UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

Faculty of Science and Technology
Department of Computer Science

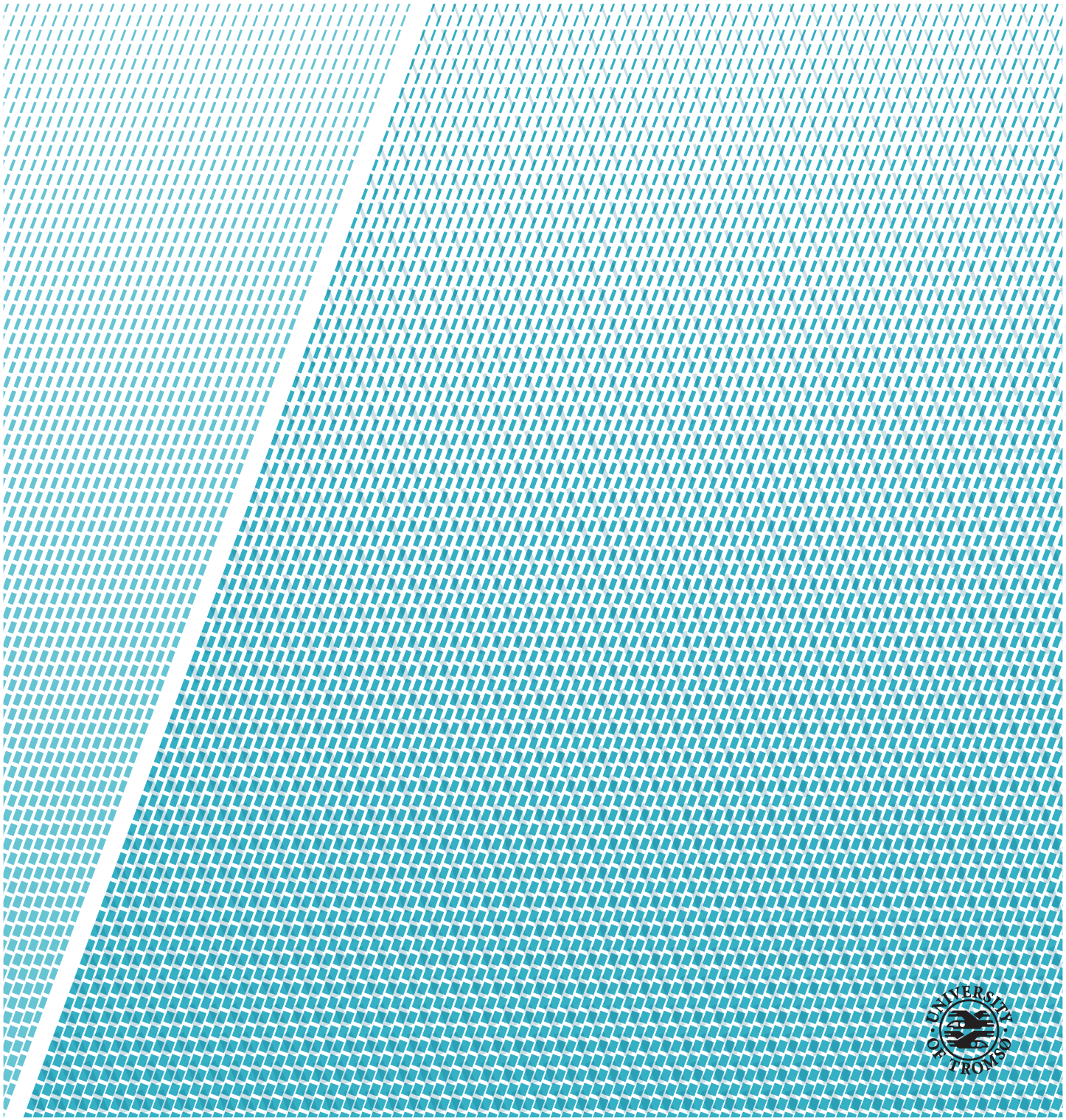
Trading Network Performance for Cash in the Bitcoin Blockchain

subtitle

—

Enrico Tedeschi

Master Thesis in Computer Science



Abstract

Nowadays blockchain systems are emerging and they are spreading each day more. Cryptocurrencies are the biggest example of a physical implementation of this protocol, having in 2012 more than 50 thousands transactions per day and reaching now in 2017 more than 350 thousands of transactions approved every day. In this thesis we evaluate the most famous blockchain system, the *Bitcoin blockchain*. Public blockchains have emerged as a plausible messaging substrate for applications that require highly reliable communication. However, sending messages over existing blockchains can be cumbersome and costly as miners require payment to establish consensus on the sequence of messages. The blockchain protocol requires an always growing size of the information stored in it so its *scalability* is the biggest problem. For that reason we collected data to be analyzed and stored in our own data frame, saving up to $x10$ space for the analysis.

This thesis will consider the network performance of the Bitcoin public ledger when used as a massaging substrate. From 2009 to 2017 a lot of analysis has been done on Bitcoin blockchain and meanwhile its block size limit changed multiple times, from 256 bytes up to 1 Mb, the Bitcoin price raised from ~ 0.7 \$ to more than 4.000 \$ and different papers were published discussing whether changing or not the block size limit or talking about the fees a miner could get from clients. We read and considered previous analysis on Bitcoin blockchain, we then present our own dataset, which contains a significant portion of the Bitcoin blockchain, updated at 09-2017, discuss our results and compare them with other evaluations from past years, then we also discuss how the fee paid to miners evolves during time and how much a client could pay for a faster approval time, plus we take into consideration transaction visibility, blockchain growth and fees paid to miners. From this we propose and evaluate, using machine learning techniques, three different cost prediction models for predicting bandwidth per Bitcoin cost of upcoming transaction. The models can be used by application to throttle network traffic to optimize message delivery. We also discuss and consider, according to the data obtained, whether the block size limit should be increased for an higher *throughput* or not.

People are using Bitcoin because it has a lower fee rate and no central authority,

we aim to find any possible relation between the fee paid from a transaction to a miner and the approval time of this transaction, plus we also noticed that the bigger is the blockchain size the more the system become centralized, since only few members, or nodes, of the Peer to Peer network can support and use the full blockchain.

Bitcoin blockchain has been analyzed with a blockchain analytics system, developed using Bitcoin's API and data were collected both by using the API and parsing `blockchain.info` HTML pages. A total of # transactions has been evaluated, more transactions than ever were considered before and useful information about the Bitcoin blockchain emerged. This thesis gives also a measurement about accuracy of data provided from `blockchain.info`.

Contents

List of Figures

List of Tables

My list of definitions

1 Introduction

In 1964, Paul Baran [31] represented a very clear topology describing the differences between a centralized, decentralized and distributed network (Figure 1.1). Since then, the attention in developing systems moved from a centralized scheme to a distributed one, leaving most of the computation to every single user in the network rather than a central coordinator. Such a change might be easy for systems that do not require much of security, where authentication or authorization is minimal. However, the more a system needs to be secure, the more the decentralization process might be tricky as it becomes very important to rely on some trusted central coordinator. Systems that more than others need to be secure are the one related to e-commerce, banking and trades, all systems that have to deal with money.

In 1983, a research paper by David Chaum introduced the idea of digital cash [37]. In 1990 he founded *DigiCash*, an electronic cash company that closed because of bankrupt in 1998. After that, other systems such as *e-gold* (1996) and *PayPal* (1998) emerged. However, these systems allowed digital money transfer while they were still relying on a central authority. In 2008 Satoshi Nakatomo has presented Bitcoin [55], the first decentralized digital currency. Until 2008 e-commerce used to rely exclusively on financial institutions serving as trusted third parties. Those are involved in the electronic payments process and they have to guarantee consistency of the transactions and security of data.

Decentralized digital currencies are not dependent on any trusted third parties and they are built over a Peer to Peer (P2P) network where every component

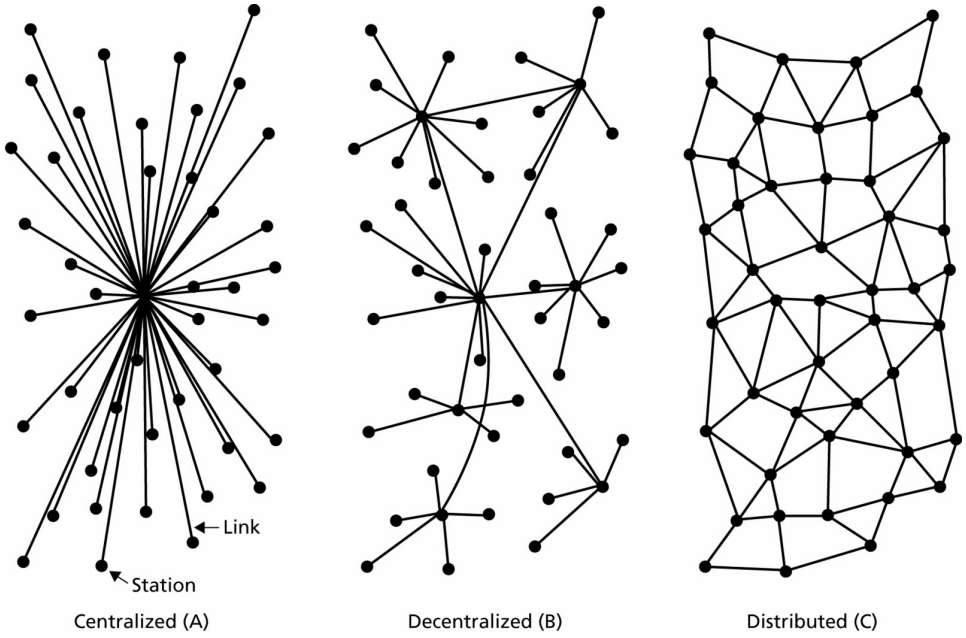


Figure 1.1: Differences between network topologies. Source: On Distributed Communication Networks, Paul Baran, 1964.

has the same privileges. These systems allow money exchange without a central authority, which means lower fees, no geographical separation and global trust among users. After Bitcoin, more decentralized digital currencies emerged, in 2011 *Litecoin*, originally based on the bitcoin protocol, then in 2013 Gavin Wood has presented *Ethereum* [64] and in 2014 *Monero* currency was released.

The order of transaction is essential in any cryptocurrency systems. However, establishing correct order can be problematic in decentralized cryptocurrency systems as they allow arbitrary nodes to join, including nodes that might be malicious. If arbitrary or Byzantine faults are allowed, the system might be left in an inconsistent or invalid state [49]. The ability to mask Byzantine faults has been implemented in various systems such as Byzantium [44], HRDB [62] and MITRA [51]. These protocol guarantees consistency of transactions having f faulty nodes, with a total of N nodes where $N = 2f + 1$ or $N = 3f + 1$, and a protocol like *Fireflies* [48] provides secure and scalable membership management and communication substrate in overlay network with Byzantine members. To guarantee an order of transaction all these cryptocurrencies rely on the *blockchain* protocol.

1.1 Blockchains

The need to tolerate malicious members was the reason for introducing the *blockchain* into cryptocurrency systems. The fundamental principle behind the blockchain is that consensus on transaction ordering is based on contributed computational power rather than number of participants. The blockchain works by appending transactions in blocks. Every block is generated after a relevant computation (*proof-of-work*), and each new block is appended to the public ledger of data, the blockchain, having in that way an ever growing chain of data containing every transaction ever happened.

Besides its use in cryptocurrency, this blockchain technology opens up to several usages in different sectors such as trading, file storage or identity management. Indeed it is already used by NASDAQ in its private socket market. If used in a P2P file sharing network, the blockchain removes the need of a centralized data base and heavy storage areas. Moreover it allows users to create tamper-proof digital identities for themselves. Blockchain technology opens up to usages in several important sectors such as trading, file storage, and identity management.

Blockchains essentially implements a distributed consensus protocols that enable a set of untrusted processes to agree on the content of an append only data structures. These ledgers are divided into blocks and linked together in sequence by hashes. They facilitate transactions between consenting individuals who would otherwise have no means to trust each other and deal with geographical separation and interfacing difficulties This technology promises a highly resilient and communication substrate where messages are kept potentially for a long time.

Nonetheless, decentralized digital currencies also have some side effects. The most relevant is *scalability*, due to the steady growth of the blockchain. It should be also considered that decentralized cryptocurrencies operate in open (or permissionless) networks in which the ledger of data could be manipulated from arbitrary adversaries and according also to the paper from University of Singapore [52] security of smart contracts has not received much attention yet. And since the only part not protected from cryptography is the *order of transactions* [16], an attacker would try to convince the network that a transaction occurred earlier than another one to gain money. The security bugs in smart contracts are classified as *Transaction-Ordering Dependence*, *Timestamp Dependence*, *Mishandled Exceptions* and *Reentrancy Vulnerability* [52]. In this thesis we refrain from explaining Bitcoin and its terminology in detail and refer the reader to already existing high-level [61, 36] or technical [55, 16]. description.

1.2 Problem Statement

While doing research, studying and reading papers related to blockchains, it turned out that the most urgent concerns are related to its scalability, performance and a profit optimization from participant in the whole system. In 2015, Möser and Böhme write [54]:

> *Bitcoin may not be as cheap for consumers as it appears.* [. . .] *Bitcoin users are encouraged to pay fees to miners, up to* 10 *cents (United States Dollar (USD)), per trnsaction, irrispective of the amount paid.*

Rizun writes in 2015 [56]:

> *The block size limit was set at one megabyte, corresponding roughly to three transactions per second.* [. . .] *The transaction rate is over three hundred times larger than when the block size limit was introduced, and rising the limit is now being seriously considered.*

Then Croman writes in 2016 [38]:

> *The current trend of increasing the block sizes on Bitcoin protends a potential problem where the system will reach its maximum capacity to clear transactions, probably by* 2017.

It is obvious then that these problems need to be taken into consideration. In our thesis we propose a longitudinal study on Bitcoin blockchain, analyzing most recent data. In particular, we discuss the scalability of the blockchain, how it affects the throughput and we present performance observations of the Bitcoin blockchain, analyzed with a blcockhain analytic system developed for this purpose. We provide detailed insights and analysis on how Bitcoin's characteristics, such as fee, block size and reward to different miners involved have changed over time, and provide an updated model describing how the Bitcoin blockchain will grow. Furthermore, we analyzed the correlation between the fee paid from a transaction and its *latency*, or the time it takes to be visible in the whole network. Three different models are proposed to describe how applications best can spend money to improve network charactertics, this affects average bandwidth available to an application.

1.2.1 Scalability

Scalability and network performances are urgent concern in existing Blockchain-based cryptocurrencies [38]. According to Ethereum white paper [16], if Bitcoin would have the same amount of transactions of a VISA circuit, its blockchain

would grow about 1 MB every 3 seconds, ~ 28GB per day, instead of the actual growth of ~ 0.12GB per day. In this thesis we discuss how much scalability affects centralization in Bitcoin network and how much it will impact in the next couple of years the blockchain growth.

1.2.2 Performance

Centralized schemes, like VISA are immediate, while having a throughput of 2000 transactions/sec up to 56 thousands transactions/sec [38]. It is true that Bitcoin has lower fees than centralized currency schemes, but these properties come at a performance and scalability cost. In the paper from Croman [38], they claim that Bitcoin achieve a throughput of 7 transactions/sec. In this thesis we also want to update at 2017 this statment and see how much a block size change might influence the whole network performance.

1.2.3 Questions

By doing transactions-wise and block-wise experiments we state and focus on three major problems of the blockchain:

1. Scalability
2. Performance
3. Fees and Tolls

1.3 Method / Context

In this thesis we analyze a considerable part of the blockchain. In the paper from 2015 written by Möser and Böhme [54], they analyze tips and tolls in Bitcoin blockchain, they collected data until 2014 and they analyze more than 9 million of transactions. At that time there were a total of 100 thousands transactions per day, while today we count about 350 thousands on daily bases, so the retrieving part turned out to be more time consuming than expected. Despite that, we aim to collect even a larger portion of the blockchain, storing data smartly in a *data frame*, which allows us to spare up to $10x$ of the space the blockchain actually requires. Then we analyze data and with *machine learning* techniques we define models, discuss about the results and how much they can be reliable in a future-wise implementation. In our data frame we store more than # transactions, with an analysis in between #date and #date. We used for the information retrieval Application Programming Interface (API) from `blockchain.info` combined with a HyperText Markup Language (HTML)

parsing on every "block-page" of the same website.

Our assumption is that we can get sufficient information about the blockchain growth, the block creation time, the time for a transaction to be visible in the public ledger of data and which miners are the more trendy and which usually requires more fee by retrieving and analyzing only a portion of the blockchain, but having in that way a finer granularity than the one represented in the Bitcoin website. In that way we hope to gain more information out of it. Moreover, sampling data from a single node in the blockchain gives statistics representative of the whole system.

For the analysis, data are retrieved block per block and part of the blockchain is saved in a text file. This finer granularity allows us to have a lot of information that may be hidden in the statistical analysis provided from Bitcoin. It is also possible to use the information retrieved to make future predictions about how much the Bitcoin blockchain will grow, using polynomial interpolation on the data. According on how many blocks ago are fetched, it is possible to have an accurate prediction on the blockchain growth for the next few years.

We are going to compare more recent data, retrieved real time, with the Bitcoin one and see the differences of the blockchain growth. Moreover, In the Bitcoin website for blockchain analysis, `blockchain.info` [9], the finer granularity shows data for the last 7 days while we are collecting and monitoring data at every block creation ($\sim$ 8-10 min). In that way is easier for us to check if there are any abnormalities in the ledger of public data.

1.4 Outline

2

Related Works - SotA

As mentioned in Chapter 1, scalability and analysis on the blockchain has been taken into consideration by many researchers in the past years. This chapter summarizes the most relevant papers or works that talks about Bitcoin, blockchain and decentralized cryptocurrencies, it gives a short view of what has already been done and the reults obtained. In our previous paper [60], we enhance the importance of paying for having a certain bandwidth in the Bitcoin network. A paper from Peter R. Rizun [56], explains how a rational Bitcoin miner should select transactions from his node mempool, when creating a new block, in order to maximize his profit. We discuss that and apply new data from more recent sources to this idea of "selecting the right transaction" in a way that a miner could select the right transactions to earn more money out of the whole mining process. Scalability has taken into consideration in the Position Paper of Kyle Croman [38], they analyze how fundamental bottlenecks in Bitcoin limit the ability of its current peer-to-peer overlay network to support substantially higher throughputs and lower latencies. We are going to test the throughput as well, comparing it with the one showed in this paper. Regarding fees and tolls paid in the Bitcoin blockchain we refer to the study done in 2014 from Möser and Böhme [54]. They analyze the entire blockchain and make assumptions about that these "fees" are supposed to substitute miners' minting rewards in the long run. This paper contributes empirical evidence from a historical analysis of agents' revealed behavior concerning their payment of transaction fees. Furthermore, to fully understand how is possible to make money out of the blockchain and mining, in necessary to have a view of how *VISA* [19] makes money as well.

2.1 Rizun - A Transaction Fee Market Exists Without a Block Size Limit

2.1.1 Problems

A pressing concern exists over the ramifications of changing (or not) a Bitcoin protocol rule called *block size limit*. This rule sets an upper bound on the network's transactional capacity, or *throughput*. The limit was set at 1 Mb, corresponding roughly to three transactions per second. When this limit was set, it was over eight hundred times greater than what was required. However in 2015, blocks were filled near capacity and users experienced delays. In 2015 the transaction rate was over three hundred times larger than when the block size limit was introduced. One of the concerns is whether, in the absence of a limit or if the limit is far above the transactional demand, a healthy transaction fee market would develop which charges users the full cost to post transactions. The object of this paper is to consider whether or not such a fee market is likely to emerge if miners, rather than the protocol, limit the block size.

2.1.2 Methods

This paper shows how a Bitcoin miner should select transactions from his node's mempool when creating a new block in order to maximize his profit in the absence of a block size limit. *Block space supply curve* and *mempool demand curve* are explained, and the paper shows how the supply and demand curves from classical economics are related to the derivatives of these two curves. In the paper Rizun claims that the block-size limit determines the transaction throughput. In this paper he derives the *miner's profit equation* and then he introduces two novel concepts called the *mempool demand curve* and the *block space supply curve*.

Miner's Profit Equation

Every time a block is mined, the miner expects to generate a revenue $\langle V \rangle$ at hashing cost $\langle C \rangle$ to earn profit per block

$$\langle \Pi \rangle = \langle V \rangle - \langle C \rangle. \tag{2.1}$$

Miner's profit equation in 2.1 shows the gain of a miner $\langle \Pi \rangle$, where the hashing cost is represented as follows:

$$\langle C \rangle = \eta h T. \tag{2.2}$$

So the hashing cost $\langle C \rangle$ is directly dependent from the miner's individual hash rate, h, the cost per hash, η, and the creation time, T. Moreover, is important to consider the expectation value of a miner's revenue per block, this value is represented with $\langle V \rangle$ and is equal to the amount he would earn if he won the block multiplied by his probability of winning. So the expected revenue would be: $\langle V \rangle = (R + M)h/H$, where the amount he would earn is the sum of the block reward, R, and the transaction fees, M. His probability of winning, assuming all blocks propagating instantly, is equal to the ratio of his hash rate, h, to the total hash rate of the Bitcoin network, H. The problem with this equation is that it does not reflect the miner's diminished chances of winning if he chooses to publish a block that propagates slowly to the other miners. If a miner finds first a valid block, but his solution is received after most miners are working on another, then his block will likely be discarded. This effect is called *orphaning*. The equation, considering the orphaning factor, $\mathbb{P}_{orphan}$, is the following:

$$\langle V \rangle = (R + M) \frac{h}{H} \left(1 - \mathbb{P}_{orphan}\right). \tag{2.3}$$

Where P_{orphan} increases with the amount of time a block takes to propagate to other miners. Indeed, if τ is the block propagation time, the probability of orphaning is defined as:

$$\mathbb{P}_{orphan} = 1 - e^{-\frac{\tau}{T}}. \tag{2.4}$$

In conclusion the *miner's profit equation* is defined as:

$$\langle \Pi \rangle = (R + M)\frac{h}{H}e^{-\frac{\tau}{T}} - \eta h T \tag{2.5}$$

A *rational miner* selects which transactions to include in his block in a manner that maximizes the expectation value of his profit. This selection is explained with the *mempool demand curve* and the *block space supply curve*.

The Mempool Demand Curve

The set of transactions that still need to be approved and included in a block is called *mempool*. The mempool set is denoted with $\mathcal{N}$ and the number of transactions contained within it as n. According to the size limit, a block can select a $b \leq n$ transactions from $\mathcal{N}$ to create a new block $\mathcal{B} \subset \mathcal{N}$. A block first includes transactions with an higher *fee density*, ρ. This last, is a ratio between the *transaction fee*, t_f and the *transaction size*, t_q. To construct the mempool demand curve, is necessary first sorting the mempool from greatest fee density to least and then associating an index $\{i : 1, 2, \ldots, n-1, n\}$ with each transaction in the resulting list. The mempool demand curve will be then a graphical representation of the sum of the fees offered by each transaction

in this sorted list:

$$M_{demand}(b) \equiv \sum_{i=1}^{b} fee_i, \tag{2.6}$$

and the sum of each transaction's size in bytes:

$$Q(b) \equiv \sum_{i=1}^{b} size_i. \tag{2.7}$$

The mempool demand curve represents then the maximum fee, $M_{demand}(b)$ a miner can claim by producing a given quantity $Q(b)$ of blockspace.

The Block Space Supply Curve

The size of the block a miner elects to produce controls the fees he attempts to claim, $M(Q)$, and the propagation time he chooses to risk, $\tau(Q)$. The block space supply curve represents the fees a miner requires to cover the additional cost of supplying block space Q. This cost grows exponentially with the propagation time. The equation which represents this curve is the following:

$$M_{supply}(Q) = R\left(e^{\frac{\Delta\tau(Q)}{T}} - 1\right), \tag{2.8}$$

where $\Delta\tau(Q) \equiv \tau(Q) - \tau(0)$. The propagation time ,$\tau$, is just an esteem from the propagation delay versus the block size.

Maximizing the Miner's Profit

To maximize his profit, the miner construct a mempool demand curve and a space supply curve. The block size Q^* where the miner's surplus, $M_{demand} - M_{supply}$, is largest represents the point of maximum profit. Considering this point Q^* of maximum profit, Rizun considers three market conditions for Bitcoin transaction fees: *healthy*, *unhealthy* and *non-existent*. In a healthy fee market, the miner's surplus is maximized at a finite quantity of block space, and thus a miner is incentivized to produce a finite block. In an unhealthy market, the miner's surplus continually increases with block space, and therefore a rational miner should produce an arbitrary large block. In a non-existent market, including *any* transactions results in a deficit to the miner, and so the miner is better off producing an empty block. A rational miner will produce a big block if his mempool is full of high fee density transactions, and will produce an empty block if no transactions pay a fee sufficient to offset the orphaning risk.

2.1.3 Results

In conclusion, they show that a transaction fee market should emerge without a block size limit if miners include transactions in a manner that maximizes the expectation value of their profit. A critical step in establishing this result was their calculation of the miner's cost to supply additional block space by accounting for orphaning risk.

2.2 Möser & Böhme - Trends, Tips, Tolls: A Longitudinal study of Bitcoin Transaction Fees

2.2.1 Problems

The Bitcoin protocol supports optional direct payments from transaction partners to miners, also called *fees*. Acknowledging their role for the stability of the system, the right level of transaction fees is a hot topic of normative debate. The actual costs of the system are not extensively studied yet. Disregarding intangible factors of (in)convenience, Bitcoin may not be as cheap for consumers as it appears. The main problems/questions that this paper focuses on are:

1. Do higher transaction fees lead to faster confirmation?
2. Do impatient users offer higher fees?
3. Do mining pools enforce strictly positive fee systematically (excluding zero-fee transactions)?

2.2.2 Methods

They enhance the definition of transaction fee, which is encoded as difference between the sum of all inputs and the sum of all outputs of a transaction. Then to study trends of Bitcoin transaction fee conventions over the past couple of years, they combine data from different sources. They load the blockchain by parsing the block files of the Bitcoin Core client [3] and extract information on the size of the block and transactions. Additional data is fetched from `blockchain.info`, such as information about miners. Furthermore, data on bitcoin exchange rate is taken from `coindesk.com`, which provides an average Bitcoin price in USD. The time range selected for the analysis is in between January 2011 and August 2014. To answer question (1), they compared time when a transaction is first seen on the network and the timestamp of the block

that includes the transaction, calculating in that way transaction latency, t_l. They analyze a representative subset of 9000 transactions randomly chosen from all eligible transaction between June 2012 and May 2013, then to answer question (2) they compute for each transaction the holding time, which is the period until the output was spent again, and compared their fees to see if they are higher. To answer question (3) is necessary get information about major mining pools. They used data from `blockchain.info` to retrieve useful information about miners and major miners were analyzed such as *AntPool*, *50BTC*, *BitMinter*, *Slush*, *ASICMiner* and more.

2.2.3 Results

Trends

Overall, they claim that Bitcoin transaction fees are lower than 0.1% of the transmitted value, which is significant below the fees charged by conventiaonal payment systems. It appeared to them that hard size limit do not (yet) significantly drive the level of transaction fees. In our thesis we want to test if this is still true. Regarding trends for the fees paid per transaction over time, the first notable change from 0 and 0.01 Ƀ fee occurs after June 2011, transactions with fee of 0.0005 Ƀ appear and account for about 20-30% of all transaction. In the second quarter of 2012 the transactions paying 0.0005 Ƀ raised to 60-70% of all transactions. In the fourth quarter of 2012, 30-40% of all transactions were paying a fee of 0.001 Ƀ. In May 2013, the nominal value of 0.001 Ƀ makes space for a tenth: 0.0001 Ƀ. This fee level stays on and gains a share of more than 70% towards 2015. In order to reason about these changes, they mapped important events in the Bitcoin ecosystem. Generally, there seem to be two main reasons for shift in trends: changes to the Bitcoin reference implementation and actions by large intermediaries in the ecosystem. The emergence of 0.0005 Ƀ fees in June 2011 can be mapped to the release of version 0.3.23 of the Bitcoin Core client, which reduced the default transaction fee from 0.01 Ƀ to 0.0005 Ƀ. The raise of these last transaction fees in the second quarter of 2012 is probably due to the launch of the gambling website *SatoshiDice* [23]. On May 2013, version 0.8.2 of Bitcoin Core was released.

Tips

There is a small share of transactions that did not offer fee to miners, most of them offered default fee amount but some of them were even willing to pay a higher fee. A plausible reason is that paying more in fee leads to a faster confirmation. After the analysis turned out that half of all zero-fee transactions had to wait more than 20 minuets for their first confirmation. In contrast to

that, paying a 0.0005 Ƀ fee lead to an inclusion into a block in half of the time. 10% of all zero-fee transactions took almost 4 hours to confirm, in contrast to 40 minutes for transactions paying a 0.0005 Ƀ fee. The difference between paying 0.0005 Ƀ or 0.001 Ƀ fee is not as pronounced, but the difference in medians are still statistically and economically significant.

Tolls

Analysis on pool behavior regarding a possible systematic exclusion of zero-fee transactions has been done. Shares have shifted between pools quite extensively. In 2013, BTC Guild had a market share of up to 40%, in 2014 both GHash.IO and Discus Fish ousted this pool. Also, the share of other pools has risen in 2014. Previous incumbents like Slush or 50BTC have lost popularity. Possible reasons include economic and technical factors, like pool fees, service availability, or robustness against attacks. Given the dominance of a few mining pools, they evaluated whether some pools systematically enforce fees. The results show that two pools, Discus Fish and Eligius, have a considerably higher share of blocks without any zero-fee transaction, with 30.6% for Eligius and 62.5% for Discus Fish, in contrast to an average of 14.4%. Over than that though, there is no clear evidence for enforcement of strictly positive transactions fees.

2.3 Croman - On Scaling Decentralized Blockchains

2.3.1 Problems

The increasing popularity of blockchain-based cryptocurrencies has made scalability a primary and urgent concern. The main question that this paper focuses on is the following:

> *Can decentralized blockchains be scaled up to match the performance of a mainstream payment processor? What does it take to get there?*

At the time of writing, the Bitcoin blockchain took 10 min or longer to confirm transactions, achieving 7 transactions/sec maximum throughput. Visa credit card confirms a transaction within seconds and processes 2000 transactions/sec ona verage with peaks of 56, 000 transactions/sec. This paper aims to place exploration of blockchain scalability on a scientific footing. Bitcoin community has put forth various proposals to modify the key systems parameters of block size and block interval. In this paper they show that such scaling by

reparametrization can achieve only limited benefits. This because because Bitcoin generates a lot of network traffic, due to its decentralization. There are a lot of peers in the network and they all have to interact. To ensure that most of the nodes in the overlay network have sufficient throughput they set two guidelines:

- **Throughput limit.** The block size should not exceed 4 MB given 10 minutes average block interval. Corresponding at maximum 27 transactions/sec.
- **Latency limit.** The block interval should not be smaller than 12 seconds.

The community also proposed radically different scaling approaches, and introduced mechanisms such as Corallo's relay network, a centralized block propagation mechanism. One of the main contribution of this paper was to *quantify* Bitcoin's current scalability limits within its decentralized components. Their findings leaded them to the position that *fundamental protocol redesign is needed for blockchains to scale significantly while retaining their decentralization*. Plus, scalability is not a single metric and measurement and understanding of many important metrics, like *fairness* or *mining power utilization*, are lacking. Monitoring and measuring a decentralized blockchain from only a few vantage points poses significant challenges. In this paper they call for better measurements techniques, by continuously monitor the health of the decentralized system to answer key questions such as: "*To what extent can we push system paramteres without sacrificing security?*".

2.3.2 Methods

In this paper they manly focused on:

- **Maximum throughput.** At the time of writing (2016) maximum throughput was 3-7 transactions/sec. Number constrained by Q and $\mathcal{T}$.
- **Latency.** Time for a transaction to confirm, t_l. A transaction is considered confirmed when it is included in a block, roughly 10 minutes expectation.
- **Bootstrap time.** The time it takes to a new node to download and process the history necessary to validate the current system state. In 2016 that was roughly 4 days.
- **Cost per Confirmed Transaction (CPCT).** The cost in USD of resources consumed by the entire Bitcoin system to confirm a single transaction. It could be summarized in:
  1. *Mining:* Expended by miners generating the proof of work for each block.
  2. *Transaction validation:* The cost of computation necessary to validate that a transaction can spend the outputs referenced by its inputs, dominated by cryptographic verifications.

3. *Bandwidth:* The cost of network resources required to receive and transmit transactions, blocks and metadata.
4. *Storage:* The cost of storing all currently spendable transactions, which is necessary for miners and full nodes to perform transaction validation, and of storing the blockchain's historical data, which is necessary to bootstrab new node that join the network.

The cost per transaction for Bitcoin was calculated performing a back-of-the-envelope calculation by summing up the electricity consumed by the network as a whole, as well as the hardware cost of mining equipment. They projected their estimates based on the *AntMiner S5+* mining hardware [21]. They assume a 1 year effective lifetime for the hardware and that the average hashing rate of the network is 450, 000, 000 GH/s. Furthermore, they assume an average price per KWh of 0.1$. Two scenarios are possible, the first is when the Bitcoin network is operating at maximum throughput of 3-7 transactions/sec. This limit is constrained by the 1 MB block size limit and the variable transactions size. The lower bound is inferred from the average transaction size of 500 bytes, while the upper bound is based on an unusually small transactions size of 250 bytes. The second scenario is based on the average throughput of Bitcoin network, which is based on statistics collected in October 2015, and it resulted to be of 1.57 transactions/sec. They show then a Bitcoin cost breakdown assuming that the entire network contains 5400 full nodes and they evince that is a fallacy to assume that transaction costs necessarily have to be offset by transaction fees. Indeed, the costs of running full nodes may be offset by financial externalities such as selling items whose costs computational time for a node or confirming a transaction without trusting third parties. Said so, is important to enhance that miners are bereft of these two factors and they need to be compensated. According to new measurements on the block propagation time, they also defined **X% effective throughput** as follows:

$$\text{X\% effective throughput} = Q/(\text{X\% block propagation delay})$$

Considering that the propagation delay is, for 1 MB block size and X% = 90% of block propagation, 2.4 minutes, they calculated an X% effective throughput of 55 Kbps ≡ 26 tx/sec having X% = 90% of block propagation.

Throughput limit

They observed that the block size Q, and interval $\mathcal{T}$, must satisfy:

$$\frac{Q}{\text{X\% effective throughput}} < \mathcal{T}$$

having in that way, for a 10 minutes block interval a block size that should not exceed 4 MB for X = 90% and 38 MB for X = 50%. Given $\mathcal{T}$ = 10 minutes the

block size should not exceed 4 MB, corresponding to a throughput of at most 27 transactions/sec.

Latency limit

To improve the system's latency it could be enough to reduce the block interval. To maintain effective throughput that would also require a reduction in the block size. Propagating a block smaller than 80 KB would not make full use of the network's bandwidth, as latency would still be a significant factor in the block's propagation time. To propagate a 80 KB block to 90% of the nodes would take roughly 12 seconds. In conclusion, to retain at least 90% effective throughput and fully utilize the bandwidth of the network, the block interval should not be smaller than 12 seconds.

2.3.3 Results

More difficult to measure metrics could also reveal scaling limitations, example *fairness*. Their measurement results suggest that top 10 % nodes receive a 1 MB block 2.4 minutes earlier than the bottom 10 %, meaning that some miners could obtain a significant lead over others solving hash puzzles. In the end, they want to rethink the design of a scalable blockchain, organizing it around a decomposition of the Bitcoin system into a set of abstraction layers that they called *planes*. In a hierarchical of dependency from bottom to the top the layers are:

1. **Network Plane.** It propagates transaction messages. Bitcoin's network protocol do not fully utilize underlying network bandwidth, making Bitcoin's Network Plane the bottleneck in transaction processing. A solution could be to avoid denial-of-service by propagation of invalid transactions, a node must fully receive and validate a transaction before further propagations.
2. **Consensus Plane.** Functionality that mines blocks and reaches consensus on their integration in the blockchain. It receives messages from Network Plane and outputs transactions ready to be insert in the system ledger. Bitcoin's blockchain protocol has a three-way-tradeoff between, *consensus speed*, *bandwidth* and *security*, and the increment of two of them leads to the loss of the third. For example, if the first two are improved then there is loss in the mining power that secures the system.
3. **Storage Plane.** It functions as a global memory that stores and provides availability for authenticated data produced by the Consensus Plane. Storage Plane in Bitocin only supports `writes` operations that append data and doesn't support *delete* operations. The only supported *read*

operation downloads the entire ledger, a process that require four days. The community has proposed ideas such as Unspent Transaction Output (UTXO) data structure.

4. **View Plane.** A view is a data structure derived from the full ledger whose state is obtained by applying all transactions. For Bitcoin miners, it is unnecessary to operate on the full ledger that stores the entire transaction history. Miners and nodes in Bitcoin locally compute and operate on a view of the ledger called UTXO set, which specifies the current balance of all entities in the system. Bitcoin requires all consensus nodes to verify all transactions, and based on the result of the computation, every node needs to update its view, e.g. UTXO sets, locally and generating then the same conclusion of all other nodes in the system, representing in that way an honest set of consensus nodes, keeping an high availability.
5. **Side Plane.** Allows off-the-main-chain consensus.

2.4 Assumptions

The main problems that we should focus on are related to blockchain scalability, performance and see whether the constant increasing amount of transactions per day affects the way miners include transactions or the way clients offer fees to miners. We assume then, having information from the previous listed papers, that analyzing the last 3 years of transactions might produce interesting results about how this (not anymore) new but still cryptic system is evolving and changing its properties according to achieve better performance even after it scaled drastically in the last couple of years and still be able to compensate miners and users.

3

Technical Background

This chapter wants to introduce the reader in blockchain systems, particularly the Bitcoin one. We show a list of symbols which might be useful for reading in Appendix B. However, we won't go in deep details explaining Bitcoin or its terminology and we refer to already existing high-level [36] or technical [64, 26, 9, 56, 55, 16] descriptions, but we will explain the concept of payments and fee in Bitcoin system, how miners profit from mining, how a user could choose whether including fee or not in its transactions and how performance and scalability is changing the way miners select transactions. We will also talk about some machine learning techniques and data structures.

Should be clear, before reading, the differences between *centralized* and *decentralized* digital currencies. These differences are summarized in Table 3.1. As mentioned in Chapter 1, more decentralized digital currencies are emerging nowadays, and they aim to facilitate transactions between consenting individuals who would otherwise have no means to trust each other and deal with geographical separation and interfacing difficulties. According to Nakamoto [55] an electronic coin is defined as follows:

The Definition 1. An *eletronic coin* is a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin.

Transactions are registered in *blocks* and all these systems use the *blockchain*

Table 3.1: Centralized vs Decentralized digital currencies

Centralized	**Decentralized**
Under the control of a central authority	Requires consensous among users to make changes
Users are dependent from the control given from the Certificate Authority (CA)	Users have full control over their assets/data
No need to enforce cryptography or to create keys	User's data is uniquely identified by private key
Only users allowed by CA can participate	Anyone can join and use ledger
Historic transactions can be changed by the CA	Historic transactions are unalterable and permanent
Can be run very efficiently using relational databases that fit the need of the applications	Inefficient for the cost of mining

technology to maintain a serial order of them. The blockchain is a consensus protocol that users run to maintain and secure a shared ledger of data and it is formed of all blocks, ordered by time and connected between each other with the *previous block* attribute. A transaction is approved only when it is included in a block, and the miner including this transactions has first solved the *proof-of-work*. Difficulty of proof-of-work is increased according to keep an average $\mathcal{T}$ of 10 minutes.

3.1 Bitcoin & Fees

We talked about miners and that they mine blocks including transactions in them. Every miner has a *mempool*, $\mathcal{N}$, containing the new, unapproved, n transactions. Miners are free to choose whether to accept or refuse a certain transaction $t \in \mathcal{N}$. Competitive miners will include transactions as long as the fee exceeds the marginal cost of inclusion. Production costs are fixed per block (but may vary between miners depending on access to technology and energy cooling) and the protocol defines a maximum block size Q. Because of that, the marginal cost of inclusion is zero if there are fewer unconfirmed transactions than the capacity left in the block. Competitive miners make positive expected profits only if transactions compete for space in the blockchain. Houy [47] argues that a maximum block size is necessary for the stability of Bitcoin. However in that way big mining pool might take advantage from less competition, having a centralization problem, since a transaction is willing to

Table 3.2: Possible Relations Between Major Attributes

	Q	$\mathcal{T}$	τ	γ	t_f	$\langle C \rangle$	$\mathbb{P}_{orphan}$	t_l
$Q \uparrow$			$\uparrow\uparrow$	$\uparrow$	$(\downarrow)$	$\uparrow$	$\uparrow\uparrow$	$(\downarrow)$
$\mathcal{T} \uparrow$	$(\uparrow)$			$\downarrow$				$\uparrow$
$\tau \uparrow$	$\uparrow\uparrow$			$\downarrow$	$(\downarrow)$	$(\uparrow)$	$\uparrow\uparrow$	$\uparrow$
$\gamma \uparrow$	$\uparrow$	$\downarrow$			$(\downarrow)$	$\uparrow$	$(\uparrow)$	$\downarrow$
$t_f \downarrow$			$(\uparrow)$			$\uparrow$		$\uparrow$
$\langle C \rangle \downarrow$	$\downarrow$				$\uparrow$			$(\downarrow)$
$\mathbb{P}_{orphan} \downarrow$	$\downarrow$					$\downarrow$		

[1] $\uparrow\uparrow/\downarrow\downarrow$: more than linear or even exponential increase/decrease
[2] $\uparrow/\downarrow$: increase/decrease.
[3] $(\uparrow)/(\downarrow)$: might increase/decrease.

compete only if the capacity is reached. In that way if the system never reaches the capacity, raising the fee won't be helpful at all. Before 2015 transactions almost never required to compete for space in blocks, since the demand was less then the offer, but nowadays we have a totally different scenario. In 2012, the number of transactions confirmed per day reached peaks of 50,000 transactions, with an average of 15,000 transactions per day [9]. Nowadays we have an average of 250,000 transactions approved per day, with peaks of 350,000. Since the throughput limitations in the system, due to the block size limit Q and the average interval $\mathcal{T}$ set at 10 minutes and with such big increment of transactions to approve every day, miners could start to choose transactions with an higher ρ or simply transactions that are willing to offer an higher fee rather than zero-fee transactions. Adding the fact that the reward R has a 50% reduction every 210,000 blocks creates a market mechanism to find the price of Bitcoin transactions.

By reading previous papers, we made some assumptions of how the following properties of the Bitcoin systems may vary if other properties are changing as well. We tried to classify the main properties of the system and make assumptions on how they may vary. This attributes are Q, $\mathcal{T}$, τ, γ, t_f, $\langle C \rangle$, $\mathbb{P}_{orphan}$ and t_l and our assumptions are summarized in Table 3.2. For each property, its row is telling what happens to other attributes if the attribute considered is increased/decreased. White space means that there might be no relation or we don't have any assumption yet. For example, if we are going to increase the block space Q, then we have a relevant increase on the propagation time, τ, the chances of orphaning are way bigger, so will be the cost to mine a block, but we have a better throughput, γ and we might have less fee from transactions, t_f, since they will have less competition to be included in a block.

3.2 Machine Learning

Once a new node first joins the Bitcoin network, it has to download all the useful information to process transactions and to verify them. These information consist in a huge quantity of data, nowadays ~ 125 GB, and it grows over time. A full node might require more than four days to downloads the entire ledger. It is obvious to think then that once you collected all the data, a proper way of analyzing them is necessary. We want to explain some of the most common and fundamental techniques of machine learning which were useful for our purpose of analyzing the Bitcoin blockchain. These information were collected from Bowles' book [34]. It is very important, to be able to verify our assumptions made in Table 3.2, to check how much a pair of attributes will be related to each other. We measure that by using *Pearson's correlation*.

3.2.1 Pearson's Correlation

The degree of correlation between two attributes can be quantified using Pearson's correlation coefficient. Pearson's correlation coefficient is defined for two equal length vectors u and v:

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \qquad v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \tag{3.1}$$

First subtract the mean value of u from all the elements of u, and the same with v, then we generate Δu and Δv as follows:

$$\overline{u} = avg(u)$$

$$\overline{v} = avg(v)$$

$$\Delta u = \begin{bmatrix} u_1 - \overline{u} \\ u_2 - \overline{u} \\ \vdots \\ u_n - \overline{u} \end{bmatrix} \qquad \Delta v = \begin{bmatrix} v_1 - \overline{v} \\ v_2 - \overline{v} \\ \vdots \\ v_n - \overline{v} \end{bmatrix} \tag{3.2}$$

The Pearson's correlation between u and v is defined as follows in Equation 3.3:

$$corr(u, v) = \frac{\Delta u^T * \Delta v}{\sqrt{(\Delta u^T * \Delta u) * (\Delta v^T * \Delta v)}} \tag{3.3}$$

3.2.2 Training a Dataset

Having a huge dataset implies having a big amount of data with different type of attributes. This attributes might have *numerical* or *categorical* values.

Numerical: Data that can be measured, quantified. For example the fee paid to a miner in Bitcoin Currency (BTC).

Categorical: Data which represent characteristics. For example the name of the miners in the Bitcoin system.

Training and predicting models involves different types of variables:

Target variables: the variable that you are attempting to predict, represented with Y;

Predictors: variables that you can use to make the prediction, represented with X.

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{bmatrix} \qquad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \tag{3.4}$$

When the targets are real numbers, the problem of training those data is called *regression problem*. If the targets are two-valued the problem is called *binary classification problem*. We need to predict then every $y_i \in Y$ using every row $x_i \in X$ and evaluate the performance of our predictions. Good performance means using the attributes x_i to generate a prediction that is *close* to y_i. For a regression problem where y_i is a real number, performance is measured in terms like the mean squared error (MSE) (Equation 3.5) or the mean absolute error (MAE) (Equation 3.6) [34].

$$\text{MSE} = \frac{1}{m} \sum_{i=1}^{m} (y_i - pred(x_i))^2 \tag{3.5}$$

$$\text{MAE} = \frac{1}{m} \sum_{i=1}^{m} |(y_i - pred(x_i))| \tag{3.6}$$

However, since MSE is in squared units, the Root Mean Square Error (RMSE) is usually a more usable number to calculate. If the problem is a classification problem, then other measure of performance must be used. One of the most used is the *misclassification error*. Classification problems generally revolve around misclassification error rates and, usually, algorithms for such kind of problems can present predictions in the form of a probability rate for the attributes rather than an attribute itself.

3.2.3 Visualizing Data

When analyzing big datasets, finding a smooth and nice way of representing information of them might be tricky. For example, if you have millions or billions of samples it will result almost impossible to get any information by simply plotting these data. When analyzing longitudinal data for example, information might be showed daily, monthly or even yearly-wise, then the *mean* for every portion is applied. However, bigger is the dataset and most likely it will contain *outliers*. In this case, calculating the mean on those data might be misleading, that is why in a longitudinal study should also be considered whether to apply a *median* value for every portion rather than the mean, since the mean is particularly susceptible to the influence of outliers.

4

Blockchain Analytics System

In order to understand digital cryptocurrencies, in particular the Bitcoin ones, to test and make experiments on the blockchain, a complete data analytic system was designed and implemented. In our thesis, thanks to this system, we were able to run a longitudinal study on the Bitcoin blcokchain, collecting useful information from 2014 to 2017, analyzing more than 100 millions of transactions corresponding to the 50% of the entire blockchain in between that period. This chapter will explain how our studies on the Bitcoin blockchain took place, clarifying how we analyzed the APIs from `blockchain.info` to get the best out of them and illustrating how we collected and manipulated our data. Then we describe how we decided to plot our data and why. In Chapter 5 again we point out our solutions compared to our assumptions and our conclusions.

4.1 Blockchain Data Sources

Many websites like `tradeblock.com` [28] or `blockchain.info` [9] are observing the Bitcoin blockchain every day. These websites provide also remote API that allows you to retrieve data directly from the website and they can be used in a Python application as a HTTP request/response. Furthermore, websites

like `coinbase.com` provide useful information about the money exchange price and they arrange API along with libraries like the one we used called *forex-python* [18]. One of the most popular system that could be used to make analysis on Bitcoin is its client, *Bitcoin Core* [3]. However, its gigantic quantity of data to download before you even get started might be disheartening, since you first have to download the whole raw data of Bitcoin blockchain, which now is about 125 GB with in total about 250 GB of space needed. Since we wanted to optimize and be able to select data we wished to analyze and we didn't have the need to run a full node, we developed a system for blockchain retrieval and analysis. This system allows to fetch a portion of the blockchain, it stores data in a pandas data frame [22] in a way to save up to $x10$ space on disk but still storing all the information we need and finally, it displays all the knowledge acquired analyzing data and applying machine learning technique on those, using Python libraries such as *matplotlib* [20] and *seaborn* [63]. Our blockchain analytic system was implemented on a MacBook Pro with MacOS Sierra *v*10.12.6, with a 2.8 GHz Intel Core i7 processor and 16 GB 1600 MHz DDR3 of RAM, using JetBrains PyCharm (*v*2017.1.4) software as text editor and *Python v*2.7.12 as programming language. Even though we know Python is not common for its computational speed, it provides nice access to the Bitcoin blockchain by using APIs arranged by `blockchain.info`, allowing the retrieval of all the information stored in a block and transaction, plus its enormous amount of libraries grants a vast flexibility on which machine learning algorithms to apply or whatever plotting system to use.

4.1.1 Data Retrieval

Bitcoin Core takes too much disk space and uses resources that we won't need and we will never use. Because of that we decided to implement a simple system for data retrieval. With this we only save the most relevant data for our purpose and create a data structure which is easy to read and to extract the information from it. The system imports Bitcoin APIs and uses them for data retrieval. The website `blockchain.info` also provides APIs to be used remotely with an HTTP request, returning JavaScript Object Notation (JSON) data. The website `blockchain.info` is monitoring 24-7 the blockchain, producing graphs and statistical analysis on data present in the actual Bitcoin network. The local application is monitoring these data as well, producing graphs that are not taken into consideration from this website, using a finer granularity that represents the data, allowing us to an in-depth analysis on those. For our analysis we select a time range from #October 2013 to September 2017, considering more than 120 million of transactions and 100 thousands of blocks. Earlier than 2013, analysis on the system were already performed and evaluated [38, 47, 54, 56] plus the popularity of the system before 2011 was low and interpreting this early data would be not useful to understand system behavior.

Table 4.1: Data sources and information gathered

Source	Entity	Information
Blockchain.info APIs	Block, Transaction	B_{ha}, t_{ha}, B_h, B_h, $\mathcal{T}$, Q, $t_{in}, t_{ou}, t_q, B_{epoch}, t_{epoch}$
Blockchain.info HTTP parsing	Miner	B_{mi}
Coindesk.com	Price	USD and Ƀ value
Data frame δ	Block, Transaction, Miner, Price	τ, t_f, t_l, γ, ρ, $t_{\%}$

To be able to study and have a general idea of the entire Bitcoin network, we combine data from different sources. As Table 4.1 shows, we use both, `blockchain.info` APIs and HTTP parsing on it to gather information about blocks, transactions and miners, plus we used `coindesk.com` APIs [7, 18] to have information about the pricing of USD and Ƀ. Even though APIs from `blockchain.info` allow us to retrieve almost every information we need, knowledge regarding miners is not included in the blockchain, then we need to get these information by parsing web pages on `blockchain.info` as already mentioned earlier. Finally, other useful information such as γ or t_f are obtained using our data structure (Chapter 4.1.2), containing all the other information put together. During our studies we had different ideas of how data could be nicely retrieved and below are listed the most used for this thesis.

First idea, Blockchain Considered Seen as Object

Data retrieval has been one of our first concern according of being able to represent significant information about the Bitcoin blockchain. Because of the relevant size and its fast growth, we considered the blockchain like an object and we thought that analyzing different parts during time following a certain pattern would give us an accurate overall of the whole system. We started retrieving data by dividing the blockchain in k portions. Let's suppose that m is the last block's height, then we divide all in k portions, having $p = m/k$. While retrieving b blocks, they are retrieved and stored in the following way:

$$\langle 1 \dots b \rangle, \langle p \dots p + b \rangle, \langle 2p \dots 2p + b \rangle, \dots, \langle kp \dots kp + b \rangle.$$

Retrieving blocks in such portions gives statistics representative of the whole system. Like Figure 4.1 shows, every new b blocks added to the blockchain will be added respectively to the indexes:

$$ip + b \ \Big\{ \text{for } i = 0 \text{ to } k. \tag{4.1}$$

Retrieving data in this way gave us a big space saving and still relevant infor-

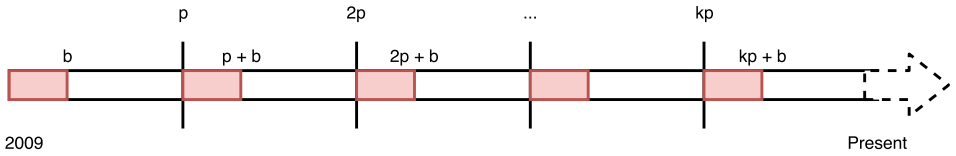


Figure 4.1: Science of Bitcoin blockchain, the entire blockchain considered as an object and divided in k portions. One of the first attempts of data retrieval on Bitcoin blokchain.

mation about the whole system. However, data collected using this methods weren't enough for our purpose of longitudinal study and analysis, but this technique gave us a spark for the next data retrieval attempt.

Fragmented Blockchain

In the other idea, to collect data we use a way smaller interval between each portion retrieved, called also jump or J. Plus we don't retrieve the earliest part of the blockchain, the one from 2009 to 2013. With retrieval of 10 blocks per time, according to the granularity of J we can have a more or less precise analysis. To balance disk space and precision we choose a $J = 10$ blocks and a fixed $b = 10$ of blocks retrieved each time, which means that every 10 blocks retrieved there is a jump of other 10 blocks, ~50% of the entire blockchain but analyzed daily. Figure 4.2 shows how our last data fetching has been implemented, so δ is generated accordingly, collecting transactions for ~100 min then ~100 min of gap. By doing that we manage to collect relevant information from 2013 to 2017 by only using ~25 GB of disk space. Then, the pattern we followed to

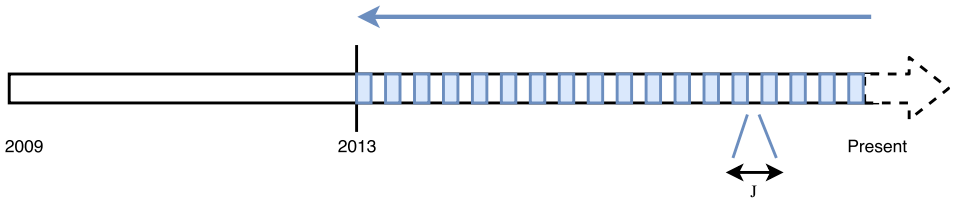


Figure 4.2: Fragmented blockchain divided in smaller portions with a jump of J blocks.

store transactions it is the following:

$$\langle m \dots m-b\rangle, \langle m-2b \dots m-3b\rangle, , \dots, \langle m-2kb \dots m-b(2k-1)\rangle.$$

Having m the height of the last block retrieved and k number of portions.

4.1.2 Data Organization

The Pandas package makes it possible to read data into a specialized data structure called *data frame* [22]. As Michael Bowles [34] says, you can think of a data frame as a table or matrix-like structure oriented with a row representing a single case or observation and columns representing particular attributes. We decided to use a data frame structure and not a matrix one becuase even though the data frame structure is matrix-like, its elements in various columns may be of different types. For statistical problems, the matrix is too confining because statistical samples typically have a mix of different types, plus, Pandas package makes it possible to automate the steps of calculating *mean*, *variance*, *median*, *group-by* elements and it offers a nice and smooth control over your data.

Finally, once data are retrieved and collected, they are saved in a data frame structure that we call δ. Derived data such as γ or ρ, useful for the blockchain analysis, are obtained using δ. This data structure is divided in multiple files with an overall size of ~25 GB and it contains both numerical and categorical variables, it saves an average of $x5$ space on disk, reaching peaks of $x10$ for some blocks. In δ information are stored in a bi-dimensional table, with transaction samples represented in rows and their attributes in columns. If we consider X our attributes set and Y our samples set of transactions, then we have

$$X = \{a_1, a_2, \ldots, a_m\}$$

and

$$Y = \{t_1, t_2, \ldots, t_n\}.$$

Our data frame δ will be then a table with $|\delta| = (m * n)$ and structured in the following way:

$$\delta = \begin{bmatrix} t_{1a_1} & t_{1a_2} & \ldots & t_{1a_m} \\ t_{21} & t_{22} & \ldots & t_{2a_m} \\ \vdots & \vdots & \ddots & \vdots \\ t_{na_1} & t_{na_2} & \ldots & t_{na_m} \end{bmatrix} \tag{4.2}$$

Once defined the structure for our dataset δ, we consider a set X having the following attributes (Appendix B):

$$X = \{t_{ha}, t_{in}, t_{ou}, t_f, t_q, t_{\%}, t_l, Q, \mathcal{T}, B_h, B_{epoch}, B_t, B_{ha}, B_{mi}\}.$$

We structure the way our data are stored for doing subsequential sampling. This is the reason why we stored transactions ordered by block and each block ordered by height/epoch, in that way we have the data set ordered by block creation with all its transactions in it and it will be much easier to make analysis or sampling data following a chronological order. Summarizing, our data frame allows us to collect all the useful information without storing the whole raw

blockchain like current systems for blockchain analysis do, see *BitcoinCore* [3]. Raw data are collected in JSON format, analyzed, printed in our data frame and then deleted. This gave us up to $x10$ space saving on the blockchain, storing in a 1 GB data frame what the raw files from `blockchain.info` store in 10 GB space. Plus, we believe that analyzing a small percentage of the blockchain everyday would give us relevant information about the whole system. Due to time limitations we could collect only 50% of the data, every day.

4.1.3 Data Visualization

Once data are retrieved and saved in our data structure δ, is finally possible to get the information we need out of them. Even though Pandas offers a pretty plotting, visualizing big data might be time consuming and, if the right information are not considered, poorly useful. Once data are ready to be analyzed, the first step is to determine the outliers. A spontaneous question that came up in our minds is: *how to deal with outliers?* We could segregate them out and train on them as a separate class or easier, instead of calculating the mean value, which is very sensitive to outliers, we calculate the median value for our data. Dealing with categorical attributes instead, as the number of attributes grows, the complexity of handle them mounts, also, most of binary tree algorithms, which are the basis for ensamble methods, have a cutoff on how many categories they can manage [34]. Random Forest package written by Breiman and Cutler has a cutoff of 32 categories [35]. Our only categorical variable is B_{mi} so we didn't deep in the analysis on categorical attributes.

To visualize our data we use Pandas libraries as well as *matplotlib* and *seaborn* [22, 20, 63]. We mostly use seaborn to plot data considering our categorical values since it offers nice methods for this kind of analysis. For example, we use it to calculate how major mining pools change their number of transactions approved over time, or to calculated the ρ for every mining pool. We also use it to calculate the linear regression and the Pearson's coefficient for every attribute $X \in \delta$. Pandas was useful to use area plots and pie plots on attributes, so wherever was necessary to manipulate our data frame δ even further.

One example of data manipulation and visualization on our dataset δ regards the calculation and classification over time of the fee density ρ. From δ we collected data we needed and we generated another data frame δ'. In that way we have a new data frame with a new set of attributes X', generated accordingly to the transformation we need, that for convenience we call $\overset{\lambda}{\rightarrow}$, having then

$$\delta \overset{\lambda}{\rightarrow} \delta',$$

$$X \xrightarrow{\lambda} X'.$$

So the transformation $\xrightarrow{\lambda}$ brings a new set of attributes with new data in it. This new X' contains a set of numerical values categorized, regarding ρ, plus the related date and the total number of transactions approved in the following way:

$$X' = \{0, <50, <100, <200, <300, >300, \text{date}, \text{total}\}.$$

Another use of data manipulation for visualization is the one we apply to the epoch. We want, accordingly to make our data more readable, to transform our epoch in date time and then get only the information we need regarding the day, the month or the year. By doing that we have to apply a function to our data frame δ that hits one attribute (in this case the epoch, B_{epoch}), converts it in date time and then parses it to get information about the day, month or year. The listing is showed in Appendix C.1.

4.2 System Architecture

We had an hint of the system architecture in Chapter 4.1.1. The Figure 4.3 shows how the informations are collected and stored in the network and in our blockchain analytic system. From the Bitcoin network data are collected and stored in databases located on different websites, eg. `blockchain.info` and `coindesk.com`. We use then their web services, APIs and HTTP parsing to get these information, create our local data structure and generate more knowledge out of it.

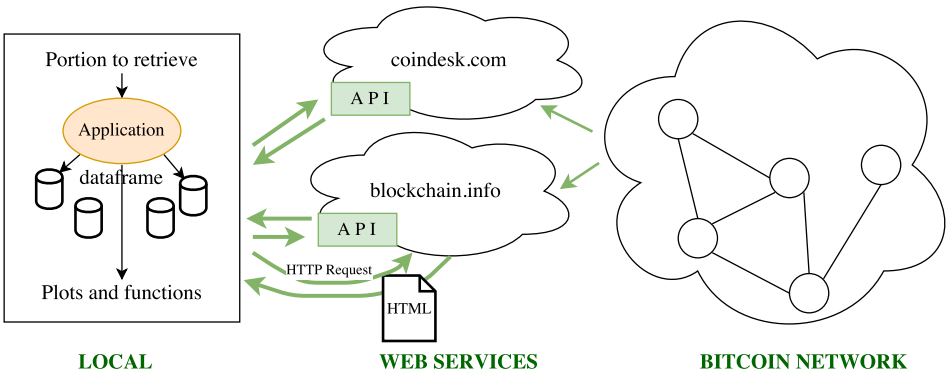


Figure 4.3: Overall architecture of the blockchain analytics system, considering how information and data travel in the network.

As Figure 4.4 shows, the information retrieval works in a combined way, *block-*

wise and *transaction-wise*, then, information gathered from both sources are written and stored in δ. The analytic system collects information from:

1. **JSON** file sources: files processed block by block and for each block transaction by transaction, then information are added to δ.
2. **HTML** file sources: files obtained via an HTTP request. The parsed information are directly added to δ.

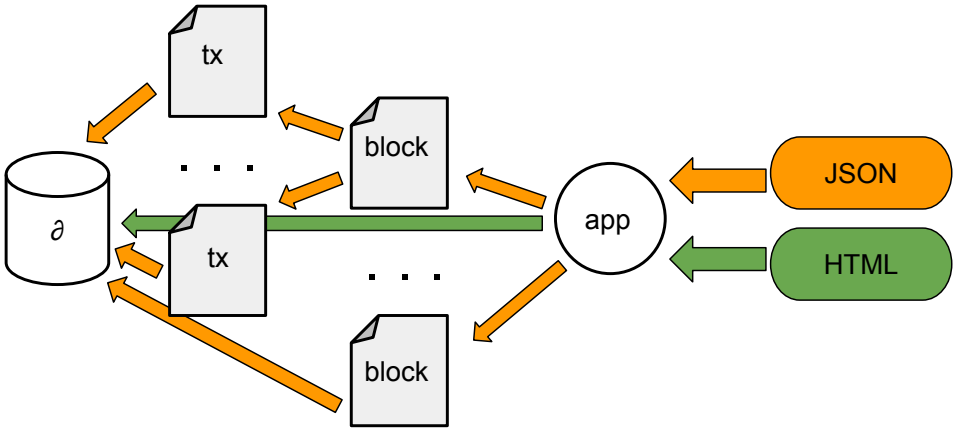


Figure 4.4: How data and informations are gathered in our data frame δ.

4.2.1 Derived Data

In Table 4.1 we talked about derived data from our data frame δ. This section aims to explain how they are calculated and why they are so important. We first focus on calculating the exact amount a transaction has to pay in order to get the payment processed and approved, the transaction fee, t_f. This fee is a difference between the sum of all input t_{in} and the sum of every output t_{ou} of a certain transaction t. In that way, if n is the number of input and m the number of output, t_f is calculated according to Equation 4.3 and measured in BTC (B).

$$t_f = \sum_{i=1}^{n} t_{in_i} - \sum_{i=1}^{m} t_{ou_i}. \tag{4.3}$$

Next, we evaluate for every transaction, the time it takes since its creation, to be visible in the whole network and ready to be spent again. We call it transaction latency, or t_l. This value is calculated following Equation 4.4 and is the difference between two epochs, the first is the block creation time in which a transaction t is approved, the second is the transaction timestamp, so when it first was created. t_l is measured in seconds.

$$t_l = B_{epoch} - t_{epoch}. \tag{4.4}$$

Another relevant derived information using δ is the throughput of the Bitcoin network, or how many transactions it can approve per second. We measure it according Equation 4.5, we call it γ and the unit used is tx/sec.

$$\gamma = \frac{t_B}{\mathcal{T}}. \tag{4.5}$$

After reading the paper from Rizun [56] we state that another important information regards the fee density. This value, that we represent with ρ, could be an important factor for miners to chose whether include or not a transaction in their next block and indicates how many BTC per bytes a transaction t has to offer. Equation 4.6 shows how ρ is calculated.

$$\rho = \frac{t_f}{t_q}. \tag{4.6}$$

We also calculate and insert in δ the percentage of fee paid compared to the total input of a transaction t. This value might be useful to have a reference when the Bitcoin price increase or decrease drastically, so we are able to monitor if the fee is steady or has some changes during time. Then we calculated $t_{\%}$ as showed in Equation 4.7:

$$t_{\%} = 100 - \frac{f_{ou} * 100}{f_{in}}. \tag{4.7}$$

The last derived data is the block propagation time, τ. The calculation for this value follows the papers from Decker and Croman [39, 38]. From 2012 to 2015 block propagation time had an increment. When it was tested from Decker and Wattenhofer in 2012 the median and 90-percentile time for Bitcoin nodes to receive a block was 6.5 seconds and 26 seconds respectively. Considering that at the time of their measurement, the average block size was 87 KB, a full 1 MB block would have taken 5 minutes to be visible from the 90% of the nodes in the network. In 2015, last measurements from Decker and Croman show that the 10%, median, and 90% block propagation times are 0.8 seconds, 8.7 seconds, and 79 seconds respectively. Further, the average block size was at the time roughly 540 KB. Projecting to a 1 MB block size, the 90%, median, and 10% block propagation times would be 2.4 min, 15.7 sec, and 1.5 sec respectively. We consider this last measurement our reference for the block propagation time τ.

5

Blockchain Observations

In this chapter we present and discuss observations of the Bitcoin blockchain captured by the analytic system outlined in Chapter 4. To evaluate and discuss our problem definition in Chapter 1.2, we focus on considerations related to performance, scalability, fees and tolls. A large number of test has been evaluated, more than 100 million of transactions over 100 thousands blocks were collected, stored and analyzed. We state that

5.1 Scalability

Scalability may concern the number of nodes in the network but also the amount of transaction's requests per second that the system faces. The constant growth of the cryptocurrencies' popularity rises not few concerns regarding the scalability of the system. The reward for miners brought a lot of people being interested in mining, and the number of nodes for this purpose highly increased during years.This brought the emerge of *mining pools*, which consists in having a lot of miners connected together to find as fast as possible the solution of the puzzle and share the earned reward afterwards. This creates a centralization problem though, only bigger mining pools nowadays, such as *AntPool* [1], *F2Pool* [17], *BTCC Pool* [10] or *BitFury* [8], will find a solution to mine new blocks, discouraging a lot of small miners to join the network.

The other scalability problem is related to the growing amount of transactions

that need to be approved every day. It is obvious that with an average t_q of 500 bytes, a fixed estimated $\mathcal{T}$ =10 minutes and a maximum block size Q fixed at 1 MB, the outside number of transactions the system can approve per second is roughly 3-4 txs/sec. Compared to systems like VISA that can easily approve 2000 transactions per second, Bitcoin or any other digital cryptocurrency using the blockchain protocol are still far in taking over digital systems such as VISA or Master Card. However we want to analyze this boundaries and understand what it possible to do according to improve the scalability.

5.1.1 Miners & Mining Pools

In our data frame δ are stored all the information regarding every single transaction and which miner approved it. We analyzed and tracked down every active miner in the system from 2013 to 2017. With our 50% of the total information we evaluate a total number of 6137 miners, where 6066 are just occasional nodes not belonging to any mining pool. In Figure 5.1 the number

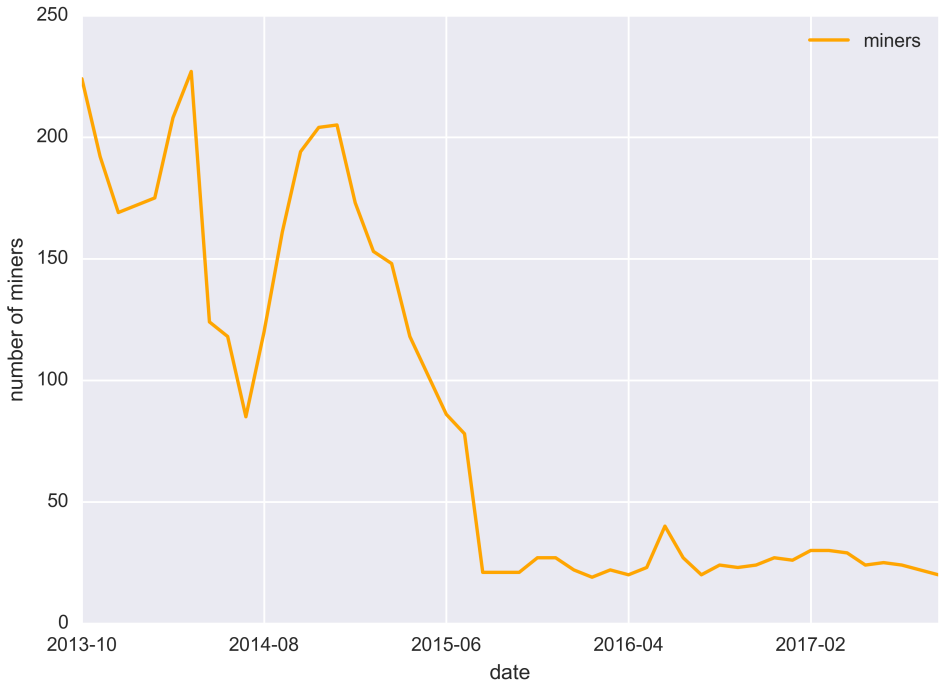


Figure 5.1: Monthly number of active miners from 2013 to 2017.

of monthly active miners from 2013 until now is displayed. Even if the number is relative to the 50% of the blockchain, we analyzed every block mined during time and we state that after December 2014 until August 2015 there was a relevant drop, caused by the coming of mining pools, KnCMiner, AntPool

and BitFury in late 2014, and the massive mining power growth of SlushPool and F2Pool in mid 2015. This is the main cause for occasional miners' drop in the system, and more mining pools will join the network, more it will be disheartening for smaller nodes to start mining, since the costs will be probably higher than the revenue. This creates a sort of centralization problem, since only the bigger mining pools could take part in the whole process of transactions approval. From our analysis we state that there are different mining pools joining and leaving the Bitcoin network during these years. As Figure 5.2

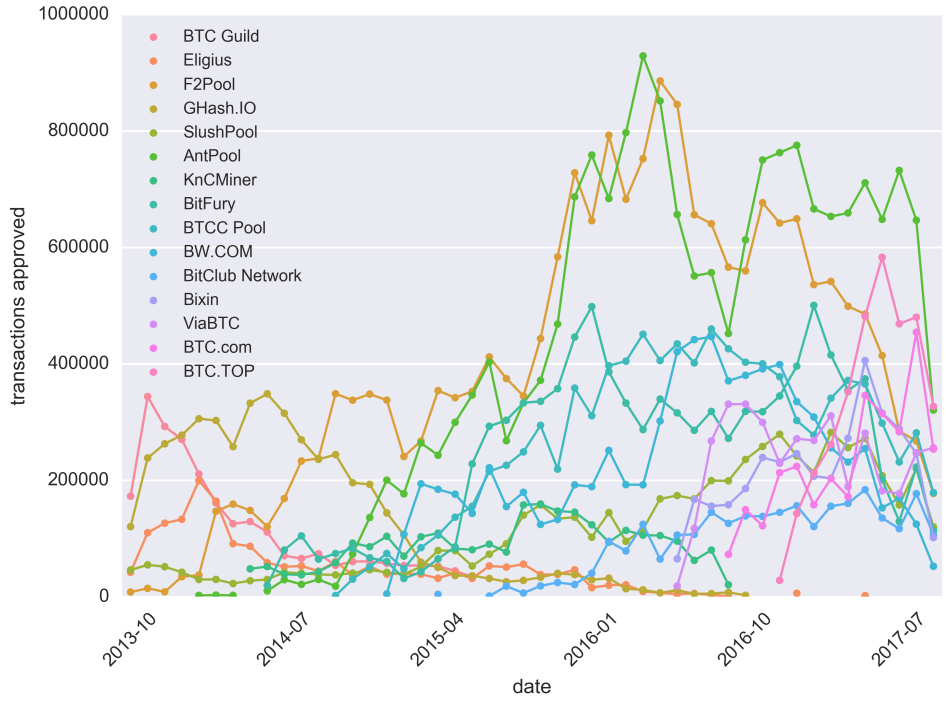


Figure 5.2: Top 15 mining pools from 2013 until 2017. In total, they approve the #% of the total Bitcoin's transactions ever occurred.

shows, the most relevant example is the *GHash.IO* pool which was one of the major mining pool until mid 2015, then its mining power started to decrease slowly until its stop in October 2016. After some research, we refrain to an article from `coindesk.com` [4] which says that Bitcoin miners ditch GHash.IO pool over fears of 51% attack. Indeed in 2014, according to `blockchain.info`, GHash.IO accounted for more than 42% of Bitcoin mining power. Nowadays as the Table 5.2 shows, AntPool has 19.53% of the mining power, and there is no centralization risk. We analyzed then the major mining pools from 2013 until 2017, we divided dates analyzed in two eras, 2013-2015 and 2016-2017, selected the first 10 miners regarding their mining power and then represented in Tables 5.1-5.2. Table 5.1 shows that a big percentage of the mining power comes from occasional miners and for the whole period between 2013 and 2015

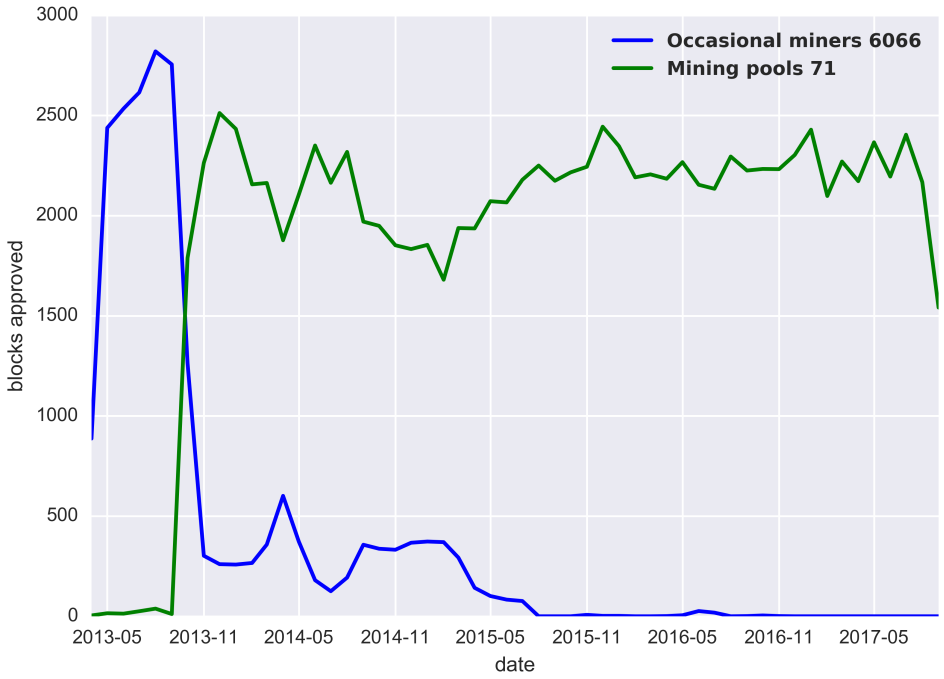


Figure 5.3: Number of transactions approved from occasional miners and mining pools from 2013 until 2017.

GHash.IO has only a 13.3% of mining power. We can see indeed in Figure 5.2 that this mining pool had only a short time peek in between 2013-2015, but anyway in an overall scheme it managed to collect the 13.3% of the whole share. Table 5.2 shows that AntPool, F2Pool and BTCC Pool are dominating the mining power in Bitcoin network from 2016 until now with values respectively of 19.53%, 16.43% and 10.5% of share. If compared with Table 5.1, AntPool is the one which is growing faster and mining pools such as GHash.IO and BTC Guild stopped their activity. We state finally that because of these results, no miner seems to get even close to the GHash.IO share in 2014 and it seems like we won't have a centralization problem in the very near future. However the mining power during the years changed from a distributed architecture to a decentralized one. We can see in Figure 5.3 that the occasional miners almost disappeared in the last two years, leaving their space to mining pools, while they were contributing for almost the 100% at the end of 2013. This brought a downside for individual users which want to buy their own hardware and start mining for themselves. Since Bitcoin always claimed its distribution as a system's strength, this scalability issue might result a centralization problem in the long run, excluding small miners from the system and letting only the biggest mining pools to have informations about the whole blockchain.

Table 5.1: Mining power distribution in a scenario from 2013 until 2015, considering only the 10 major mining pools.

Mining Pool	Mining Power (%)	Blocks Mined
F2Pool	13.53	10,548
GHash.IO	13.3	10,370
BTC Guild	7.37	5,749
AntPool	6.82	5,323
Eligius	4.99	3,889
BitFury	4.73	3,693
BTCC Pool	3.9	3,042
SlushPool	3.37	2,627
KNCMiner	3.12	2,432
BW.COM	2.67	2,064
Total Analyzed	63.8	49,737
Other Miners	36.2	28,225

Table 5.2: Mining power distribution in a scenario from 2016 until 2017, considering only the 10 major mining pools.

Mining Pool	Mining Power (%)	Blocks Mined
AntPool	19.53	9,078
F2Pool	16.43	7,635
BTCC Pool	10.5	4,874
BitFury	8.71	4,050
BW.COM	8	3,717
SlushPool	5.26	2,447
ViaBTC	4.8	2,222
Bixin	4.12	1,947
BTC.TOP	4	1,857
BTC.com	3.55	1,648
Total Analyzed	84.9	39,475
Other Miners	15.1	6,996

6

Conclusions

6.1 Discussion

6.2 Future Implementation

6.3 Comments

show bibliography [55], [64], [30], [42], [52], [40], [16], [24], [53], [57], [46], [48], [26], [44], [62], [51], [2], [12], [33], [13], [59], [31], [58], [9], [14], [15], [25], [28], [27], [50], [29], [20], [38], [43], [45], [21], [5], [6], [56] [11], [60], [41], [47], [19], [32], [54], [22], [63].

References

[1] Antpool, mining pool in bitcoin network. https://www.antpool.com/,.

[2] Bitcoin api's, api-v1-client-python. https://github.com/blockchain/api-v1-client-python.

[3] Bitcoin client application. https://bitcoin.org/en/bitcoin-core/.

[4] Bitcoin miners ditch ghash.io pool over fears of 51% attack. https://www.coindesk.com/bitcoin-miners-ditch-ghash-io-pool-51-attack/.

[5] Bitcoin mining hashing rate. https://blockchain.info/charts/hash-rate.

[6] Bitcoin nodes. https://bitnodes.21.co/.

[7] Bitcoin prices calculated every minute. https://coindesk.com.

[8] The bitfury group, your leading full service blockchain technology company. http://bitfury.com/.

[9] Bitocoin blockchain analytic website. https://blockchain.info.

[10] Btcc mining pool in bitcoin network. https://www.btcc.com/.

[11] Construct a linear, no-fork, best version of the bitcoin blockchain. https://github.com/bitcoin/bitcoin/tree/master/contrib/linearize.

[12] Ethereum api's, pyethereum. https://github.com/ethereum/pyethereum.

[13] Ethereum wiki/patricia tree. https://github.com/ethereum/wiki/wiki/Patricia-Tree.

[14] Ethereum's blockchain analysis website. https://etherscan.io.

[15] Ethereum's blockchain website. https://etherchain.org/.

[16] Ethereum's white paper. https://github.com/ethereum/wiki/wiki/White-Paper.

[17] F2pool, mining pool in bitcoin, ethereum and litecoin network. https://www.f2pool.com/.

[18] Foreign exchange rate and currency conversion, based on coindesk.com. https://pypi.python.org/pypi/forex-python.

[19] How visa make money. https://revenuesandprofits.com/how-visa-makes-money-understanding-visa-business-model/.

[20] Matplotlib for data plotting. https://matplotlib.org.

[21] Mining hardware comparison. https://en.bitcoin.it/wiki/Mining_hardware_comparison.

[22] pandas: Python Data Analysis Library. http://pandas.pydata.org/, 2012.

[23] Satoshi dice - gambling with bitcoin. https://satoshidice.com/rules, 2012.

[24] Ethereum foundation - the solidity contract-oriented programming language. Technical report, https://solidity.readthedocs.io/en/develop/, 2014.

[25] Bitcoin mining process. http://bitcoinminer.com/, 2015.

[26] Bitcoin website – mining. https://www.bitcoinmining.com, 2016.

[27] Ethereum project – website. https://www.ethereum.org, 2016.

[28] tradeblock.com – analysis on the blockchain. https://tradeblock.com, 2016.

[29] Alfred V. Aho and Jeffrey D. Ullman. *Foundations of Computer Science*. Computer Science Press, Inc., New York, NY, USA, 1992.

[30] Adam Back. Hashcash - a denial of service counter-measure. Technical report, 2002.

[31] Paul Baran. *On Distributed Communications: Introduction to Distributed*

Communication Networks. The Rand Corporation, 1964.

[32] Massimo Bartoletti, Andrea Bracciali, Stefano Lande, and Livio Pompianu. A general framework for bitcoin analytics. *CoRR*, abs/1707.01021, 2017.

[33] Georg Becker. *Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis*. 2008.

[34] M. Bowles. *Machine Learning in Python: Essential Techniques for Predictive Analysis*. Wiley, 2015.

[35] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[36] Rainer Böhme, Nicolas Christin, Benjamin Edelman, and Tyler Moore. Bitcoin: Economics, technology, and governance. *Journal of Economic Perspectives*, 29(2):213–38, May 2015.

[37] David Chaum. Blind signatures for untraceable payments. In D. Chaum, R.L. Rivest, and A.T. Sherman, editors, *Advances in Cryptology Proceedings of Crypto 82*, pages 199–203, 1983.

[38] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, Dawn Song, and Roger Wattenhofer. *On Scaling Decentralized Blockchains*, pages 106–125. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[39] C. Decker and R. Wattenhofer. Information propagation in the bitcoin network. In *IEEE P2P 2013 Proceedings*, pages 1–10, Sept 2013.

[40] Kevin Delmolino, Mitchell Arnett, Ahmed Kosba, Andrew Miller, and Elaine Shi. *Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab*, pages 79–94. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[41] Jacob Donnelly. Bitcoin network still backlogged with tens of thousands of unconfirmed transactions, causing delays. *Bitcoin Magazine*, July 2015.

[42] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '92, pages 139–147, London, UK, UK, 1993. Springer-Verlag.

[43] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert van Renesse. Bitcoin-ng: A scalable blockchain protocol. *CoRR*, abs/1510.02037, 2015.

[44] Rui Garcia, Rodrigo Rodrigues, and Nuno Preguiça. Efficient middleware for byzantine fault tolerant database replication. In *Proceedings of the Sixth Conference on Computer Systems*, EuroSys '11, pages 107–122, New York, NY, USA, 2011. ACM.

[45] Begnaud Francis Hildebrand. *Introduction to Numerical Analysis: 2Nd Edition*. Dover Publications, Inc., New York, NY, USA, 1987.

[46] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.

[47] Nicolas Houy. The economics of bitcoin transaction fees. Working Papers 1407, Groupe d'Analyse et de Théorie Economique (GATE), Centre national de la recherche scientifique (CNRS), Université Lyon 2, Ecole Normale Supérieure, 2014.

[48] Håvard D Johansen, Robbert van Renesse, Ymir Vigfusson, and Dag Johansen. Fireflies: A secure and scalable membership and gossip service. *ACM Transactions on Computer Systems (TOCS)*, 33(2):5:1–5:32, May 2015.

[49] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982.

[50] Craig Larman. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition)*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.

[51] Aldelir Fernando Luiz, Lau Cheuk Lung, and Miguel Correia. Mitra: Byzantine fault-tolerant middleware for transaction processing on replicated databases. *SIGMOD Rec.*, 43(1):32–38, May 2014.

[52] Loi Luu, Duc-Hiep Chu, Hrishi Olickel, Prateek Saxena, and Aquinas Hobor. Making smart contracts smarter. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 254–269, New York, NY, USA, 2016. ACM.

[53] Loi Luu, Jason Teutsch, Raghav Kulkarni, and Prateek Saxena. Demystifying incentives in the consensus computer. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 706–719, New York, NY, USA, 2015. ACM.

[54] Malte Möser and Rainer Böhme. *Trends, Tips, Tolls: A Longitudinal Study of*

Bitcoin Transaction Fees, pages 19–33. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[55] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system," http://bitcoin.org/bitcoin.pdf, 2008.

[56] Peter R. Rizun. A transaction fee market exists without a block size limit. Technical report, 2015.

[57] Chaitya B. Shah and Drashti R. Panchal. Secured hash algorithm-1: Review paper. Technical report, Indus Institute of Technology and Engineering, Gujarat Technological University, 2014.

[58] William Stallings. *Cryptography and Network Security: Principles and Practice*. Pearson Education, 3rd edition, 2002.

[59] M. Swan. *Blockchain: Blueprint for a New Economy*. O'Reilly Media, 2015.

[60] Enrico Tedeschi. Paying for bandwidth in blockchain internet applications. Technical report, UiT Arctic University of Norway, 2016.

[61] Sarah Underwood. Blockchain beyond bitcoin. *Commun. ACM*, 59(11):15–17, October 2016.

[62] Ben Vandiver, Hari Balakrishnan, Barbara Liskov, and Samuel Madden. Tolerating Byzantine Faults in Transaction Processing Systems Using Commit Barrier Scheduling. In *ACM SOSP*, Stevenson, WA, October 2007.

[63] Michael Waskom, Olga Botvinnik, drewokane, Paul Hobson, David, Yaroslav Halchenko, Saulius Lukauskas, John B. Cole, Jordi Warmenhoven, Julian de Ruiter, Stephan Hoyer, Jake Vanderplas, Santi Villalba, Gero Kunter, Eric Quintero, Marcel Martin, Alistair Miles, Kyle Meyer, Tom Augspurger, Tal Yarkoni, Pete Bachant, Mike Williams, Constantine Evans, Clark Fitzgerald, Brian, Daniel Wehner, Gregory Hitz, Erik Ziegler, Adel Qalieh, and Antony Lee. seaborn: v0.7.1 (june 2016), June 2016.

[64] Dr. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. Technical report, 2014.

A

Terminology

RLP: Stands for recursive length prefix. It is a serialization method for encoding arbitrary structured binary data (byte arrays).

KEC-256: Another serialization method generating a 256-bit hash.

full node: A full node in a decentralized digital currency peer-2-peer network, is a node that stores and processes the entirety of every block, storing locally the entire size of the blockchain.

light node: A light node in a decentralized digital currency peer-2-peer network, is a node that only stores the part of the blockchain it needs.

satoshi: Unit of the Bitcoin currency. 100,000,000 satoshi are 1 BTC (Bitcoin).

B

List of Symbols

t_B	number of transaction approved in a block B.
t_{in}	transaction input in bitcoin (Ƀ). All the money sent.
t_{ou}	transaction output (Ƀ). All the money received.
t_f	transaction fee (Ƀ) (Equation 4.3).
t_q	transaction size, in bytes.
t_l	commit latency of a single transaction (Equation 4.4).
t_h	transaction height. Height of the block in which the transaction is included.
t_{ha}	transaction hash.

$t_{\%}$	percentage of t_{in} paid in fee, t_f.
$\mathcal{T}$	expected block interval time (~ 10 min)
$\mathbb{P}_{orphan}$	probability that given a block is orphaned.
τ	block solution propagation time, we consider a $\tau = 10$ seconds according to Decker [39].
t_{epoch}	timestamp of a transaction t. Epoch of when t was first seen in the network
η	cost per hash.
$\langle \Pi \rangle$	expectation value of a miner's profit per block.
$\langle V \rangle$	expectation value of a miner's revenue per block.
$\langle C \rangle$	expectation value of a miner's hashing cost per block.
R	block reward, currently at 12.5 Ƀ.
h	miner's individual hash rate.
H	total hash rate of Bitcoin network.
Q	block size or block space in bytes.
Q^*	the block size that maximizes the miner's expected profit.
ρ	fee density, or the price per byte for block space.

M	money, bitcoin (B).
$M_{demand}(b)$	partial sum of the b transaction fees in mempool in order of descending fee density.
$M_{supply}(Q)$	miner's cost due to orphaning to produce a certain block size Q.
$\mathcal{N}$	the set of transactions in a miner's mempool.
n	number of transactions in a miner's mempool.
B	single block.
B_t	transaction root that links to every transaction in a block B.
B_{epoch}	timestamp of a block B. Epoch of when the block was included in the blockchain
B_h	block height.
B_{ha}	block hash.
B_{mi}	miner which mined the block B.
γ	throughput of Bitcoin network, measured in txs/sec.
δ	data frame generated by the analytic system.

C Listing

Listing C.1: Function for data manipulation. It creates a new column ('date'), from another ('B_ep') containing the respective B_{epoch} transformed in date time value with days as granularity.

```
1  def epoch_date_dd(df):
2  # get a df with a column of epoch 'B_ep', returns
3  # another column with the date yyyy-mm-dd so it
4  # orders the date by day
5  # :param df:  dataframe in input
6  # :return:    new dataframe containing the 'date'
       attribute
7
8  df['date'] = df['B_ep'].apply(epoch_datetime)
9  df['date'] = df['date'].apply(revert_date_time)
10
11 df['date'] = df['date'].str.slice(start=0, stop=10)
12 return df
```