \geq N$ such that

$$5^{l'} \equiv 5^l\left[10^{N-1}\right].$$

This equation means that the numbers $5^{l'}$ and $5^{l}$ have the same last $N-1$ digits. This means that

$$\exists \alpha \in \mathbb{N} \quad 5^{l'} = 5^{l} + \alpha\, 10^{N-1} \Leftrightarrow \exists \alpha \in \mathbb{N} \quad \alpha\, 10^{N-1} = 5^{l'} - 5^{l}.$$

Therefore, 5 divides $\alpha$, which means that the unit digit of $\alpha$ necessarily is 0 or 5.

– If it is 0, then there exists $\beta \in \mathbb{N}$ such that $\alpha = 10\beta$, hence

$$5^{l'} \equiv 5^{l}\left[10^{N}\right].$$

The above steps can be repeated with $N$ instead of $N-1$, and this stops at some point to necessarily reach the second possibility below.

– If it is 5, then there exists $\gamma \in \mathbb{N}$ such that $\alpha = 5 + 10\gamma$, hence

$$\frac{5^{l'} - 5^{l}}{10^{N-1}} = \alpha = 5 + 10\gamma.$$

This proves that

$$\frac{5^{l'} - 5^{l}}{10^{N-1}} \equiv 5[10],$$

which is precisely the expected result. ∎

Elaborating an algorithm to predict the Satoshi digits still is an open question, but the above shall be an opportunity for a starting point. Any curious reader is more than welcome for a discussion.

# Chapter 6
# On Improving the Merkle Trees: The *n*-Trees


In the context of Bitcoin Simplified Payment Verification (SPV) and Solvency, there are tree configurations where the path to the root is shorter than the one given by the usual binary tree, for the same computational memory – which is an essential strength. The $n$-trees have such properties, for $n \geq 2$ (binary trees are for $n = 2$). We show that, for some values of number of transactions $N \geq 3$, the system could be much faster, plus gain computational power, memory, and significantly reduce the computational time.

## 1 Description of an *n*-Tree

The usual tree in the Bitcoin and Ethereum is the binary tree, i.e., the 2-tree: a parent node has two children, from the first floor up to the root of the Tree (see Sect. 3, Chap. 2).

More generally, for a natural integer $n \in \mathbb{N}^* \backslash \{1\}$, an $n$-tree is a tree whose nodes have $n$ children. Figure 6.1 shows an example of 3-tree, with the usual notations as in Sect. 3, Chap. 2.

Focusing on the particular case shown in Fig. 6.1, the complete 3-tree has a number of leaves, which is $N = 3^H$, where $H$ is a natural integer (here $H = 2$, giving $3^2 = 9$ leaves). From one floor to another, the number of nodes is divided by 3, and so on until reaching the root. (Here, the ground floor has 9 leaves, the first floor has $9/3 = 3$ nodes, and the last floor has $3/3 = 1$ node, i.e., the root.)

What would become the Authentication path? As there are 3 children for each parent node, the child node of interest has 2 brothers. For the verification purposes, the client will need the brothers' nodes to concatenate and hash to get the parent's node. Therefore, the Authentication path must contain the two brothers, for each floor. It turns out that the Authentication path is a matrix, which has as many rows as floors in the tree (counting the ground floor but not the root's), and the number of columns is going to be 2. In Fig. 6.1, the Authentication path is a matrix of



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_6

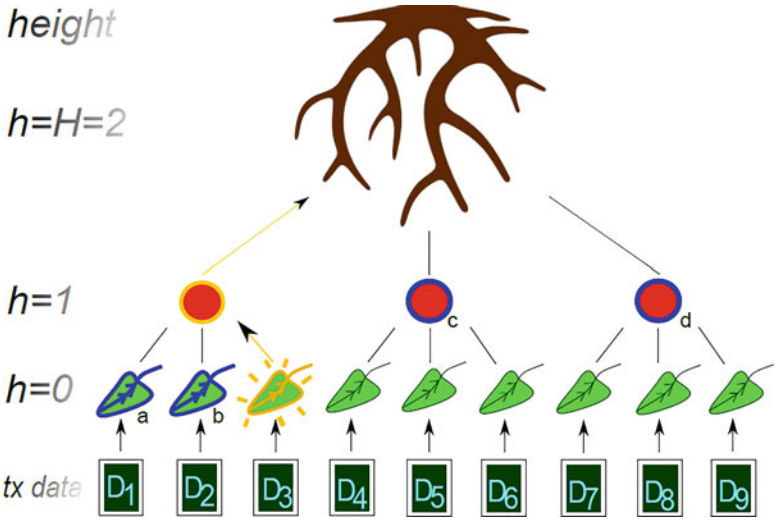


**Fig. 6.1** A 3-tree. Supposing we have 9 transactions (TXs), there are only two floors from the leaves to the root. The Authentication path (in blue) is a matrix of dimension $2 \times 2$, the first row is associated with the ground floor $h = 0$, the second row is associated with the first floor $h = 1$, and the first column is the most-left neighbor, while the second column is the most-right neighbor. The Root path (in yellow) is a vector of two components. See text for the reasons. (Source: Julien Riposo, 2023)

dimension $2 \times 2$. The first row corresponds to the ground floor $h = 0$, and the element in the first column of this row is the first Transaction ID (TXID) of this floor (node a – see Fig. 6.1), while the element in the second column of this row is the second TXID of this floor (node b). The second row corresponds to the first floor $h = 1$, and the element in the first column of this row is the second node of this floor (node c), while the element in the second column of this row is the third node of this floor (node d).

What would be the "Merkle" path, or, the *Root* path? This still is the shortest path from the TXID of interest up to the root. This therefore is a vector whose components (each of them corresponds to one floor) are the nodes connecting the TXID to the root. There are $N = 3^H$ nodes, the Root path has length $H = \ln N / \ln 3 = \log_3 N$, and the verification has complexity $O(\log_3 N)$.

We generalize the above approach now. Suppose that parent nodes have $n \geq 2$ children. See Fig. 6.2 for a graphic illustration.

The complete $n$-tree is a tree with a number of transations of $N = n^H$, with some natural integer $H > 0$. From one floor to another, the number of nodes is divided by $n$, and so on and so forth until reaching the root.

What would be the Authentication path? As there are $n$ children for each parent node, the child node of interest has $n - 1$ brothers. For the verification purposes, the client will need the brothers' nodes to concatenate and hash the parents' node. Therefore, the Authentication path must contain the $n - 1$ brothers for each floor.

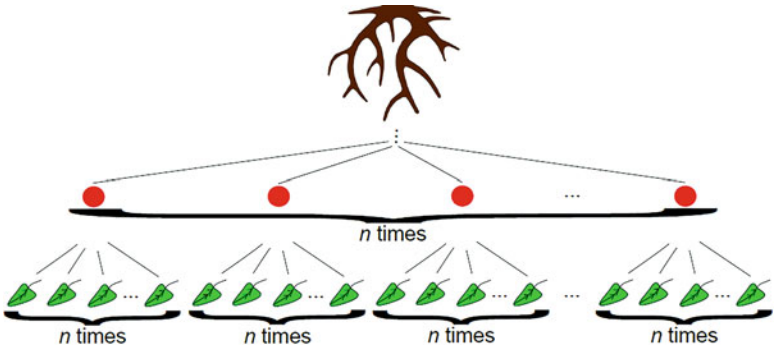


**Fig. 6.2** An $n$-tree. Supposing we have $N = n^H$ TXs, there are $H + 1$ floor from the leaves to the root. The Authentication path is a matrix of dimension $H \times (n - 1)$, the $i^{\text{th}}$ row is associated with the floor $i - 1$, and the $j^{\text{th}}$ column is the $j^{\text{th}}$ brother from the left. The Root path is a vector of $H$ components. (Source: Julien Riposo, 2023)

It turns out that the Authentication path is a matrix, which has as many rows as floors in the tree (counting the ground floor but not the root's), and the number of columns is going to be $n - 1$. The Authentication path is a matrix of dimension $H \times (n - 1)$.

What would be the Root path? As usual, this is the shortest path from the TXID of interest up to the root. As there are $N = n^H$ nodes, the Root path has length $H = \ln N / \ln n = \log_n N$, and the verification has complexity $O(\log_n N)$.

As a straightforward example, the $N$-tree (i.e., $n = N$) is a tree with leaves all connected to its root (indeed, we must have $N = N^H$, therefore $H = 1$), but as it implies the Authentication path to store $N - 1$ leaves, it is too memory demanding and thus represents no particular interest.

## 2 Developments on an *n*-Tree Implementation

The *equivalent* 2-*tree* of the 3-tree in Fig. 6.1 is given in Fig. 6.3.

We now compare the Authentication paths in both trees. For the 2-tree, the Authentication path is the vector

$$\text{Auth}^{2-\text{tree}} = (a', b', c', d'),$$

while for the 3-tree, the Authentication path is the matrix

$$\text{Auth}^{3-\text{tree}} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

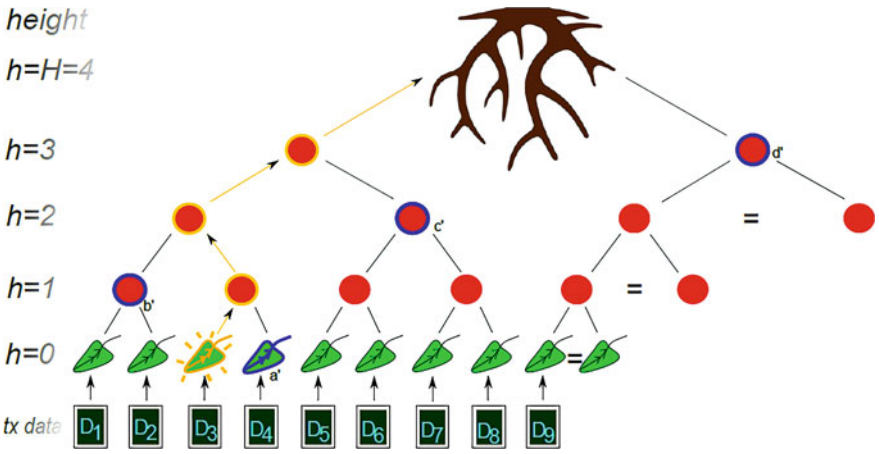


**Fig. 6.3** Equivalent 2-tree of the 3-tree shown in Fig. 6.1. Although the Authentication paths in both configurations store the same quantity of memory (see text), the height is twice reduced in the 3-tree, as well as in the Root path. (Source: Julien Riposo, 2023)

We see that $\text{Auth}^{2-\text{tree}}$ and $\text{Auth}^{3-\text{tree}}$ store the same amount of information. At this point, taking the 2-tree or the 3-tree does not make any difference.

However, the Root path for the 2-tree is a vector of 4 elements, while the one for the 3-tree is a vector of only 2 elements. This is twice reduced and allows the client to verify twice as fast as for the 2-tree case.

As an illustration, when the number of transactions is $3^{13}$, the Root path computation time is reduced by 1/3 from the 2-tree (on average 1.73 min) to the 3-tree (on average 1.13 min), see next section. The high number of transactions is here validating the asymptotic limit settlement.

This 1/3 reduction might be understood from the fact that, from the 2-tree to the 3-tree, the number of floors decreases from $\log_2 N$ to $\log_3 N$, which represents a relative decrease of $|(\log_3 N - \log_2 N)/\log_3 N|$ being roughly 1/3. This approach applies only if $N$ is (close to) a power of 3.

## 3 Empirical Comparison of Computation Times

Figure 6.4 shows the computational of the 3-tree with respect to the 2-tree, for varying number of transactions, from 1k to 1M.

The empirical evidence gives a slope of 3/4. This is significantly lower than 1, which is a satisfactory and sufficient fact for what we would like to achieve. Note that the reason we do not observe a slope of 2/3 is simply because of the number of transactions is not necessarily close to a power of 3.

We also show the comparison for computational times for the 20-tree vs the 2-tree, see Fig. 6.5.

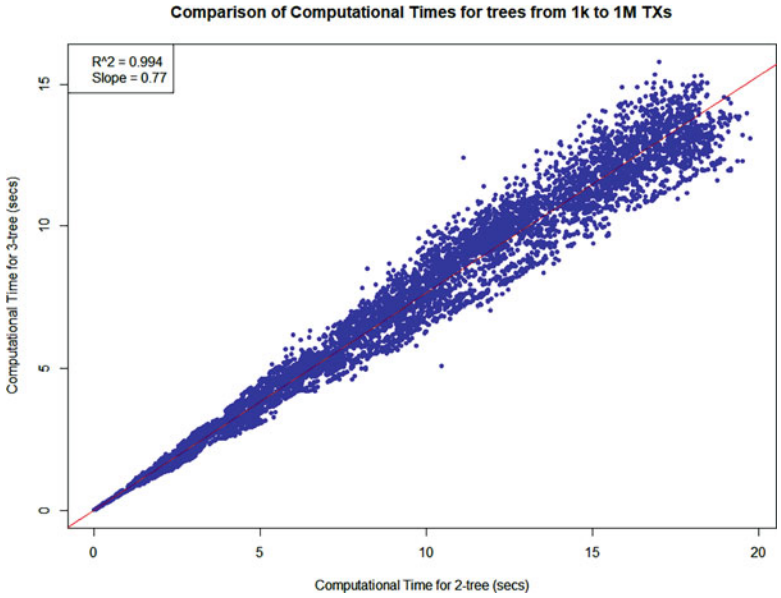


**Fig. 6.4** Linear trend for the computational times of the 3-tree with respect to the 2-tree, for varying number of transactions, from 1k to 1M. Here $R^2 = 0.994$. We do not observe 2/3, but rather a slope of 3/4. (Source: Julien Riposo, 2023)

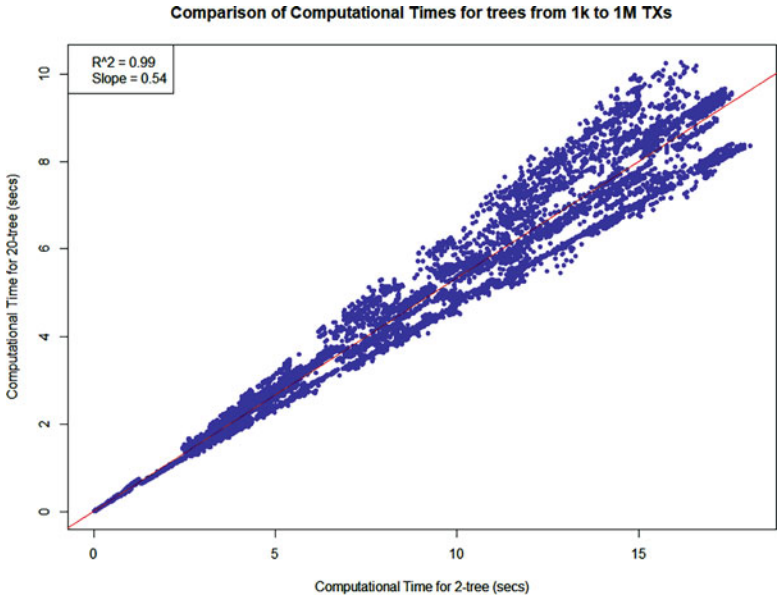


**Fig. 6.5** Linear trend for the computational times of the 20-tree with respect to the 2-tree, for varying number of transactions, from 1k to 1M. Here $R^2 = 0.990$. (Source: Julien Riposo, 2023)

## 4 A Mathematical Challenge for Finding the Right $n$

In fact, for the algorithm to be effective, given the number $N$ of leaves, appropriate numbers for $n$ and $H$ must be chosen.

### 4.1 Problem Position

To gain intuition from this previous statement, consider again the example above, of the 3-tree and its equivalent 2-tree. The number of nodes is $N = 3^2 = 9$. The closest power of 2 is $2^3 = 8$, therefore the last transaction – number $9^{\text{th}}$ – is on the second part of the 2-tree of size $2^4 = 16$. This enhances many more computational power than for a more complete tree, here the 3-tree of size $3^2$. In addition, the quantity of memory for the Authentication path for both trees is the same, see above. Henceforth, the best choice between the 2-tree of size $2^4$ and the 3-tree of size $3^2$ clearly is the latter.

Another very close power of number to $N = 9$ is $9^1$. The 9-tree is simply all the leaves connected to the root. This does present no interest for the reason that the Authentication path stores 8 numbers, two times more than the two previous trees. Finally, there is no other power of number, rather than $2^2$ or $3^2$, which are closer to $N = 9$ and shown interest.

We therefore should find the closest upper nontrivial power of number to get the best tree and which stores optimal memory in Authentication and Root paths.

Thus, we pose the problem as follows.

***Mathematical Challenge 6.4.1*** *Given the integer number $N \in \mathbb{N}^*$, find the closest integer number $n^H \in \mathbb{N}^*$ such that*

1. $H \in \mathbb{N}^* \backslash \{1\}$,
2. $n^H \geq N$, *and*
3. *optimal memory is stored in both Authentication and Root paths.*

The three conditions must be satisfied. A quick remark though: $H$ is an integer number and $H > 1$. If indeed $H = 1$, the trivial solution $n = H$ is allowed but presents no interest, as discussed.

The solution to this problem will allow to choose the $n$-tree with height $H$, in the most optimal computation speed and memory storing environment.

### 4.2 Partial Solution

Since $n^H$ must be closest upper number to $N$, we must have

$$n^{H-1} < N \leq n^H.$$

Applying the ln function, we have

$$(H-1)\ln n < \ln N \leq H \ln n \Leftrightarrow H - 1 < \frac{\ln N}{\ln n} \leq H.$$

Calling c(·) the ceiling integer part function, we then have

$$H = \mathrm{c}(\log_n N).$$

We have that $\log_n N > 1$ at all time (since $n < N$), and therefore $H > 1$ at all time, as require. In addition, this proves that if there are $N$ leaves, the uncomplete $n$-tree is systematically completed to form a complete $n$-tree, with $H = \mathrm{c}(\log_n N)$ floors.

This gives that the points 1 and 2 of the Mathematical Challenge 6.4.1 reduces to the following.

***Mathematical Challenge 6.4.2*** *Given the integer number $N \in \mathbb{N}^*$, find the closest integer number $n^{\mathrm{c}(\log_n N)} \in \mathbb{N}^*$ such that $n^{\mathrm{c}(\log_n N)} \geq N$.*

We now need to focus on an appropriate range for the variable $n$, where the optimal value will be. To do so, for $N \in \mathbb{N}^* \backslash \{1\}$, consider the function $f$ defined by

$$f \; : \; \begin{cases} \mathbb{N}^* \backslash \{1\} \longrightarrow \mathbb{R}_+^* \\ n \longmapsto n^{\mathrm{c}(\log_n N)} \end{cases}$$

The function $f$ is increasing from at least a given rank $n_0 \in \mathbb{N}^* \backslash \{1\}$ (the interest of $n_0$ is shown later below).

Indeed, the function

$$g \; : \; \begin{cases} \mathbb{N}^* \backslash \{1\} \longrightarrow \mathbb{R}_+^* \\ n \longmapsto \mathrm{c}(\log_n N) \end{cases}$$

is a nonincreasing function. Thus, there exists $n_0 \in \mathbb{N}^* \backslash \{1\}$ such that $g(n) = \mathrm{c}(\log_n N) \leq 2$, $\forall n \geq n_0$. Since $g(n) = 1$ if and only if $n = N$ (proving that $n_0 < N$), we have that $g(n) = 2$, $\forall n \in \{n_0, \ldots, N-1\}$. Bearing this in mind and that $f(n) = n^{g(n)}$ for any $n \in \mathbb{N}^* \backslash \{1\}$, therefore $f$ is increasing on $\forall n \in \{n_0, \ldots, N-1\}$.

In practice, the ratio $n_0/N$ decreases with $N$, see Fig. 6.6 for numerical illustrations with $N \in \{50, 100, 1000\}$.

The previous point demonstrates that there is no need to focus on large values of $n$, because otherwise $f(n)$ would be too large and likely higher than $N$. Indeed, we have

$$1 < \frac{\ln N}{\ln n_0} \leq 2 \Leftrightarrow n_0 < N \leq n_0^2,$$

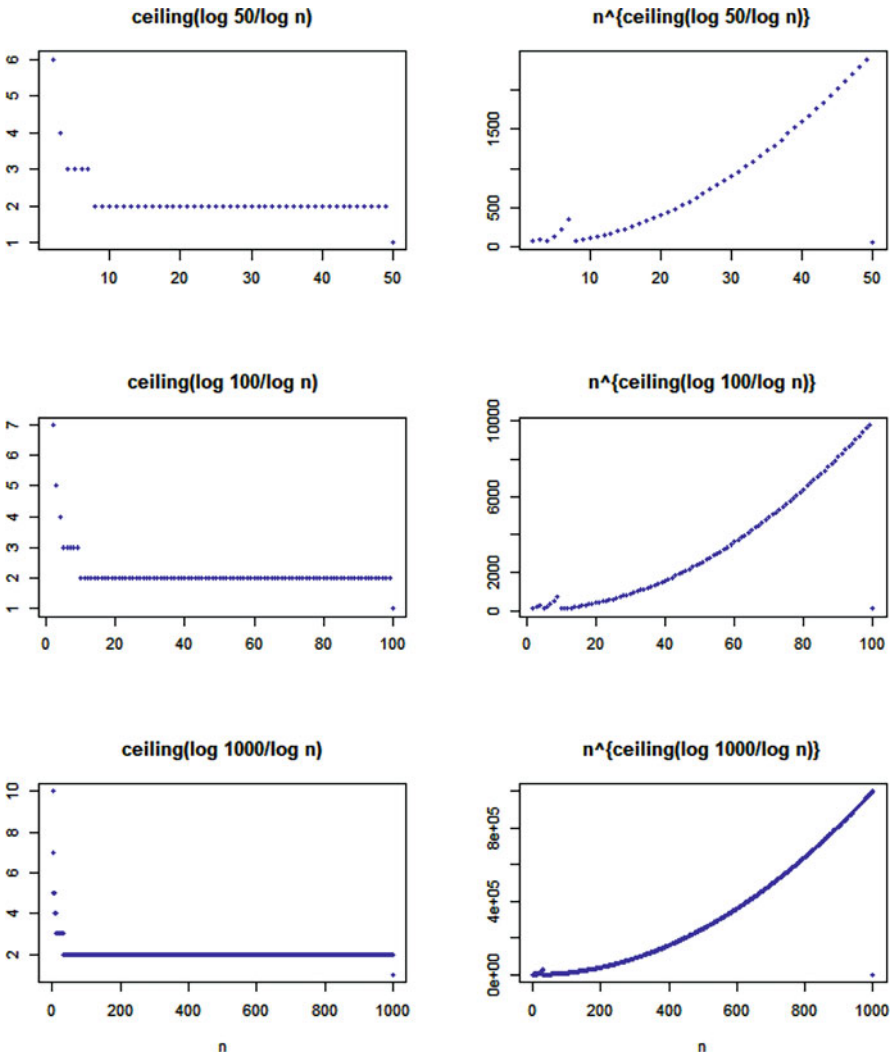


**Fig. 6.6** Representations for the functions $g$ (left panel) and $f$ (right panel). The range of values for which $g$ takes the value 2 increases with $N$, which implies $f$ to be strictly increasing on this range (see text for details). (Source: Julien Riposo, 2023)

so focusing on $\{2, \ldots, n_0 - 1\}$ is sufficient (hence the interest of $n_0$). A proxy for $n_0$ is $\sqrt{N}$. We will retain the following two ideas to answer the Mathematical Challenge 6.4.2.

***Idea 6.4.3*** *It is enough to focus on the intervalley* $n \in \left\{2, \ldots, \sqrt{N}\right\}$ *to find the optimal value of n.*

We therefore can state the main idea to solve the Mathematical Challenge 6.4.2.

***Idea 6.4.4*** *The optimal value of n is the one which minimizes the function*

$$\mathcal{L}: \left\{2, \ldots, \mathrm{E}\left(\sqrt{N}\right)\right\} \longrightarrow \mathbb{R}_+^*, \quad \mathrm{n} \mapsto \mathrm{n}^{\mathrm{c}(\log_{\mathrm{n}} \mathrm{N})} - \mathrm{N}$$

It is worth pointing out that the function $\mathcal{L}$ always is nonnegative. Indeed, we have

$$n^{\ln N} = N^{\ln n} \Leftrightarrow n^{\frac{\ln N}{\ln n}} = N,$$

therefore, since $2 \leq n \leq \mathrm{E}\left(\sqrt{N}\right) \leq \sqrt{N} < N$, we have

$$n^{\mathrm{c}(\log_n N)} \geq N \Leftrightarrow \mathcal{L}(n) \geq 0.$$

We found a solution for the points 1 and 2 of the Mathematical Challenge 6.4.1.

# 5 A Full Solution to the Mathematical Challenge for Finding the Right *n*

## *5.1 The Cost Function*

We want to solve the point 3 for the Mathematical Challenge 6.4.1. We need to calculate the total number of elements to be stored in client's memory, among the Authentication and Root paths.

Consider an $n$-tree with $N$ nodes.

- For the Authentication path, we have seen that we need to store $H \times (n - 1)$ elements;
- For the Root path, we have seen that we need to store $H$ elements.

Therefore, the total number of elements to be stored is $H + H(n - 1) = nH = n\mathrm{c}(\log_n N) > 0$. This number should ideally be the minimal.

***Idea 6.5.1*** *The optimal value of n is the one that minimizes the following cost function:*

$$\overline{\mathcal{L}}: \left\{2, \ldots, \mathrm{E}\left(\sqrt{N}\right)\right\} \longrightarrow \mathbb{R}_+^*, \quad \mathrm{n} \mapsto \mathrm{n}^{\mathrm{c}(\log_{\mathrm{n}} \mathrm{N})} - \mathrm{N} + \mathrm{nc}(\log_{\mathrm{n}} \mathrm{N})$$

Here the cost function $\overline{\mathcal{L}}$ is the sum of the cost $n^{\mathrm{c}(\log_n N)}$ due to being far from $N$, and of the memory cost to be stored at the client's.

## 5.2 *Considering the* n*-Tree Nodes in the Cost Function*

If we also would like to consider the memory stored in a company, we just need to count the number of nodes $\mathcal{N}$ in an $n$-tree. This can be done as follows (we generalize the approach in Theorem II.3.4). Suppose we have $N$ leaves. Then the number of nodes on the floor $h \in \{0, \ldots, H\}$ is

$$u_h = \mathrm{c}(\ldots \mathrm{c}(\mathrm{c}(N/n)/n) \ldots /n), \quad h \text{ times},$$

with the convention $u_0 = N$. If $N = n^H$ (perfect $n$-tree), then the total number of nodes is

$$\mathcal{N} = \sum_{h=0}^{H} u_h = \sum_{h=0}^{H} n^{H-h} = \frac{n^{H+1} - 1}{n - 1}.$$

Hence, more generally, if $n^{H-1} < N \leq n^H$, then

$$\frac{n^H - 1}{n - 1} < \mathcal{N} \leq \frac{n^{H+1} - 1}{n - 1},$$

with, of course, $H = \mathrm{c}(\log_n N)$. For the convenience of the presentation, we can state that there exists $t \in ]0, 1]$ such that

$$\mathcal{N} = \frac{n^{H+t} - 1}{n - 1}.$$

We ignored the nodes of the Merkle tree stored in memory above, since the client does not need to hold the full tree in her computer. Thus, bearing the previous section and all above in mind, the total cost function should be given by $\overline{\mathcal{L}}$ in the Idea 6.5.1.

## 5.3 *Empirical Analysis*

At this stage, an empirical analysis for finding optimal values of $n$ with respect to the number of transactions $N$ could be performed. Figures 6.7 and 6.8 show optimal values for a number of transactions from 100 to 1M.

We observe a diversity in the findings. Specifically, for the range of transactions from $N = 100$ to $N = 250{,}000$, the most relevant optimal values of $n$ are 20, 3, and 22. It is worth pointing out that the optimal value $n = 3$ is three times more frequent than $n = 2$. For the range of transactions from $N = 100$ to $N = 1{,}000{,}000$, although the value $n = 2$ is one of the most important, it represents only less than 4% of all the found optimal values (no significance above $n = 30$, so we did not show values above). This empirically proves that, in more than 96% of the cases, the binary tree is not optimal in terms of memory saving, for this range of transactions.

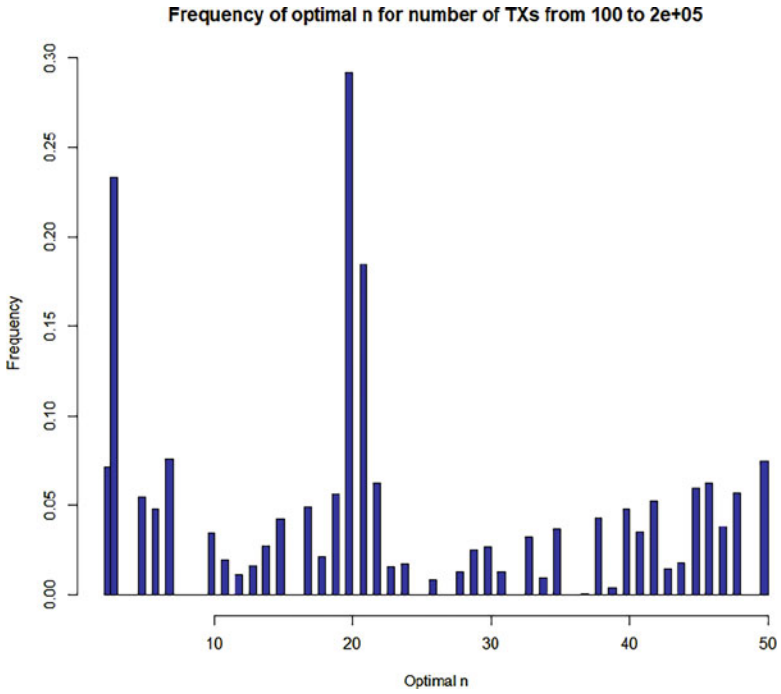


**Fig. 6.7** Frequency (in proportion) of optimal values of $n$ (i.e., minimal cost function $\overline{\mathcal{L}}$) for varying number $N$ of transactions (from 100 to 250,000). (Source: Julien Riposo, 2023)

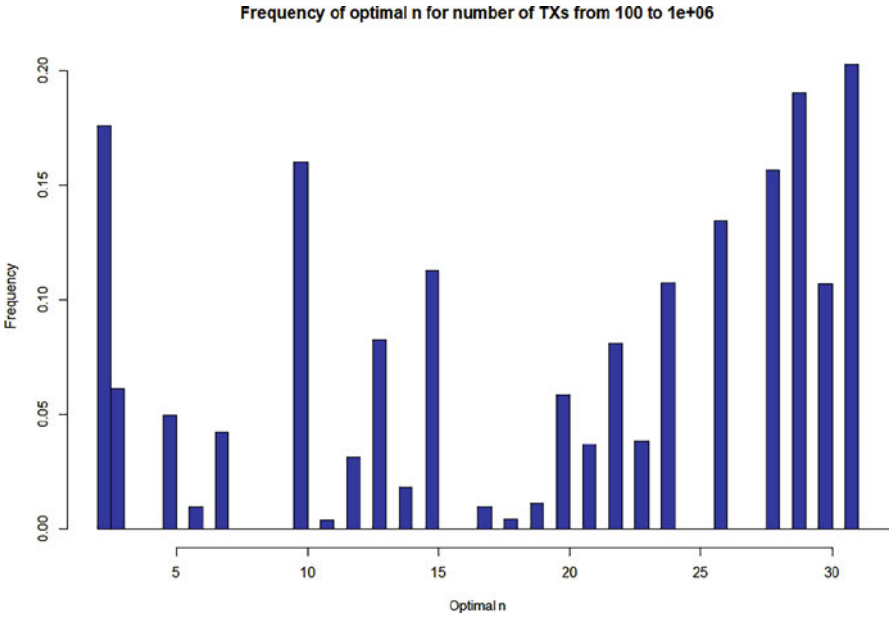


**Fig. 6.8** Frequency (in proportion) of optimal values of $n$ (i.e., minimal cost function $\overline{\mathcal{L}}$) for higher varying $N$ of transactions (from 100 to 1M). (Source: Julien Riposo, 2023)

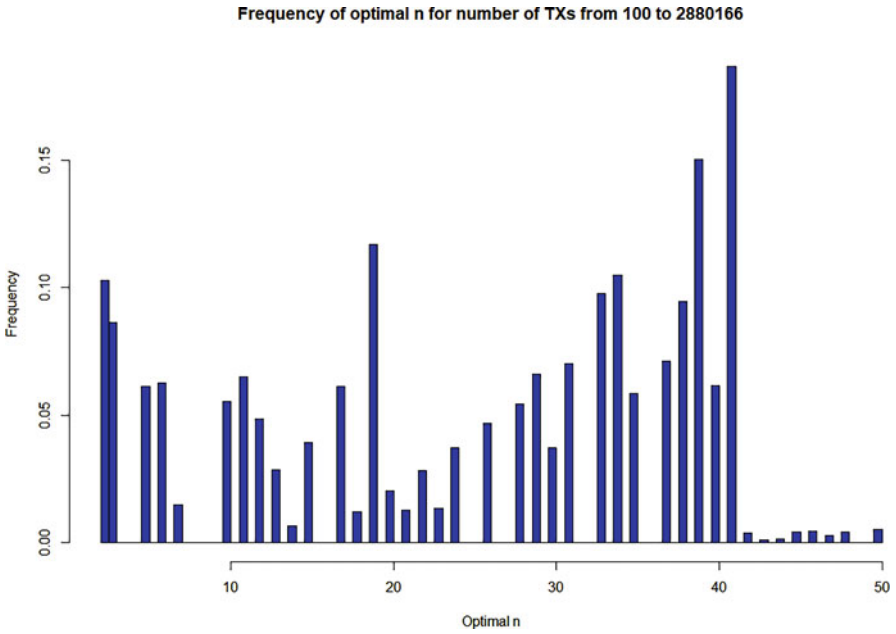


**Fig. 6.9** Frequency (in proportion) of optimal values of $n$ (i.e., minimal cost function $\overline{\mathcal{L}}$) for higher varying $N$ of transactions (from 100 to a bit less than 3M). (Source: Julien Riposo, 2023)

As further illustrated, we show in Figures 6.8 and 6.9 the strong variability of the optimal values of $n$ for number of transactions in a higher range. All these figures empirically prove that the 2-tree is not necessarily optimal.

## *5.4 A Word on an Analytical Derivation of the Optimal Value*

It is worth pointing out that an analytical determination for the optimal value of $n$ looks to be uneasy, because the ceiling interger part function is not derivable on $\mathbb{Z}$. We could have thought about generalized functions, in which case the derivative of $\overline{\mathcal{L}}$ in the generalized function context is, for any $x \in \mathbb{R}_+^*$, given by

$$\bar{\mathcal{L}}'(x) = \left(\mathrm{c}\left(\frac{\ln N}{\ln x}\right) - \frac{\ln N}{\ln x}\,\mathrm{III}\left(\frac{\ln N}{\ln x}\right)\right) x^{\mathrm{c}\left(\frac{\ln N}{\ln x}\right)-1} + \mathrm{c}\left(\frac{\ln N}{\ln x}\right) + \frac{\ln N}{(\ln x)^2}\,\mathrm{III}\left(\frac{\ln N}{\ln x}\right),$$

where III is the Dirac-comb generalized function, i.e.,

$$\mathrm{III}(x) = \sum_{k\in\mathbb{Z}} \delta(x-k),$$

and $\delta$ is the Dirac function.

We cannot simplify this function when $\ln N/\ln x \in \mathbb{N}$, because $Ш(\ln N/\ln x) \neq 0$, that is, when $N = x^H$ for some $H \in \mathbb{N}$, or, equivalently, when $x = N^{1/H}$ for some $H \in \mathbb{N}$. In the case where $\ln N/\ln x \notin \mathbb{N}$, the above expression simplifies to

$$\overline{\mathcal{L}}'(x) = \mathrm{c}\left(\frac{\ln N}{\ln x}\right)\left(x^{\mathrm{c}\left(\frac{\ln N}{\ln x}\right)-1}+1\right), \quad \forall x \in \mathbb{R}_+^* \backslash N^{1/\mathbb{N}}.$$

Hence, the function $\overline{\mathcal{L}}$ never cancels on $\mathbb{R}_+^* \backslash N^{1/\mathbb{N}}$. Henceforth, this analytical study does not look to be relevant here?

## 6 The Dynamic *n*-Tree

If a miner adopts the $n$-tree construction instead of the usual Merkle tree, building the block necessitates to update the tree each time a new list of transactions is received. In case of a high transaction seed (several thousands of transactions per second), the number of children $n$ may vary and should be dependent on it. Thus, at a given time, we expect that $n$ is the number of transactions hit to the miner.

However, the number $n$ should not vary from floors higher than the leaves, which, in this sense, the constructed tree is an $n$-tree.

### *6.1 Principles*

Bearing the above in mind, when the miner receives a list of transactions, she must update her tree as soon as possible, and she has in practice one second to do so. Figure 6.10 shows a dynamical way an $n$-tree should be updated every second, approximately. We write $h$ the floor number in the tree.

We could still divide the tree into several parts as a binary tree, and this is a matter of convention. This model is actually quite simple, as, for round $i \in \mathbb{N}^*$, the parent node will be at floor $h = 1$ if $i \in \{1, 2\}$, and at floor $h = i - 1$ otherwise, as indicated in Fig. 6.10.

At floor $h = 0$, each node forms a part of a set that has arrived at a certain time in the history of the construction of the tree. We could organize this succession of transactions in a vector $\mathcal{V}$ ordered as follows (see Fig. 6.10):

$$\mathcal{V} = (1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, \ldots). \qquad (*)$$

Besides, component $i$ of $\mathcal{V}$ is equal to the $\mathcal{V}_i^{\text{th}}$ set of transactions, i.e., the transaction arriving at second $\mathcal{V}_i$, and whose parent node will be at floor $h = 1$ if $i \in \{1, 2\}$, and at floor $h = i - 1$ otherwise.

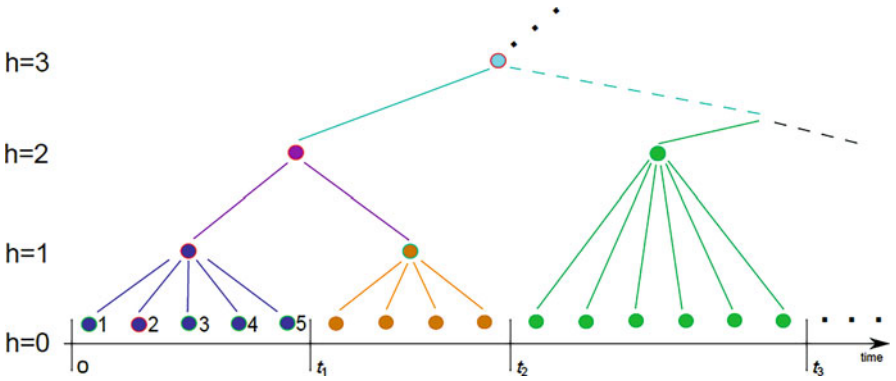


**Fig. 6.10** Dynamical *n*-tree. Every second, approximately, the miner receives a list of transactions. She is adding these transactions in her candidate block and is updating her *n*-tree. Here, between times 0 and $t_1 > 0$, the miner receives five transactions and forms the parent node. Between times $t_1$ and $t_2 > t_1$, the miner receives four more transactions, forms the parent node from them, and forms the parent node (purple) from the blue and yellow nodes (thus reaching floor $h = 2$). Finally, a new list of six transactions is received and their parent node is concatenated in a way to form an *n*-tree (here $n = 6$). The process goes on and on until a nonce is found. (Source: Julien Riposo, 2023)

## 6.2 *Static Tree: Authentication Path*

In a static tree, the number of children $n$ does not vary, it is the same all the time. We focus on an $n$-tree with $N$ nodes and $H$ floors (+ the ground floor of leaves). From Sect. 5.2, we know that the number of nodes $u_h$ on floor $h$, with $h \in \{0, \ldots, H\}$, is

$$u_h = \mathrm{c}(\ldots \mathrm{c}(\mathrm{c}(N/n)/n) \ldots /n), \quad h \text{ times}.$$

We note that there is a relationship between $H$, $N$, and $n$. The root of the tree is the unique node at floor $H$, so $u_H = 1$, that is, $H = \underset{h \in \mathbb{N}}{\operatorname{argmin}}(\mathrm{c}(\ldots \mathrm{c}(\mathrm{c}(N/n)/n) \ldots /n) \ (h \text{ times}) = 1)$. As an example, if $N = 15$ and $n = 3$, we have $\mathrm{c}(\mathrm{c}(15/3)/3) = 2$ and $\mathrm{c}(\mathrm{c}(\mathrm{c}(15/3)/3)/3) = 1$, so that $H = 3$. Another example: if $N = 17$ and $n = 4$, we have $\mathrm{c}(\mathrm{c}(17/4)/4) = 2$ and $\mathrm{c}(\mathrm{c}(\mathrm{c}(17/4)/4)/4) = 1$, so that $H = 3$.

This nontrivial formula gives a unique value for $H$. Figure 6.11 is an illustration.

In Fig. 6.11, the path to the root for transaction $a$ is $(a, b, c, d)$, and the Authentication path, which cannot be a matrix, could be defined as a *family* of vectors, here given by $\mathrm{Auth} = \{(1, 3, 4), (2', 3', 4'), (2'')\}$. We construct such path in the following. For any floor $h$, we index the nodes in order from left to right (e.g., node $a$ is indexed by 2).

The transaction of interest is indexed by 2 (transaction $a$). Let $j = 2$. We perform the Euclidean division of $j$ with respect to $n = 4$ to simply have $j = qn + r$, i.e., here $2 = 0 \times 4 + 2$. It is clear that the first vector of the Authentication path is given by $\mathrm{Auth}_0 = (0 \times 4 + 1, 0 \times 4 + 3, 1 \times 4 + 0) = (1, 3, 4)$.

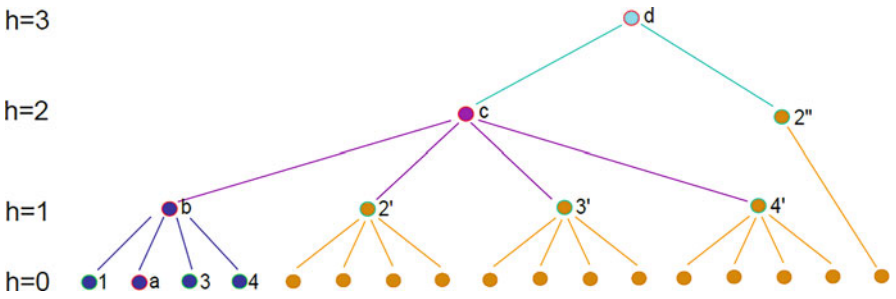


**Fig. 6.11** A static $n$-tree, where $N = 17$ and $n = 4$. As we can see, there are $H = 3$ floors (+ the ground floor). Given values for $N$ and $n$ gives a unique value for $H$. (Source: Julien Riposo, 2023)

We now go to the next floor, $h = 1$. The parent node $b$ is in the first position, that is at position $\mathrm{c}(j/n) = \mathrm{c}(2/4) = \mathrm{c}(0.5) = 1$. We can repeat the above lead to obtain the next vector for the Authentication path: there are $u_1 = \mathrm{c}(N/n) = 5$ nodes, so that $\mathrm{Auth}_1 = (\mathrm{Auth}_0, (0 \times 4 + 2, 0 \times 4 + 3, 1 \times 4 + 0)) = (\mathrm{Auth}_0, (2, 3, 4))$. The final relevant floor, $h = 2$, before reaching the root has $u_2 = \mathrm{c}(\mathrm{c}(N/n)/n) = 2$ nodes (note that the last node automatically comes to the above floor, i.e., is the parent of itself). The index of the parent node in the path to the root is $\mathrm{c}(1/4) = 1$, thus $\mathrm{Auth}_2 = (\mathrm{Auth}_1, (2))$, and since $h = 2$ and there are 3 floors, we finally have $\mathrm{Auth} = \mathrm{Auth}_2$, that is, $\mathrm{Auth} = \{(1, 3, 4), (2', 3', 4'), (2'')\}$ as stated above (the prime and double prime are just for the reader's convenience to specify the different flooring).

Algorithm 6.6.1 describes the generalization of the above.

**Algorithm 6.6.1 Authentication Path Computation – Static $n$-tree**
***Result:*** *Authentication path from a static n-tree*
*We focus on an n-tree with N nodes and H floors (+ the ground floor);*
*Set the client of interest is number j, for* $j \in \{1, \ldots, N\}$, *and* $\mathrm{Auth}_{-1} = \emptyset$;
*For* $h \in \{0, \ldots, H - 1\}$ *do*

- *Calculate the quotient q and the rest* $r \in \{0, \ldots, n - 1\}$ *of the Euclidean division of j with n, so that* $j = qn + r$;
- *If* $j \neq N$ *then*
  $\mathrm{Auth}_h = (\mathrm{Auth}_{h-1}, (qn + i)_{i \in \{1, \ldots, n\} \setminus \{r\}})$;
  *Else*
  $\mathrm{Auth}_h = (\mathrm{Auth}_{h-1}, (qn + i)_{i \in \{1, \ldots, n-1\}})$;
  *End*
- $j \mapsto \mathrm{c}(j/n)$.

*End*
*Output* $\mathrm{Auth}_{H-1}$.

## 6.3 Dynamic Tree: Authentication Path

The derivation of the Authentication path for the $n$-tree in a dynamical environment needs to use the vector of the style of Equation $(*)$, Sect. 6.1. The index $t$ of component $\mathcal{V}_t$ of such vector is the index of transaction $t$ stacked in the block, in chronological order. The value $\mathcal{V}_t$ is the second at which the associated family of transactions has arrived at the miner. The vector $\mathcal{V}$ is to play a memory role to allow remembering faster the family of a transaction.

Thus, for transaction $t$, the task consists in identifying where the transaction is. A simple search can identify the position of the brothers of the transaction $t$ in the same family. We thus refer to

$$\text{Auth}_0 = \underset{l \in \{1, \ldots, N\}}{\arg} (\mathcal{V}_l, \mathcal{V}_l = \mathcal{V}_t)$$

as the enhanced Authentication path at floor $h = 0$.

As an example, in Fig. 6.10, if $t = 2$, then $\text{Auth}_0 = (1, 3, 4, 5)$, and the parent node will have index $\mathcal{V}_2 = 1$. The parents' nodes then are playing the roles of leaves for a static tree, with $N^{'}$ nodes and $H^{'} = H - 1$ floors. Algorithm 6.6.1 then applies to the rest of the tree. The whole above approach is generalized in the following Algorithm 6.6.2.

**Algorithm 6.6.2 Authentication Path Computation – Dynamical *n*-tree**

***Result:*** *Authentication path from a dynamical n-tree*

*We focus on an n-tree with N nodes and H floors (+ the ground floor), with the available vector* $\mathcal{V}$*;*

*Set the client of interest is number t, for* $t \in \{1, \ldots, N\}$*, and* $\text{Auth}_0 = \underset{l \in \{1, \ldots, N\}}{\arg} (\mathcal{V}_l, \mathcal{V}_l = \mathcal{V}_t)$*;*

*Set* $j = \mathcal{V}_t$*;*

*For* $h \in \{1, \ldots, H-1\}$ *do*

- *Calculate the quotient q and the rest* $r \in \{0, \ldots, n-1\}$ *of the Euclidean division of j with n, so that* $j = qn + r$*;*
- *If* $j \neq N$ *then*
  $\text{Auth}_h = (\text{Auth}_{h-1}, (qn + i)_{i \in \{1, \ldots, n\} \setminus \{r\}})$*;*
  *Else*
  $\text{Auth}_h = (\text{Auth}_{h-1}, (qn + i)_{i \in \{1, \ldots, n-1\}})$*;*
  *End*
- $j \mapsto \text{c}(j/n)$.

*End*

*Output* $\text{Auth}_{H-1}$.

# Chapter 7
# Entropy of Peer-to-Peer Network


A strong flow of information 24/7 occurs between the agents of the peer-to-peer network. This intensive activity is a mark of perpetual evolution. One famous quantitative metric of evolution is entropy. We are going to synthesize three representative graphs of information flow. Entropy could be defined in the three of them, but by the equivalence of their information flow representation, one network might be more appropriate depending on the goal. One direct application of entropy is the increase of privacy of users: the Particles Model is explained and closes this chapter.

## 1 Blockchain Network, Blockchain Transaction Network, and Peer-to-Peer Network

There are at least three different networks to represent the same flow of information between the actors of a blockchain.

- the first way is through the Blockchain Network, as we have seen thus far. This is modeled through a graph, where each vertex represents a block, thus endowed with block header and transactions list information (see Fig. 7.1, top panel). Note that this graph is unidirectional (i.e., from left to right). Transaction $t_1$ occurs before transaction $t_2$, thus either the two appears in the same block, and $t_1$ is above $t_2$, or they are not, and $t_1$ is in an older block, i.e., a block on the left from $t_2$'s.
- the second way is through the Blockchain Transaction Network (see Sect. 4, Chap. 2 and Fig. 2.4) and for instance Sect. 2, Chap. 8 for an application. In this graph, the vertices are inputs (in the sense of Definition 2.4.2) endowed with the bitcoins it possesses. When an input has not been outflowed, it is an absorbing state in the graph, thus an Unspent Transaction Output (or UTXO), and constitutes a source of cash. Spent outputs cannot be used anymore. If not all the cash in an input is used, the remaining cash is stored in another input, hence unidirectionality again. In Fig. 7.1, middle panel, two transactions are



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_7

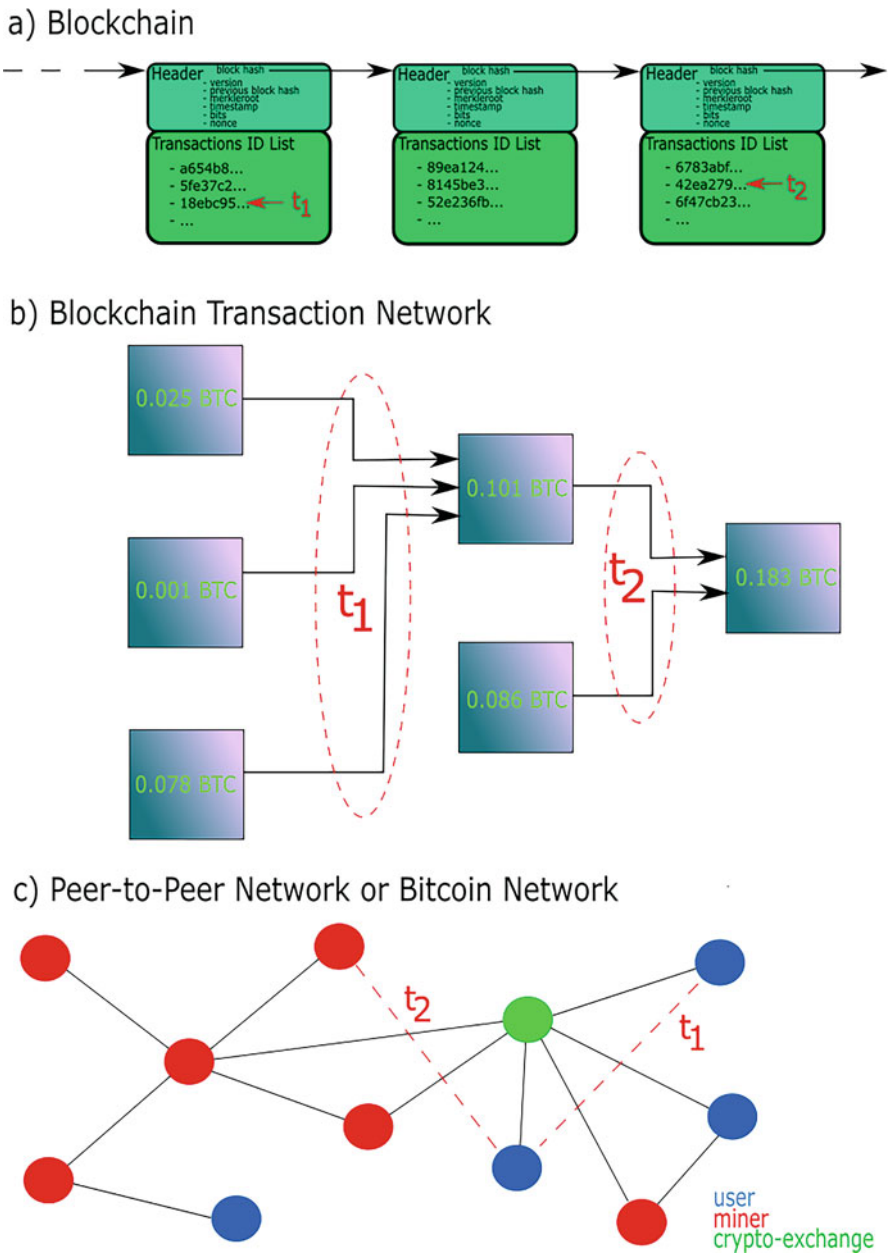


**Fig. 7.1** Equivalence of graphs. Each graph represents the same information, but in a different fashion. We have illustrated two particular transactions ($t_1$ first occurred, then $t_2$). See text for explanations. (Source: Julien Riposo, 2023)

represented. Transaction $t_1$ occurs before transaction $t_2$, thus the former appears on the left in the graph and the latter on the right.

- the third way is through the Peer-to-Peer Network, see Sect. 2, Chap. 4 and in particular [12] for a deep Diffusion Theory construction on a graph. In Fig. 7.1, bottom panel, there are (at least) three different types of vertex:

  1. the users, who are coding their transaction, to be sent to a miner;
  2. the miners, who are constructing the blockchain through the Proof-of-Work (see Definition 2.2.4);
  3. the crypto-exchanges, who are coding the user (here client)'s transaction.

  In this panel, transaction $t_1$ occurs between two users through the crypto-exchange (transfer from the crypto-exchange to a miner not represented), and transaction $t_2$ occurred between two users (not shown) and is then transferred from the user to a miner without crypto-exchange intervention. In this example, the top-right user utilizes three inputs (right-hand side in middle panel), the bottom user utilizes two inputs, and the top-left user has one input credited. In general, there is no particular reason that one transaction considers only two users, so that we may find more than two edges in the Peer-to-Peer Network for one transaction. The important aspect to note is that one transaction always forms a connected bipartite subgraph in the Peer-to-Peer Network (the sender vs the receivers).

  It is worth stressing that this type of representation is more common than the two others.

The evolution description of the flows of cash could be done through the definition of the entropy on the second and third graphs in Fig. 7.1. The object of diffusion could for instance be a Satoshi unit (irreducible value of Bitcoin). Examples of graph approaches are studied in [26, 27].

## 2 The Shannon Entropy and the Entropy Rate for the Peer-to-Peer Network

We are mostly inspired by [17] in this section.

### *2.1 Shannon Entropy and Entropy Rate*

Let $(\Omega, \mathcal{T}, \mathbb{P})$ be the usual probability space.

***Definition 7.2.1*** *Let X be a discrete random variable with value in $\chi$, a countable space which is summable. The* Shannon Entropy *of X is defined as*

$$\mathcal{S}(X) = -\sum_{x\in\chi}\mathbb{P}(X=x)\, ln\, \mathbb{P}(X=x).$$

By *summable*, we mean that all the sums on $\chi$ seen here make sense. The Shannon Entropy is thus defined as a sum of nonnegative numbers in the finite space $\chi$. In general, Card($\chi$) can be infinite (but $\chi$ is countable) while the sum in Definition 7.2.1 needs to be finite. In this definition and in case $0 \in \chi \subset \mathbb{R}$, we use the convention $0 \times \ln 0 = 0$. We can only consider the values of $x$ for which $x > 0$. More generally, we have the following definition.

***Definition 7.2.2*** *Let $X$ and $Y$ be discrete random variables with value in $\chi$, a countable space which is summable. The multivariate* Shannon Entropy *of $X$ and $Y$ is defined as*

$$\mathcal{S}(X,Y) = -\sum_{x,\,y\in\chi}\mathbb{P}(X=x, Y=y)\, ln\, \mathbb{P}(X=x, Y=y).$$

This can be generalized to any number of discrete random variables.

***Definition 7.2.3*** *Let $X$ and $Y$ be discrete random variables with value in $\chi$, a countable space which is summable. The* Conditional Entropy *of $X$ given $Y$ is defined as*

$$\mathcal{S}(X|Y) = \sum_{y\in\chi}\mathbb{P}(Y=y)\mathcal{S}(X|Y=y),$$

*where*

$$\mathcal{S}(X|Y=y) = -\sum_{x\in\chi}\mathbb{P}(X=x|Y=y)\, ln\, \mathbb{P}(X=x|Y=y).$$

These definitions straightforwardly generalize when we have more than two variables. We have the following useful Chasles relationship.

***Proposition 7.2.4*** *Let $X$ and $Y$ be discrete random variables with value in $\chi$, a countable space which is summable. We have*

$$\mathcal{S}(X,Y) = \mathcal{S}(Y) + \mathcal{S}(X|Y).$$

***Proof***
We may use directly the definitions above. By using the Bayes law, we have

$$\mathcal{S}(Y) + \mathcal{S}(X|Y) = \sum_{y\in\chi} \mathbb{P}(Y=y)\, ln\, \frac{1}{\mathbb{P}(Y=y)}$$
$$+ \sum_{x,\, y\in\chi} \mathbb{P}(X=x, Y=y)\, ln\left(\frac{\mathbb{P}(Y=y)}{\mathbb{P}(X=x, Y=y)}\right).$$

By using the total probability formula, we have

$$\mathcal{S}(Y) + \mathcal{S}(X|Y) = \sum_{x,\, y\in\chi} \mathbb{P}(X=x, Y=y)(\ln \frac{1}{\mathbb{P}(Y=y)} + \ln(\frac{\mathbb{P}(Y=y)}{\mathbb{P}(X=x|Y=y)})) =$$
$$- \sum_{x,\, y\in\chi} \mathbb{P}(X=x, Y=y) \ln \mathbb{P}(X=x, Y=y) = \mathcal{S}(X, Y). \ \blacksquare$$

***Corollary 7.2.5*** *Let X, Y, and Z be discrete random variables with value in $\chi$, a countable space which is summable. We have*

$$\mathcal{S}(X, Y|Z) = \mathcal{S}(Y|Z) + \mathcal{S}(X|Y, Z).$$

***Proof***
Assume $z \in \chi\backslash\{0\}$. Then $X \mid \{Z = z\}$ and $Y \mid \{Z = z\}$ are discrete random variables with value in $\chi$, and we apply Proposition 7.2.4 to have

$$\mathcal{S}(X, Y|Z=z) = \mathcal{S}(Y|Z=z) + \mathcal{S}(X|Y, Z=z).$$

We multiply by $\mathbb{P}(Z = z)$ for all $z \in \chi\backslash\{0\}$, use the convention $0 \times \ln 0 = 0$ and Definition 7.2.3 to have

$$\mathcal{S}(X, Y|Z) = \mathcal{S}(Y|Z) + \sum_{z\in\chi\backslash\{0\}} \mathbb{P}(Z=z)\mathcal{S}(X|Y, Z=z).$$

We note that, by again using Definition 7.2.3, we have

$$\mathcal{S}(X|Y, Z=z) = \sum_{y\in\chi} \mathbb{P}(Y=y|Z=z)\mathcal{S}(X|Y=y, Z=z),$$

therefore

$$\begin{aligned}
&\sum_{z\in\chi}\mathbb{P}(Z=z)\mathcal{S}(X|Y,Z=z)\\
&\quad=\sum_{(y,z)\in\chi\times\chi\setminus\{0\}}\mathbb{P}(Z=z)\mathbb{P}(Y=y|Z=z)\mathcal{S}(X|Y=y,Z=z)\\
&\quad=\sum_{(y,z)\in\chi\times\chi\setminus\{0\}}\mathbb{P}(Y=y,Z=z)\mathcal{S}(X|Y=y,Z=z)\\
&\quad=\sum_{(y,z)\in(\chi\setminus\{0\})^2}\mathbb{P}(Y=y,Z=z)\mathcal{S}(X|Y=y,Z=z).
\end{aligned}$$

By using the convention $0\times\ \ln 0=0$, we deduce that

$$\sum_{z\in\chi}\mathbb{P}(Z=z)\mathcal{S}(X|Y,Z=z)=\mathcal{S}(X|Y,Z).\ \blacksquare$$

***Corollary 7.2.6*** *Let $(X_i)_{i\in\{1,\ldots,n\}}$, $n\in\mathbb{N}^*$, be a family of discrete random variables with value in $\chi$, a countable space which is summable. We have*

$$\mathcal{S}(X_1,\ldots,X_n)=\sum_{i=1}^{n}\mathcal{S}(X_i|X_1,\ldots,X_{i-1}),$$

*with "$\mathcal{S}(X_1|X_0)=\mathcal{S}(X_1)$".*

If the random variables are all independent, then for any $(x_1,\ldots,x_{i-1})\in\chi^{i-1}$, we have (see Definition 7.2.3)

$$\begin{aligned}
&\mathcal{S}(X_i|X_1=x_1,\ldots,X_{i-1}=x_{i-1})\\
&=-\sum_{x_i\in\chi}\mathbb{P}(X_i=x_i|X_1=x_1,\ldots,X_{i-1}=x_{i-1})\ln\mathbb{P}(X_i=x_i|X_1=x_1,\ldots,X_{i-1}=x_{i-1})\\
&\qquad=-\sum_{x_i\in\chi}\mathbb{P}(X_i=x_i)\ln\mathbb{P}(X_i=x_i)=\mathcal{S}(X_i).
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\mathcal{S}(X_i|X_1,\ldots,X_{i-1})=\sum_{x_i\in\chi}\mathbb{P}(X_i=x_i)\mathcal{S}(X_i|X_1=x_1,\ldots,X_{i-1}=x_{i-1})\\
&=\mathcal{S}(X_i)\sum_{x_i\in\chi}\mathbb{P}(X_i=x_i)=\mathcal{S}(X_i).
\end{aligned}$$

***Proof***

From Corollary 7.2.5, we have

$$\begin{aligned}
\mathcal{S}(X_1,\ldots,X_n)&=\mathcal{S}(X_1)+\mathcal{S}(X_2,\ldots,X_n|X_1)=\mathcal{S}(X_1)+\mathcal{S}(X_2|X_1)\\
&\quad+\mathcal{S}(X_3,\ldots,X_n|X_1,X_2),
\end{aligned}$$

and we iteratively obtain the desired equation. ∎

The previous Corollary 7.2.6 will be useful in the sequence of this section. We now introduce the Entropy Rate.

***Definition 7.2.7*** *Let $\boldsymbol{X} = (X_i)_{i \in \mathbb{N}}$ be a discrete stochastic process with value in $\chi$. The* Entropy Rate *of $\boldsymbol{X}$ is given by*

$$\boldsymbol{\mathcal{S}}(\boldsymbol{X}) = \lim_{n \to +\infty} \frac{1}{n+1} \mathcal{S}(X_0, \ldots, X_n).$$

Suppose all the random variables are independent, then by Corollary 7.2.6 and the remark that follows, we have

$$\boldsymbol{\mathcal{S}}(\boldsymbol{X}) = \lim_{n \to +\infty} \frac{1}{n+1} \sum_{i=0}^{n} \mathcal{S}(X_i),$$

which proves that $\boldsymbol{\mathcal{S}}(\boldsymbol{X})$ is in this case the Cesàro limit of the sequence $(\mathcal{S}(X_i))_{i \in \mathbb{N}}$. If now the random variables are all identically distributed, then $\mathcal{S}(X_i) = \mathcal{S}(X_0)$ for any $i \in \mathbb{N}$, and $\boldsymbol{\mathcal{S}}(\boldsymbol{X}) = \mathcal{S}(X_0)$.

***Definition 7.2.8*** *Let $\boldsymbol{X} = (X_i)_{i \in \mathbb{N}}$ be a discrete stochastic process with value in $\chi$. It is (strongly)* stationary *if, for any $m \in \mathbb{N}$ and $(x_0, \ldots, x_m) \in \chi^{m+1}$, we have*

$$\mathbb{P}(X_{n+m} = x_m, \ldots, X_n = x_0) = \mathbb{P}(X_m = x_m, \ldots, X_0 = x_0), \quad \forall n \in \mathbb{N}.$$

***Proposition 7.2.9*** *Let $\boldsymbol{X} = (X_i)_{i \in \mathbb{N}}$ be a stationary discrete stochastic process with value in $\chi$. Then*

$$\boldsymbol{\mathcal{S}}(\boldsymbol{X}) = \lim_{n \to +\infty} \mathcal{S}(X_n | X_{n-1}, \ldots, X_0).$$

***Proof***
For any $n \in \mathbb{N}$ and $(x_0, \ldots, x_n) \in \chi^{n+1}$, we note that

$$\mathcal{S}(X_{n+1} | X_n = x_n, \ldots, X_0 = x_0) \leq \mathcal{S}(X_{n+1} | X_n = x_n, \ldots, X_1 = x_1),$$

hence

$$\mathcal{S}(X_{n+1} | X_n, \ldots, X_0) \leq \mathcal{S}(X_{n+1} | X_n, \ldots, X_1).$$

The stationarity implies that

$$\mathcal{S}(X_{n+1} | X_n, \ldots, X_1) = \mathcal{S}(X_n | X_{n-1}, \ldots, X_0).$$

Thus, the sequence $(\mathcal{S}(X_n | X_{n-1}, \ldots, X_0))_{n \in \mathbb{N}}$ is decreasing and positive. It therefore converges to an element written as $\boldsymbol{\mathcal{S}}'(\boldsymbol{X})$. Therefore, for any $n \in \mathbb{N}$, we have, by Corollary 7.2.6,

$$\mathcal{S}(X_0, \ldots, X_n) = \sum_{i=0}^{n} \mathcal{S}(X_i | X_1, \ldots, X_{i-1}),$$

hence the sequence $\left(\frac{1}{n+1}\mathcal{S}(X_0, \ldots, X_n)\right)_{n\in\mathbb{N}}$ is the Cesàro average of the sequence $(\mathcal{S}(X_n|X_{n-1}, \ldots, X_0))_{n\in\mathbb{N}}$. Therefore, it converges to its limit, and $\boldsymbol{\mathcal{S}}'(\boldsymbol{X}) = \boldsymbol{\mathcal{S}}(\boldsymbol{X})$. ∎

## 2.2 *Markov Traceability*

We are now focusing on a graph of the Peer-to-Peer Network style. An example is shown in Fig. 7.2. This is a graph $G$ with vertices $V$ and edges $E$ (see Sect. 1, Chap. 4).

We introduce the stochastic process $\boldsymbol{X} = (X_i)_{i \in \mathbb{N}}$ with value in $V$, representing the position of the Satoshi unit in the graph. More specifically, for any $x \in V$, $\{X_n = x\}$ is the event that the Satoshi unit is on node $x$ at "time" $n \in \mathbb{N}$. Thus, the stochastic process $\boldsymbol{X}$ is describing a *random walk* on the graph $G$.

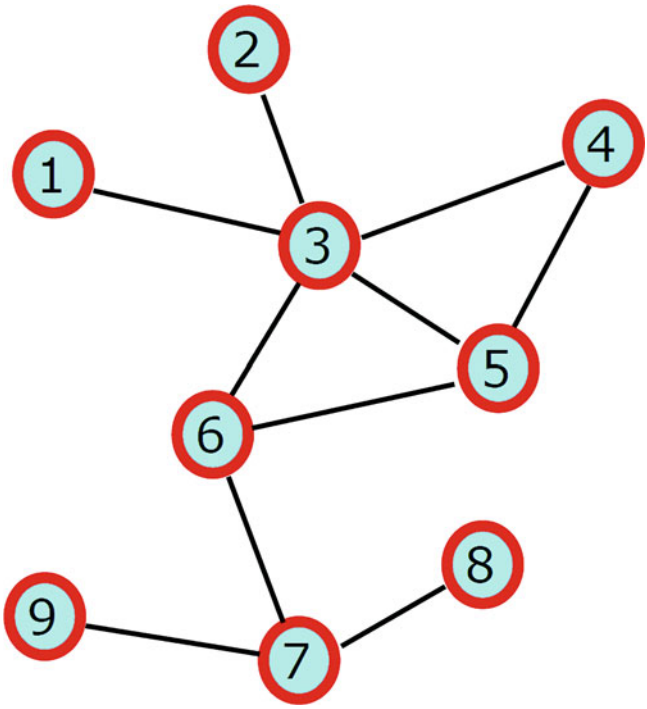


**Fig. 7.2** A typical Peer-to-Peer Network graph, where each vertex is a miner, a user, or a crypto-exchange. We are describing the evolution of a Satoshi unit diffusion throughout this network

***Definition 7.2.10*** *A* random walk *is a stochastic process* $\boldsymbol{X} = (X_i)_{i \in \mathbb{N}}$ *with value on a countable set V such that* $G = (V, E)$ *for some edges E, and there exists a function f such that*

$$X_n = X_{n-1} + f(X_{n-1}, \ldots, X_0), \quad \forall n \in \mathbb{N}^*.$$

*We say that* $\boldsymbol{X}$ *is a random walk on G.*

In other words, the position of the walker at time $n$ is depending on the position at previous time $n-1$, and is also function of its history on the chain, as well as other potential parameters associated with $G$.

In practice, $V$ is finite, therefore we set $V = \{1, \ldots, m\}$ without any loss of generality.

***Definition 7.2.11*** *Let* $G = (V, E)$ *be a graph and* $\boldsymbol{X}$ *be a random walk on G. We say that* $\boldsymbol{X}$ *is a* Markov chain *if, for any* $n \in \mathbb{N}$ *and family* $(x_0, \ldots, x_n) \in V^{n+1}$*, we have*

$$\mathbb{P}(X_n = x_n | X_{n-1} = x_{n-1}, \ldots, X_1 = x_1) = \mathbb{P}(X_n = x_n | X_{n-1} = x_{n-1}).$$

In other words, the walker does not remember its position before time $n-1$.

***Definition 7.2.12*** *Let* $G = (V, E)$ *be a graph and* $\boldsymbol{X}$ *be a Markov chain on G. We say the* $\boldsymbol{X}$ *is* homogeneous *if*

$$\mathbb{P}(X_n = x_n | X_{n-1} = x_{n-1}) = \mathbb{P}(X_1 = x_n | X_0 = x_{n-1}), \quad \forall n \in \mathbb{N}^*.$$

In other words, the probability for the walker to get from vertex $x_{n-1}$ to vertex $x_n$ is the same at any time of the walk.

***Definition 7.2.13*** *Let* $G = (V, E)$ *be a graph and* $\boldsymbol{X}$ *be a homogeneous Markov chain on G. A* Transition Probability Matrix $P = (p_{ij})_{1 \leq i, j \leq m}$ *is a matrix such that, for all* $1 \leq i, j \leq m$*, the element* $p_{ij}$ *is the probability that a walker goes to* $j \in V$ *from* $i \in V$*.*

It turns out that

$$\sum_{j=1}^{m} p_{ij} = 1, \quad \forall i \in \{1, \ldots, m\}.$$

Therefore, $P$ is a stochastic matrix. In particular, we have

$$\mathbb{P}(X_1 = j | X_0 = i) = p_{ij}.$$

***Definition 7.2.14*** *Let* $G = (V, E)$ *be a graph and* $\boldsymbol{X}$ *be a homogeneous Markov chain on G. We say that* $\boldsymbol{X}$ *is a Markov chain* of order $l \in \mathbb{N}^*$ *if, for all vertex* $i \in V$*, the walker has equal chances to run through any path of length l from i in the graph. Calling* $\mathcal{P}_l(i)$ *the collection of all paths of length l from i, this means that*

$$\mathbb{P}(X_{n+l}=j|X_n=i)=\frac{1_{j \text{ terminates a path in } \mathcal{P}_l(i)}}{\text{Card}(\mathcal{P}_l(i))}, \quad \forall n \in \mathbb{N}^*.$$

We shall call such a chain an $l$-Markov chain. The sentence "$j$ terminates a path in $\mathcal{P}_l(i)$" designates any path of the style $(i, *, *, \ldots, *, j)$ and the length is $l$ (i.e., the previous family has $l + 1$ elements).

As an example, the transition matrix $P$ for the graph in Fig. 7.2 related to a 1-Markov chain is given by

$$P=\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/5 & 1/5 & 0 & 1/5 & 1/5 & 1/5 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 0 & 1/3 & 0 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/3 & 0 & 1/3 & 1/3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Regarding a 2-Markov chain process, the vertex 7 has five paths of length 2: $(7, 6, 3)$, $(7, 6, 5)$, $(7, 6, 7)$, $(7, 8, 7)$, and $(7, 9, 7)$. Therefore,

$$p_{7j}=\frac{1_{j \text{ terminates a path in } \mathcal{P}_2(7)}}{5}.$$

## 2.3 *Maximal Entropy Random Walk*

This section is inspired by [18]. The following content should be a starting point for designing a diffusion that maximizes the entropy rate.

***Definition 7.2.15*** *Let $G = (V, E)$ be a graph and $\boldsymbol{X}$ be a homogeneous Markov chain on $G$. We say that $\boldsymbol{X}$ is a* Maximal Entropy Random Walk (MERW) *if it is a $+\infty$-Markov chain (or an $l$-Markov chain when $l$ tends to infinity).*

***Theorem 7.2.16*** *Let $G = (V, E)$ be a graph and $\boldsymbol{X}$ be a homogeneous Markov chain on $G$. If the graph is* primitive *(see Definition 4.4.3 – $G$ is primitive if its adjacency matrix is primitive), then the stochastic process $\boldsymbol{X}$ has a unique stationary law $(\mu_i)_{i \in V}$ and the entropy rate is given by*

$$\mathcal{S}(\boldsymbol{X})=-\sum_{(i,j)\in V^2} \mu_i\, p_{ij} \ln p_{ij}.$$

***Proof***
The first statement comes from Theorem 4.4.4. Regarding the second statement, let $p_n = (\mathbb{P}(X_n = i))_{i \in V}$ be the vector for the probability law at time $n \in \mathbb{N}$. We have the equation

$$p_n = P' p_{n-1}, \quad \forall n \in \mathbb{N}^*.$$

This equation is the *Kolmogorov* equation, and is straightforward to show by using the total probability formula. Indeed, for any $n \in \mathbb{N}^*$ and $i \in V$, we have

$$\mathbb{P}(X_n = i) = \sum_{j \in V} \mathbb{P}(X_n = i | X_{n-1} = j) \mathbb{P}(X_{n-1} = j).$$

Hence,

$$p_n = (P')^n p_0.$$

Next, from Definition 7.2.3, we have

$$\mathcal{S}(X_n | X_{n-1}) = \sum_{i \in V} \mathbb{P}(X_{n-1} = i) \mathcal{S}(X_n | X_{n-1} = i) = -\sum_{i,j \in V} \mathbb{P}(X_{n-1} = i)\, p_{ij} \ln p_{ij}.$$

Hence,

$$\lim_{n \to +\infty} \mathcal{S}(X_n | X_{n-1}) = -\sum_{i,j \in V} \lim_{n \to +\infty} (\mathbb{P}(X_{n-1} = i))\, p_{ij} \ln p_{ij} = -\sum_{i,j \in V} \mu_i\, p_{ij} \ln p_{ij}.$$

We will show that $\lim_{n \to +\infty} \boldsymbol{\mathcal{S}}(X_n | X_{n-1}) = \boldsymbol{\mathcal{S}}(\boldsymbol{X})$. Indeed, for any $(x_1, \ldots, x_{n-1}) \in \chi^{i-1}$, we have (see Definition 7.2.3)

$$\begin{aligned}
&\mathcal{S}(X_n | X_1 = x_1, \ldots, X_{n-1} = x_{n-1}) \\
&\quad = -\sum_{x_n \in \chi} \mathbb{P}(X_n = x_n | X_1 = x_1, \ldots, X_{n-1} = x_{n-1}) \ln \mathbb{P}(X_n = x_n | X_1 = x_1, \ldots, X_{n-1} = x_{n-1}) \\
&\quad = -\sum_{x_n \in \chi} \mathbb{P}(X_n = x_n | X_{n-1} = x_{n-1}) \ln \mathbb{P}(X_n = x_n | X_{n-1} = x_{n-1}) \\
&\quad = \mathcal{S}(X_n | X_{n-1} = x_{n-1}).
\end{aligned}$$

We therefore have

$$\mathcal{S}(X_n | X_{n-1}, \ldots, X_0) = \mathcal{S}(X_n | X_{n-1}),$$

and conclude with Proposition 7.2.9. ∎

We conclude this section with the following important theorem.

***Theorem 7.2.17*** *Let $G = (V, E)$ be a primitive graph and $\boldsymbol{X}$ be a homogeneous Markov chain on G, with adjacency matrix M with highest eigenvalue $\lambda$. Assume that M has elements in $\{0, 1\}$. Then the Entropy Rate $\boldsymbol{\mathcal{S}}(\boldsymbol{X})$ is maximal and equal to $\ln\lambda$, if and only if $\boldsymbol{X}$ is a MERW.*

***Lemma 7.2.18*** *Let $G = (V, E)$ be a graph, and its adjacency matrix is M with simple and dominant highest eigenvalue $\lambda$ (it is positive and strictly higher than the others). Let u and v be the left and right eigenvectors for the eigenvalue $\lambda$, i.e., $Mu = \lambda u$ and $v^{'}M = \lambda v^{'}$. Then we have:*

$$\lim_{k\to+\infty}\left(\frac{M}{\lambda}\right)^k = \frac{1}{v'u}\,uv'.$$

***Proof***

Let $A = M - \frac{\lambda}{v'u}uv'$. The set $v^{\perp}$ is the orthogonal space enhanced by all vectors that are orthogonal with $v$, i.e., $z \in v^{\perp}$ if and only if $z^{'}v = 0$. We can decompose $\mathbb{R}^m$ such that $\mathbb{R}^m = \mathbb{R}u \bigoplus v^{\perp}$.

Note that $M(\mathbb{R}u) \subset \mathbb{R}u$ and $M(v^{\perp}) \subset v^{\perp}$, thus we can define the restriction $M_{|v^{\perp}}$. We have $A = M_{|v^{\perp}}$ (since $Au = Mu - \dfrac{\lambda}{v'u}(v'u)u = \lambda u - \lambda u = 0$, hence $A(\mathbb{R}u) = \{0\}$). Therefore, the eigenvalues of $A$ are strictly comprised between $-\lambda$ and $\lambda$.We can write

$$M = \begin{pmatrix} M_{|\mathbb{R}u} & * \\ 0 & A \end{pmatrix}.$$

From the Jordan Normal form theorem (see Sect. 2, Chap. 4), there exists $(P, J) \in \mathcal{GL}_m(\mathbb{C}) \times \mathcal{M}_m(\mathbb{C})$ such that $A = PJ^{\perp}P^{-1}$, where

$$J^{\perp} = \begin{pmatrix} \lambda_2 & 1 & \cdots & 0 \\ 0 & \lambda_3 & \cdots & 0 \\ 0 & 0 & \ddots & 1 \\ 0 & 0 & \cdots & \lambda_m \end{pmatrix},$$

with $\lambda = 1 > \lambda_i$, $\forall i \in \{2, 3, \ldots, m\}$. Next, for any $k \in \mathbb{N}$, we have $\left(\frac{A}{\lambda}\right)^k = P\left(\frac{J^{\perp}}{\lambda}\right)^k P^{-1}$, and $\left(\frac{J^{\perp}}{\lambda}\right)^k$ is an upper-triangular matrix that writes as monomials of the $\lambda_i/\lambda$'s to the power of $k$. Since $\lambda > \lambda_i$, $\forall i \in \{2, 3, \ldots, n\}$, we therefore have $\lim_{k\to+\infty}\left(\frac{J^{\perp}}{\lambda}\right)^k = 0$ and $\lim_{k\to+\infty}\left(\frac{A}{\lambda}\right)^k = 0$.

From the equation $A = M - \frac{\lambda}{v'u}uv'$, we have, for any $k \in \mathbb{N}$

$$\left(\frac{M}{\lambda}\right)^k = \left(\frac{A}{\lambda}\right)^k + \frac{1}{v'u}uv'.$$

We conclude the lemma. ∎

***Proof of Theorem 7.2.17***

We note that the MERW is following the *Principle of Maximal Entropy* for paths of infinite length: there is an equal probability that the walker chooses one path of infinite length (see Definitions 7.2.14 and 7.2.15). Henceforth, for any two $i, j \in \{1, \ldots, m\}$, any path from $i$ to $j$ has an equal probability to be taken. The entropy rate is maximized by definition, as a limit of entropy of multivariate random variables associated with such random walk (this was mostly "heuristic," but intuitive though). It remains to prove that the entropy for a MERW is equal to $\ln\lambda$.

Suppose $\boldsymbol{X}$ is a MERW. Let $u$ and $v$ be the left and right eignevectors for the eigenvalue $\lambda$, i.e., $Mu = \lambda u$ and $v^{'}M = \lambda v^{'}$. Lemma 7.2.18 implies that

$$M^k \underset{k\to+\infty}{\sim} \frac{\lambda^k}{v'u} uv'.$$

Henceforth, by setting $\mathcal{B} = (e_1, \ldots, e_n)$ the canonical basis of $\mathbb{R}^n$, and by multiplying the previous relationship with adequate sums of elements in $\mathcal{B}$ from the left and/or right, we deduce that

$$\begin{cases} \sum_{i=1}^{n} (M^k)_{il} \underset{k\to+\infty}{\sim} \frac{\lambda^k}{v'u} \left(\sum_{i=1}^{n} u_i\right) v_l, & \forall l \in \{1, \ldots, n\} \\ \sum_{j=1}^{n} (M^k)_{lj} \underset{k\to+\infty}{\sim} \frac{\lambda^k}{v'u} \left(\sum_{j=1}^{n} v_j\right) u_l, & \forall l \in \{1, \ldots, n\} \end{cases}$$

We deduce that, by multiplying the two above equivalents, the number of paths $\mathcal{N}_k(l)$ from $l$ and of length $2k$ is given by

$$\mathcal{N}_k(l) = \sum_{i,j=1}^{n} (M^k)_{il} (M^k)_{lj} \underset{k\to+\infty}{\sim} \frac{\lambda^{2k}}{(v'u)^2} \left(\sum_{j=1}^{n} v_j\right) \left(\sum_{i=1}^{n} u_i\right) u_l v_l.$$

We set

$$m_k(l) = \frac{\lambda^{2k}}{(v'u)^2} \left(\sum_{j=1}^{n} v_j\right) \left(\sum_{i=1}^{n} u_i\right) u_l v_l.$$

The metric $m_k(l) / \sum_{l'=1}^{n} m_k(l')$ must coincide with the unique stationary law of the $+\infty$-Markov chain, since the matrix $G$ is primitive. Since

$$\frac{m_k(l)}{\sum_{l'=1}^{n} m_k(l')} = \frac{u_l v_l}{\sum_{l'=1}^{n} u_{l'} v_{l'}},$$

we therefore deduce that the stationary law $(\mu_l)_{l \in \{1,\dots,n\}}$ is given by

$$\mu_l = \frac{u_l v_l}{\sum_{l'=1}^{n} u_{l'} v_{l'}} = \frac{u_l v_l}{v' u}.$$

It remains to calculate

$$\begin{aligned} p_{ij} &= \mathbb{P}(X_1 = j | X_0 = i) = \lim_{k \to +\infty} \mathbb{P}(X_k = j | X_{k-1} = i) \\ &= \lim_{k \to +\infty} \frac{\mathbb{P}(X_k = j, X_{k-1} = i)}{\mathbb{P}(X_{k-1} = i)}, \quad \forall k \\ &\in \mathbb{N}, \end{aligned}$$

by stationarity. First, $\lim_{k \to +\infty} \mathbb{P}(X_{k-1} = i) = \mu_i$. Second, the probability $\lim_{k \to +\infty} \mathbb{P}(X_k = j, X_{k-1} = i)$ must be proportional to $v_i M_{ij} v_j$, and we therefore write

$$\lim_{k \to +\infty} \mathbb{P}(X_k = j, X_{k-1} = i) = \frac{v_i M_{ij} u_j}{\sum_{i',j'=1}^{n} v_{i'} M_{i'j'} u_{j'}} = \frac{v_i M_{ij} u_j}{v' M u} = \frac{v_i M_{ij} u_j}{\lambda v' u}.$$

Therefore,

$$p_{ij} = \frac{\lim_{k \to +\infty} \mathbb{P}(X_k = j, X_{k-1} = i)}{\lim_{k \to +\infty} \mathbb{P}(X_{k-1} = i)} = \frac{\frac{v_i M_{ij} u_j}{\lambda v' u}}{\frac{u_i v_i}{v' u}} = \frac{M_{ij}}{\lambda} \frac{u_j}{u_i}.$$

Henceforth, from Theorem 7.2.16, we have

$$\mathcal{S}(X) = -\sum_{(i,j)\in V^2} \mu_i\, p_{ij} \ln p_{ij} = -\sum_{i,j=1}^{n} \frac{u_i v_i}{v'u}\,\frac{M_{ij}}{\lambda}\,\frac{u_j}{u_i} \ln\left(\frac{M_{ij}}{\lambda}\,\frac{u_j}{u_i}\right) = -\frac{1}{\lambda v'u}$$

$$\times \sum_{i,j=1}^{n} v_i M_{ij} u_j \ln\left(\frac{M_{ij}}{\lambda}\,\frac{u_j}{u_i}\right) = \frac{v'Mu}{\lambda v'u}\ln\lambda + \frac{1}{\lambda v'u}$$

$$\times \sum_{i,j=1}^{n} v_i M_{ij} u_j (\ln(M_{ij}u_i) - \ln(M_{ij}u_j)) = \ln\lambda + \frac{1}{\lambda v'u}\sum_{i,j=1}^{n} v_i M_{ij} u_j (\ln u_i - \ln u_j)$$

$$+\frac{1}{\lambda v'u}\sum_{i,j=1}^{n} v_i M_{ij} \ln(M_{ij}) u_j = \ln\lambda + \frac{1}{\lambda v'u}\sum_{i,j=1}^{n} (v_i \lambda u_i \ln u_i - \lambda v_j u_j \ln u_j) + \frac{1}{\lambda v'u}$$

$$\times \sum_{i,j=1}^{n} v_i M_{ij} \ln(M_{ij}) u_j = \ln\lambda + \frac{n}{\lambda v'u}\sum_{i=1}^{n} (v_i \lambda u_i \ln u_i - v_i \lambda u_i \ln u_i) + \frac{1}{\lambda v'u}$$

$$\times \sum_{i,j=1}^{n} v_i M_{ij} \ln(M_{ij}) u_j = \ln\lambda + \frac{1}{\lambda v'u}\sum_{i,j=1}^{n} v_i M_{ij} \ln(M_{ij}) u_j.$$

Since $M$ has elements in $\{0,1\}$, then $M_{ij} \ln(M_{ij}) = 0$ for any $(i,j) \in \{1,\ldots,n\}^2$ and using the convention $0 \times \ln 0 = 0$. We conclude that $\mathcal{S}(X) = \ln\lambda$. ∎

In practice, by choosing the relevant adjacency matrix $M$ (or equivalently the relevant graph $G$), then one can benefit from considering the stationary law to model a random walk behavior at equilibrium (i.e., steady state), by just calculating the highest eigenvector's components, and at maximal Entropy Rate. This technic actually is quite theoretically involving but has the surprisingly virtue to be straightforwardly implementable!

## 3 Privacy and the Blockchain Transaction Graph

We introduce the Shannon Entropy to the Blockchain Transaction Graph, that is the Privacy Entropy. This is another type of entropy and it is describing all the different backward paths from an UTXO up to from where a Satoshi unit could come from. This metric is particularly adapted for traceability, but also for the amount of information about the possible sources of cash finally stored in an UTXO. We will see that a way to improve privacy, i.e., anonymity of addresses' owner on the blockchain, is to increase the Privacy Entropy. The approach will lead to an optimal cost of transactions, in a way we are going to define.

## 3.1 The Privacy Issue

Many Bayesian technics exist to identify a user through the information released on the blockchain. From an address, a privacy attacker can go backward through the Blockchain Transaction Graph, and identify the history of the transactions of a particular user. See [19, 20] for more information.

## 3.2 A Privacy Violation Model

As elaborated in [20], privacy needs to be modeled by considering a bunch of addresses. In fact, an attacker cannot deduce anything with only one transaction: she will identify the transaction in which an initially chose UTXO is an output, and will appropriately select the inputs she thinks are associated with the same user, in a *direct* (the same user appears in the input and output) or *indirect* (the user does not appear in the output, but, e.g., mainly may appear in the inputs of previous transactions) way. The selection input(s) is (are) the output(s) of a previous transaction, so she will iteratively repeat the previous dynamic to get the bunch of addresses she thinks are associated with one unique user.

As an illustration, suppose an attacker is targeting a specific UTXO. This UTXO is the output of a transaction which has, say, two inputs. Each input is the output of an already spent output, but a direction should mainly refer more to the targeted user. Thus, the attacker has one chance over two to get the appropriate path leading to the bunch of addresses that gives the optimal amount of information about the user: the probability of choosing the optimal path, i.e., the right one, is 1/2. Since each choice is considered to be independent from the previous ones, one can multiply the probabilities to get the probability of a path. This model actually is related to a star tree, where the middle of the tree is the chosen UTXO. Thus, in Fig. 7.3, if the chosen UTXO is $\text{UTXO}_1$, then the probability the attacker choses the red path is $\left(\frac{1}{2}\right)^3$ as indicated.

If the set of UTXOs was indexed, we simply have $\sum_{k=1}^{K} p_i(k) = 1$, where $p_i(k)$ is the probability for path $k$ from the $i^{\text{th}}$ UTXO and $K$ is the total number of paths from the considered UTXO. We give the probabilities for the paths in Fig. 7.3:

$$
\begin{aligned}
p_1(1) &= 1/4, \\
p_1(2) &= p_1(3) = 1/8, \\
p_1(4) &= p_1(5) = 1/8, \\
p_1(6) &= p_1(7) = p_1(8) = 1/12, \\
p_2(1) &= p_2(2) = 1/4, \\
p_2(3) &= p_2(4) = p_2(5) = 1/6.
\end{aligned}
$$

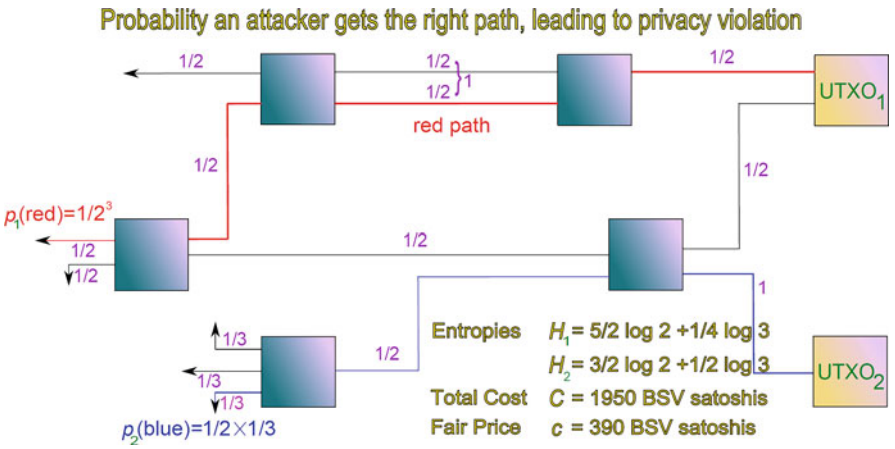


**Fig. 7.3** Model for the violation of privacy. An attacker chooses one UTXO and selects a bunch of spent addresses backwardly from it. The randomly selected bunch gathers addresses that the attacker believes are the most informative toward one unique targeted user. From the blockchain – as schematized above – there is an optimal path in the sense that the attacker will gather specific addresses to get the most possible information regarding the user. Thus, if the attacker chooses $UTXO_1$ above and the optimal path is the red bold one, then she will have $1/2^3$ probability to get the optimal bunch of addresses. Thus, we can derive a complete set of events associated with one UTXO. To complement this model, we define in Sect. 4.1 the entropy of privacy associated with each UTXO, which quantitatively assesses the difficulty to violate privacy. Finally, the total cost and fair price for an exchange to create this system also are specified. (Source: Julien Riposo, 2023)

# 4 The Particles Model and the Entropy of Privacy

In this section, we directly propose a solution for the previous problem.

## *4.1 Entropy of Privacy*

We focus on the Blockchain Transaction Graph, as described in Fig. 7.1b, and the related section. As explained, the graph ends in absorbing states, which represent UTXOs. At each time, there are a given number of UTXOs. We focus on the $i^{\text{th}}$ UTXO and introduce the following Privacy Entropy.

***Definition 7.4.1 Privacy Entropy*** *Let $p_i(k)$ be the probability for path k from the $i^{\text{th}}$ UTXO, with a total of $K \in \mathbb{N}$ different paths. The* Entropy of Privacy *is the Shannon Entropy associated with this UTXO:*

$$\mathcal{S}_i = -\sum_{k=1}^{K} p_i(k) \ln p_i(k).$$

The total entropy is

$$\mathcal{S} = \sum_i \mathcal{S}_i.$$

Indeed, this model supposes that the process of selecting the addresses starts from one unique UTXO, and does not depend on any other UTXO. Thus, we have an independency of UTXOs and of the selection processes. Here, this entropy is a metric of privacy, that is, it indicates how difficult it is for the attacker to find the right path for the user identification. In fact, we have

$$\mathcal{S} \geq 0,$$

and if $\mathcal{S} = 0$, then the attacker will surely identify the *optimal* path to reach maximum information related to one user. Besides, the higher the privacy entropy, the higher the *disorder* in the system and the more difficult to find the optimal path.

As an illustration, we give the entropies for the two UTXOs in Fig. 7.3.

## *4.2 The Particles Model*

One possibility is to *dilute* privacy through an institution, e.g., an exchange, when a client is ordering a position. Dilution could mean *partition* of an amount, say $X$ BTC ($X > 0$) into distinct UTXOs to be stacked into the blockchain. Specifically, the exchange makes a transaction where the input is $X$ BTC, and the output is a set of $n$ UTXOs ($n \geq 1$). The first UTXO has amount of $Z_1$ BTC, the second one has amount of $Z_2$ BTC, and so on. This process may result in millions of UTXOs per such transaction ($n$ is a big number).

Such process may be realized for all the buy orders, i.e., for each customer and each order, resulting in millions of millions of distinct UTXOs. It is worth stressing to call these UTXOs *particles*: each of them has a sufficiently low amount of BTC so that only one particle may not be used as a proper transaction. However, a set of particles might constitute a proper transaction: a sell order may result in selling an amount of BTC to the source of BTCs (inverse path), by means of gathering a set of particles to form a unique tangible BTC amount. This model is close to an order book model, but the orders are divided into particles, making the match more efficient. The exchange would thus become a system of particles, where forces of repulsion (split the amount $X$ BTC into particles) and attraction (construct an amount $Y$ BTC from particles) generate the whole stability. Fig. 7.4 is illustrating the model.

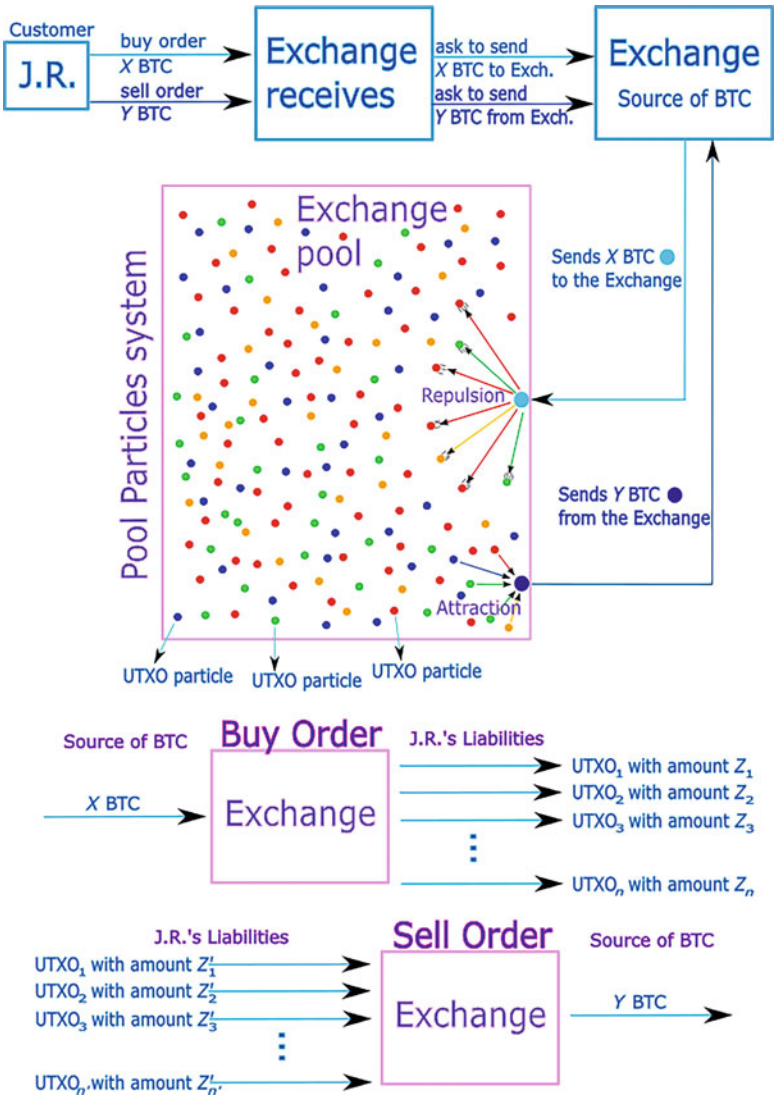


**Fig. 7.4** The *Particles model*: a customer (say J.R. for Julien Riposo) balance is *decomposed* into several particles, i.e., UTXOs of very small BTC amounts. The particles constitute the matter of each amount, and they are gravitating in an exchange system ruled by forces of repulsion (giving the particles in case of a buy order) and of attraction (in case of a sell order). This results in significant improvement of privacy, as it becomes more difficult to follow the identity of customers. (Source: Julien Riposo, 2023)

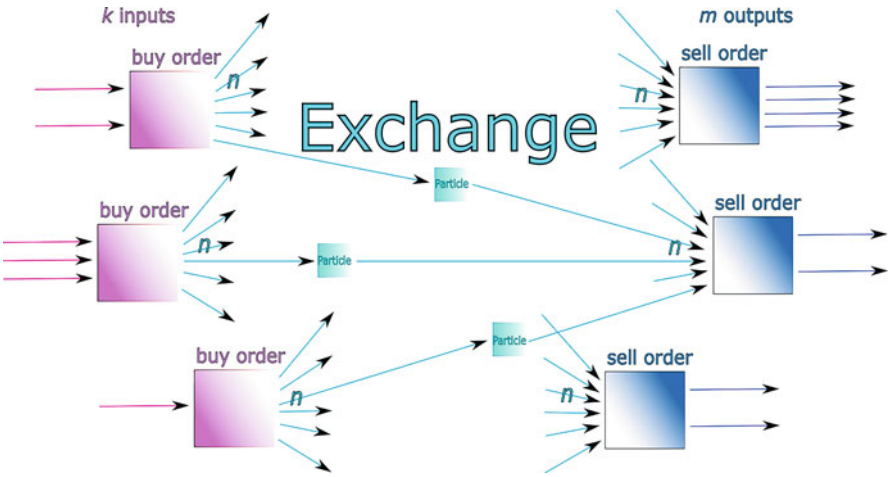


**Fig. 7.5** Application of the Privacy Violation model to an exchange's system utilizing the Particles model. There are buy order transactions (left-hand side, purple boxes) having in total $k$ inputs and sell order transactions (right-hand side, blue boxes) having in total $m$ outputs. We suppose, to simplify, that there are systematically $n$ particles involved for explosions and implosions. We can derive a fair price-optimal calibrated strategy, which makes the entropy increase. (Source: Julien Riposo, 2023)

## *4.3 Average Entropy*

We now apply the above modeling to an exchange that uses the Particles model. The exchange is represented in Fig. 7.5.

Let $I \geq 1$ and $J \geq 1$ be the number of buy orders (purple blocks in Fig. 7.5, left-hand side) and sell orders (blue blocks in Fig. 7.5, right-hand side), respectively, each indexed by the variables $i \in \{1, \ldots, I\}$ and $j \in \{1, \ldots, J\}$. Let $k^{(i)}$ be the number of distinct inputs for the $i^{\text{th}}$ buy order block and $n_{i,j}$ be the number of particles going from the $i^{\text{th}}$ buy order to the $j^{\text{th}}$ sell order.

***Definition 7.4.2*** *The* Average Condition for the Entropy *refers to when* $k^{(i)} = \overline{k}$ *(independent of i) and* $n_{i,j} = \mathrm{C}_j^{\text{te}}$ *(same number of paths from the* $i^{\text{th}}$ *buy order to the* $j^{\text{th}}$ *sell order, for any i).*

***Theorem 7.4.3*** *The* Average *entropy of privacy, i.e., the entropy derived from the* Average Condition*, is given by*

$$\mathcal{S} = \frac{J}{\overline{k}} \ln\left(I\overline{k}\right).$$

***Proof***

Let $j \in \{1, \ldots, J\}$. We focus on the $j^{\text{th}}$ sell order block. One path essentially is given by the pair or numbers $(i, j)$, for some $i \in \{1, \ldots, I\}$, as the attacker leaves from the $j^{\text{th}}$

sell order block to reach the $i^{\text{th}}$ buy order block. The number of possibilities to do that is given by $n_{i,j}$, and there are $n_j := \sum_{i=1}^{I} n_{i,j}$ total possibilities for the attacker to choose any buy order block. Once this latter is chosen, the attacker has $k^{(i)}$ choices for the backward next spent outputs. Therefore, the probability for this path to be the right one, when the attacker chooses the route $(i,j)$, is given by

$$p_{i,j} = \frac{n_{i,j}}{n_j} \times \frac{1}{k^{(i)}}.$$

The entropy associated with the $j^{\text{th}}$ sell order is given by

$$\mathcal{S}^{(j)} = -\sum_{i=1}^{I} p_{i,j} \ln p_{i,j} = \sum_{i=1}^{I} \frac{n_{i,j}}{n_j k^{(i)}} \ln \frac{n_j k^{(i)}}{n_{i,j}},$$

hence the total entropy of the system is given by

$$\mathcal{S} = \sum_{j=1}^{J} \mathcal{S}^{(j)} = \sum_{j=1}^{J} \sum_{i=1}^{I} \frac{n_{i,j}}{n_j k^{(i)}} \ln \frac{n_j k^{(i)}}{n_{i,j}}.$$

Following the Average Condition, we have $n_{i,j}/n_j = 1/I$ and $k^{(i)} = \overline{k}$, so that the average entropy is finally given by the desired result. ∎

This deserves a comment. In practice, this number could be used for implementation purposes. In addition, the entropy of privacy increases with the number of explosions $I$ (as a log) and implosions $J$ (linearly): having many sell orders further masks privacy.

In addition, and perhaps less intuitively, we observe that when $\overline{k}$ increases, the entropy decreases: there are less paths to the previous spent outputs, generating the buy order transactions. It is worth pointing out that the average entropy does not depend on the number of outputs for the sell orders, since this number does not depend on the privacy generated by the exchange.

## 4.4 Privacy Entropy Rate

This subsection is a connection between the Entropy of Privacy and the Entropy Rate discussed in Sect. 2. The Blockchain Transaction Graph $G = (V, E)$ can be decomposed into several intersected graphs. If there are $\mathfrak{I} \in \mathbb{N}^*$ UTXOs in total, and focusing on the $i^{\text{th}}$ UTXO, $i \in \{1, \ldots, \mathfrak{I}\}$, the graph $G_i = (V_i, E_i)$ enhanced by the $i^{\text{th}}$ UTXO clearly is a star graph, with the end being "genesis" addresses, i.e., exclusive inputs that are not outputs. We have

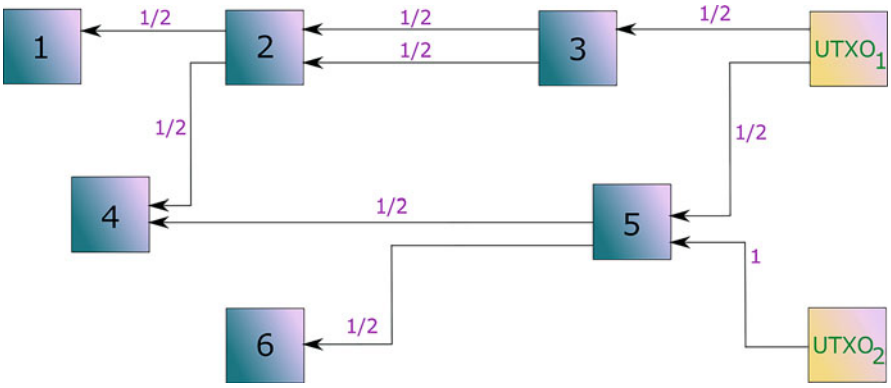


**Fig. 7.6** A particular Blockchain Transaction Graph, with "genesis" addresses 1, 4, 6. (Source: Julien Riposo, 2023)

$$G = \bigcup_{i=1}^{\mathfrak{I}} G_i,$$

but there is no reason that $G_i \cap G_j = \emptyset$, for $i \neq j$. For any $i \in \{1, \ldots, \mathfrak{I}\}$, let $P_i$ be the backward transition probability matrix for a system, e.g., as illustrated in Fig. 7.6.

In Fig. 7.6, the transition probability matrix for $\text{UTXO}_1$ and $\text{UTXO}_2$ are respectively (noting 0 the vertex $\text{UTXO}_i$, $i \in \{1, 2\}$ – the second matrix is made from vertices $\{0, 4, 5, 6\}$)

$$P_1 = \begin{pmatrix} 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}; P_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Fix $i \in \{1, \ldots, \mathfrak{I}\}$, we introduce the stochastic process $\boldsymbol{X} = (X_j)_{j \in \mathbb{N}}$, where $X_j$ is the position of the attacker at step $j$. We set $X_0 = 0$ (the attacker is at the considered UTXO). Then $\boldsymbol{X}$ clearly is a stepwise decision process. If there are $n + 1$ vertices in total (including the UTXO), then $\boldsymbol{X}$ is constant from rank $n$, i.e., $X_j = \mathrm{C}^{\mathrm{te}}$ for any $j \geq n$, and the constant depends on the chosen path. This is because the "genesis" addresses are absorbing states. In addition, we can assume that the stochastic process $\boldsymbol{X}$ is a homogeneous Markov chain. Therefore, for any $i \in \{1, \ldots, \mathfrak{I}\}$, for any $j \in \mathbb{N}^*$, and family $(i_1, \ldots, i_j) \in V_i$, we have

$$\mathbb{P}\big(X_j = i_j, \ldots, X_1 = i_1, X_0 = 0\big) = p_{i_{j-1} i_j} \cdots p_{0\, i_1},$$

in particular, for any $j - 1 \geq n$, since $p_{i_{j-1}i_j} = 1$ if $i_{j-1} = i_j = i_n$ is a "genesis" address, then

$$\begin{aligned}\mathbb{P}\big(X_j = i_n, \ldots, X_n = i_n, \ldots, X_1 = i_1, X_0 = 0\big) &= \mathbb{P}(X_n = i_n, \ldots, X_1 = i_1, X_0 = 0) \\ &= p_{i_{n-1}i_n} \cdots p_{0\, i_1},\end{aligned}$$

for all $j \geq n$. Therefore, the Shannon Entropy is such that

$$\begin{aligned}\mathcal{S}\big(X_j, \ldots, X_n, \ldots, X_1, X_0\big) &= -\sum_{i_1, \ldots, i_n} p_{i_{n-1}i_n} \cdots p_{0\, i_1} \ln\big(p_{i_{n-1}i_n} \cdots p_{0\, i_1}\big) \\ &= \mathcal{S}(X_n, \ldots, X_1, X_0).\end{aligned}$$

Henceforth, the entropy rate is

$$\begin{aligned}\boldsymbol{\mathcal{S}}(\boldsymbol{X}) &= \lim_{j\to\infty} \frac{1}{j+1} \mathcal{S}\big(X_j, \ldots, X_n, \ldots, X_1, X_0\big) \\ &= \left(\lim\nolimits_{j\to\infty} \frac{1}{j+1}\right) \mathcal{S}(X_n, \ldots, X_1, X_0) = 0.\end{aligned}$$

This is an interesting result, since it shows that the entropy rate really reflects the long-run behavior of the random walker, who will stop her exploration at genesis blocks for a finite blockchain (provided she is faster than the construction of the blockchain).

It is worth pointing out that the attacker might stop before the genesis blocks, if she judges the targeted user is not concerned with blocks from a given rank (backward). In this case, the entropy rate is still 0.

# Chapter 8
# Applications: Selection of Research Studies


Many studies have been developed around the blockchain, and it is obviously impossible to deal with all of them. In this final section, we would like to highlight key blockchain research topics through the viewpoint of the developed theory above. The selected articles bore the attention of the author, but are far from exhaustively reflecting the overall research, and the below could specifically be seen as examples whose underlying blockchain theory, the one developed previously in this book, is general, and can gather a priori distinct blockchain topics.

## 1 Satoshi Nakamoto's White Paper [7]: On the Probability of a 51% Attack

It is worth pointing out that the content of this book can be seen as a theoretical elaboration and extension of this white paper. There is an additional problem which we have not considered thus far, and would like to get deeper down to, which is the one of a 51% attack, explained in Section 11 in [7]. In fact, we could more rigorously assess the content of this section, and especially we could derive the probability that a 51% attacker "replaces" blocks already mined, and eventually adds more, with the help of Sect. 2.2, Chap. 2.

We introduce the Poisson random variables $N_t^{\mathrm{a}}$ and $N_t$, respectively, be the number of blocks that the *a*ttacker has reached at time $t$ and the number of blocks in the blockchain at time $t$, with respective intensities $\lambda_{\mathrm{a}}$ and $\lambda$. We are interested in the case where the number of blocks of the attacker is lower than the ones for the blockchain, i.e., the even $\left\{N_t \geq N_t^{\mathrm{a}} + z\right\}$, which is a safe configuration against 51% attack. Thus, we want to compute the probability $\mathbb{P}\left(N_t \geq N_t^{\mathrm{a}} + z\right)$. See Fig. 8.1 for an illustration.

The following theorem is a more formal elaboration of Satoshi Nakamoto's argument, based on the study developed thus far.



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3_8

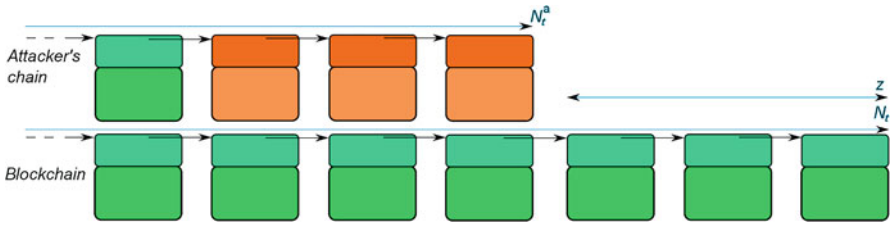


**Fig. 8.1** A safe configuration against 51% attack. The blockchain has $z$ more blocks than the attacker's; hence, according to the *consensus* rule (see Sect. 2, Chap. 4), the miners will not choose the attacker's blocks. (Source: Julien Riposo, 2023)

***Theorem 8.1.1*** *If we assume that $\lambda_a < \lambda$, then, following some other straightforward assumptions, for any $z \geq 0$, we have:*

$$\mathbb{P}\left(N_t \geq N_t^a + z\right) = \left(\frac{\lambda_a}{\lambda}\right)^z.$$

The assumption $\lambda_a < \lambda$ means the attacker has less resource than all the miners populating the blockchain.

***Proof***
Let $(t_i)_{i \in N}$ be the *realized* times that a new block appears either on the blockchain or in the attacker's chain. We introduce

$$b_i = \begin{cases} +1 & \text{if } \Delta N_{t_i} = 1 \\ -1 & \text{if } \Delta N_{t_i}^a = 1 \end{cases},$$

where we set $N_{t_{-1}} = N_{t_{-1}}^a = 0$ and $\Delta N_{t_i} = N_{t_i} - N_{t_{i-1}}$ for all $i \in \mathbb{N}$, same for $N_{t_i}^a$. In other words, $b_i$ is +1 if a block was added to the blockchain, or $-1$ if the attacker added a block to her/his chain.

We now make the claims that

- $(b_i)_{i \in N}$ are all independent and identically distributed;
- $\nexists i \in \mathbb{N} \quad \Delta N_{t_i} = \Delta N_{t_i}^a$.

The first bullet point allows to stress that $\mathbb{P}(\Delta N_{t_i} = 1)$ is the same, for any $i \in \mathbb{N}$. From the second one, we set

$$B_i = N_{t_i} - N_{t_i}^a = \begin{cases} \sum_{j=0}^{i} b_j & \text{if } i > 0 \\ 0 & \text{otherwise} \end{cases}$$

Therefore, given that $\left\{\Delta N_{t_i} = 1 \cap \Delta N_{t_i}^a = 0\right\} \cup \left\{\Delta N_{t_i} = 0 \cap \Delta N_{t_i}^a = 1\right\}$ is an almost sure event for any $i \in \mathbb{N}^*$, the *conditional law* of $B_i$ given the realized set $(t_i)_{i \in N}$ is a Binomial random walk, of parameters $(i, \mathbb{P}(\Delta N_{t_i} = 1))$. We now

introduce the set $\mathcal{A}_z^i = \{B_i \geq z\}$, which is the event that the attacker can have the same number of blocks as the blockchain, given s/he has $z$ blocks to catch up with, from the $i^{\text{th}}$ block. We note that $\mathbb{P}\left(\mathcal{A}_z^i\right)$ should not depend on $i$, by the memoryless of the process, so we call $\mathcal{A}_z$ such event. Thus, our goal is to derive $\mathbb{P}(\mathcal{A}_z)$, but the probability $\mathbb{P}(\Delta N_{t_i} = 1)$ needs to be calculated first.

Let $\tau_i$ be the *random* time that the $i^{\text{th}}$ block appears on the blockchain, and $\tau_i^{\text{a}}$ be the *random* time that the $i^{\text{th}}$ block appears on the attacker's chain. According to Sect. 2.2, Chap. 2, $\tau_i$ (resp. $\tau_i^{\text{a}}$) is an exponential random variable with parameter $\lambda$ (resp. $\lambda_{\text{a}}$). Therefore, we have, for any $i \in \mathbb{N}$:

$$\mathbb{P}(\Delta N_{t_i} = 1) = \mathbb{P}\left(\tau_i < \tau_i^{\text{a}}\right).$$

Indeed, for the $i^{\text{th}}$ round, the blockchain's block appears *before* the attacker's. Since $\tau_i$ and $\tau_i^{\text{a}}$ are independent, we have

$$\mathbb{P}\left(\tau_i < \tau_i^{\text{a}}\right) = \int_0^{+\infty} \int_0^x \lambda \, \mathrm{e}^{-\lambda y} \times \lambda_{\text{a}} \, \mathrm{e}^{-\lambda_{\text{a}} x} \, \mathrm{d}x\mathrm{d}y = \frac{\lambda}{\lambda + \lambda_{\text{a}}}.$$

Next, note that $\mathbb{P}(\mathcal{A}_0) = 1$, since $B_0 = 0$. Suppose now that $z \in \mathbb{N}^*$. We use the following recursive equation:

$$\mathbb{P}(\mathcal{A}_z) = \mathbb{P}(\Delta N_{t_i} = 1) \times \mathbb{P}(\mathcal{A}_{z+1}) + \mathbb{P}\left(\Delta N_{t_i}^a = 1\right) \times \mathbb{P}(\mathcal{A}_{z-1}),$$

which is true, since if there is one additional block added to the blockchain, this means the attacker has to catch up with one additional block, but also if one block is added by the attacker, there remains one less block to catch up with. Thus, we have

$$\mathbb{P}(\mathcal{A}_z) = \frac{\lambda}{\lambda + \lambda_{\text{a}}} \times \mathbb{P}(\mathcal{A}_{z+1}) + \frac{\lambda_{\text{a}}}{\lambda + \lambda_{\text{a}}} \times \mathbb{P}(\mathcal{A}_{z-1}).$$

Given the initial condition $\mathbb{P}(\mathcal{A}_0) = 1$, then the solution for the above recursive equation is given by

$$\mathbb{P}(\mathcal{A}_z) = 1 + C\left(\left(\frac{\lambda_{\text{a}}}{\lambda}\right)^z - 1\right), \qquad \text{with} \quad C \in [0, 1].$$

Another condition is that it is impossible that the attacker catches up with an infinite number of blocks in the blockchain. This means that the event $\mathcal{A}_{+\infty}$ is impossible: $\mathbb{P}(\mathcal{A}_{+\infty}) = 0$. Therefore, $C = 1$ and

$$\mathbb{P}(\mathcal{A}_z) = \left(\frac{\lambda_{\text{a}}}{\lambda}\right)^z, \qquad \forall z \in \mathbb{N}.$$

We indeed see that $\mathbb{P}(\mathcal{A}_z)$ does not depend on $i$. This ends the proof. ∎

Note that if $\lambda_a > \lambda$, the only acceptable value of $C$ is $C = 0$, thus $\mathbb{P}(\mathcal{A}_z) = 1$ for any $z$: if the attacker has more resource than the rest of the miners, then s/he will catch up with the blockchain almost surely.

In addition, we note that, as stated by Satoshi, but proven above, the higher the number $z$ of blocks the attacker has to catch up with, the less likely s/he will indeed catch up with the full blockchain, which makes the blockchain more and more secure with time, provided honest miners possess more than 50% of the total hash power (i.e., $\lambda > \lambda_a$).

Therefore, the probability $\mathcal{P}_z$ of an attack is given by

$$\mathcal{P}_z = \begin{cases} \left(\frac{\lambda_a}{\lambda}\right)^z & \text{if } z \geq 0 \\ 1 & \text{if } z < 0 \end{cases}$$

Indeed, when $z < 0$, this means that the attacker has more blocks than the blockchain and will thus proceed to an attack. Bearing in mind the notations of Section 11 in [7] (page 7), and the fact that a sequence of binomial random variables, under some assumptions, converges in law to a Poisson process, we have that $\left(\frac{\lambda_a}{\lambda}\right)^z \approx \left(\frac{q}{p}\right)^z$, thus the probability obtained in [7] (page 7) is just the convolution of a Poisson law with $\mathcal{P}_z$.

As a conclusion, we have linked the theory developed in Sect. 2.2, Chap. 2 with the main attack's result on the white paper.

## 2 On forecasting Bitcoin Price with Chainlets [8, 9]

Forecasting the price of the Bitcoin cryptocurrency, or, equivalently, its one-day logarithm return, is a task full of obstacles. The main one being the fact that they do not satisfy the necessary conditions for being described by the usual econometric models. The returns are far from being stationary, they are empirically strongly correlated, and signals are usually tough to discover.

Even though they are as well difficult to identify, bitcoins' "risk factors" (as in the traditional banking jargon) are not quite understood. For instance, in the environment of equity derivatives, the risk factors are what drives the derivative prices: these are mostly the equity prices (stock prices). However, for Bitcoin, what could be risk factors?

They may be nothing else but *blockchain data*: block size, hash rate, number of transactions per second/minute/day, inputs, outputs, or even the coefficient $D$ of a fitted model, see Sect. 3.6, Chap. 4.

Thus, a *risk factor* model remains possible. This is the object of [8, 9], with a particular risk factor: the *chainlet*. See Fig. 8.2 for an illustration.

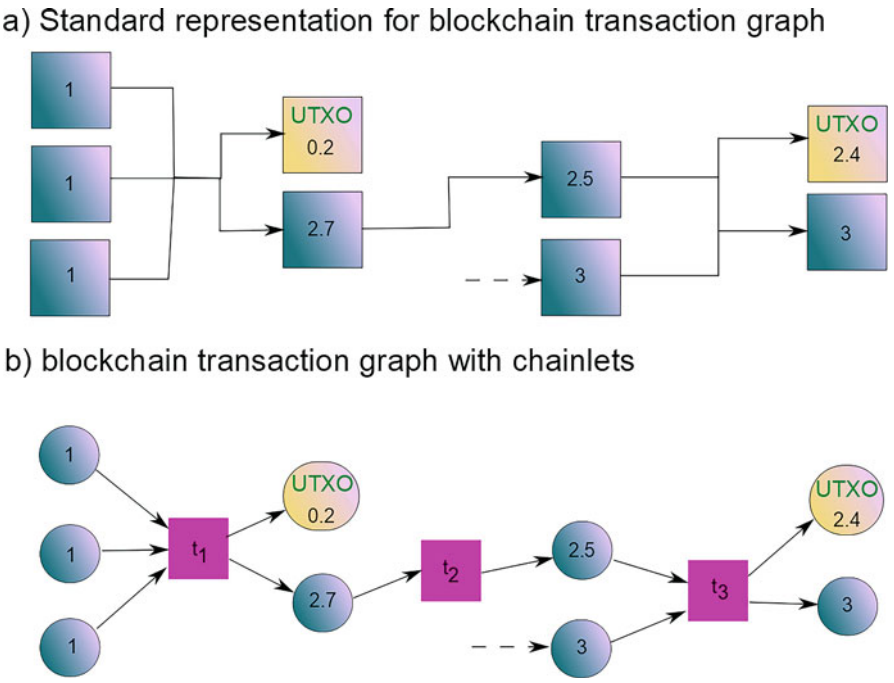


**Fig. 8.2** Two different ways to represent the one-directional graph of transactions on the blockchain: (**a**) usual representation – there are three transactions, and vertices are outputs (transaction fees not represented), also see Fig. II.4.1; (**b**) another way to represent the transactions, by means of a second type of vertex. This is a particular case of 3-chainlet (there are three transactions). (Source: Julien Riposo, 2023)

***Definition 8.2.1*** *A graph of blockchain transactions $G = (V, E)$ is a graph where $V$ is the set of outputs (see Definition II.4.1) and $E$ the set of pairs whose first element is an* input *(see Definition II.4.2) and a related* output *as a second element, thus forming a part of transaction (see Definition II.4.3).*

Besides, a transaction is defined as the set of pairs with inputs and outputs appearing several times. For instance, in Fig. 8.2.a, there are three represented transactions. On the left-hand side, there are three inputs $a$, $b$, $c$ and two outputs $a^{'}$, $b^{'}$. Thus, this transaction is defined by the family of pairs $\{(a, a^{'}), (a, b^{'}), (b, a^{'}), (b, b^{'}), (c, a^{'}), (c, b^{'})\}$. Thus, if there are $n$ inputs and $n^{'}$ outputs, a transaction will be a set of all $nn^{'}$ ordered pairs.

We can now define a $k$-chainlet.

***Definition 8.2.2*** *Let $G = (V, E)$ be the graph of blockchain transactions. A* $k$-chainlet *($k \in \mathbb{N}^*$) is a subgraph $G^{'} = (V^{'}, E^{'})$ of $G$ containing $k$ transactions (as defined in Definition 8.2.1).*

An alternative definition is as follows (this is presented differently from [8]).

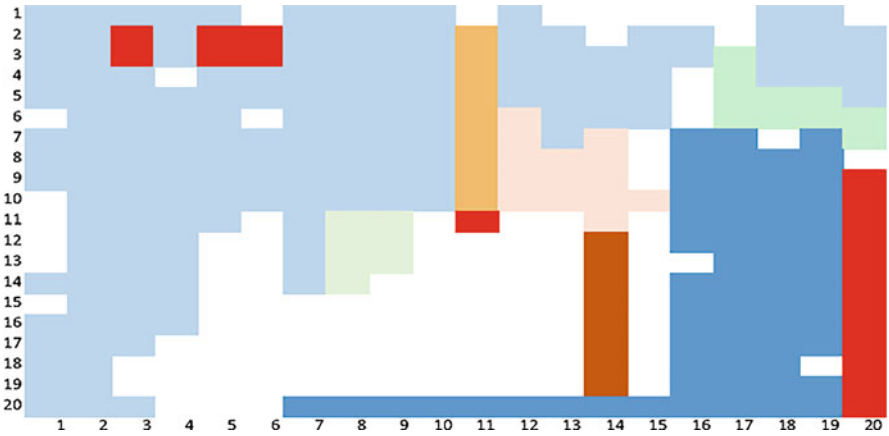


**Fig. 8.3** 1-Chainlet matrix $O$ (daily clusters from 2009 to 2018), with $n = 20$. (Source: [8])

***Definition 8.2.2 bis*** *A* $k$-chainlet *could be seen as a subgraph* $G = (V \cup T, E)$ *of the blockchain transaction graph, where* $T$ *are vertices of type "Transaction," and the elements of* $V$ *(the outputs) are connected with the ones of* $T$*, i.e., an element of* $E$ *is of the form* $(u, v) \in V \times T$ *(*$u$ *is an input) or* $T \times V$ *(*$v$ *is an output). When* $k = 1$*, we introduce the notation* $C_{i\to j}$ *for one transaction with* $i \in \mathbb{N}^*$ *inputs and* $j \in \mathbb{N}^*$ *outputs.*

In order to represent the information on the number of 1-chainlets, [8, 9] have introduced the matrix $O$ of 1-chainlets whose element $(i, j)$ is defined as (with $n > 0$)

$$O_{i,j} = \begin{cases} \mathrm{Card}(C_{i\to j}) & \text{if } i < n \text{ and } j < n \\ \sum_{l=n}^{+\infty} \mathrm{Card}(C_{i\to l}) & \text{if } i < n \text{ and } j = n \\ \sum_{l=n}^{+\infty} \mathrm{Card}(C_{l\to j}) & \text{if } i = n \text{ and } j < n \\ \sum_{l=n}^{+\infty} \sum_{l'=n}^{+\infty} \mathrm{Card}(C_{l\to l'}) & \text{if } i = n \text{ and } j = n \end{cases} 0$$

where $\mathrm{Card}\left(C_{i\to j}\right)$ is the number of 1-chainlets $C_{i\to j}$. By means of hierarchical clustering, a result is shown in Fig. 8.3 (which is Figure 4.a in [8]).

This representation has the advantage to perceive some clusters among the 1-chainlets of different kinds. In particular, an appropriate choice for $n$ needs to be done to reveal the clusters. Some further elaborations depicted in particular in [9] could allow to plot an *indicator* of transaction flows mined in the blockchain (see Figure 3 in [9]).

A $k$-chainlet thus defined is an interesting risk factor. In fact, [8, 9] have performed some econometric studies to reveal the dependency of the Bitcoin price and one-day log return with respect to them:

- [8] has defined, in this context, a Granger Causality (a time series $X$ Granger-causes $Y$ if the history of $X$ contains some information on the future of $Y$). Their table 1 reveals that there is causality significance of 1-chainlets for predicting the Bitcoin price tomorrow, given the history of all the features.
- [9] applied an ARMA-GARCHX model, including 1-chainlet activity, and backtests they have performed showed more significance than without including them (see their table 3).

## 3 On Valuing Bitcoin Price with Metcalfe's Law [10]

Valuing the price of Bitcoin has been a topic under investigation. Researchers and economists have been proposing some known laws, modeling price with specific attributes and according to a certain networking rule. For instance, Sarnoff values the price as a function of the number $n$ of users in the Bitcoin Network. Reed values it as a function of $2^n$, or Odlyzko as a function of $n \log n$ (see [10] and references cited therein). As explained in [10] (page 12), Bitcoin is best modeled as a network of peer-to-peer interactions, and its price should reflect those interactions.

Thus, forming a network of $n$ users, there are $\frac{n(n-1)}{2}$ total possible pairs in the network (basically, the dimension of a symmetric matrix, excluding the diagonal, which is, again, the number of all non-diagonal upper elements). Thus, Metcalfe's law assesses that the price $P_t$ to be modeled is given by

$$P_t = A\ \frac{n_t(n_t-1)}{2},$$

where $n_t$ is the number of users at time $t$, and $A \in \mathbb{R}_+^*$, but which is susceptible to change through time.

The author in [10] has performed a quick statistical analysis of the comparisons and observed strong deviations in 2013–2014, see Fig. 8.4 (which is Exhibit 5 in [10]).

The author cannot explain this deviation but has strong suspicions of price manipulation. Digging a little bit this possibility, other authors shared the same suspicions (see [11]) where it is explained that Bitcoin suffered from price manipulation in the period of Mt. Gox bankruptcy, due to individuals having several accounts and exchanging bitcoins illicitly.

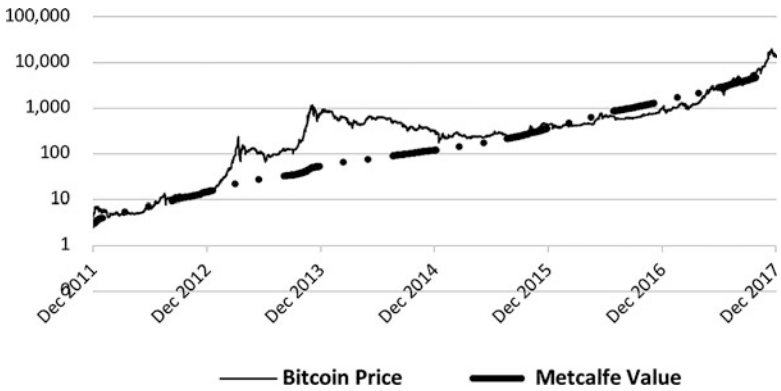


**Fig. 8.4** Log plot of the price and comparison with an optimal fit of the Metcalfe law. (Source: [10])

## 4 On Valuing Bitcoin Price from an Equilibrium Approach

This section has not been the object of a previous study to the best of the author's knowledge. The pricing of financial instruments is the essential means for circulation of goods and services within communities. When it comes to new financial instruments such as Bitcoin, a review of Financial Pricing Theory becomes crucial. Over the past 50 years, Probability Theory has vigorously developed pricing, especially Option Pricing. Calculating the *fair price* of an instrument (mainly based on a zero present value, i.e., gains = costs) and selling the related contract to a community sets the level of liquidity in a financial market. The fair price is thus not exclusively determined by the law of supply and demand, but by a model – considering supply and demand, to some extent. Such a model makes economical and financial assumptions, setting an agreed set of hypotheses, or mathematical assertions. From these hypotheses follow a path of deductions up to a final equation: the fair price with respect to appropriate parameters, but also time.

Pricing Theories mainly apply to derivatives. They are sophisticated financial products, and the underlying mathematics often requires advanced stochastic calculus. To complement Futures and Swap Pricing, the Black-Scholes formula of 1973 has made Option Pricing popular. There are many models derived from this very first equation. Thus, pricing a swaption (Swap Option) requires juggling between different probability spaces through changes of measure: the risk-neutral measure, the forward measure, the terminal measure, the spot LIBOR measure, etc. Each measure enables a specific calculation of the price. A common rule of thumb (which in fact is related to the Fundamental Theorem of Asset Pricing) for giving a price $P$ for the contracts is given by the following equation:

$$P = \mathbb{E}\Big(B(t,T)^{-1}\,\Psi(T)|\mathcal{F}_t\Big),$$

which is the conditional expectation today (at time $t$, given the "information" $\mathcal{F}_t$ we know) of the price $\Psi(T)$, in an appropriate measured space, of the instrument at maturity (time $T$), actualized to today by the factor $B(t, T)$ (related to an interest rate, which can be zero to simplify).

Pricing obviously relates not only to derivatives but also, in particular, to commodities. The price of commodities, however, is mainly governed by supply and demand, instead of by the underlying support where it is materially issued. This means that equation of the above type may not be relevant – perhaps with the exception of Bitcoin. Indeed, it is difficult to model a commodity price based on factors such as its essence, on the technology required to extract it, the impact of climate change on it, etc. However, if Bitcoin is considered as a commodity, it is not a "natural" commodity since it has been created by mankind [7]. Different criteria should apply.

Bitcoin does not currently have a dedicated pricing method or model, since one cannot rely on simple geometric Brownian motion – even if one chooses the "most" appropriate drift and volatility – since its returns (1D, 10D, 20D, etc.) do not relate to normal random variables. In fact, contrary to any "natural" commodity, we *know* the underlying (technological) environment of Bitcoin: its blockchain.

In the below, we discuss a short model for pricing Bitcoin based on the equilibrium pricing (gains = costs) being verified every 10 min, approximately.

The idea is that all the computational efforts need to be rewarded, and this reward should be evaluated at its *fair price*. In other words, we should be able to identify an equation of the form "Cost = Reward" (financially, this is zero present value), which is exactly an *equilibrium equation*, since the losses due to hashing must be compensated with the rewards obtain from mining a block. Thus, we can introduce the following parameters:

- $t$ – time, supposed to be in $]0, T]$, e.g., $T = 10$ min;
- $x$ – a parameter, which is a witness variable of block and miner dependency.

The following functions of interest, depending on $t$ and $x$, are also introduced:

- $r_b$ – block reward for block $b$;
- $c$ – (electrical) cost per hash (e.g., in USD);
- $P_t$ – price of one Bitcoin (e.g., in USD) at time $t$;
- $f_i$ – fee for transaction $i$ in block $b$;
- $N_b$ – number of hashes needed to hash the block $b$.

From the above, we can easily compute the revenues $R_b$ for block $b$. The total gain is the block reward, plus the sum of the fees collected by the winning miner, in the appropriate money unit:

$$R_b = \left( r_b + \sum_i f_i \right) P_T.$$

We can compute the costs, more precisely the expected costs $C_b$, for block $b$ as well. This is the cost per hash, times the number of hashes needed to be computed to get a correct key:

$$C_b = c\, N_b.$$

We obtain the following conservation equation: $R_b = C_b$, or

$$\left(r_b + \sum_i f_i\right) P_T = c\, N_b.$$

We possibly could obtain a pricing model for Bitcoin, which has the advantage of directly considering its underlying technology, by further elaborating a Pricing Equation-type, which can be written as

$$P_T = \frac{c\, N_b}{r_b + \sum_i f_i}.$$

Three variables can henceforth be modeled: the cost per hash $c$, the number of needed hashes $N_b$, and the fee $f_i$. A probabilistic and/or statistical approach may be relevant in this context. Suppose we defined an appropriate filtered probability space $\left(\Omega, \mathcal{T}, \mathbb{P}, (\mathcal{F}_t)_{t\in\mathbb{R}_+}\right)$.

The above leads to a model for the price each time a block is added. More generally, we could get the price at any time $t$ in $]0, T[$ as:

$$P_t = \mathbb{E}(P_T|\mathcal{F}_t) \Leftrightarrow P_t = \beta\, \frac{c\, N_{b,t}}{r_b + \sum_i f_{i,t}} + \epsilon_t,$$

where $\beta$ and $\epsilon_t$ are appropriately fitted and the ratio $\frac{c\, N_{b,t}}{r_b + \sum_i f_{i,t}}$ becomes time-dependent. Note how close this equation is to the pricing one, and we used a *linear* form of the conditional mathematical expectation (true for normal random variables).

Additionally, modeling the diffusion of information $I$ should allow connection with $N_{b,\,t}$, since the more the diffused information within the network, the more needed power for finding a correct key, and the higher $N_{b,\,t}$. This property has a strong impact on the Bitcoin price, and may be an additional factor to consider for pricing Bitcoin.

One possible model for $I$ is the one given by the usual creation–diffusion equation:

$$\frac{\partial I}{\partial t} = D\Delta I + \Gamma,$$

where $\Gamma$ is the rate of transactions introduced in the mining network and $D$ is the diffusion coefficient within the network. Note that the Laplacian $\Delta$ needs to be defined appropriately, see Sect. 3.6, Chap. 4.

As a conclusion, the set of possibilities to build a mathematical description of the blockchain seems gigantic. Further research is needed to elaborate on a pricing equation in order to get full construction and implementation of new models and allow us to get closer to the understanding of this fascinating environment.

For a foundation of the mathematics of diffusion on the peer-to-peer network, see [12].

## 5 Weights Contribution of a Crypto Portfolio through the Euler Allocation [14]

From the knowledge of the price of $n$ ($n \in \mathbb{N}^*$) cryptos, it is possible for an investor to analyze the contribution of the weights describing her investment capital allocation. This is performed using the Euler Allocation framework [14].

Let $O$ be a function from $\mathbb{R}^n$ to $\mathbb{R}$ which describes a portfolio characteristic. This function could be a total cost, or the portfolio variance, or again the tracking error with respect to a benchmark portfolio. It is a function of a *vector of weights* $w \in \mathbb{R}^n$. For any $i \in \{1, \ldots, n\}$, the number $w_i$ represent the percentage of capital allocation contribution to crypto $i$. It is common to assume that $0 \leq w_i \leq 1$ and $\sum_{i=1}^{n} w_i = 1$, but it is not compulsory. Thus, the previous information could be represented as constraints for an optimization process: the objective is to minimize (or maximize) the function $O$, *subject* to the above constraints. If $O$ is an objective function, the usual mathematical framework to perform such an exercise is by the means of the Lagrange multipliers, and it returns weight contribution satisfying the above constraints. The reader who is interested to discover the mathematical framework of the Lagrange multipliers can focus on [15].

The disadvantage of this method is that the function $O$ needs to verify certain properties to apply the method (e.g., being of class $C^2$, convex, or others), but also the constraints. Many constraints, as well as their tightness, could lead to an infeasible case, which is that *no solution* exists. In practice, it is easy to arrive to this possibility, making the investor stuck in her investment.

In the case of analyzing the weights contribution, we propose a brand-new method, which considers the function $O$ itself as a function of direct market parameters (as the returns, or sensitivities as the Greeks). This method finds its foundation by the means of the *Return Of Risk Adjusted Capital* (RORAC) [14]. Heuristically, the underlying allocation methodology is based on the *Euler Allocation*. For instance, if $O(x_1, x_2) = x_1 + 2x_2$, we have

$$x_1 \frac{\partial O}{\partial x_1}(x_1, x_2) = x_1; \quad x_2 \frac{\partial O}{\partial x_2}(x_1, x_2) = 2x_2.$$

Then the contribution from the variable $x_2$ is twice as much as the one from the variable $x_1$, on $(x_1, x_2) = (1, 1)$. This methodology has multiple applications to Finance, e.g., Funding Costs Minimization (in the context of Margin Valuation Adjustments, the allocation would be used to identify the highest contributing sensitivities, and therefore clearly indicate a possible reinvestment under constraints). For an application to the Standard Initial Margin Model (SIMM), see [14].

Applied to our case above, one assumption we need to make on the function $O$ is to be differentiable on a semi-cone $U \subseteq \mathbb{R}^n$, with $0_{\mathbb{R}^n} \in U$, where a set of $n \in \mathbb{N}^*$ parameters $u \in U$ (e.g., returns, sensitivities) can be defined. For the reader's information, a semi-cone of $\mathbb{R}^n$ is an open subset of $\mathbb{R}^n$ such that

$$\forall u \in U, \quad \forall \alpha \in \mathbb{R}_+^*, \quad \alpha u \in U.$$

It is equivalent to state that

$$\forall u \in U, \quad u\mathbb{R}_+ \subseteq U.$$

In fact, the function $O : U \to \mathbb{R}$ needs to be a *homogeneous function of degree l,* which means that $O$ is a differentiable on $U$, and

$$\exists l \in \mathbb{R}_+^* \quad \forall u \in U \quad \forall \alpha \in \mathbb{R}_+^* \quad O(\alpha u) = \alpha^l O(u).$$

It is easy to prove that $O$ is a homogeneous function of degree $l$ if and only if $O$ is differentiable and

$$\forall u \in U \quad O(u) = \frac{1}{l} \sum_{i=1}^{n} u_i \, \frac{\partial O}{\partial u_i}(u).$$

It is possible to define the *Euler Allocation Contributions* as

$$c_i = \frac{u_i \, \frac{\partial O}{\partial u_i}(u)}{\sum_{j=1}^{n} u_j \, \frac{\partial O}{\partial u_j}(u)}, \qquad \forall i \in \{1, \ldots, n\}, \quad \forall u \in U.$$

On the one hand, not only do we have that

$$\sum_{i=1}^{n} c_i = 1,$$

but also the sign of $c_i$ is mainly governed by the one of $u_i \, \frac{\partial O}{\partial u_i}(u)$. Thus, when $U$ describes intrinsic instrument characteristics (e.g., factors as scores), then $c$ may here indicate a specific portfolio construction. Supposing $O(u) > 0$ (we expect this is mostly the case in practice) for any $u \in U$, with $u$ the vector of returns, we have the following strategy:

- if $u_i > 0$ (positive return) and $\frac{\partial O}{\partial u_i}(u) > 0$ (positive contribution to $O$), then $c_i > 0$ and the position is long;
- if $u_i > 0$ (positive return) and $\frac{\partial O}{\partial u_i}(u) < 0$ (negative contribution to $O$), then $c_i < 0$ and the position is short;
- if $u_i < 0$ (negative return) and $\frac{\partial O}{\partial u_i}(u) > 0$ (positive contribution to $O$), then $c_i < 0$ and the position is short;
- if $u_i < 0$ (negative return) and $\frac{\partial O}{\partial u_i}(u) < 0$ (negative contribution to $O$), then $c_i > 0$ and the position is long.

Rigorously speaking, it is a necessity to verify that the chosen function $O$ is *RORAC-compatible*... For a mathematical foundation of the RORAC methodology coupled with the Euler Allocation, see [14].

Apart from the Euler Allocation, the methodology described in [14], proving the convergence for a sequence of iterative estimations of a linear factor model, could be a starting point for deriving a pure factor portfolio whose constituents are cryptos.

## 6 On Tracing Knowledge Publicly: The MathCoin [16] as an Example

The author in [16] proposes to construct a blockchain whose content is mathematically based. This idea may deserve attention if we would like to expand to more global knowledge, perhaps by creating another coin (a proof-of-stake shall be considered, rather than a proof-of-work).

If we extrapolate this vision, the blockchain could be the mean of a 2.0 human encyclopedia, where all the different knowledge, with their inventor(s), could be stored and immutably be traced throughout the evolution of humanity. In fact, with such a developed technology, humanity would ineluctably trace the origins of each of its creation. Thus, there would not remain any obscure area in the history of knowledge, of humanity. This tool could be an opportunity for studying the evolution of thinking in full transparency, having access to the subtle and deep details of the history of a thought, of a developed technology, or of basically *anything* that is intrinsically related to mankind. In this context, the incentive is clear: culture departments of states should be interested in financing such potential change for humanity for a research and development division elaborates on this, as the object of this fascinating project may be way beyond what Bitcoin or Ethereum propose...

# Conclusion

In this book, we have provided a framework for a general viewpoint of the mathematics of blockchain. This viewpoint should further assist the mathematician, more generally the curious reader, who is not used to the precise content of a blockchain. More specifically, the reader is invited to bear in mind:

- The content and the connection of the blocks in the blockchain,
- The proof of control of bitcoins through digital signature, and
- The overall interactions between all the agents involved in the construction of the blockchain.

In addition, we have seen that the developed theory in this book can be extended to the most influencing papers in this domain. Besides, for each of these points, we have seen that the engaged mathematics were (i) algebra, for the description of the blocks' content as well as their interactions, but also for digital signature and its classical cryptography means, (ii) probability theory for describing laws around blockchain, and (iii) further linear algebra through the implications of the users' network modeling.

This book is an introduction to a further generalization, i.e., it might be interesting to apply the same approach to other important cryptos, as Ethereum. Generally speaking, the world of Crypto is going to extend. Thus, we do hope that the content of this book is going to inspire the research community in order to develop the environment and infrastructure of tomorrow.



J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3

# References


1. Pierre-Edouard Thiery, Copernic. *How to represent a Blockchain through a mathematical model?* April 2020.
2. Alexander Lipton, and Adrien Treccani, World Scientific. *Blockchain and Distributed Ledger*. 2021.
3. Joseph H. Silverman, Springer. *The Arithmetic of Elliptic Curves*. 2016.
4. Kazuyuki Fujii, and Hiroshi Oike, Advances in Pure Mathematics. *An Algebraic Proof of the Associative Law of Elliptic Curves*. December 2017.
5. Philippe Blanchard, and Dimitri Volchenkov, Springer. *Random Walks and Diffusion on Graphs and Databases*. 2011.
6. Dimitrios Noutsos, Linear Algebra and its Applications. *On Perron-Frobenius property of matrices having some negative entries*, 412 132–153, 2006.
7. Satoshi Nakamoto, White Paper. *Bitcoin: A Peer-to-Peer Electronic Cash System.* 2008.
8. Cuneyt G. Akcora, Asim Kumer Dey, Yulia R. Gel, and Murat Kantarcioglu. *Forecasting Bitcoin Price with graph Chainlets*. PaKDD. 2018
9. Cuneyt G. Akcora, Matthew F. Dixon, Yulia R. Gel, and Murat Kantarcioglu. *Bitcoin Risk Modeling with Blockchain Graphs*. ArXiV. 2018.
10. Timothy F. Peterson. *Metcalfe's Law as a Model for Bitcoin's Value*. CAIA. 2018.
11. Neil Gandal, J. T. Hamrick, Tyler Moor, and Tali Oberman. *Price Manipulation in the Bitcoin Ecosystem*. Journal of Monetary Economics. 2017.
12. Julien Riposo. *Diffusion on the Peer-to-Peer Network*. Journal of Risk and Financial Management, 15(2), 47. 2022. Doi: https://doi.org/10.3390/jrfm15020047.
13. Matthew O. Jackson. *Social and Economic Networks*. Princeton University Press. 2010.
14. Julien Riposo, Gabriel Medina. *SIMM Risk Allocation*. Wilmott Magazine. 2022.
15. Wilhelm Forst, and Dieter Hoffmann, Springer. *Optimization – Theory and Practice*. 2010.
16. Borching Su. *MathCoin: A Blockchain Proposal that Helps Verify Mathematical Theorems in Public.* ArXiV. 2018.
17. Antoine Chambert-Loir. *Théorie de l'information*. Calvage & Mounet. 2022.
18. Jarosław Duda. *Extended Maximal Entropy Random Walk*. Ph.D. Thesis, Kraków 2012.
19. Wagner and Eckhoff. *Technical Privacy Metrics: A Systematic Survey.* ACM Computing Surveys, Vol 51, No 3, Article 57. 2018.
20. Juhasz, Stéger, Kondor, and Vattey. *A Bayesian approach to identify Bitcoin users.* PLoS One 13(12): e0207000. 2018.
21. https://jlopp.github.io/bitcoin-transaction-size-calculator/
22. Arnault et al. *Quantum simulation of quantum relativistic diffusion via quantum walks*. Journal of Physics A: Mathematical and Theoretical. 53 205303. 2020.
23. J. Zarrin, H.W. Phang, L.B. Saheer, and B. Zarrin. *Blockchain for decentralization of internet: prospects, trends, and challenges.* Springer 24, 2841–2866. 2021.

24. A. Firdaus, M.F.A. Razak, A. Feizollah, I.A.T. Hashem, M. Hazim, and N.B. Anuar. *The rise of "blockchain": bibliometric analysis of blockchain study.* Springer 120; 1289–1331. 2019.
25. Y.M. Know and C. Lustig. *Imaginaries and Crystallization Processes in Bitcoin Infrastructuring.* Spring 27, 209–232. 2017.
26. L. Serena, S. Ferretti, and G. D'Angelo. *Cryptocurrencies activity as a complex network: Analysis of transactions graphs.* Spring 15, 839–853. 2021.
27. D. Goldsmith, K. Grauer, and Y. Shmalo. *Analyzing hack subnetworks in the bitcoin transaction graph.* Springer, 5, article number: 22. 2020.
28. A.P. Motamed, and B. Bahrak. *Quantitative analysis of cryptocurrencies transaction graph.* Springer, 4, article number: 131. 2019.

# Index





J. Riposo, *Some Fundamentals of Mathematics of Blockchain*,
https://doi.org/10.1007/978-3-031-31323-3