Extended Essay:

-Computer Science-

Can crypto-currency replace traditional currency?

- An investigation into the long-term viability of blockchain

Research question: How changing the target hash difficulty affects block frequency?

Date: May 2020

Word Count: 3827

Candidate Code: hxk369

Contents

1. Introduction

1.1 Rational and Explanation of the Question

Blockchain is a relatively new technology and it is still growing and developing. It has gained popularity thanks to the adoption by Crypto-currencies which are currencies that use it to store and transfer digital money in a transparent and secure manner (Explained further in 1.2).

As of September 7th, 2019, there are 2697 crypto-currencies with a Total market cap $271,394,978,593 and Total trade volume $49,631,174,438 [1]. The latest crypto-currency comes courtesy of Facebook whose crypto-currency Libra is set to be released in 2020. It aims to use the decentralized nature of crypto-currencies to allow people without bank accounts, to transfer and store money. China is also looking at developing its own currency using blockchain technology and many other countries are expected to follow.

With this increase in the adoption of blockchain and its short history and no proven track record, it is evident why there is skepticism surrounding its success as the financial infrastructure of the future.

Therefore, this paper aims to test one element of the blockchain technology, used to ensure stability and long-term viability – The Target Hash Difficulty by investigating

How changing the target hash difficulty affects block frequency?

[1] CoinLore, Accessed from https://www.coinlore.com/all_coins on September 7th 2019

How changing the target hash difficulty affects block frequency?

The Difficulty of the Target hash is used to limit the frequency of new blocks being created and added to the chain. Each crypto-currency sets its own block frequency it wishes to sustain and uses the difficulty to keep the block frequency the same, as computing power increases. It is essential to have a stable block frequency as it limits the supply of new bitcoin in circulation which is explained in more detail in part 1.2.

It is important to answer this question as it helps analyze why Bitcoin has a target frequency of 10 blocks per minute and what this means for the long-term success of Bitcoin.

1.2 What is Blockchain, as used in Bitcoin?

Blockchain is a decentralized and distributed database used to maintain a continuously expanding list of records, called blocks [2]. In this research paper the focus is on the application of blockchain technology in crypto-currencies, such as Bitcoin. A crypto-currency uses blockchain technology as "an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way" [2]. It stores and transfer the money of users by recording all new transactions in blocks, which are then added to the blockchain which contains the previous transactions.

[2] A. Shanti Bruyn, August 26, 2017, "Blockchain an introduction", Accessed from https://beta.vu.nl/nl/Images/werkstuk-bruyn_tcm235-862258.pdf on July 24, 2019

How changing the target hash difficulty affects block frequency?

Each block is made up of:

1. A block number
2. The Hash of the previous block
3. The transaction data
4. Timestamp
5. A nonce
6. The hash of the current block

Notice the hash of the previous block is used in the creation of the new block's which creates the "chain".

Visually this is shown in Figure 1.1

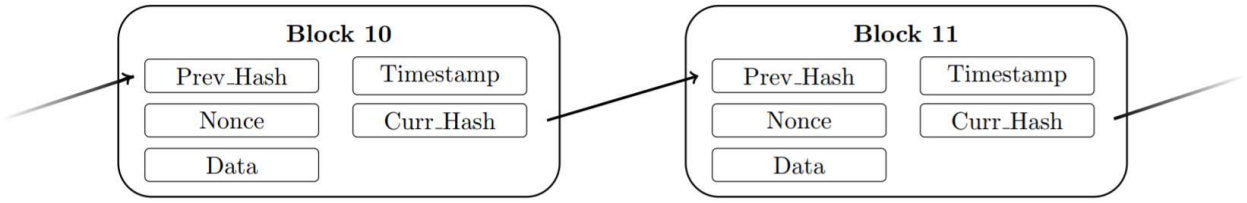


Figure 1.1 Two sequential blocks in a blockchain and the data they convey. Source [1]

The Data of a Block contains 2400 transactions. These include transferring Bitcoin from one person to another, but also the creation of new Bitcoin. The process of creating blocks is further analyzed in the next section.

How changing the target hash difficulty affects block frequency?

1.3 Mining and Proof of Work

Mining refers to the process of creating new blocks in the blockchain. It is the most expensive process as it takes large amounts of processing power from miners to create a block. It can be described as a competition between all miners on the network to solve a complicated mathematical problem and the first one to solve it is rewarded with new Bitcoins. This is the process by which new Bitcoins are added into circulation. The reward was 50BTC in 2009 and has been halved every 4 years. It will continue to decrease until there is 21,000,000BTC in circulation which is the limit set by the protocol. However, this does not mean miners will not receive any reward after that thanks to additional fees. These optional fees allow miners to receive payments from users and in return miners priorities their transaction in the 2400 transaction limit that can be stored in a single block.

This means by mining new blocks, the total number of Bitcoin in circulation also increases. This is why the block frequency is important because it controls the rate at which new Bitcoins are created. This stops Bitcoin from experiencing hyperinflation as there can never be a surge in money supply, making it viable in the long term in theory.

Mining is done by brute force, by changing the nonce – a number added to the block data that completely changes the result of the SHA256 cryptographic function (explained further in 1.4). The aim of mining is to get an output that is less or equal to the current Target Hash(explained in 1.5). For example, if the Target Hash starts with 10 zeros, for example 0000000000101011l…, the output Hash of the SHA256 must also start with the same or a greater number of zeros in order to be accepted.

How changing the target hash difficulty affects block frequency?

Miners continuously use the trial and error method until a valid hash is found. When it is found, the new block is added to the chain, and reward is payed to the miner.

This process is also called Proof of Work. It is used to make the network secure as in theory it makes the network secure as long as 51% of the network is honest. This is because the proof of work makes it necessary to alter all previous blocks in order to create false blocks and this cannot be done without 51% of the network agreeing on the new blockchain.

The following steps explain the process by which mining work as explained by Satoshi Nakamoto the founder of Bitcoin as follows [3] ;

1. New transactions are broadcast to all nodes.
2. Each node collects new transactions into a block.
3. Each node works on finding a difficult proof-of-work for its block.
4. When a node finds a proof-of-work, it broadcasts the block to all nodes.
5. Nodes accept the block only if all transactions in it are valid and not already spent.
6. Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

[3] Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System Accessed from https://bitcoin.org/bitcoin.pdf on July 24, 2019

How changing the target hash difficulty affects block frequency?

1.4 SHA256

SHA256 is in simple terms a cryptographic function that produces a 256-bit hash from a given input. Detailed explanation on this subject is beyond this paper. However, it is important to understand that SHA256 comes from the SHA2 family of cryptographic functions developed by the NSA [4] and has proven to be one of the strongest encryption methods to date. It takes inputs of any length and produces an output of a fixed length. SHA256 is a one-way function, meaning it is impossible to decode the original message from an outputted hash.

1.5 Target Hash

The Target Hash Difficulty is a 256-bit number that is shared by all Bitcoin nodes (users) on the network. As explained earlier in 1.3, generating blocks is similar to a lottery. Miners input a nonce together with the rest of the data into the SHA256 and the algorithm then outputs a 256-bit hash in the range of 0 and 2^{256} number.

The target hash is the number that the SHA256 output is compared to. The output hash must be lower or equal to the Target hash in order to "win the lottery". Therefore, as the target hash changes, in theory, the time taken to mine a new block will change as well the PoW difficulty changes [5].

[4] Wouter Penard and Tim van Werkhoven, On the Secure Hash Algorithm family, Accessed from https://web.archive.org/web/20160330153520/http://www.staff.science.uu.nl/~werkh108/docs/study/Y5_07_08/infocry/project/Cryp08.pdf on July 25, 2019

[5] Bitcoin.org, 15 January 2016, Accessed from en.bitcoin.it/wiki/Target on July 25, 2019.

How changing the target hash difficulty affects block frequency?

In the case of Bitcoin, the Target Hash is adjusted every 2016 blocks. The goal of the network is to mine a block every 10 minutes. This means it should take two weeks to mine 2016 blocks which is when a new adjustment takes place. This limit is created for stability and reducing latency in transactions which has already become a problem with the expanded adoption of Bitcoin and to control the creation of new Bitcoins.

The current target hash of Bitcoin is 9,013,786,945,891 [6] and is estimated to increase by 0.84%.

How is the Difficulty calculated?

The Difficulty adjustment is briefly explained by Satoshi Nakamoto as follows [2];

"To compensate for increasing hardware speed and varying interest in running nodes over time, the proof-of-work difficulty is determined by a moving average targeting an average number of blocks per hour. If they're generated too fast, the difficulty increases."

In other words, the protocol calculates the block frequency (blocks per minute) and uses the average across the previous 2016 blocks. It then adjusts the difficulty to keep the block frequency at 6 blocks per hour which is the goal set by the protocol.

[6] BitMain, Accessed from https://btc.com/stats/diff on July 25, 2019

How changing the target hash difficulty affects block frequency?

2. Experimentation

2.1 Method

In order to test the relationship between the Target Hash and the block frequency, I decided to simulate the process the Bitcoin network goes through when mining. This is done using a script explained in 2.3 that creates a blockchain of 10000 blocks by the same process a crypto-currency creates blocks.

Each blockchain is created at a different Target Hash, starting from 2^{256} and decreasing the Target Hash by multiples of two. In other words, $Target\ Hash = 2^{256-Difficult}$, where the difficulty is set at 0 and increases in increments of 1 up to 20 after every 10000 blocks.

The time it takes for the blockchain to be created at each difficulty is then measured to calculate the average block frequency for that difficulty.

The equation to calculate block frequency is as follows;

$$Block\ Frequency\ (Blocks/second) = \frac{10000\ blocks}{Time\ taken\ in\ seconds} \quad (1.1)$$

How changing the target hash difficulty affects block frequency?

2.2 Variables

Independent Variable: Target hash in range of 2^{236} to 2^{256} in increments of 2^{1}

Dependent Variable: Block frequency in blocks per second

Controlled Variables:	Why?	How?
Hardware	The processing power of the network affects the block frequency as more hashes can be hashed per second	Use the same computer for all experimentation
Blocks per run	Block frequency is measuring the time to mine 10000 blocks at each difficulty then dividing the 10000 by the time taken in second	Keeping the limit in the source code the same
Temperature of hardware	As temperature of the processor increases, the clock speed could decrease resulting in a change of hash power and therefore unfair test.	Running test runs to increase the temperature of the CPU to a stable temperature and using an aggressive cooling curve in the fan settings.
Core used	Each CPU consists of cores made up of silicon, and the slight impurities can affect different cores so each core could have different performance	Lock the script to run on one specific core in the operating system

How changing the target hash difficulty affects block frequency?

2.3 Source Code

In order to explain how the Target hash effects the block frequency, a script by howCodeORG was used. (available on their GitHub repository[7]). This script was forked and modified to measure run times and some other changes were made necessary to insure a fair test. The complete code used is available in Appendix however the key elements are explained in 2.1.

The Figure 2.1 shows the contents of one Block with the previous_hash (which is 0 in the first block) which is the key to linking the blocks in the blockchain and below the figure are all the elements that are added into one string that makes up the input hash of a block, including the nonce.

```
#Create class Block
        class Block:
            blockNo = 0
            data = None
            next = None
            hash = None
            nonce = 0
            previous_hash = 0x0
            timestamp = datetime.datetime.now()

#Initializing a data for class Block
            def __init__(self, data):
                self.data = data

#Initializing a hash with the encrypted string containing all components of a block
            def hash(self):
                h = hashlib.sha256()
                h.update(
                str(self.nonce).encode('utf-8') +
                str(self.data).encode('utf-8') +
                str(self.previous_hash).encode('utf-8') +
                str(self.timestamp).encode('utf-8') +
                str(self.blockNo).encode('utf-8')
                )
                return h.hexdigest()
```

[7] howCodeORG, "Simple-Python-Blockchain", Accessed from https://github.com/howCodeORG/Simple-Python-Blockchain on July 27th, 2019

How changing the target hash difficulty affects block frequency?

Figure 2.1

The figure 2.2 below is the mine() function that uses the variable n as the nonce, hashes the block that is generated using the lowest nonce, compares the resulting hash to the target hash and if the resulting hash is lower or equal to the target, the block is mined and added to the chain. Otherwise the nonce is incremented by one.

```
def mine(self, block):
    for n in range(self.maxNonce):
        guess = int(block.hash(), 16)
        if guess <= self.target:
            self.add(block)
            print(block)
            break
        else:
            block.nonce += 1
```

Figure 2.2

In this blockchain, the target hash is increased as shown below:

```
target = 2 ** (256-difficulty)
```

where the difficulty is increased by 1. This means that an increase in the difficulty decreases the target hash therefore making it harder to find the correct hash as the range of an acceptable output hash is decreased.

This incrementation of the difficulty is done after 10000 blocks.

Figure 2.3 shows how the time is measured and how the block frequency is calculated.

```
start = time.time()
for n in range(0,10000):
    blockchain.mine(Block("Block " + str(n+1)))
end = time.time()
blockfrequency = str(float(10000/(end - start)))
```

How changing the target hash difficulty affects block frequency?

Figure 2.3

The time is measured as the difference between the start time and the time after the last block (Block #10000) is mined. Then the 10000 blocks are divided by the total time in order to calculate the average time per block, block frequency.

3. Results

3.1 Raw Results

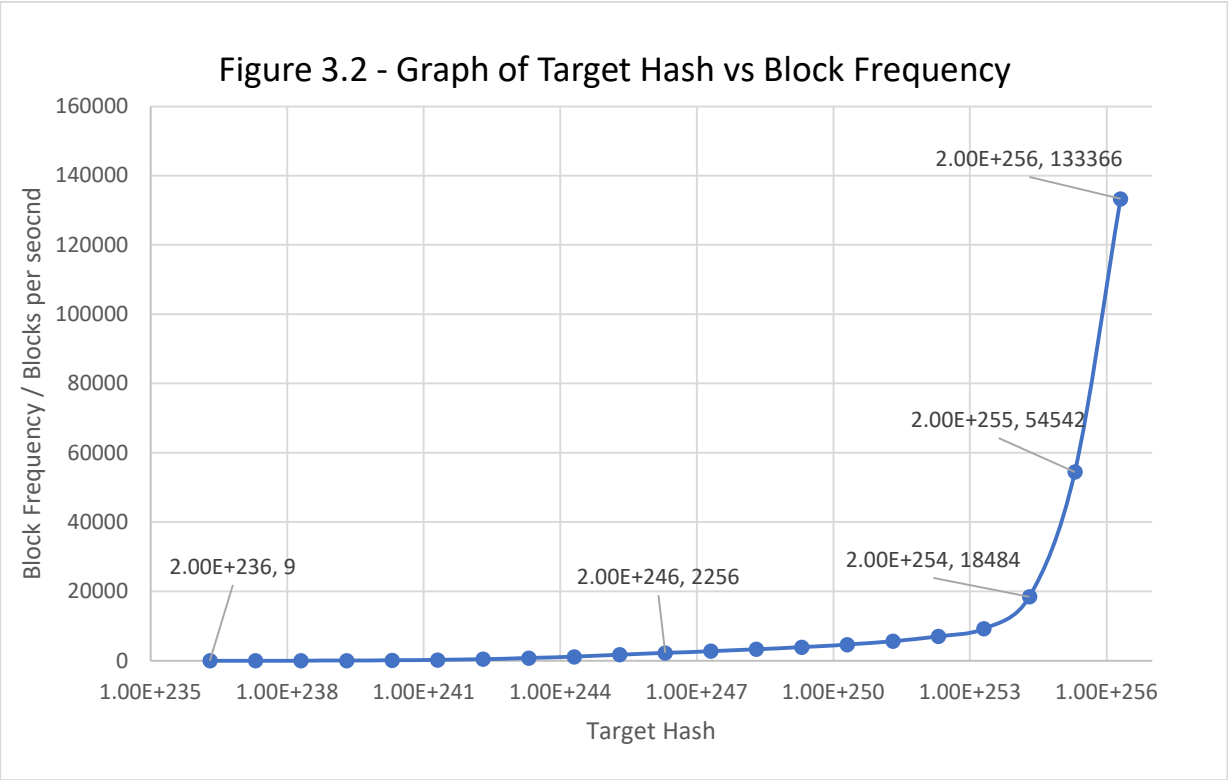
Figure 3.1 – Table of raw results

Difficulty	Target hash	Block Frequency / Blocks per second		Difficulty	Target hash	Block Frequency / Blocks per second
0	2.00E+256	133366		10	2.00E+246	2256
1	2.00E+255	54542		11	2.00E+245	1753
2	2.00E+254	18484		12	2.00E+244	1177
3	2.00E+253	9263		13	2.00E+243	786
4	2.00E+252	7039		14	2.00E+242	490
5	2.00E+251	5648		15	2.00E+241	257
6	2.00E+250	4662		16	2.00E+240	143
7	2.00E+249	3941		17	2.00E+239	78
8	2.00E+248	3328		18	2.00E+238	37
9	2.00E+247	2796		20	2.00E+236	9

These raw results from figure 3.1 are plotted on a graph in Figure 3.2

How changing the target hash difficulty affects block frequency?

3.2 Data representation



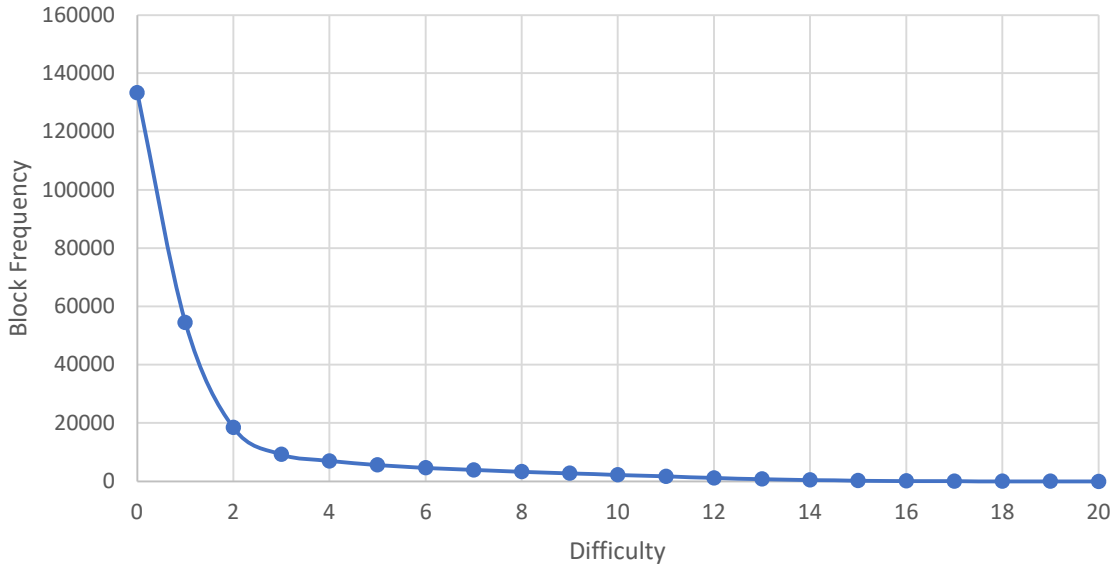
Note that not all data points are shown as they could not all be visible.

There is an obvious casual positive correlation between the Target Hash and the Block frequency.

Given that the Target hash and Difficulty are inversely proportional, we can take the inverse of the Figure 3.2 to show how the change in difficulty affects the block frequency.

How changing the target hash difficulty affects block frequency?

Figure 3.3 - Graph of Difficulty vs Block Frequency



Finally, Figure 3.3 shows us the relationship we were looking for.

It is important to understand that the Difficulty values displayed are just for reference as the Target Hash has a real value, but the difficulty as used in this experiment is defined as decreasing the Target hash in increments of $2^{Difficulty}$ as explained previously in 2.3.

3.3 Analysis of Data

Firstly, we can see that at the lowest difficulty where the Target Hash = 2^{256} the Block Frequency = 133366 blocks/second. This tells us that the total Hash rate of the CPU used is 1.33×10^5 hashes per second and because at that difficulty, all hashes are accepted.

This information allows us to compare our results to the whole Bitcoin network which has a Hash rate of 8.46×10^{19} hashes per second [8] as of September 7th, 2019.

$$\text{Scale factor of experiment } = \frac{8.46 \times 10^{19}}{1.33 \times 10^{5}} = 6.34 \times 10^{14}$$

Therefore, the Bitcoin network is 6.34×10^{14} times greater than our simulated network in terms of hash rate.

This would mean that for example at the Target Hash of 2^{240} our Block Frequency was 143 however the whole Bitcoin network should theoretically have the block frequency of $143\times(6.34\times10^{14}) = 9.07\times10^{16}$ Blocks per second.

Furthermore, by observing figure 3.3, it is clear that as the difficulty was increased (target hash decreased) the block frequency decreased. Therefore, we can reach a conclusion that:

The Block frequency is inversely proportional to the Difficulty.

Moreover, it is visible that the first 3 increases of difficulty had a strong effect on the decrease in block frequency however, after the 3rd increase the effect was drastically decreased and after the 12th increase, the effect was barely visible, although the block frequency was still decreasing slightly.

The mathematical way of conveying the relationship from Figure 3.3 is:

$$Block\ Frequency = \frac{1}{Difficulty} \qquad (2.1)$$

[8] Blockchain.com, Accessed from https://www.blockchain.com/en/charts/hash-rate on the September 7, 2019

How changing the target hash difficulty affects block frequency?

However, in a Bitcoin protocol that does not have a constant Hash rate but rather ever-changing total hash rate we must account for the total hash rate change. This is done by multiplying the block frequency by the new total hash rate as explained in the "scale factor of experiment calculation".

This gives us that the $Block\ Frequency = \frac{Total\ Hash\ Power}{Difficulty}$. (2.2)

Keeping in mind this scale factor it is possible to compare our Figure 3.3 with the relationship shown in Figure 3.4 [1]

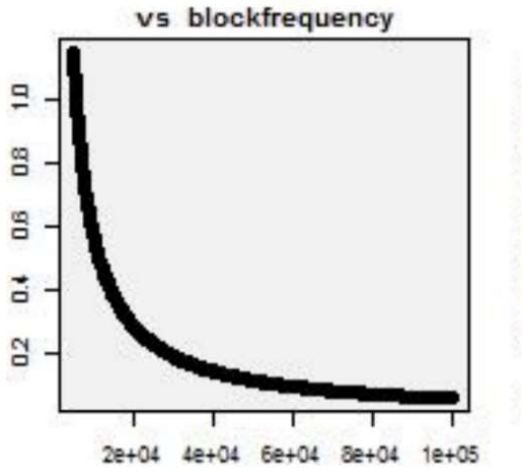


Figure 3.4 Theoretical relationship between Difficulty and block frequency [2]

Our results in Figure 3.3 seem to match the general shape of the curve, however there is a clear difference at the lower difficulties and possible reasons for these differences are discussed in the 4.2 Evaluation.

How changing the target hash difficulty affects block frequency?

4. Conclusion and Evaluation

4.1 Conclusion

Given the results of the experiment it can be said that there is a positive relationship between the Hash Target and Block frequency, seen on Figure 3.2.

Similarly, there is an inverse relationship between the Target hash Difficulty and the Block Frequency in blockchain mining seen on Figure 3.3.

In order to apply this to the Bitcoin protocol let's first look at the state of the protocols different variables.

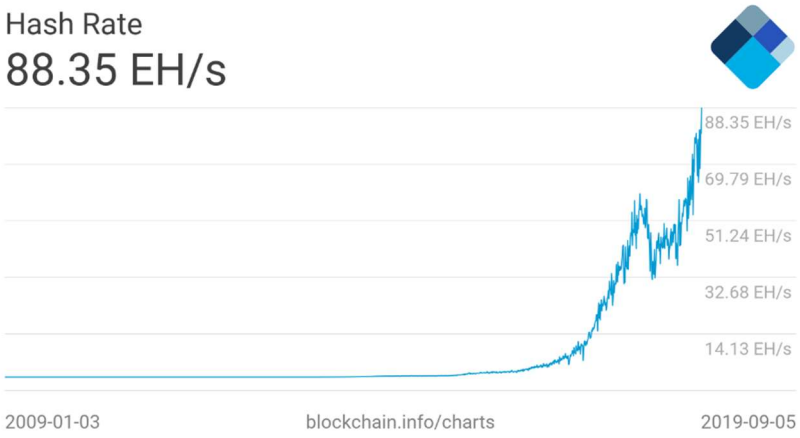


Figure 4.1 Hash rate of Bitcoin

How changing the target hash difficulty affects block frequency?

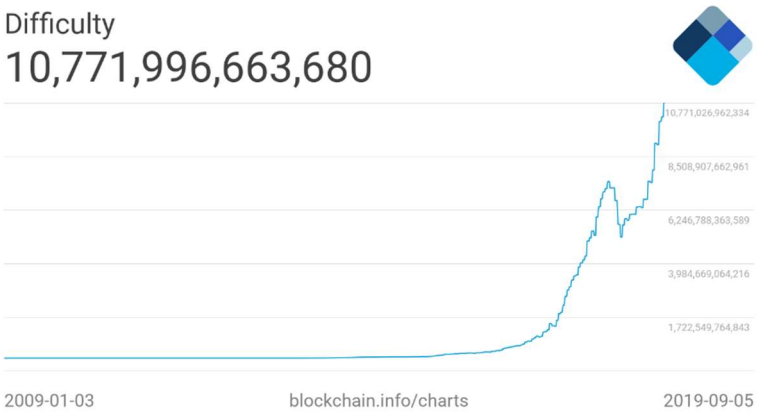


Figure 4.2 Difficulty of Bitcoin

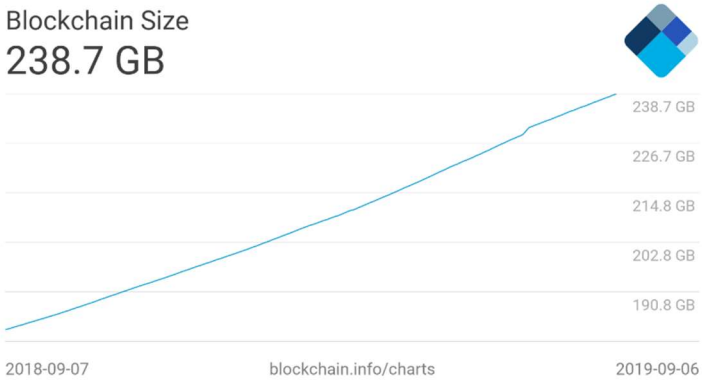


Figure 4.3 Blockchain size of Bitcoin

By analyzing the Figures, we can see that the difficulty has been increasing sharply as a response to the sharp increase in hash rate. Furthermore we can see that the Bitcoin protocol has been mostly successful at keeping the Blockchain frequency constant (Block Frequency is the gradient of the Block size curve) with the only increase is block frequency visible in the time when the hash rate dropped dramatically as seen in Figure 4.1 when it took the protocol two weeks to adjust to a lower difficulty.

How changing the target hash difficulty affects block frequency?

Rearranging the equation 2.1 to $Difficulty = \frac{Total\ Hash\ Power}{Block\ Frequency}$.

It is visible that depending on what the Block Frequency target is, the difficulty will have to change for different amounts in order to react to the difference in Total Hash Power. For example, if the target frequency is one block per two weeks and the Hash power doubles, the Difficulty must double as well.

Comparing this to a crypto-currency whose block frequency is 2016 blocks per two weeks, if the Total Hash power doubles then the Difficulty must increase by a factor of 2016 times 2. This greater difficulty increase means that the Difficulty will fluctuate significantly if the block frequency is high, and if it is low the difficulty will fluctuate less to maintain the same block frequency.

In other words, for an increased total hash rate, the protocol will have to increase its difficulty by a greater factor each time, relative to the total hash rate in order to achieve the same block frequency.

Figure 3.3 also shows an asymptote at y = 0 which suggests that there will never be a point at which the network cannot produce any more blocks. Further, it suggests that as the network increases, the difficulty will always have the ability to increase, never reaching a point at which the average block frequency is over 2016.

The data also suggests that a new blockchain with a small network of little miners must do very slight, accurate corrections, as the effect on the block frequency is much greater than at the higher difficulties. Moreover, if the total hash rate of the network increases

drastically between two corrections in the difficulty there could be an unfair profit for the miners in that cycle as they will be mining at a much lower difficulty than the network would aim for at that hash rate, and it will take a large correction to return to the target block frequency. This issue is seen in the Figures 4.x at the point of drastic drop of Hash Rate.

In conclusion, the way Bitcoin uses the difficulty to manage block frequency and Bitcoin supply seems to be effective. The results and equations derived suggest that Bitcoin will have to keep increasing its difficulty by the same factor for each increase in hash rate of the network. This creates an issue for the long-term ability for the technology to succeed. Because it assumes the network will never reach a Hash Rate higher than $10x2^{256}$ hashes per minute, as at that point, the network could mine blocks at the frequency of 10 per minute at the highest difficulty possible given that the maximum difficulty is 2^{256}. At what point, the protocol will have to change to a cryptographic function with more digits such as SHA 512, which potentially lead to new problems such as porting the existing blockchain to a new protocol standard.

4.2 Evaluation

The data collected in the experimentation seemed to be reliable as there were no anomalies or obvious systematic or random errors. Furthermore, the relationship between Target Hash Difficulty and the block frequency shown by the collected data followed the theoretical and hypothesized relationship.

How changing the target hash difficulty affects block frequency?

The main difference between the collected results and data seen in [1] was the gradient of the curve in Figure 3.3 and Figure 3.4 where our results did not show as great of a gradient as the curve in Figure 3.4. This was most likely due to the limitations in the increments of difficulty by which we could test. The increment by which the script increased the difficulty was by a factor of two. Due to the limited hardware, the hash rate was not great enough to have smaller increments as the amount of hashes in total the computer would have to do would grow greatly. This ment that the gradient between our data points was largely assumed using the trendline by connecting the data points that were far apart in some cases.

Furthermore, an assumption made was that there would be no Bitcoin forks. This means there is a single blockchain at all times, while in the Bitcoin network, two acceptable blocks can be found at the same time. Therefore, it creates a disagreement until one chain is accepted by the majority of the network. The probability of forks decreases with an increase in difficulty [2] which is another possible reason why at low difficulties our data did not follow the scientific data from [1] shown in Figure 3.4.

Another possible factor that could have affected the results is the fact that a single core on the CPU was used to mine the blocks, since it was the most controlled way possible. However crypto-currencies are usually mined using the GPUs which have different fundamental architecture than CPUs and are able to execute much more instructions per clock (on average).

Some precautions that increased the reliability of the data include:

How changing the target hash difficulty affects block frequency?

Controlling the variables explained in 2.2.

The number of blocks per blockchain, given that 10000 blocks is close to 5 times what the actual protocol mines before readjusting the difficulty again. This reduces random errors and has a more accurate reading for the average time per block, and therefore the block frequency.

The range of results was enough to show a clear trend. This is visible by the asymptote found at the y = 0 axis suggesting any greater range would not change the trend in the data.

How changing the target hash difficulty affects block frequency?

REFERENCES:

[1] COINLORE, ACCESSED FROM HTTPS://WWW.COINLORE.COM/ALL_COINS, ON SEPTEMBER 7TH, 2019

[2] A. SHANTI BRUYN, AUGUST 26, 2017, "BLOCKCHAIN AN INTRODUCTION", ACCESSED FROM HTTPS://BETA.VU.NL/NL/IMAGES/WERKSTUK-BRUYN_TCM235-862258.PDF ON JULY 24TH, 2019

[3] SATOSHI NAKAMOTO, 2008, BITCOIN: A PEER-TO-PEER ELECTRONIC CASH SYSTEM ACCESSED FROM HTTPS://BITCOIN.ORG/BITCOIN.PDF ON JULY 24TH, 2019

[4] WOUTER PENARD AND TIM VAN WERKHOVEN, ON THE SECURE HASH ALGORITHM FAMILY, ACCESSED FROM HTTPS://WEB.ARCHIVE.ORG/WEB/20160330153520/HTTP://WWW.STAFF.SCIENCE.UU.NL/~WERKH108/DOCS/STUDY/Y5_07_08/INFOCRY/PROJECT/CRYP08.PDF ON JULY 25TH, 2019

[5] BITCOIN.ORG, JANUARY 15, 2016, ACCESSED FROM EN.BITCOIN.IT/WIKI/TARGET ON JULY 25TH, 2019.

[6] BITMAIN, JULY 25,2019, ACCESSED FROM HTTPS://BTC.COM/STATS/DIFF ON JULY 25, 2019

[7]HOWCODEORG, MARCH 19 2018, "SIMPLE-PYTHON-BLOCKCHAIN" ACCESSED FROM HTTPS://GITHUB.COM/HOWCODEORG/SIMPLE-PYTHON-BLOCKCHAIN ACCESSED ON JULY 27TH, 2019

[8] BLOCKCHAIN.COM, SEPTEMBER 7, 2019 HTTPS://WWW.BLOCKCHAIN.COM/EN/CHARTS/HASH-RATE , ACCESSED ON THE SEPTEMBER 7, 2019

[9] BITCOIN.ORG, 15 JANUARY 2016, ACCESSED FROM EN.BITCOIN.IT/WIKI/DIFFICULTY ON JULY 25, 2019.

[10] MICHAEL CROSBY, NACHIAPPAN, PRADHAN PATTANAYAK, SANJEEV VERMA AND VIGNESH KALYANARAMAN, BLOCKCHAIN TECHNOLOGY, OCTOBER 16, 2015, ACCESSED FROM HTTPS://SCET.BERKELEY.EDU/WP-CONTENT/UPLOADS/BLOCKCHAINPAPER.PDF ON JULY 20TH, 2019

[11] MARKO V., THE QUEST FOR SCALABLE BLOCKCHAIN FABRIC: PROOF-OF-WORK VS. BFT REPLICATION. OPEN PROBLEMS IN NETWORK SECURITY, OCTOBER 2015, ACCESSED FROM HTTPS://ALLQUANTOR.AT/BLOCKCHAINBIB/PDF/VUKOLIC2015QUEST.PDF ON JULY 24TH 2019

How changing the target hash difficulty affects block frequency?

Appendix

Source Code used

```
#import modules
import datetime
import hashlib
import time

#assign files to output results to
file  = open('output.txt', 'w')

#define function that will mine n number of blocks at one difficulty (pdiffic
ulty)
def miner(pdifficulty):

#Create class Block
        class Block:
            blockNo = 0
            data = None
            next = None
            hash = None
            nonce = 0
            previous_hash = 0x0
            timestamp = datetime.datetime.now()

#Initializing a data for class Block
            def __init__(self, data):
                self.data = data

#Initializing a hash with the encrypted string containing all components of a
 block
            def hash(self):
                h = hashlib.sha256()
                h.update(
                str(self.nonce).encode('utf-8') +
                str(self.data).encode('utf-8') +
                str(self.previous_hash).encode('utf-8') +
                str(self.timestamp).encode('utf-8') +
                str(self.blockNo).encode('utf-8')
                )
                return h.hexdigest()

#Returning the information about the last block that was just mined
            def __str__(self):
                return "Block Hash: " + str(self.hash()) + "\nBlockNo: " + st
r(self.blockNo) + "\nBlock Data: " + str(self.data) + "\nHashes: " + str(self
.nonce) + "\n--------------"
```

How changing the target hash difficulty affects block frequency?

```
#defining Blockchain class
        class Blockchain:

            difficulty = pdifficulty
            maxNonce =  4294967296 #2**32
            target = 2 ** (256 -difficulty)
#the final block hash must be less than this value

            block = Block("Genesis") #first block is special as it has hash o
r data from previous block
            pass_ = head = block

            def add(self, block):

                block.previous_hash = self.block.hash() #block at the top of
the linked list
                block.blockNo = self.block.blockNo + 1  #next block

                self.block.next = block #pointer to the next block to add at
the end of list
                self.block = self.block.next #move pointer up


            def mine(self, block):
                for n in range(self.maxNonce):
                    guess = int(block.hash(), 16)
                    if guess <= self.target:
                        self.add(block)
                        print(block)
                        break
                    else:
                        block.nonce += 1

        blockchain = Blockchain()

        start = time.time()
        for n in range(0,10000):
            blockchain.mine(Block("Block " + str(n+1)))
        end = time.time()


        while blockchain.head != None:
                print(blockchain.head)
                blockchain.head = blockchain.head.next

        difficulty = str(int(blockchain.difficulty))
        blockrate = str(float(10000/(end - start)))
        message = "\n" + difficulty + "," + blockrate
        file.write(message)
```

How changing the target hash difficulty affects block frequency?

```
        print("Time taken for 10000 blocks: ", end - start, "Average blockrat
e: ", 10000/(end - start), "Difficulty: ", blockchain.difficulty)

pdifficulty = 0
while pdifficulty <= 20:
    miner(pdifficulty)
    pdifficulty = pdifficulty + 1
```