IMPROVING THE ENERGY EFFICIENCY OF BITCOIN MINING

Conference Paper · November 2019

CITATIONS
0

READS
15

3 authors, including:

Riaan Bezuidenhout
University of the Free State
5 PUBLICATIONS **1** CITATION
SEE PROFILE

Wynand Nel
University of the Free State
19 PUBLICATIONS **6** CITATIONS
SEE PROFILE

IMPROVING THE ENERGY EFFICIENCY OF BITCOIN MINING

R. Bezuidenhout[1], W. Nel[2], A.J. Burger[3]
[1,2,3]Computer Science and Informatics, University of the Free State, 205 Nelson Mandela Drive, Bloemfontein, South Africa
E-mail: rbez@mweb.co.za, nelw@ufs.ac.za, burgeraj@ufs.ac.za

Abstract: Bitcoin is probably the most well-known blockchain system in existence. It employs the proof-of-work (PoW) consensus algorithm to add transactions to the blockchain. This process is better known as Bitcoin mining. PoW requires miners to compete in solving a cryptographic puzzle before being allowed to add a block of transactions to the blockchain. This mining process is energy intensive and results in high energy wastage. The underlying cause of this energy inefficiency is the result of the current implementation of the PoW algorithm. This PoW algorithm assigns the same cryptographic puzzle to all miners, creating a linear probability of success between the miner's computational power as a proportion of the total computational power of the network. In order to address this energy inefficiency of the PoW mining process, the researchers investigated whether a nonlinear probability of success, between the miner's computation power and its probability of success, will result in better energy usage. This nonlinear proof-of-work (nlPoW) algorithm required that a random difficulty be assigned to each miner through the process of pseudo-random number generation. In order to generate this pseudo-random number, a seed value is needed for generating a specific difficulty needed in the nlPoW algorithm. Initially, the requirements for this seed value needed to be determined. The researchers proceeded with a simulation of the Bitcoin mining process, using the proposed nlPoW algorithm. Preliminary results, simulating a network of 1000 miners with identical computational power, indicate that nlPoW may reduce the amount of hash computations required by Bitcoin mining. This provides a foundation from where refinements to the nlPoW algorithm and mining simulations can be made in the future.

Keywords: Bitcoin, bitcoin mining, consensus algorithm, nonlinear proof-of-work, proof-of-work

1. INTRODUCTION

Blockchain systems such as Bitcoin is a public, indelible, transactional data structure shared on a distributed computer network without central authority (Mulár, 2018). The nodes of the distributed, decentralised computer network follow a pre-determined set of rules (consensus algorithm) to add transactions and reach consensus on the single correct version of the transaction history in the blockchain. This entire process is called Bitcoin mining (Nakamoto, 2008).

The Bitcoin mining process is extremely energy intensive and results in high energy wastage. The underlying cause of this energy inefficiency is the result of the original implementation of the proof-of-work (PoW) algorithm, as discussed in Section 3. Different blockchain systems all share certain basic components although they differ in their intended application and in the architecture used to achieve their aims and objectives. Section 2 describes the basic blockchain components as introduced by Nakamoto for what was to become Bitcoin (Nakamoto, 2008) and elaborated upon by Zheng, Xi, Dai, Chen, & Wang (2017).

2. BITCOIN BLOCKCHAIN DATA SRUCTURE

The blockchain data structure consist of a sequential series of data blocks that are cryptographically linked. Each block consists of three main components or sub-blocks, namely the transaction block, block header and block hash (Bitcoin.org, n.d.). These components will be discussed next.

2.1 Transaction Block

The transaction sub-block contains a set of transactions to be added to the blockchain. In Bitcoin, this set of transactions is structured as a Merkle tree (Figure 1) where each parent hash contains a hash of its children. This process culminates in a root hash that contains a signature unique to the data contained in all the transactions and which would fail to be reproducible if any of the information in any of transactions were to be changed (Nakamoto, 2008).

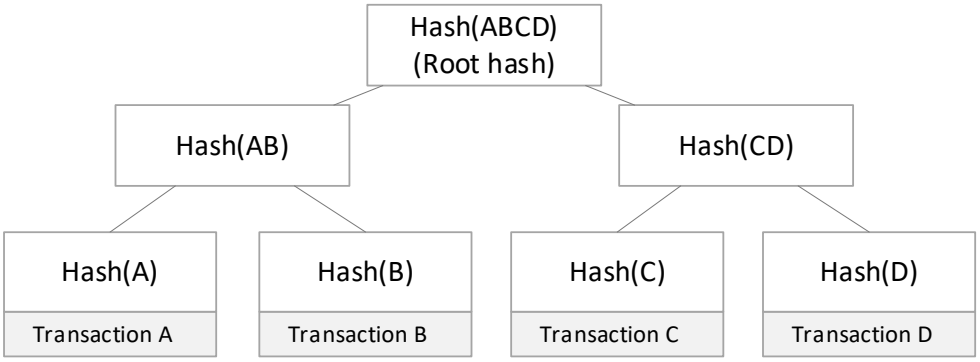

Figure 1: Transaction Merkle tree (Nakamoto, 2008)

In Bitcoin, the first transaction in the transaction Merkle tree is called the coinbase transaction and it has the specific function of creating new Bitcoins and paying them to the mining node as a mining reward (Bitcoin.org, n.d.).

2.2 Block Header

In its simplest form, a blockchain is a tamper-evident log of transactions. This means that each block must contain the transactions, a hash pointer to the previous block and a timestamp (Mulár, 2018). Nakamoto (2008) showed that it is not necessary to calculate the hash of the entire block, including all the transactions, but that a block header can be constructed that contains only the root hash of the transaction Merkle tree and the other header fields (previous block hash, timestamp and depending on the specific blockchain, other block data). Note that the block hash is computed only from the block header. The Bitcoin block header contains the fields as indicated in Table 1.

Table 1: Bitcoin block header fields (Bitcoin.org, n.d.)

Field	**Contents**
Previous block hash	Block hash calculated during previous block round.
Transaction Merkle tree root hash	Root hash of the transaction Merkle tree.
Timestamp	Timestamp when the miner started mining the block – according to the miner's clock.
Consensus algorithm software version	Specifies the set of validation rules used to create the block.
Number used only once (nonce)	A number, normally starting at zero and increased by one for every hash calculation. This produces a new hash, while keeping all the other header fields unchanged.
nBits field / Difficulty parameter (static target that all miners must solve)	A binary number that indicates the maximum value the hash may be. The miner must keep on changing the nonce and re-hashing the block header until the hash is strictly smaller than the difficulty parameter.

2.3 Block Hash

The block hash is the hash of the block header data and must conform to the rules of the consensus algorithm (Section 3). These rules include a calculation difficulty that the mining node must prove it reached during the construction of the block. Although it is very difficult for the mining node to compute a suitable hash value, it is very easy for any other node to inspect the blockchain and confirm that the mining node did indeed do the required work (Zheng et al., 2017).

Figure 2 depicts the components of a Bitcoin block and shows how the cryptographic link between blocks are formed by including the hash of the previous block header in the hash of each new block.

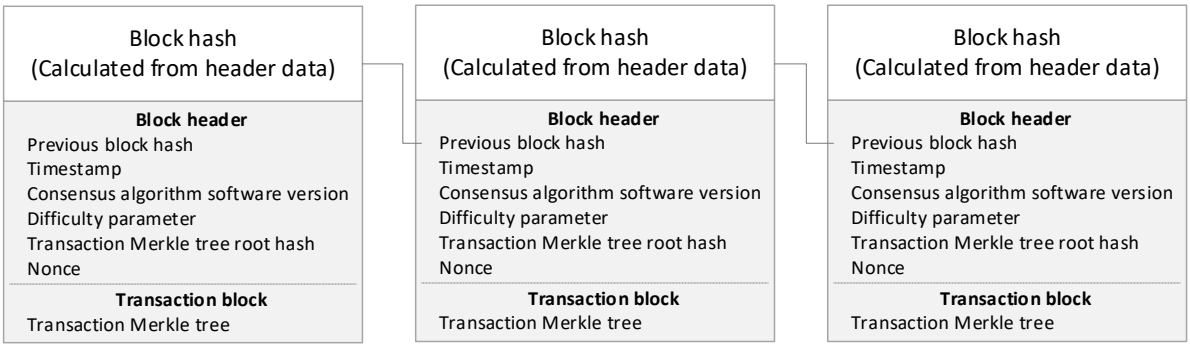

Figure 2: Structure of the Bitcoin blockchain (Zheng et al., 2017)

3. BITCOIN CONSENSUS ALGORITHM

A blockchain consensus algorithm is a detailed set of rules that miners follow in order to mine new blocks and to agree amongst themselves which version of the blockchain is considered the valid one (Zheng et al., 2017). Bitcoin uses the PoW consensus algorithm as discussed below (Nakamoto, 2008).

3.1 Proof-of-Work

PoW is an algorithm for determining which miners managed to create valid new blocks. This entails the calculation of a hash value of sufficient difficulty from the block header data, including the transaction Merkle tree root hash, which serves as a "cryptographic summary" of all the transactions in the block. The difficulty is selected so that it requires each miner to calculate an enormous number of hashes, each time using a new nonce to render a different hash, before finding a suitable block hash (Nakamoto, 2008).

One of the built-in rules of Bitcoin's PoW is that the network recalculates the difficulty every 2016 blocks so that it takes on average 600 seconds (10 minutes) to mine a block. The calculation adjusts the difficulty by the factor that the average block addition time is less than or more than 10 minutes and has the effect that the difficulty changes as the computational power of the network changes. Both the 10-minute average block addition time and the 2016 block difficulty recalculation interval (at 10-minute intervals the mining of 2016 blocks equates to two weeks) are arbitrary values that were introduced by Nakamoto in the original Bitcoin software (Nakamoto, 2008). These values could be substituted by any other values. Historically, the Bitcoin block addition times mostly started at 10 minutes and decreased over the span of 2016 blocks (which then also took less time to complete than the theoretical two weeks), as new computational power were continuously added to the network. This decrease in block addition times continued until such time that the difficulty was recalculated (Narayanan, Bonneau, Felten, Miller, & Goldfeder, 2016).

One of the core innovations of Bitcoin is that any miner is allowed to add a block with the simple rule that the first valid block announced, is the correct one. In practice it does happen, due to the latency of data propagation, that more than one valid block is found by the network,

bringing multiple branches of the blockchain into existence. This is called a fork in the blockchain and has the result that different partitions of the network may mine onto different branches (Eyal, Gencer, Sirer, & Van Renesse, 2016).

As one of the forks eventually becomes longer than the other it will eventually replace all other versions of the blockchain on the network (Bonneau et al., 2015). It can be inferred that transactions that were contained in the shorter fork, will then be included in future blocks on the longer chain.

3.2 Valid Chain Selection

A critical requirement of any blockchain system is that the network of miners eventually reaches an agreement on only one valid version of the blockchain, discarding all other forks. In Bitcoin, this is done through the longest chain rule. The fork that is mined by the largest network partition (by computing power) will eventually grow longer (will contain more blocks) than forks that are mined by smaller partitions (by computing power) (Nakamoto, 2008). Nakamoto (2008) referred to this principle in the context of the proportion of honest to dishonest miners, but it applies to network partitions in general.

When a miner receives a version of the blockchain that is longer than the one it is currently mining, it will discard its current blockchain and replace it with the longer one. This process ensures that only the longest blockchain survives and becomes the only valid blockchain accepted by all miners (Zheng et al., 2017).

3.3 The Side-effects of Proof-of-Work

Ma, Gans and Tourky (2018) note that finding a suitable hash requires no strategy by a miner, but relies on a form of brute force guessing (making a large number of guesses, each time with another nonce in the header, until a suitable hash is discovered) to solve a computational puzzle. The more computational power commanded by a miner, the more likely it is to solve the puzzle first and claim the prize, although the process remains probabilistic and does not guarantee that the most powerful miner will prevail.

Dimitri (2017) presents a framework whereby each round of the Bitcoin mining process can be viewed as an all-pay contest. An all-pay contest is a competition where the award or prize is known in advance and require everyone to make an investment to participate. Having miners play this type of game has two important desirable effects on the blockchain ecosystem, whereby *firstly*, it results in very few (preferably one) suggested new blocks to be found during each round and *secondly*, reducing the ability for bad actors to take back already spent Bitcoins. This is important as there is no prohibition on anyone to participate as a Bitcoin miner (Ma et al., 2018).

On the other hand, brute force guessing and unprohibited participation have led to one of the main undesirable side effects of Bitcoin mining, namely energy wastage.

3.3.1 Energy Wastage by Bitcoin Mining

The incentive structure of Bitcoin's PoW consensus scheme requires that all the miners in the network calculate an enormous amount of hashes in the race to be the first to produce one suitable block hash (Ma et al. 2018). This translates into an unsustainable level of energy usage. As the computing power of the network increases, resulting in a higher hash rate by the network in general, the difficulty of the block hash is automatically adjusted upward by the consensus algorithm. This is done to keep the average block creation time constant, but in turn results in mining entities adding ever more computing power to remain competitive (Tschorsch & Scheuermann, 2016). Ma et al. (2018) point out that as the value of Bitcoin has risen over the years (2009 to 2018 were the years covered by their research), the reward from mining has become larger, resulting in drawing more participants and consequently, increasing energy usage.

According to the Bitcoin Energy Consumption Index (2019), the Bitcoin network consumed an estimated 73 terrawatt-hour (TWh) of electricity per year in July 2019. This is comparable with the energy consumption of Austria, a country with 8.79 million inhabitants (July 2018 estimate) (Central Intelligence Agency, n.d.).

Only a single block, mined by a single miner eventually survives to permanently become part of the blockchain and all the energy expended by unsuccessful miners, which is the vast majority, goes to waste (Zheng et al., 2017). Tschorsch and Scheuermann (2016) state that, given the same block hash difficulty for each miner and given that all miners aim to come up with a solution, the chance of solving the block hash is proportional to the miner's fraction of the total computing power. In other words, the probability of mining a block successfully is linear to the computational power as explained by (Dimitri, 2017):

Assume a network where each miner is denoted as i and:

$$i = 1, 2, .., n$$

The hash power of each miner is denoted as:

$$h_i$$

Thus, the hash power of the entire network is denoted as:

$$h_n$$

Then, the probability that a miner is the first to solve the block hash is:

$$\mathrm{P}(h_i) = \frac{h_i}{h_n}$$ resulting in a linear equation

4. PROPOSED SOLUTION

In Section 3, high energy wastage of the PoW consensus algorithm was identified as a major concern for the long-term sustainability of Bitcoin. The energy consumption is related to the amount of computation required and is measured in hashes per second per watt (H/s/W) (Narayanan et al., 2016). This energy wastage was linked to the fact that each miner's probability of success is linear to the proportion of its computational power. If this probability of success could be changed to be nonlinear to the computational power of each miner, it is reasonable to argue whether it would have an effect on the energy consumption of the network

as a whole. In other words, if every miner received a random target during each block round, will it reduce the energy consumption of the mining network?

The researchers propose a nonlinear proof-of-work (nlPoW) algorithm for assigning a dynamic target (random target) to each miner during each block round (Section 4.1) and to provide initial indications of the viability of the solution through simulation of the mining process under nlPoW (Section 4.2).

4.1 Assigning a Random Target

Assigning a dynamic difficulty to each miner entails generating a uniformly distributed random number in the range $0\ to\ 2^{256}$ (the possible outcomes of the SHA256 hash algorithm). Stark & Ottoboni (2018) propose that cryptographic hash functions may be useful as pseudo-random number generators. The unpredictability and collision resistant properties of hash functions make them suitable for pseudo-random number generators.

For the purposes of this research, the cryptographic hash function provides an elegant solution to generating a pseudo-random number, but it will require a seed (input value) with specific characteristics. It is important that the seed is determined during the design of the algorithm and that it conforms to a number of guidelines that the algorithm must adhere to when selecting the seed.

Firstly, the seed must be unique to each miner so that the target calculated for each miner is different. *Secondly,* the information for selecting the seed must be encoded on the blockchain to allow other miners to confirm the difficulty calculated by the successful miner and by implication, the validity of the block hash produced. *Thirdly*, the seed must change every time a new block is constructed so as not to provide a permanent advantage or disadvantage to any individual miner. *Finally*, since each miner is free to choose any strategy to its own advantage, it must not be possible for the miner to manipulate the seed (generate many seeds) until it produces an easy target for itself.

One solution to this problem is to construct the seed from the miner's coinbase address and the previous block hash with the proviso that the coinbase address must contain a previous transaction on the blockchain (the coinbase address may not be created at the same time as the block containing it).

4.2 Simulation of Dynamic Target Assignment

In order to investigate the proposed solution that assigning a dynamic target from a uniform distribution to each miner, during each block round, will reduce the number of computations performed by the network to mine a new block, the mining process was simulated. During the simulation a random difficulty was awarded to each miner by calculating the SHA256 hash of the miner's address in conjunction with a fictional previous block hash. The address for each miner was unique and the previous block hash was identical.

By dividing the result of the SHA256 hash function above by 10000 and taking the remainder, a uniformly distributed random number ranging from zero to 9999 is produced. This number can be standardised to produce a uniformly distributed random number (to four decimals) in the range (0,1] (zero excluded and 1 included) by adding one and dividing by 10000.

For the purpose of the following example, the state of the Bitcoin network was selected from block 592583 on 31 August 2019 (BTC.com, 2019). The static target (as specified by the nBits field in Table 1) in Bitcoin block 592583 is then divided by the standardised uniform random variable to produce a dynamic target that is different for each miner, unpredictable and changes with each block round (as the previous block hash changes and is unpredictable).

Table 2 shows a sample of the results from a simulation of nonlinear targets for 1000 miners from a uniform distribution.

Table 2: Sample of simulated results for nonlinear target

Miner number	**Standardised random variable**	**Nonlinear target**	**Estimated number of hashes required by miner**	**Static target**	**Estimated number of hashes required by static target**
	A random value in the range (0,1]	The dynamic target for this miner in this block round. Static target divided by standardised random variable	Estimated number of hashes required by miner to find a hash smaller than the nonlinear target	Original target encoded in the block header - nBits parameter in Table 1	Estimated number of hashes required by miner to find a hash smaller than the static target
0	0.5514	4.8015E+054	2.41E+022	2.65E+054	4.37E+022
1	0.3021	8.7643E+054	1.32E+022	2.65E+054	4.37E+022
…	…	…	…	…	…
998	0.9309	2.8438E+054	4.07E+022	2.65E+054	4.37E+022
999	0.5652	4.6839E+054	2.47E+022	2.65E+054	4.37E+022

The standardised random variables must be tested for randomness. The Kolmogorov-Smirnov test is suitable for this purpose (Accord, 2017). The hypothesis is stated as:

H_0: Series of standardised random variables (column 2 in Table 2) is random

At the confidence level of $\alpha = 0.05$ the null hypothesis cannot be rejected and the series of standardised random variables are assumed to be random.

The total estimated number of hashes that will be done by the entire network under the existing PoW algorithm is

$$4.3738 * 10^{25}\ hashes$$

In contrast the total estimated number of hashes that will be done by the entire network under the nlPoW algorithm is

$$2.2486 * 10^{25}\ hashes$$

This produces a saving of

$$2.215 * 10^{25} \; hashes \; or \; 48.59\%$$

4.3 Adjusting the Static Target

Section 3.1 explained the process whereby the static target (nBits parameter in the block header) is adjusted after every 2016 blocks to account for changes in the computational power of the network. In this respect the static target is nothing other than the estimation of the hash rate of the network. If the block creation time over 2016 blocks is faster than the theoretical ten minutes multiplied by 2016, the network's hash rate is deemed to have increased and the static target is adjusted downward (difficulty increased), proportionally.

Under nlPoW the mean of the dynamic targets over 2016 blocks represents a factor whereby the block creation time should be faster than ten minutes. This is the same factor whereby the block creation time should decrease under nlPoW (because proportionally fewer hashes are required). This estimated nonlinear completion time over 2016 blocks can be compared to the actual completion time to compute a factor for the estimated increase or decrease of the network's computational power over the last 2016 blocks. By multiplying the static target by this factor, the new static target to be used as the basis for assigning dynamic targets for the next 2016 blocks, can be established.

5. CONCLUSION

The approach described in this paper establishes a new direction of research, whereby the energy wastage of PoW type consensus algorithms in blockchain systems can be addressed. It proposes the key requirements for an algorithm that randomly distributes the difficulty of mining blocks on the Bitcoin blockchain between the population of miners. The aim of this approach is to lower the number of hash calculations required by the network to create new blocks on the blockchain.

Early results from simulations of Bitcoin mining under nlPoW shows promising results, but further research is needed to establish the optimal distribution of dynamic targets between the population of miners. The results, as presented above, distributes dynamic targets uniformly between miners, which serves the purpose of casting light on the viability of the concept. The researchers are currently investigating other types of distributions to investigate if it may yield more optimal results.

BIBLIOGRAPHY

Accord. (2017). Accord.NET Framework. Retrieved November 12, 2019, from http://accord-framework.net/docs/html/T_Accord_Statistics_Testing_KolmogorovSmirnovTest.htm

Bitcoin.org. (n.d.). Bitcoin Developer Reference. Retrieved May 21, 2019, from https://bitcoin.org/en/developer-reference#block-headers

Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J. A., & Felten, E. W. (2015). SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy*. Retrieved from https://ieeexplore.ieee.org/abstract/document/7163021

BTC.com. (2019). Pool Distribution. Retrieved October 11, 2019, from https://btc.com/stats/pool

Central Intelligence Agency. (n.d.). Austria. Retrieved July 22, 2019, from https://www.cia.gov/library/publications/the-world-factbook/geos/au.html

Digiconomist. (2019). Bitcoin Energy Consumption Index. Retrieved April 7, 2019, from https://digiconomist.net/bitcoin-energy-consumption

Dimitri, N. (2017). Bitcoin Mining as a Contest. *Ledger*, *96*.

Eyal, I., Gencer, A. E., Sirer, E. G., & Van Renesse, R. (2016). Bitcoin-NG: A Scalable Blockchain Protocol. In *13th USENIX Symposium on Networked Systems Design and Implimentation*. Santa Clara.

Ma, J., Gans, J. S., & Tourky, R. (2018). *Market Structure in Bitcoin Mining* (No. 24242). Cambridge, MA.

Mulár, B. O. (Masaryk U. (2018). *Blockchain Technology in the Enterprise Environment*.

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*.

Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). *Bitcoin and Cryptocurrency Technologies*. Princeton: Princeton University Press.

Stark, P. B., & Ottoboni, K. (2018). Random Sampling: Practice Makes Imperfect. *ArXiv.Org*. Retrieved from https://arxiv.org/pdf/1810.10985.pdf

Tschorsch, F., & Scheuermann, B. (2016). Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. *IEEE Communications Surveys & Tutorials*, 2084–2123.

Zheng, Z., Xi, S., Dai, H., Chen, X., & Wang, H. (2017). An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. In *2017 IEEE 6th International Congress on Big Data*.