BLOCKASTICS

Stochastic models for blockchain analysis

Pierre-O Goffard

updated on April 19, 2022

Chapter 1

Introduction

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

A blockchain is

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

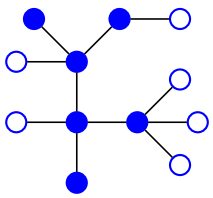


Figure 1.1: A network made of full nodes (blue) and light nodes (white)

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

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

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

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

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

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

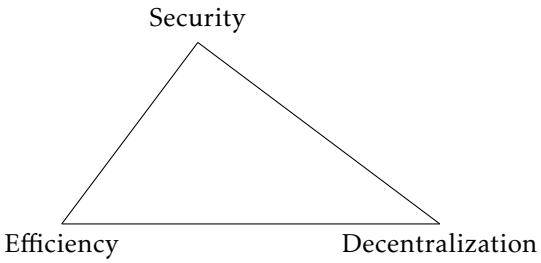


Figure 1.2: The blockchain trilemma

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

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

1. No central authority (Decentralized network)

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

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

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

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

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

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

We focus here on the use of mathematics to optimize blockchain systems which makes our problems very close to those encountered in operations research. These notes are organized as follows. Chapter 2 presents the various consensus algorithms. Chapter 3 focuses on the security aspects. In Chapter 3, we take a look at decentralization in Chapter 4. We close on efficiency with Chapter 5.

Chapter 2

Consensus protocol

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

- bandwidth
- computational power
- storage

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

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

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

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

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

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

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

2.1 Voting system

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

2.1.1 Two generals problem

Two generals wish to attack a city but they must agree on a timing to attack a city. They communicate via a messenger who must cross enemy territory at the risk of being intercepted.

The first general G_1 sends a message to the second one G_2 saying

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

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

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

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

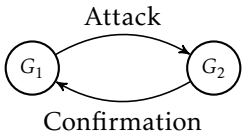


Figure 2.1: Message and confirmation loop

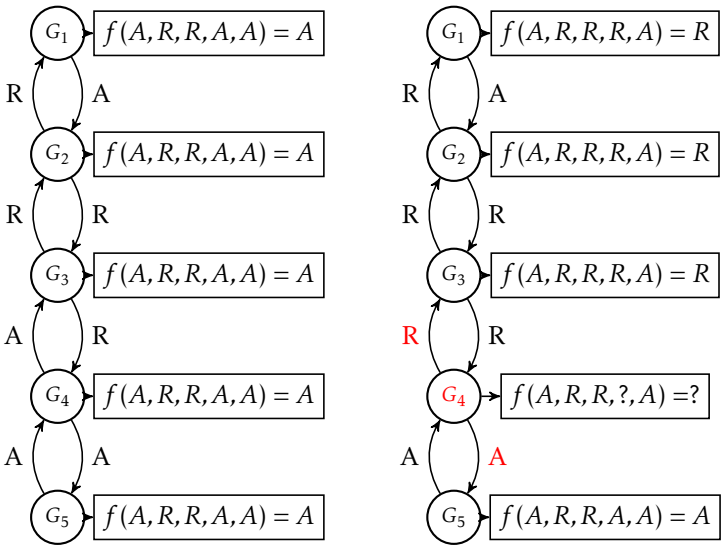
deemed unsolvable from a theoretical point of view and corresponds to a situation where two nodes communicate through an unreliable link. A practical solution for generals is to send many messengers hoping that at least one of them will succeed. This is only a thought experiment leading to the several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

2.1.2 Byzantine General problem

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

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

If there are no traitors, each general is communicating the same value to all the peers and consensus is reached as in Figure 2.2a. If one general is traitor, then he might not communicate the same value to all the generals and no consensus can be reached. It is the case for G_4 in Figure 2.2b. To handle such a situation, roles are given to the general. One of them become the leader and the other are the lieutenants. We aim at finding an algorithm such that



(a) No traitor (b) One traitor

Figure 2.2: Majority vote with or without a traitor

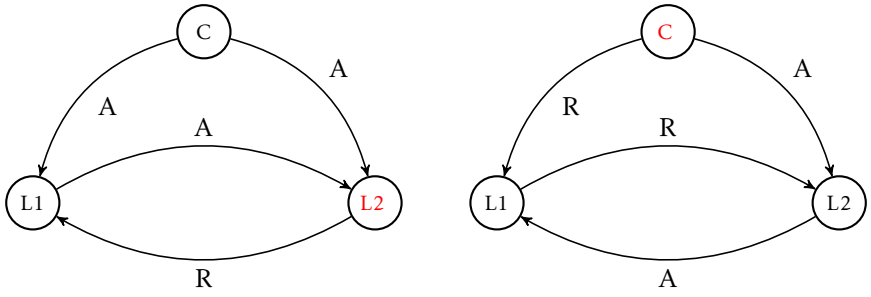
C1 All the loyal lieutenants obey the same order

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

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

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

Proof. Consider the situation where $n = 3$ and $m = 1$. The traitor is either the commander or one of the lieutenants as shown in Unfortunately for Lieutenant 2, there is no way for her to tell



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

Figure 2.3: Majority vote with or without a traitor

apart the situation pictured in Figure 2.3a and Figure 2.3b and therefore no way to ensure both C1 and C2. We prove the result for $n > 3$ by contradiction. Assume that there is a way to verify both C1 and C2 with $3 < n < 3m + 1$. We then construct a solution with generals by having one general simulate the commander plus at most $m - 1$ generals, and the other two simulating at

most m generals. One of the generals gather all the traitors and is therefore a traitor. The other two are loyal generals as they only simulate loyals general. We have built a solution with three generals that we know is impossible. □

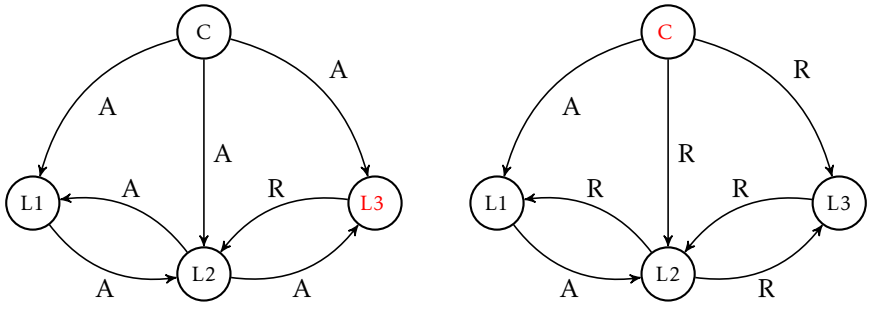
Now we need an algorithm that allows $n > 3m+1$ generals to deal with m traitors. The 'Oral Message' algorithm denoted by OM(m) and summarized in Algorithm 1 can handle m traitors if the number of generals verifies $n > 3m+1$. Before looking into the theoretical justification of

Algorithm 1 The Oral message algorithm OM(m)

```
1:  if m = 0 then;
2:      for i = 1 → n − 1 do
3:          Commander sends v_i = v to lieutenant i
4:          Lieutenant i set their value to v
5:      end for
6:  end if
7:  if m > 0 then;
8:      for i = 1 → n − 1 do
9:          Commander sends v_i to lieutenant i
10:         Lieutenant i uses OM(m-1) to communicate v_i to the n − 2 lieutenants
11:     end for
12:     for i = 1 → n − 1 do
13:         Lieutenant i set their value to f(v_1, ..., v_{n−1})
14:     end for
15: end if
```

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

Example 1. *Consider the situation where $n = 4$ and $m = 1$ shown in Figure 2.4. If the commander is*



(a) Commander is loyal

(b) Commander is a traitor

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

loyal then one of the lieutenant is a traitor, see Figure 2.4a. The commander gives the order to attack to all the lieutenant 3 *tells the other that she heard retreat from the commander. The loyal lieutenants*

then apply the map f to agree on their value

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

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

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

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

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

Proof. The proof follows from simple induction.

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

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

and both IC1 and IC follow.

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

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

which leads to the verification of IC1. □

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

2.2 Leader system

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

2.2.1 Proof-of-Work

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

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

The header usually includes

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

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

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

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

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

- quick to compute
- one way
- deterministic

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

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

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

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

98b1146926548f6b57c4347457713ff2f035beda9c93f12fbc9b202e9c512e80.

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

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

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

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

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

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

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

which means that with a difficulty of $D = 2^4$ it takes on average 16 trials. The protocol adjusts the difficulty automatically every 2,016 block discoveries so as to (globally) maintain one block discovery every 10 minutes on average. The time between two block discoveries depends on the number of hash values computed by the network at a given instant. At the time of writing, the network computes 182.58 Exahashes per second and the difficulty is 27,967,152,532,434.[1] For an exhaustive overview of the mining process in the bitcoin blockchain, we refer the reader to the book of Antonopoulos [2017, Chapter 10]. As each trial (of the system) for mining a block is independent of the others and leads to a success with very small probability, the overall number of successes is binomially distributed and will be very well approximated by a Poisson random variable. This justifies the Poisson process assumption made in the sequel to model the block arrival and the reward collecting processes. Empirical studies of the block inter-arrival times data tend to confirm this hypothesis, see the work of Bowden et al. Bowden et al. [2020]. The information recorded in a public blockchain may be retrieved by anyone and can be accessed through a blockchain explorer such as blockchain.com, the content of the block of height #724724 may be viewed through the following link block content.

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

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

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

[1] Source: bitcoinblockhalf.com

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

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

Proof-of-Capacity

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

Proof-of-Spacetime

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

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

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

2.2.3 Proof-of-Interaction

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

2.2.4 Proof-of-Stake

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

coin appends a new block and collect the reward.

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

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

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

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

where

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

Two potential issues needs to be studied

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

The "rich get richer" problem will be extensively studied in Chapter 4. Regarding the NaS problem, the nodes when chosen by a branch decide whether they want to add a block, it is an option. The cryptocoin value comes from its use as a medium of exchange. A long lasting disagreement results in a useless cryptocoin with no value.

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

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

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

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

  which corresponds to a minimum stake condition.

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

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

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

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

Chapter 3

Security of blockchain systems

The security evaluation of blockchain systems consists in calculating the probability of a successful attack on the blockchain. We will focus, in Section 3.1, on the double spending attack which is concern for PoW powererd cryptocurrency like teh bitcoin one. Security is also at risk when the node have an incentive to deviate from the prescribed protocol. Section 3.2 discusses the opportunity for miner of PoW equipped blockchain to resort to blockwithholding strategy to optimize their revenue.

3.1 Double-spending in PoW

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

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

The race between the two competing branches of the blockchain is summarized on Figure 3.1.

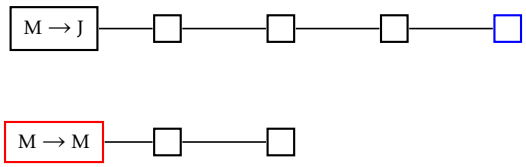


Figure 3.1: Double spending race illustrated, here we have $\alpha = 4$ and $z = 2$

3.1.1 Random walk model

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

$$R_0 = z, \text{ and } R_n = z + Y_1 + \ldots + Y_n.$$

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

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

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

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

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

$$\mathcal{F}_n = \sigma(Y_1, \ldots, Y_n), \; n \geq 1.$$

An illustration of this first-hitting time problem is provided in Figure 3.2. Let us denote by

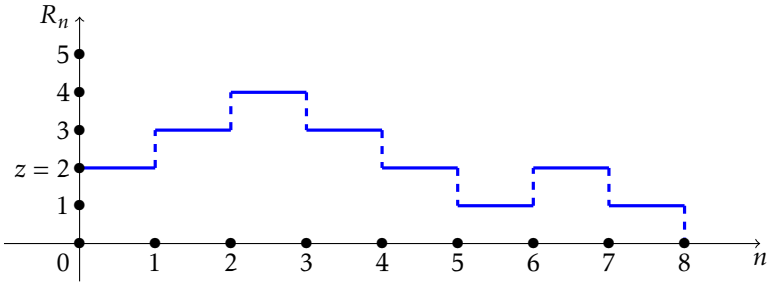


Figure 3.2: Illustration of the first-hitting time problem of a double spending attack.

$$\mathbb{P}_z(\cdot) = \mathbb{P}(\cdot | R_0 = z) \text{ and } \mathbb{E}_z(\cdot) = \mathbb{E}(\cdot | R_0 = z)$$

We are interested for now in the conditional distribution of τ_0 provided that $R_0 = z$.

Double spending probability

The double spending probability is defined as

$$\phi(z) = \mathbb{P}_z(\tau_0 < \infty),$$

and given in the following result

Theorem 3. *If $p > q$ then*

$$\phi(z) = \left(\frac{q}{p}\right)^z.$$

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

Proof 1:

Using a first step analysis, we have

$$\phi(z) = p\phi(z+1) + (1-p)\phi(z-1),\ z \geq 1. \tag{3.1}$$

We also have the boundary conditions

$$\phi(0) = 1 \text{ and } \lim_{z\to+\infty} \phi(z) = 0 \tag{3.2}$$

Equation (3.1) is a linear difference equation of order 2 associated to the following characteristic equation

$$px^2 - x + 1 - p = 0$$

which has two roots on the real line with

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

The solution of (3.1) is given by

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

where A and B are constant. Using the boudary conditions (3.2), we deduce that

$$\phi(z) = \left(\frac{1-p}{p}\right)^z$$

as announced.

For the second proof we need the notion of martingale

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

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

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

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

and the optional stopping theorem.

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

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

in each of the following situations

(i) T is bounded almost surely

(ii) There exists $c > 0$ such that $|X_{T\wedge n}| < c$ for every $n > 0$.

(iii) $\mathbb{E}(T) < \infty$, and, for some $K > 0$ we have that

$$|X_n(\omega) - X_{n-1}(\omega)| \leq K, \ \forall (n, \omega).$$

Proof 2:

Define the process

$$X_n = \exp[sR_n - n\kappa_Y(s)], \text{ for } n \in \mathbb{N} \text{ and } s \in \mathbb{R},$$

where

$$\kappa_Y(s) = \log\left[\mathbb{E}\left(e^{sY}\right)\right],$$

is the cumulant generating function of Y.

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

Proof. Denote by $M_Y(s) = \mathbb{E}(e^{sY})$ the moment generating function of Y, we have that

$$\begin{aligned}
\mathbb{E}(X_n|\mathcal{F}_n) &= \mathbb{E}\{\exp[sR_n - n\kappa_Y(s)]|\mathcal{F}_n\} \\
&= \exp[sR_{n-1} - n\kappa_Y(s)]\,\mathbb{E}[\exp(sY_n)|\mathcal{F}_n] \\
&= \exp[sR_{n-1}]M_Y(s)^{-n}M_Y(s) \\
&= X_{n-1}.
\end{aligned}$$

□

The equation $\kappa_Y(s) = 0$ is equivalent to

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

which admits $\gamma = \log(q/p)$ as only non-zero solution. The process $(\mathrm{e}^{\gamma R_n})_{n\geq 0}$ is a $\mathcal{F}_n$-Martingale. Define $\tau_a = \inf\{n \geq 0 \ ; \ R_n = a\}$, for $a > z$. Consider the stopping time $\tau = \tau_0 \wedge \tau_a$, we have that for any $n > 0$,

$$\mathbb{P}(\tau = \infty) \leq \mathbb{P}(\tau > n) < \mathbb{P}(|R_n| \leq a) \underset{n\to\infty}{\longrightarrow} 0;$$

We can therefore apply the optional stopping time theorem at τ to get

$$\begin{aligned}
\mathbb{E}(X_\tau) = \mathbb{E}(X_0) &\Leftrightarrow \mathbb{P}(\tau = \tau_0) + [1 - \mathbb{P}(\tau = \tau_0)]\mathrm{e}^{az} = \mathrm{e}^{\gamma z} \\
&\Leftrightarrow \mathbb{P}(\tau = \tau_0)] = \frac{\mathrm{e}^{\gamma z} - \mathrm{e}^{az}}{1 - \mathrm{e}^{az}}.
\end{aligned}$$

We then let $a \to \infty$ in the above equation to conclude that

$$\mathbb{P}(\tau = \tau_0) = \left(\frac{q}{p}\right)^z.$$

Exercise 1. *Just for fun*

- *What is $\phi(z)$ when $p \leq q$?*
- *Compute $\mathbb{P}(\tau_0 \leq \tau_a)$ if $p = q = 1/2$*

In practice the number of blocks z is actually random variable

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

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

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

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

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

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

Double spending time

In the block mining world time is money. Every hour spent in computing hashes is costly in terms of energy. It is then very interesting to know whether a double spending attack is meant to last long or not. Intuitively, we can think that if it must occur the it should at a earlier stage because as $p > 1/2$ our random walk $(R_n)_{n \geq 0}$ will eventually drift toward $+\infty$. The following result provides the probability distribution of τ_0 when $R_0 = z$.

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

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

and 0 otherwise.

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

Lemma 2.

$$\mathbb{P}_z(\tau_0 = n) = \frac{z}{n} \mathbb{P}_z(R_n = 0) \tag{3.3}$$

Proof. If $z = 0$ then $\tau_0 = 0$ almost surely and both sides of (3.3) equal to 0. Assume that $z \geq 1$, we have that $\mathbb{P}_z(\tau_0 = n) = 0$ and $\mathbb{P}_z(R_n = 0) = 0$ whenever $n < z$ and $n - z$ is odd. The rest of the proof is by induction on $n \geq 1$, when $n = 1$ we have that

$$\mathbb{P}_z(\tau_0 = 1) = 0 = \frac{z}{1}\mathbb{P}_z(R_1 = 0), \text{ for } z > 1,$$

and

$$\mathbb{P}_1(\tau_0 = 1) = q = \frac{1}{1}\mathbb{P}_1(R_1 = 0), \text{ for } z = 1.$$

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

$$\begin{aligned}
\mathbb{P}_z(\tau_0 = n+1) &= \sum_{y\in\{-1,1\}} \mathbb{P}_z(\tau_0 = n+1 | Y_1 = y)\mathbb{P}(Y_1 = y) \\
&= \sum_{y\in\{-1,1\}} \mathbb{P}_{z+y}(\tau_0 = n)\mathbb{P}(Y_1 = y) \\
&= \sum_{y\in\{-1,1\}} \frac{z+y}{n}\mathbb{P}_{z+y}(R_n = 0)\mathbb{P}(Y_1 = y)
\end{aligned}$$

The second equality holds thanks to the strong Markov property. We further undo the law of total probability.

$$\begin{aligned}
\mathbb{P}_z(\tau_0 = n+1) &= \sum_{y\in\{-1,1\}} \frac{z+y}{n}\mathbb{P}_z(Y_1 = y | R_{n+1} = 0)\mathbb{P}_z(R_{n+1} = 0) \\
&= \frac{\mathbb{P}_z(R_{n+1} = 0)}{n}\left[z + \mathbb{E}(Y_1 | R_{n+1} = 0)\right] \qquad (3.4)
\end{aligned}$$

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

$$\mathbb{E}(Y_1 | R_{n+1} = 0) = \mathbb{E}(Y_i | R_{n+1} = 0),\ i = 1, \ldots, n+1,$$

and it follows that

$$\mathbb{E}(Y_1 | R_{n+1} = 0) = \frac{1}{n+1}\sum_{i=1}^{n+1} \mathbb{E}(Y_i | R_{n+1} = 0) = \frac{-z}{n+1}.$$

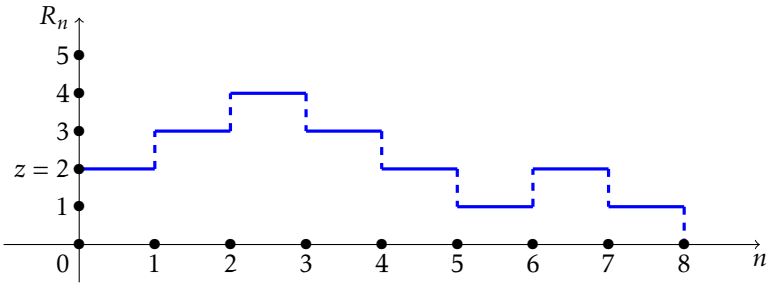
Inserting the above expression in (3.4) yields

$$\mathbb{P}_z(\tau_0 = n+1) = \frac{z}{n+1}\mathbb{P}_z(R_{n+1} = 0).$$

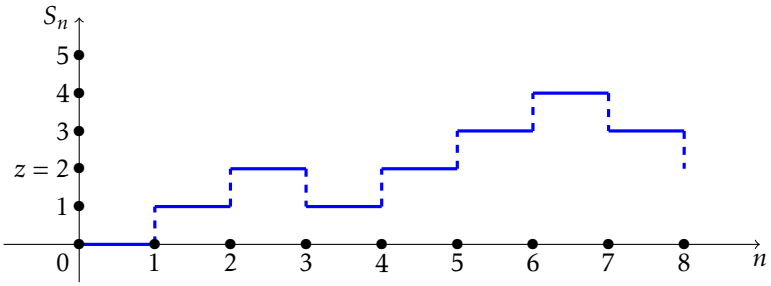
□

Remark 2. *This proof is direct, simple and inspired from van der Hofstad and Keane [2008]. It is possible to make it shorter taking advantage of the ballot theorem. Indeed consider again the first hitting problem on Figure 3.2 and reverse the timeline. It corresponds to that of a random walk $(S_n)_{n\geq 0}$ that starts at 0, make upward jumps with probability q, and aims at reaching the level z without crossing the X axis, see Figure 3.3. Note that Figure 3.3b is the reflection of Figure 3.3a with respect to the Y axis. We have equivalently*

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



(a) Original first hitting problem



(b) Time reversed first hitting problem

Figure 3.3: Another look at the first hitting time problem.

and

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

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

To complete the proof, we just note that

$$\mathbb{P}_z(R_n = 0) = \binom{n}{(n-z)/2} p^{(n-z)/2} q^{(n+z)/2}$$

as it corresponds to a trajectory of $(R_n)_{n\geq 0}$ starting at $R_0 = z$ ending at 0 made of $(n-z)/2$ upward jumps and $(n+z)/2$ downward one. □

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

3.1.2 Counting process model

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

Poisson process, Exponential distributions and friends

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

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

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

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

A trajectory of a counting process is given in

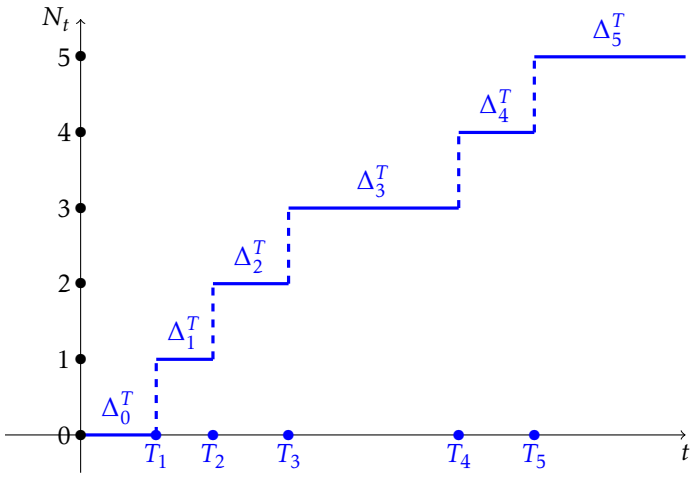


Figure 3.4: Trajectory of the counting process $(N_t)_{t\geq 0}$.

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

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

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

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

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

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

$$U_{(1)} \leq U_{(2)} \leq \ldots \leq U_{(n)}$$

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

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

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

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

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

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

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

It then follows that

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

□

We require some additional result about the gamma distribution.

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

1. *The* T_k*'s have an Erlang distribution. We denote by* $T_k\sim Erl(k,\lambda)$ *with*

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

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

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

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

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

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

Exercise 2. *Can you propose another way to show this result? Without using induction.*

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

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

Let us apply the following change of variable

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

minding the change in the integration domain as

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

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

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

3. We have that

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

□

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

Proposition 3. *The following statements are equivalent*

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

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

The stochastic processes with independent and stationnary increments are called Levy processes.

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

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

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

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

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

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

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

$\underline{2 \Rightarrow 1}$

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

$$f_{T_1, \ldots, T_n}(t_1, \ldots, t_n) = \lambda^n e^{-\lambda t_n} \mathbb{I}_{0 < t_1 < \ldots < t_n}. \tag{3.5}$$

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

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

We have

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

Divide by h^n and let h go to 0 to get (3.5). After applying a change of variable (reciprocal of that used in the proof of Proposition 2) to recover the joint distribution of $(\Delta_1^T, \ldots, \Delta_n^T)$, we see that the later is actually that of an i.i.d. sample of size n of exponential random variables. □

Last but not the least, we establish the order statistic property of the Poisson process.

Proposition 4. *Provided that* $\{N_t = n\}$, *The jump times* $T_1, \ldots, T_n$ *have the same distribution as the order statistic of an i.i.d. sample of n uniform random variable on* $(0, t)$, *namely it holds that*

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

Proof. We have

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

□

The order statistic property of the Poisson process will be useful to solve the first hitting time problem arising later on. To fully benefit from this nice property, we need to introduce Appell and Abel-Gontcharov polynomials.

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

Definition 7. *Starting with $A_0(x|U)=1$, the $A_n(x|U)$'s satisfy the differential equations*

$$A_n^{(1)}(x|U)=nA_{n-1}(x|U), \tag{3.6}$$

with the border conditions

$$A_n(u_n|U)=0,\quad n\ge 1. \tag{3.7}$$

So, each A_n has the integral representation

$$A_n(x|U)=n!\int_{u_n}^{x}\left[\int_{u_{n-1}}^{y_n} dy_{n-1}\dots\int_{u_1}^{y_1} dy_1\right]dy_n,\quad n\ge 1. \tag{3.8}$$

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

Definition 8. *Starting with $G_0(x|U)=1$, the $G_n(x|U)$'s satisfy the differential equations*

$$G_n^{(1)}(x|U)=nG_{n-1}(x|\mathcal{E}U), \tag{3.9}$$

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

$$G_n(u_1|U)=0,\quad n\ge 1. \tag{3.10}$$

So, each G_n has the integral representation

$$G_n(x|U)=n!\int_{u_1}^{x}\left[\int_{u_2}^{y_1} dy_2\dots\int_{u_n}^{y_{n-1}} dy_n\right]dy_1,\quad n\ge 1. \tag{3.11}$$

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

$$G_n(x|u_1,\dots,u_n) = A_n(x|u_n,\dots,u_1), \quad n \geq 1. \tag{3.12}$$

The two families (i.e. for all $n \geq 0$), however, are distinct and enjoy different properties. From (3.8) and (3.11), it is clear that the polynomials A_n and G_n can be interpreted in terms of the joint distribution of the vector $(U_{1:n},\dots,U_{n:n})$.

Proposition 5. *For* $0 \leq u_1 \leq \dots \leq u_n \leq x \leq 1$,

$$P[U_{(1)} \geq u_1,\dots,U_{(n)} \geq u_n \text{ and } U_{(n)} \leq x] = A_n(x|u_1,\dots,u_n), \quad n \geq 1. \tag{3.13}$$

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

$$P[U_{(1)} \leq u_1,\dots,U_{(n)} \leq u_n \text{ and } U_{(1)} \geq x] = (-1)^n G_n(x|u_1,\dots,u_n), \quad n \geq 1. \tag{3.14}$$

The representations (3.13) and (3.14) will play a key role for solving first-hitting problem that involve Poisson processes. Numerically, it will be necessary to evaluate some special values of the polynomials. To this end, it is convenient to use the following recusive relations.

Proposition 6. *The Appell polynomials are computed through the expansion*

$$A_n(x|U) = \sum_{k=0}^{n} \binom{n}{k} A_{n-k}(0|U)x^k, \quad n \geq 1, \tag{3.15}$$

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

$$A_n(0|U) = -\sum_{k=1}^{n} \binom{n}{k} A_{n-k}(0|U)u_n^k, \quad n \geq 1. \tag{3.16}$$

The A-G polynomials are computed through the recursion

$$G_n(x|U) = x^n - \sum_{k=0}^{n-1} \binom{n}{k} u_{k+1}^{n-k} G_k(x|U), \quad n \geq 1. \tag{3.17}$$

Proof. The Maclaurin expansion of $A_n(x|U)$ gives (3.15) as

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

Evaluation at $x = u_n$ then provides (3.16). Regarding (3.17), first note that

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

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

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

By expanding x^n one gets (3.17). □

Hereafter are a couple of useful properties

Proposition 7. *1. For any $a, b \in \mathbb{R}$, it holds that*

$$A_n(x|a+bU) = b^n A_n((x-a)/b\,|U), \quad n \geq 1, \tag{3.18}$$

with the same identity for G_n.

2. We have

$$A_n(x|1,\ldots,n) = x^{n-1}(x-n), \tag{3.19}$$

$$G_n(x|0,\ldots,n-1) = x(x-n)^{n-1}. \tag{3.20}$$

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

$$\mathbb{E}\,[A_n(x|S_1,\ldots,S_n)|S_n] = x^{n-1}(x-S_n), \tag{3.21}$$

$$\mathbb{E}\,[G_n(x|S_0,\ldots,S_{n-1})|S_n] = x(x-S_n)^{n-1}. \tag{3.22}$$

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

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

and

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

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

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

so the property holds for $n+1$. No need to do the job for the $G_n(.|U)$'s thanks to (3.12).

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

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

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

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

For identity (3.20), write

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

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

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

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

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

For identity (3.22), write

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

□

A tiny bit of insurance risk theory

In what follows we will use classical result from insurance risk theory and so we have to provide some context. The financial reserve of a nonlife insurance company can be modelled by a continuous time stochastic process $(R_t)_{t\geq 0}$ defined as

$$R_t = u + P_t - L_t,\ t \geq 0 \tag{3.23}$$

where

- $R_0 = u > 0$ are the initial reserves,
- $(P_t)_{t\geq 0}$ is the premium income process,
- $(L_t)_{t\geq 0}$ is the liability of the insurance company, that is the sum of all the compensations paid to to the policyholders.

The classical assumes that the premium are collected linearly in times at some rate c per time unit. Namely, we have

$$P_t = c \times t.$$

The number of claims is governed by a counting process $(N_t)_{t\geq 0}$, usually a standard Poisson process with intensity λ. To each claim is associated a randomly sized compensation distributed as U. The claim sizes $U_1, U_2, \dots$ form a sequence of i.i.d. random variables independent from N_t. We therefore have

$$L_t = \sum_{k=1}^{N_t} U_k.$$

The name of the game is to compute the ruin probabilities

$$\psi(u,T) = \mathbb{P}(\tau_0 \leq T), \text{ and } \psi(u) = \mathbb{P}(\tau_0 < \infty), \tag{3.24}$$

where $\tau_0 = \inf\{t \geq 0\ ;\ R_t \leq 0\}$ is the ruin time. This first hitting time problem is illustrated on Section 3.1.2 This enables to select an initial reserves level u so that the ruin probability is low enough. The premium rate is usually set to compensate the losses due to claims with

$$\mathbb{E}(P_t) = (1+\eta)\mathbb{E}(L_t) \text{ or equivalently } c = (1+\eta)\lambda\mathbb{E}(U).$$

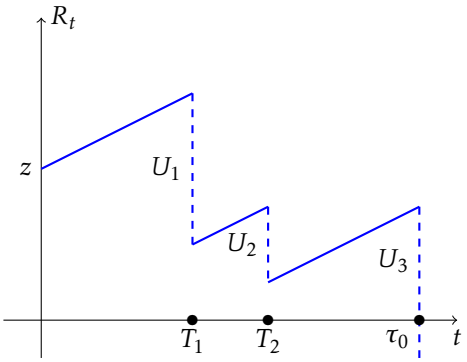


Figure 3.5: Illustration of the classical insurance ruin problem

Define the claim surplus process as

$$S_t = u - R_t = L_t - P_t,\ t \geq 0.$$

The following result is the counterpart of Lemma 1 for Lévy processes.

Proposition 8. *If $(S_t)_{t\geq 0}$ is a Lévy process then*

$$M_t = \exp\left[sS_t - t\kappa(s)\right],\ t \geq 0,$$

where $\kappa(s) = \log \mathbb{E}\left(e^{sS_1}\right)$, is a Martingale.

The infinite time horizon ruin probability may be evaluated using the following result.

Proposition 9. *If $S_t \overset{a.s.}{\to} -\infty$, and there exists $\gamma > 0$ such that $\left(e^{\gamma S_t}\right)_{t\geq 0}$ is a martingale then*

$$\mathbb{P}(\tau_0 < \infty) = \frac{e^{-\gamma u}}{\mathbb{E}\left[e^{\gamma \xi(u)} | \tau_0 < \infty\right]},$$

where $\xi(u) = S_{\tau_0} - u$ denotes the deficit at ruin.

Proof. We apply the optionnal stopping theorem on $(e^{\gamma S_t})_{t\geq 0}$ at time $T \wedge \tau_0$ for some $T > 0$ to get

$$\begin{aligned} 1 = \mathbb{E}\left(\mathrm{e}^{\gamma S_0}\right) &= \mathbb{E}\left(\mathrm{e}^{\gamma S_{T\wedge\tau_0}}\right) \\ &= \mathbb{E}\left(\mathrm{e}^{\gamma S_{\tau_0}} \mathbb{I}_{\tau_0 \leq T} + \mathrm{e}^{\gamma S_T} \mathbb{I}_{\tau_0 > T}\right). \end{aligned}$$

Letting $T \to \infty$ yields[1]

$$\begin{aligned} 1 &= \mathbb{E}\left(\mathrm{e}^{\gamma S_{\tau_0}} \mathbb{I}_{\tau_0 < \infty}\right) \\ &= \mathrm{e}^{\gamma u} \mathbb{E}\left(\mathrm{e}^{\gamma \xi(u))} | \tau_0 < \infty\right) \psi(u). \end{aligned}$$

□

We will use Proposition 9 to derive the double spending probability within a continuous time framework in the next sections.

[1] Thanks to $S_t \to -\infty$, monotone and dominated convergence.

Double spending probability

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

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

$$\tau_z = \inf\{t \geq 0 \, ; \, z + N_t = M_t\}.$$

Assume that $(N_t)_{t\geq 0}$ and $(M_t)_{t\geq 0}$ are Poisson processes with intensity λ and μ such that $\lambda > \mu$. Let $(T_i)_{i\geq 0}$ and $(S_i)_{i\geq 0}$ be the arrival times of $(N_t)_{t\geq 0}$ and $(M_t)_{t\geq 0}$ The first-hitting problem along with its notation is illustrated in Figure 3.6. The double spending probability is given by the

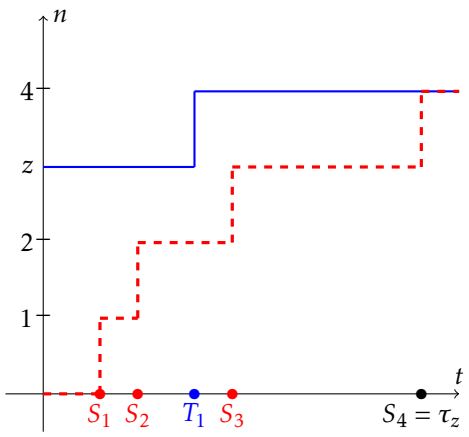


Figure 3.6: Illustration of the double spending problem within a continuous time framework.

following result

Theorem 6. *The double spending probability is given by*

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

Proof. We make an analogy with Section 3.1.2, where z is the inital reserves, N_t is the premium income and M_t is the liability of an insurance company. Define the claim surplus process as

$$S_t = M_t - N_t, \; t \geq 0.$$

Note that $(S_t)_{t\geq 0}$ is a Lévy process such that $S_t \to +\infty$ beacuse $\lambda > \mu$. The equation

$$\kappa(s) = \log \mathbb{E}\left(e^{sS_1}\right) = 0$$

is equivalent to

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

which has a unique non negative solution given by

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

It follows from Proposition 8 that $(e^{\gamma S_t})_{t\geq 0}$ is a martingale, and applying Proposition 9 yields the double spending probability

$$\mathbb{P}(\tau_z < \infty) = \left(\frac{\mu}{\lambda}\right)^z,$$

since $\xi(z) = 0$ as ruin occurs exactly. □

Double spending time

Just like in Section 3.1.1, we are interested in the time required to complete a double spending attack. Accounting for the cost of electricity, we can approximate the operational cost per time unit by

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

where

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

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

Theorem 7. *If $(N_t)_{t\geq 0}$ is a Poisson process and $(M_t)_{t\geq 0}$ is a renewal process then the **p.d.f.** of τ_z is given by*

$$f_{\tau_z}(t) = \mathbb{E}\left[\frac{z}{z+N_t} f_{S_{N_t+z}}(t)\right], \text{ for } t \geq 0. \tag{3.25}$$

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

$$\{\tau_z \in (t, t+\mathrm{d}t)\} = \bigcup_{n=0}^{+\infty} \{\tau_z \in (t, t+\mathrm{d}t)\} \cap \{N_t = n\}. \tag{3.26}$$

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

$$\{\tau_z \in (t, t+\mathrm{d}t)\} \cap \{N_t = 0\} = \{S_z \in (t, t+\mathrm{d}t)\} \cap \{N_t = 0\}, \tag{3.27}$$

and consequently

$$f_{\tau_z|N_t}(t|0) = f_{S_z}(t),\ t \geq 0, \tag{3.28}$$

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

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

Applying the law of total probability yields

$$\begin{aligned} & \mathbb{P}\left(\{\tau_z \in (t, t+\mathrm{d}t)\} \cap \{N(t) \geq 1\}\right) \\ = & \sum_{n=1}^{+\infty} \mathbb{P}\left[\bigcap_{k=1}^{n} \{T_k \leq S_{z+k-1}\} \cap \{S_{z+n} \in (t, t+\mathrm{d}t)\} \middle| N(t) = n\right] \mathbb{P}[N(t) = n]. \end{aligned} \tag{3.29}$$

In virtue of the order statistic property, the successive jump times $(T_1, \ldots, T_n)$ are distributed as the order statistics $(V_{1:n}, \ldots, V_{n:n})$ of a sample of n i.i.d. random variables uniformly distributed on $(0, t)$. The conditional probability in (3.29) may be rewritten as

$$\begin{aligned} & \mathbb{P}\left[\bigcap_{k=1}^{n} \{V_{k:n} \leq S_{z+k-1}\} \cap \{S_{z+n} \in (t, t+\mathrm{d}t)\}\right] \\ = & \mathbb{P}\left[\bigcap_{k=1}^{n} \{U_{k:n} \leq S_{z+k-1}/t\} \cap \{S_{z+n} \in (t, t+\mathrm{d}t)\}\right] \\ = & \mathbb{P}\left[\bigcap_{k=1}^{n} \{U_{k:n} \leq S_{z+k-1}/t\} \middle| S_{z+n} \in (t, t+\mathrm{d}t)\right] \mathbb{P}[S_{z+n} \in (t, t+\mathrm{d}t)] \\ = & \mathbb{E}\left\{(-1)^n G_n[0 \middle| S_z/t, \ldots, S_{z+n-1}/t] \middle| S_{z+n} \in (t, t+\mathrm{d}t)\right\} \mathbb{P}[S_{z+n} \in (t, t+\mathrm{d}t)], \\ = & (-1/t)^n \mathbb{E}\left\{G_n[0 \middle| S_z, \ldots, S_{z+n-1}] \middle| S_{z+n} \in (t, t+\mathrm{d}t)\right\} \mathbb{P}[S_{z+n} \in (t, t+\mathrm{d}t)], \end{aligned} \tag{3.30}$$

where $U_{1:n}, \ldots, U_{n:n}$ denote the order statistics of a sample of n i.i.d. uniform random variables on $(0, 1)$, and $G_n(.|.)$ correspond to the sequence of A-G polynomials as defined in Section 3.1.2. The last equation in (3.30) follows from using the first identity of Proposition 7. Inserting (3.30) into (3.29) and letting $\mathrm{d}t$ be small enough yields

$$\begin{aligned} f_{\tau_z|N(t)\geq 1}(t) &= \sum_{n=1}^{+\infty} (-1/t)^n \mathbb{E}\left\{G_n[0 \middle| S_z, \ldots, S_{z+n-1}] \middle| S_{z+n} = t\right\} \\ &\times f_{S_{z+n}}(t) \mathbb{P}[N(t) = n]. \end{aligned} \tag{3.31}$$

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

$$\begin{aligned} \mathbb{E}\left\{G_n(0 \middle| S_z, \ldots, S_{z+n-1}) | S_{z+n} = t\right\} &= \mathbb{E}\left\{G_n(-S_z | 0, \ldots, S_{z+n-1} - S_z) | S_{z+n} = t\right\} \\ &= \mathbb{E}\left\{\mathbb{E}\left[G_n(-S_z | 0, \ldots, S_{z+n-1} - S_z) | S_{z+n} - S_z, S_{z+n}\right] | S_{z+n} = t\right\} \\ &= \mathbb{E}\left\{(-S_z)(-S_z - S_{n+z} + S_z)^{n-1} | S_{z+n} = t\right\} \\ &= (-1)^n \mathbb{E}\left\{S_z S_{n+z}^{n-1} | S_{z+n} = t\right\} \\ &= (-1)^n t^{n-1} \frac{z}{z+n} t = (-t)^n \frac{z}{z+n} \end{aligned} \tag{3.32}$$

Inserting (3.32) into (3.31) yields

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

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

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

which is equivalent to the announced result (3.25). □

Remark 4. *Just like in the random walk framework of Section 3.1.1, the number z is actually a random variable defined as*

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

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

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

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

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

3.2 Blockwitholding in PoW

The constant operational cost and the infrequent capital gains make mining blocks on a PoW blockchain a risky activity. One way to tackle this problem is to engage in blockwitholding strategy. The goal is to build up a stock of blocks and release them publicly at well chosen times so as to fork the chain and make a part of the mining activities of competitors worthless, depending on which branch is followed up in the longer run. We focus here on a simplified version[2] of the original blockwitholding strategy called selfish mining and described in Eyal and Sirer [2014].

We start by introducing a risk model to follow the financial reserves of a miner over time. The ruin probability and expected profit given that ruin did not occur plays teh role of risk and performace indicators which will allow us to compare the solo mining strategy to the selfish mining strategy.

[2] For the sake of tractability.

3.2.1 Ruin and expected profit of a miner

A miner, referred to as Sam, starts operating with an initial surplus level $u > 0$. We consider the following stochastic model for the surplus process over time. Mining blocks generates steady operational costs of amount $c > 0$ per unit of time, most notably due to electricity consumption. The entire network of miners appends blocks to the blockchain at an exponential rate λ, which means that the length of the blockchain over time is governed by a Poisson process $(N_t)_{t\geq 0}$ with intensity λ. We assume that Sam owns a fraction $q \in (0, 1/2)$ of the overall computing power, which implies that each block is published by Sam with a probability q. The number of blocks found by Sam and therefore the number of rewards of size $b > 0$ he collects up to time $t \geq 0$ is a thinned Poisson process $(\tilde{N}_t)_{t\geq 0}$ with intensity λ and thinning parameter q. Sam's surplus R_t at time t is then given by

$$R_t = u + b \cdot \tilde{N}_t - c \cdot t, \qquad t \geq 0. \tag{3.33}$$

The stochastic process $(R_t)_{t\geq 0}$ is referred to as the dual risk process in risk theory, see for instance Avanzi et al. [2007]. Our goal is to measure the profitability of mining blocks on the blockchain, but subject to a ruin constraint. In this context, the time of ruin $\tau_u = \inf\{t \geq 0 : \ R_t = 0\}$ is defined as the first time when the surplus reaches zero, i.e. the miner runs out of money and cannot continue to operate. The riskiness of the mining business may be assessed via the finite-time

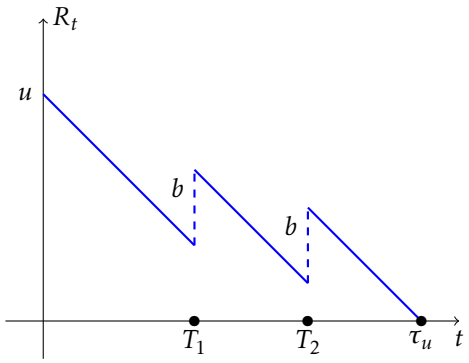


Figure 3.7: Ruin time in the miner risk model

and infinite-time horizon ruin probabilities defined as

$$\psi(u, t) = \mathbb{P}(\tau_u \leq t), \text{ and } \psi(u) = \mathbb{P}(\tau_u < \infty), \tag{3.34}$$

respectively. The rewards earned through mining must in expectation exceed the operational cost per time unit. The latter condition translates into $q\lambda b > c$, and is referred to as the net profit condition in standard risk theory, see Asmussen and Albrecher [2010]. In particular, this implies $\psi(u) < 1$, i.e. ruin does not occur almost surely. The profitability for a time horizon $t > 0$ is now measured by

$$V(u, t) = \mathbb{E}(R_t \cdot \mathbb{I}_{\tau_u > t}), \tag{3.35}$$

Correspondingly, $V(u, t)$ is the expected surplus level at time t, where in the case of ruin up to t this surplus is 0 (i.e., due to the ruin event the surplus is frozen in at 0). In terms of a conditional

expectation, one can equivalently express $V(u,t)$ as the probability to still be alive at t times the expected value of the surplus at that time given that ruin has not occurred:

$$V(u,t) = (1-\psi(u,t)) \cdot \mathbb{E}(R_t | \tau_u > t).$$

The following result provides formulas for the ruin probabilities and $V(u,t)$ in this setting.

Proposition 10.

1. *For any $u \geq 0$, the infinite-time ruin probability is given by*

$$\psi(u) = e^{-\theta^* u}, \tag{3.36}$$

where θ^* is the positive solution in θ of the equation

$$c\theta + q\lambda(e^{-b\theta} - 1) = 0. \tag{3.37}$$

2. *For any $u \geq 0$, the finite-time ruin probability is given by*

$$\psi(u,t) = \sum_{n=0}^{\infty} \frac{u}{u+bn} \mathbb{P}\left[\tilde{N}_{\frac{u+bn}{c}} = n\right] \mathbb{I}_{\left\{t > \frac{u+bn}{c}\right\}}. \tag{3.38}$$

3. *For any $u \geq 0$, the expected surplus at time t in case ruin has not occurred until then, can be written as*

$$V(u,t) = \mathbb{E}\left[\left(u + b\tilde{N}_t - ct\right)_+ (-1)^{\tilde{N}_t} G_{\tilde{N}_t}\left(0 \,\middle|\, \left\{\frac{u}{ct} \wedge 1, \ldots, \frac{u + (\tilde{N}_t - 1)b}{ct} \wedge 1\right\}\right)\right], \tag{3.39}$$

where $(.)_+$ denotes the positive part, $\wedge$ stands for the minimum operator and $(G_n(\cdot|\{\ldots\})_{n\in\mathbb{N}}$ is the sequence of Abel-Gontcharov polynomials defined in Section 3.1.2.

Proof. 1. Define the process

$$S_t = ct - b \cdot \tilde{N}_t, \ t \geq 0.$$

It is Lévy and such that $S_t \to -\infty$. Note that

$$\kappa(\theta) = \log\left\{\mathbb{E}\left[e^{\theta(c - bN_1)}\right]\right\} = \theta c + q\lambda(e^{-b\theta} - 1).$$

The equation $\kappa(\theta) = 0$ has only one nonnegative solution θ^*. The process (e^{θ^*}) is a martingale, we then apply Proposition 9 to get

$$\psi(u) = e^{-\theta^*},$$

noting that $\xi(u) = 0$ as ruin occurs exactly.

2. The ruin time τ_u may be rewritten as

$$\tau_u = \inf\left\{t \geq 0 \,;\, \tilde{N}_t = ct/b - u/b\right\}.$$

Note that ruin can only occur at the specific times

$$t_k = \frac{u + bk}{c}, \ k \geq 0,$$

when the function $t \mapsto ct/b - u/b$ reaches integer levels. For $t > 0$, define the set of indices $\mathcal{I} = \{k \geq 0 \, ; \, t_k \leq t\}$. The finite-time ruin probability can then be written as

$$\psi(u,t) = \sum_{k \in \mathcal{I}} \mathbb{P}(\tau_u = t_k)$$

We have that

$$\begin{aligned}
\mathbb{P}(\tau_u = t_k) &= \mathbb{P}\left(\bigcap_{l=1}^{k} \{T_l \leq t_{l-1}\} \cap \{T_{k+1} > t_k\} \right) \\
&= \mathbb{P}\left(\bigcap_{l=1}^{k} \{T_l \leq t_{l-1}\} \cap \{T_k \leq t_k\} \cap \{T_{k+1} > t_k\} \right) \\
&= \mathbb{P}\left(\bigcap_{l=1}^{k} \{T_l \leq t_{l-1}\} \cap \{N_{t_k} = k\} \right) \\
&= \mathbb{P}\left(\bigcap_{l=1}^{k} \{T_l \leq t_{l-1}\} | N_{t_k} = k \right) \mathbb{P}(N_{t_k} = k) \\
&= \mathbb{P}\left(\bigcap_{l=1}^{k} \{U_{(l)} \leq t_{l-1}/t_k\} | \right) \mathbb{P}(N_{t_k} = k) \\
&= (-1)^k G_k(0 | t_0/t_k, \ldots, t_{k-1}/t_k) \mathbb{P}(N_{t_k} = k) \\
&= (-1)^k G_k\{0 | u/(u+bk), \ldots, [u+b(k-1)]/(u+bk)\} \mathbb{P}(N_{t_k} = k) \\
&= (-1)^k \left(\frac{1}{u+bk} \right)^k G_k\{0 | u, \ldots, [u+b(k-1)]\} \mathbb{P}(N_{t_k} = k) \\
&= (-1)^k \left(\frac{b}{u+bk} \right)^k G_k\{-u/b | 0, \ldots, k-1\} \mathbb{P}(N_{t_k} = k) \\
&= (-1)^k \left(\frac{b}{u+bk} \right)^k \left(-\frac{u}{b} \right) \left(-\frac{u}{b} - k \right)^{k-1} \mathbb{P}(N_{t_k} = k) \\
&= \frac{u}{u+bk} \mathbb{P}(N_{t_k} = k)
\end{aligned}$$

3. Using the tower property, we can express the value function (3.35) as

$$\begin{aligned}
V(u,t) &= \mathbb{E}\left[\mathbb{E}\left(R_t \mathbb{I}_{\tau_u > t} \middle| \tilde{N}_t \right) \right] \\
&= \mathbb{E}\left[\left(u + b\tilde{N}_t - ct \right) \mathbb{E}\left(\mathbb{I}_{\tau_u > t} \middle| \tilde{N}_t \right) \right] \\
&= \mathbb{E}\left[\left(u + b\tilde{N}_t - ct \right) \mathbb{E}\left(\prod_{k=1}^{\tilde{N}_t} \mathbb{I}_{\{T_k \leq t_{k-1} \wedge t\}} \mathbb{I}_{\{u + b\tilde{N}_t - ct > 0\}} \middle| \tilde{N}_t \right) \right] \\
&= \mathbb{E}\left[\left(u + b\tilde{N}_t - ct \right)_+ \mathbb{P}\left(\bigcap_{k=1}^{\tilde{N}_t} \{T_k \leq t_{k-1} \wedge t\} \middle| \tilde{N}_t \right) \right] \\
&= \mathbb{E}\left[\left(u + b\tilde{N}_t - ct \right)_+ \mathbb{P}\left(\bigcap_{k=1}^{\tilde{N}_t} \{U_{1:k} \leq \frac{t_{k-1}}{t} \wedge 1\} \right) \right], \qquad (3.40)
\end{aligned}$$

where $U_{1:n}, \ldots, U_{n:n}$ denote the order statistics of n i.i.d. standard uniform random variables. Using the interpretation of Abel-Gontcharov polynomials as the joint probabilities of uniform order statistics then yields (3.39).

□

The reader who wants to learn more on the use of Appell and Abel-Gontcharov polynomials to solve first passage problems involving point processes is referred to Goffard and Lefèvre [2017]. The main issue with formula (3.39) is the lack of tractability. Formula (3.39) is not a closed form expression because of the infinite serie. Note also that the evaluation of the higher order AG polynomials can result in numerical instabilities when using the recurrence relationship. The work around is to replace the fixed time horizon t by an independent exponential random time horizon T with mean t. We hence consider the ruin probability and expected profit at $T \sim \text{Exp}(t)$, defined by

$$V(u,t) := \mathbb{E}[V(u,T)] = \mathbb{E}(R_T \mathbb{I}_{\tau_u > T}). \tag{3.41}$$

Simple expressions for these quantities are given in the following result

Theorem 8. *For any $u \geq 0$, we have*

$$\widehat{\psi}(u,t) = e^{\rho^* u},$$

and

$$\widehat{V}(u,t) = u + (q\lambda b - c)t\left(1 - e^{\rho^* u}\right),$$

where ρ^* is the negative solution of the equation

$$-c\rho + q\lambda(e^{b\rho} - 1) = 1/t. \tag{3.42}$$

The solution ρ^* of (3.42) is given by

$$\rho^* = -\frac{q\lambda t + 1}{ct} - \frac{1}{b} \mathrm{W}\left[-\frac{q\lambda b}{c} e^{-b\left(\frac{q\lambda t+1}{ct}\right)}\right],$$

where $W(.)$ denotes the Lambert function.

Proof. Let $0 < h < u/c$, so that ruin can not occur in the interval $(0,h)$. We distinguish three cases:

(i) $T > h$ and there is no block discovery in the interval $(0,h)$,

(ii) $T < h$ and there is no block discovery in the interval $(0,T)$,

(iii) There is a block discovery before time T and in the interval $(0,h)$.

Let T_1 be the arrival time of the first block. Applying the strong Markov property at $S = T \wedge h \wedge T_1$, we see that

$$\widehat{\psi}(u,t) = e^{-h(1/t+q\lambda)} \widehat{\psi}(u - ch, t) + \int_0^h q\lambda\, e^{-s(1/t+q\lambda)} \widehat{\psi}(u - cs + b, t) ds,$$

and

$$\begin{aligned}\widehat{V}(u,t) \quad = \quad & e^{-h(1/t+q\lambda)} \widehat{V}(u - ch, t) + \int_0^h \frac{1}{t} e^{-s(1/t+q\lambda)} (u - cs) ds \\ & + \int_0^h q\lambda\, e^{-s(1/t+q\lambda)} \widehat{V}(u - cs + b, t) ds.\end{aligned}$$

Now we take the derivative with respect to h and set $h = 0$ to obtain

$$c\widehat{\psi}'(u,t) + \left(\frac{1}{t} + q\lambda\right)\widehat{\psi}(u,t) - q\lambda\widehat{\psi}(u+b,t) = 0, \tag{3.43}$$

and

$$c\widehat{V}'(u,t) + \left(\frac{1}{t} + q\lambda\right)\widehat{V}(u,t) - q\lambda\widehat{V}(u+b,t) - \frac{u}{t} = 0, \tag{3.44}$$

Note that the derivative is with respect to the first argument and that equations (3.43) and (3.44) are advanced functional differential equations. Equation (3.43) is actually the homogeneous counterpart of (3.44). For (3.43) consider the form

$$\widehat{\psi}(u,t) = A_1 e^{\rho u} + B_1, \ u \geq 0. \tag{3.45}$$

Substituting in (3.43), together with the boundary condition $\widehat{\psi}(0,t) = 1$, yields

$$\begin{cases} 0 &= ct\rho + (1 + q\lambda t) - q\lambda t e^{\rho b}, \\ 0 &= B_1, \\ 0 &= A_1 + B_1, \end{cases}$$

where A_1, B_1 and ρ are constant to be determined. We have that $B_1 = 0$ and $A_1 = 1$. The equation

$$ct\rho + (1 + q\lambda t) - q\lambda t e^{\rho b} = 0,$$

have two solutions on the real line, one being negative, the other being positive. To ensure that $\lim_{u\to\infty} \widehat{\psi}(u,t) = 1$, we must take the negative solution that we denote by ρ^*. We finally get

$$\widehat{\psi}(u,t) = \mathrm{e}^{\rho^* u}.$$

Now for (3.44) consider form

$$\widehat{V}(u,t) = A_2 e^{\rho u} + B_2 u + C_2, \ u \geq 0, \tag{3.46}$$

where A_2, B_2, C_2 and ρ are constants to be determined. Substituting (3.46) in (3.44) together with the boundary condition yields the system of equations

$$\begin{cases} 0 &= ct\rho + (1 + q\lambda t) - q\lambda t e^{\rho b}, \\ 0 &= B_2(1 + tq\lambda) - q\lambda t B_2 - 1, \\ 0 &= B_2 ct + C_2(1 + tq\lambda) - q\lambda t B_2 b - q\lambda t C_2, \\ 0 &= A_2 + C_2. \end{cases}$$

We then have $A_2 = -t(q\lambda b - c)$, $B_2 = 1$, $C_2 = t(q\lambda b - c)$. As $A < 0$, we have to choose the negative solution $\rho^* < 0$

$$ct\rho + (1 + q\lambda t) - q\lambda t e^{\rho b} = 0,$$

in order to ensure $\widehat{V}(u,t) > 0$. Substituting A_2, B_2, C_2 and ρ^* in (3.46) yields the result. □

3.2.2 Ruin and expected profit of a selfish miner

Let $(N_t)_{t\geq 0}$ be the homogeneous Poisson process with intensity λ that governs the block discovery process of the entire network. As in the previous section, we consider a miner named Sam who owns a share $q \in (0,1)$ of the computing power so that a newly found block belongs to Sam with probability q. We keep track of Sam's lead over the honest chain in terms of number of blocks via a Markov jump process

$$X_t = Z_{N_t},\ t \geq 0,$$

where $(Z_k)_{k\geq 0}$ is a homogeneous Markov chain with finite state space $E = \{0, 1, 0^*\}$ which we define now. Consider some time $n \geq 0$,

- If Sam is not hiding any block, then $Z_n = 0$,
  - if Sam finds the next block, he stores it in a buffer and $Z_{n+1} = 1$,
  - if the other miners discover a block then Sam's buffer remains empty $Z_{n+1} = 0$,

  In both cases, Sam is not collecting any reward.
- If Sam is hiding one block at that time, then $Z_n = 1$,
  - if he then finds a new block, then he broadcasts both blocks immediately (which resets the Markov chain to $Z_{n+1} = 0$), and he collects two rewards,
  - if the others find a block, then Sam also releases his block leading to a fork situation characterized by $Z_{n+1} = 0^*$. At that moment Sam is not collecting any rewards.
- If a fork situation is present at that time ($Z_n = 0^*$), then
  - if Sam finds a new block then he appends it to his branch of the chain and collects the reward for two blocks and $Z_{n+1} = 0$.
  - if the others find a block then
    - they append it to Sam's branch with a probability $0 \leq \gamma \leq 1$, in which case Sam gets the reward for one block.
    - If the block is mined on top of the competing branch, then Sam earns nothing.

    In both cases, the number of hidden blocks then becomes $Z_{n+1} = 0$.

Let $(\xi_n)_{n\geq 1}$ be a sequence of i.i.d. Bernouill variables with parameter q, we have that

$$Z_n = g[Z_{n-1}, \xi_n] = \begin{cases} 0, & \text{if } Z_{n-1} = 0^*, \\ 0, & \text{if } Z_{n-1} = 1 \ \& \ \xi_n = 1, \\ 0, & \text{if } Z_{n-1} = 0 \ \& \ \xi_n = 0, \\ 0^*, & \text{if } Z_{n-1} = 1 \ \& \ \xi_n = 0, \\ 1, & \text{if } Z_{n-1} = 0 \ \& \ \xi_n = 1. \end{cases} \tag{3.47}$$

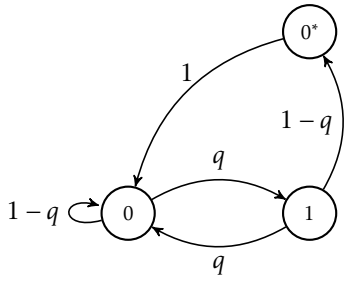


Figure 3.8: Transition graph of the Markov chain $(Z_k)_{k\geq 0}$ representing the stock of blocks retained by Sam when implementing the simplified selfish mining strategy.

The process $(Z_n)_{n\geq 0}$ is a Markov chain with transition graph provided in Figure 3.8. The selfish mining strategy alters the reward collecting process. The surplus process of Sam introduced in (3.33) now becomes

$$R_t = u - c \cdot t + b \cdot \sum_{n=1}^{N(t)} f\left[Z_{n-1}, \xi_n, \zeta_n\right], \tag{3.48}$$

where the $(\xi_n)_{n\geq 1}$ and $(\zeta_n)_{n\geq 1}$ are i.i.d. Bernoulli random variables with parameter q and γ, respectively, and

$$f\left[Z_{n-1}, \xi_n, \zeta_n\right] = \begin{cases} 0, & \text{if } Z_{n-1} = 0, \\ 0, & \text{if } Z_{n-1} = 0^* \ \&\ \xi_n = 0 \ \&\ \zeta_n = 0, \\ 1, & \text{if } Z_{n-1} = 0^* \ \&\ \xi_n = 0 \ \&\ \zeta_n = 1, \\ 2, & \text{if } Z_{n-1} = 0^* \ \&\ \xi_n = 1, \\ 2, & \text{if } Z_{n-1} = 1 \ \&\ \xi_n = 1. \end{cases} \tag{3.49}$$

It is interesting to see whether selfish mining is still profitable for Sam if the possibility of ruin is included in the analysis. His average earning per time unit is now given by

$$\frac{b}{t}\,\mathbb{E}\left[\sum_{k=1}^{N(t)} f(Z_{k-1}, \xi_k, \zeta_k)\right] - c. \tag{3.50}$$

This quantity can be determined if we assume that Sam has been mining in a selfish way for quite some time already, so that we can consider the Markov chain to be in stationarity with stationary probabilities

$$\mathbb{P}(Z = 0) = \frac{1}{1 + 2q - q^2},\ \mathbb{P}(Z = 1) = \frac{q}{1 + 2q - q^2},\ \text{and } \mathbb{P}(Z = 0^*) = \frac{q(1-q)}{1 + 2q - q^2}.$$

The quantities $U_k := f\left(Z_{k-1}, \xi_k, \zeta_k\right),\ n \geq 1$ then have a p.m.f. $p_U(\cdot) := \mathbb{P}(U = \cdot)$ given by

$$p_U(0) = \frac{1 + q(1-q) + q(1-q)^2(1-q)}{1 + 2q - q^2},\ p_U(1) = \frac{p\gamma(1-p)^2}{1 + 2q - q^2},\ \text{and } p_U(2) = \frac{q^2 + q^2(1-q)}{1 + 2q - q^2},$$

and the net profit condition correspondingly reads

$$b\lambda \frac{\gamma q(1-q)^2 + 4q^2 - 2q^3}{1 + 2q - q^2} - c > 0.$$

The profitability of selfish mining consequently depends on the interplay between the probabilities q and γ, and not all values of q and γ will lead to positive expected profit. Define

$$\widehat{\psi}_z(u,t) \equiv \mathbb{E}\left[\psi_z(u,T)\right] = \mathbb{E}\Big(\psi(u,T)\Big|Z_0 = z\Big) \text{ and } \widehat{V}_z(u,t) \equiv \mathbb{E}[V_z(u,T)] = \mathbb{E}\Big(R_T \mathbb{I}_{\tau_u > T}\Big|Z_0 = z\Big).$$

Theorem 9. *For any $u \geq 0$, the ruin probability and expected profit of a selfish miner are given by*

$$\widehat{\psi}_0(u,t) = C_1\, \mathrm{e}^{\rho_1 u} + \mathrm{e}^{\rho_2 u}\left[C_2 \cos(\rho_3 u) + C_3 \sin(\rho_3 u)\right],$$

and

$$\widehat{V}_0(u,t) = A_1\, \mathrm{e}^{\rho_1 u} + \mathrm{e}^{\rho_2 u}\left[A_2 \cos(\rho_3 u) + A_3 \sin(\rho_3 u)\right] + u + C,$$

Proof. See Albrecher and Goffard [2021]. □

The truth is that following the protocol is always more profitable on average, the question is then why selfish mining?

Three reasons:

1. The relative revenue of selfish miner can be greater than their fair share
2. Honest miners waste resources, therefore they might quite making malicious miners more prominent
3. Selfish mining slows down the pace of block arrivals leading to a downward adjustment of the cryptopuzzle difficulty

Reason 1 is the one invoked in the paper of Eyal and Sirer [2014]. Reason 2 is probably difficult to study as we would have to model the resilience of honest miners, it probably requires a game theoretic framework. We will try to illustrate numerically reason 3 in Section 3.2.3.

3.2.3 Solo mining versus selfish mining when including a difficulty adjustment

Let the time unit be the hour. Since the Bitcoin blockchain protocol is designed to ensure that one block of confirmed transactions is added to the blockchain about every ten minutes, this renders the block arrival intensity in our model to be $\lambda = 6$. The reward b is determined by the number n_{BTC} of bitcoins earned when finding a block and the price π_{BTC} of the bitcoin. For the illustrations in this paper, we use the data of January 1, 2020, when $n_{BTC} = 12.5$ and $\pi_{BTC} = \$7,174.74$[3], so that the reward amounts to

$$b = n_{BTC} \times \pi_{BTC} = \$89,684.30.$$

We assume that the operational cost of mining reduces to the electricity consumed when computing hashes. On January 1, 2020, the yearly consumption of the network was estimated

[3] Source:blockchain.com

by the Cambridge Bitcoin Electricity Consumption Index[4] to 72.1671 TWh.[5] We denote by

$$W = \frac{72.1671 \times 10^9}{365.25 \times 24}$$

the electricity consumption of the network expressed in kWh. We let the operational cost c of a given miner be proportional to its share $p \in (0,1)$ of the network computing power with

$$c = p \times W \times \pi_W,$$

where π_W denotes the price of the electricity where the miner is located, expressed in USD per kWh. Mining (at least on the bitcoin blockchain) boils down to drawing random numbers, referred to as hashes uniformly inside the set $\{0, \dots, 2^{256} - 1\}$. The block is mined if the computed hash is smaller than some target T. Calibrating T helps to maintain a steady flow of blocks in the blockchain. In the bitcoin blockchain, the target T is set so as to ensure that 6 blocks are generated per hour. The target may be estimated by comparing 2^{256} to the number of hashes computed by the network in ten minutes. On January 1, 2020, the network was computing 97.01[6] exahashes per second. The number of hashes required on average to mine a block is given by the difficulty defined as $D = T_{\max}/T$. Define $H = 97.01 \times 10^{18} \times 3600$ as the number of hashes computed per hour by the network. We then set the target so that $D/H = 6$ which is equivalent to

$$T = \frac{T_{\max}}{6H}$$

The difficulty/target is adjusted every 2,016 blocks by changing the target T to

$$T^* = T \times \frac{t^*}{336},$$

where t^* is the time (in hours) it took to mine 2,016 blocks (here 2016/6 = 336 hours is the time it should have taken to mine 2,016 blocks). The difficulty adjustment was studied in Bowden et al. [2020] and led them to conclude that the block arrival process is well captured by a non-homogeneous Poisson process. Following their terminology, we refer to the time elapsed between two difficulty adjustments as a *segment*.

We now want to quantitatively address whether selfish mining is worthwhile when considering the possibly implied adjustment of the cryptopuzzle difficulty. We do so in a simplified setup, where only Sam may switch between selfish mining and following the protocol, whereas everyone else follows the protocol. Concretely, we compute the ruin probability and expected profit of Sam over two segments when

(i) he is following the protocol during both segments,

[4] Source:CBEI

[5] The choice of this concrete date for the illustrations in this paper is somewhat arbitrary. Choosing another date and estimate of the Bitcoin price will, however, not crucially change the conclusions as long as the reward for finding a block compensates the operational cost.

[6] Source: blockchain.com

(ii) he applies selfish mining during the first segment and resumes following the protocol during the second segment.

For (i), we compute the expected profit using the result of Theorem 8 by setting the average time horizon to $t = 672$ (the number of hours in four weeks). We assume that the arrival intensity of the blocks in the blockchain remains unchanged over the two segments with $\lambda = 6$, as does the cryptopuzzle difficulty T.

For (ii), we proceed as follows. Selfish mining slows down the pace at which the blocks are added to the blockchain, and the number of blocks wasted exactly corresponds to the number of passages of the Markov chain $(Z_n)_{n\geq 0}$ through the state 0^*. We know that once stationarity is reached, the probability for $(Z_n)_{n\geq 0}$ to be in state 0^* is

$$\frac{p(1-p)}{1+2p-p^2}.$$

We therefore approximate the arrival process in this first segment by a homogeneous Poisson process with intensity

$$\lambda_1 = \lambda \times \left(1 - \frac{p(1-p)}{1+2p-p^2}\right).$$

When blocks are being withheld, the average time required to mine 2,016 blocks increases from 336 to $t_1 = 2016/\lambda_1$. We hence compute the ruin probability and expected surplus over the first segment using Theorem 9 respectively with time horizon t_1. Selfish mining during the first segment then leads to a downward adjustment of the cryptopuzzle difficulty to $T_2 = T \times t_1/336$ that will be in force during the second segment. As the miner resumes following the protocol on that second segment, the block arrival process becomes again a proper homogeneous Poisson process with intensity λ_2 to be determined as follows. Let H be the number of hashes computed per hour (hashrate) by the network. The number of blocks mined per hour is then given by

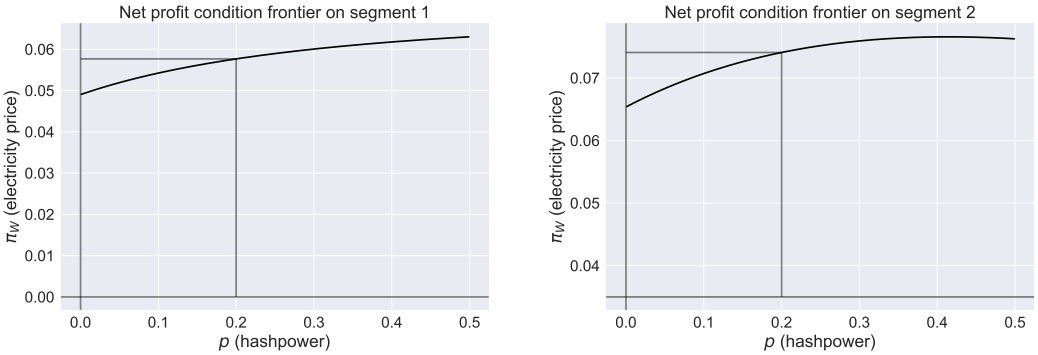
$$\lambda_2 = \frac{2^{256}}{L_2 \times H}.$$

The miner's ruin probability and surplus over the second segment are now computed using the formulas of Theorem 8 using as initial wealth the miner's expected surplus over the first segment

$$u_2 = \widehat{V}_0(u,t),$$

a block arrival intensity equal to λ_2 and a reduced time horizon equal to $t_2 = 2016/\lambda_2$.

We consider a miner who owns a share $p = 0.2$ of the network computing power. If he follows the protocol, then the net profit condition holds if $\pi_W < 0.065$. If he withholds blocks on the first segment, then the net profit condition frontier depends on the electricity price and the hashpower provided that his connectivity is fixed, for instance set to $q = 0.75$. Figure 3.9 shows the net profit condition frontiers for a selfish miner on Segment 1 who starts following again the protocol on Segment 2. In absence of ruin considerations, selfish mining on Segment 1 is



(a) Net profit condition frontier on Segment 1

(b) Net profit condition frontier on Segment 2.

Figure 3.9: Net profit condition frontier on Segment 1 and 2 for a selfish miner.

profitable only if the price of electricity is lower than 0.058, see Figure 3.9a. Only when the selfish miner owns the totality of the hashpower, is selfish mining as profitable as following the protocol. On Segment 2, the profitability is always greater than when following the protocol. The profitability on Segment 2 holds in our case if the electricity price is lower than 0.074, see Figure 3.9b. It is interesting to note that by increasing the hashpower beyond a certain threshold we actually lower the profitability during Segment 2. At higher hashpower levels, the probability of the Markov chain $(Z_n)_{n\geq 0}$ visiting state 0^* becomes small. In that case fewer blocks are being wasted, which in turn reduces the downward adjustment of the cryptopuzzle difficulty and hence the profitability during Segment 2.

Figure 3.10 shows the ruin probability of a selfish miner and a miner following the protocol as a function of initial wealth for a range of electricity prices. From a ruin probability perspective, reducing the difficulty adjustment does not compensate for the risk involved in implementing selfish mining.

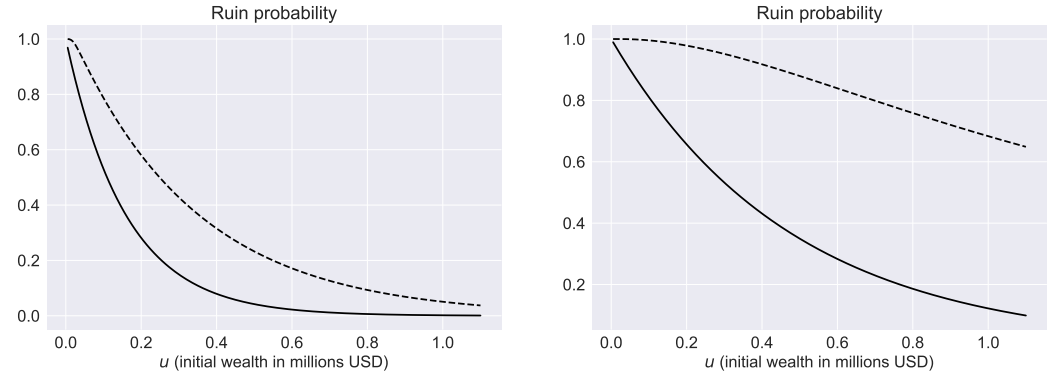
Figure 3.11 displays the expected profit of a selfish miner and a miner following the protocol as a function of initial wealth for a range of electricity prices. One can observe the different profit and loss profile of a selfish miner compared to that of a miner following the protocol on both segments. If the net profit condition holds when following the protocol, then it also holds for the second segment when blocks were withheld during the first segment. For electricity prices $\pi_W = 0.05, 0.06$, the expected profit as a function of u reaches a plateau of level

$$(c - b\lambda p) \times t \tag{3.51}$$

when following the protocol, and

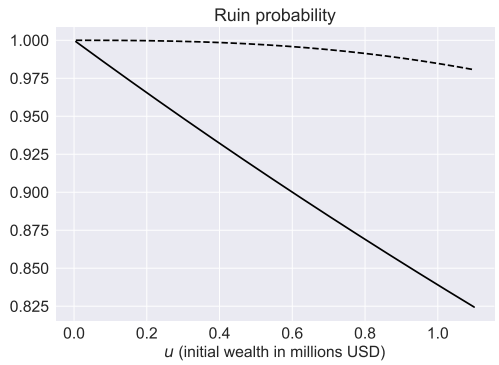
$$(c - b\lambda_2 p) \times t_2, \tag{3.52}$$

when selfish mining is applied during the first segment. For $\pi_W = 0.05$, the plateau when following the protocol (3.51) is higher than the plateau when withholding blocks (3.52), but the expected profit at lower initial wealth is greater for the selfish miner, see Figure 3.11a. The

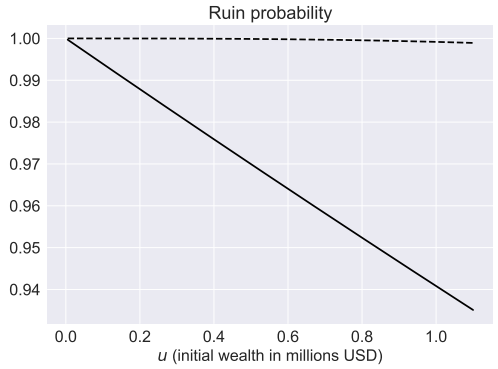


(a) $\pi_W = 0.05$.

(b) $\pi_W = 0.06$.



(c) $\pi_W = 0.07$.

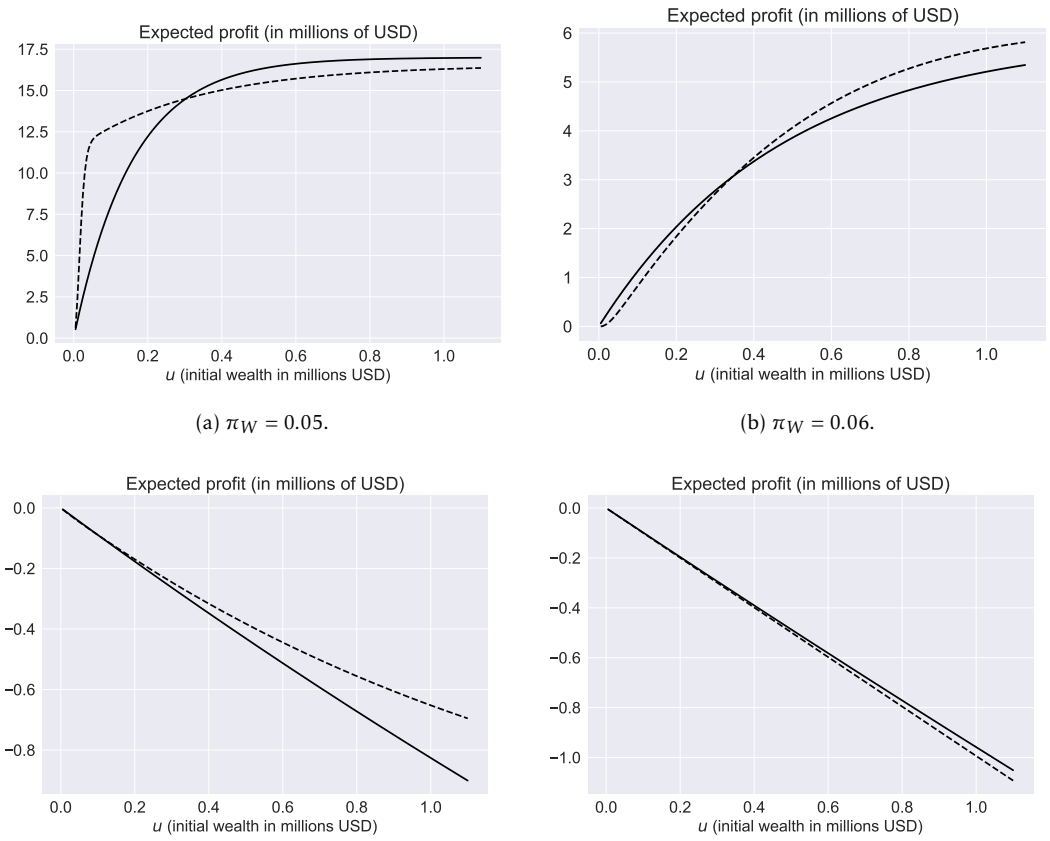


(d) $\pi_W = 0.08$.

Figure 3.10: Ruin probability over two segments as a function of initial wealth of a miner following the protocol (solid) and a selfish miner (dashed) for various electricity prices with hashpower $p = 0.2$ and connectivity $q = 0.75$.

exact opposite holds when $\pi_W = 0.06$, see Figure 3.11b. The latter is probably the most desirable situation for a selfish miner. For $\pi_W = 0.07$, the net profit condition no longer holds when following the protocol which entails a loss, it holds however on the second segment of the selfish miner but it does not compensate for the loss incurred during the first segment, see Figure 3.11c. Selfish mining helps at least in slightly mitigating the losses in this case. For electricity prices $\pi_W > 0.074$, the net profit condition breaks down in each case, resulting in huge losses for both the selfish and the honest miner (cf. Figure 3.11d).

The above analysis allowed us to distinguish situations where selfish mining can be considered worthwile and when it may not. In particular, it turns out that selfish mining can be advisable when following the protocol is not profitable.



(a) $\pi_W = 0.05$.

(b) $\pi_W = 0.06$.

(c) $\pi_W = 0.07$.

(d) $\pi_W = 0.08$.

Figure 3.11: Expected profit over two segments as a function of initial wealth of a miner following the protocol (solid) and a selfish miner (dashed) for various electricity prices with hashpower $p = 0.2$ and connectivity $q = 0.75$.

Chapter 4

Decentralization of blockchain system

Decentralization represents th efairness of the distribution of the accouunting right of the nodes in thee blockchain network. The consensus protocol must be designed so that the decision power does not eventually concentrate on a few nodes.

Section 4.1 focuses on the PoS protocol by modelling the evolution of the stakes of the nodes by a stochastic process with reinforcement. Section 4.2 presents the concept of mining pool and discusses the threat they represent for the decentralized aspect of the network.

4.1 Decentralization in PoS

Rich get richer? Polya's urn

4.1.1 Average stake own by each peer

4.1.2 Distribution of the stakes

4.2 Decentralization in PoW

4.2.1 Mining pools and reward systems

4.2.2 Mining pool risk analysis

Chapter 5

Efficiency of blockchain systems

5.1 A queueing model with bulk service

5.2 Latency and throughputs computation

Bibliography

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

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

Hansjörg Albrecher and Pierre-Olivier Goffard. On the profitability of selfish blockchain mining under consideration of ruin. *To appear in Operations Research*, May 2021. URL https://hal.archives-ouvertes.fr/hal-02649025. https://arxiv.org/abs/2010.12577.

Andreas Antonopoulos. *Mastering Bitcoin*. O'Reilly UK Ltd., July 2017. ISBN 1491954388. URL https://www.ebook.de/de/product/26463992/andreas_antonopoulos_mastering_bitcoin.html.

Søren Asmussen and Hansjörg Albrecher. *Ruin Probabilities*. WORLD SCIENTIFIC, sep 2010. doi: 10.1142/7431.

Benjamin Avanzi, Hans U. Gerber, and Elias S.W. Shiu. Optimal dividends in the dual model. *Insurance: Mathematics and Economics*, 41(1):111–123, jul 2007. doi: 10.1016/j.insmatheco.2006.10.002.

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

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

Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security*, pages 436–454. Springer Berlin Heidelberg, 2014. doi: 10.1007/978-3-662-45472-5_28.

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

Pierre-Olivier Goffard and Claude Lefèvre. Boundary crossing of order statistics point processes. *Journal of Mathematical Analysis and Applications*, 447(2):890–907, mar 2017. doi: 10.1016/j.jmaa.2016.10.044.

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

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

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

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

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

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

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

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

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

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