Bazzanella and Gangemi
Financial Innovation (2023) 9:91
https://doi.org/10.1186/s40854-023-00505-2

Financial Innovation

RESEARCH Open Access

Bitcoin: a new proof-of-work system with reduced variance

Danilo Bazzanella[1] and Andrea Gangemi[1*]

*Correspondence:
andrea.gangemi@polito.it

[1] Dipartimento di Scienze Matematiche, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Turin, Italy

Abstract

Since its inception, bitcoin has used the popular consensus protocol proof-of-work (PoW). PoW has a well-known flaw: it distributes all rewards to a single miner (or pool) who inserts a new block. Consequently, the variance of rewards and the mining enterprise risk are extremely high. In 2016, Shi proposed addressing this problem with a theoretical algorithm. We introduce an easily-implemented PoW variant that improves Shi's idea. The network must not find a single nonce but a few to insert a block. This simple change allows for a fairer distribution of rewards and also has the effect of regularizing the insertion time of blocks. This method would facilitate the emergence of small pools or autonomous miners.

Keywords: Proof-of-work, Bitcoin, Consensus protocol

Introduction

Proof-of-work (PoW) is a cryptographic proof in which one party (the prover) demonstrates a significant amount of computational work to the others (the verifiers). An essential condition is that verification must require little computational effort.

In Dwork and Naor (1992) proposed the basic idea of deterring Denial of Service attacks and email spam. The idea is that before being able to access the service, the service user must perform a certain amount of computation, which is time-consuming to perform but quick to verify.

PoW was formalized a few years later in a 1999 paper by Jakobsson and Juels (1999). However, the most successful version of PoW comes from Adam Back, who proposed the protocol called *hashcash* in a 2002 technical report entitled *A Denial of Service Counter-Measure* (Back et al. 2002).

Hashcash is very simple, powerful, and flexible. The idea is to take any text, add a small piece of random text (called the *nonce)*, and require the hash of the composed text to be below a certain target. If the hash is not below the target, one must try again with a new nonce until obtaining a hash below the target. Predicting the hash function's output as the input text changes are impossible. Hence, the protocol works by randomly extracting a number in a certain range and requiring the hash value to fall below a certain value, making it easy to check the probability of success. Verifying completion of a large amount of calculation is straightforward. That

is, compute a single hash with the nonce shown by the person who performed the PoW and check that the hash is below the set target. For an overview of PoW-based consensus protocols, see Meneghetti et al. (2020) and Rani and Bhambay (2023).

Bitcoin's consensus protocol is based on the hashcash protocol, which works as follows. Each miner builds a possible block of transactions and then tries to add it to the blockchain. The miner searches for a nonce to insert into the block so that the hashing of the block header is less than a public target set by the network protocol. The first miner to obtain a valid nonce has the right to insert the new block and receive the reward (6.25BTC in 2022). For every block, the target is modified to keep the block generation time at 10 min on average, regardless of the network's total computing power, roughly two weeks (Antonopoulos 2017).

This consensus protocol guarantees high security. A user would need more computing power than the rest of the network to attack the network, the so-called *51% attack*. The drawback is that adding a block to the blockchain without a major investment in adequate equipment, such as Application Specific Integrated Circuits, is very complicated. This limitation dramatically discourages autonomous mining and pushes all users to join a mining pool, paying fees to reduce the business risk.

One problem with this natural tendency is that the five largest pools currently account for more than 80% of the entire network's computing power, making the network very poorly decentralized. These data are from December 1, 2022, from the following link https://btc.com/stats/pool, a daily updated resource to consult this kind of data.

Another problem stems from the fact that, in Bitcoin, the miner (or pool) inserting the block takes the whole reward, whereas all the other miners take nothing. Therefore, the variance in miners' profit becomes very high. The only way to address this effect is to become a huge miner with the associated investments or work for a pool.

Finally, in Bitcoin, the insertion time of a block has a high variance. It is easy to verify, mathematically and in practice, that only 7% of blocks are produced between 9 and 11 min, whereas 14% of blocks are created between 8 and 12 min. The high variance of block insertion time is at the root of critical attacks on the PoW blockchain (Bissias and Levine 2020).

Modifying the Bitcoin consensus protocol would be very important for the network's security to promote autonomous mining even by not large mining farms while reducing the concentration of mining pools and the variance in blocking times.

This study proposes a flexible PoW system that can provide better network decentralization, redistributing rewards more fairly, and lowering the time variance to insert a new block.

The paper is organized as follows. In "Related work" section describes the state-of-the-art and highlights the research gap that led to the writing of this study. In "New Proof-of-Work systems" section theoretically explains the idea for a new consensus protocol, and "Analysis of the new systems" section reports a mathematical analysis of the new model. Then, "Blockchain forks" section is devoted to the study of blockchain forks, and "Numerical simulations" section shows the results of our simulations. Finally, "Conclusions" section concludes the study.

Related work

The idea of a more equitable reward distribution for successfully inserting a block is not new, and Sarkar (2019) introduced the *Multi-stage Proof-of-Work* in 2019. The PoW for mining a block consists of multiple stages that divide the block reward into an equal number of stage rewards.

Once a block gets onto the blockchain, the miner who completed a particular stage can claim the reward. D'Arco et al. (2020) analyzed in detail these types of protocols. They proved that the mining probability might not be strictly related to the miner hashing power, which could open potential fairness issues in mining. In addition, some proposed attacks seem to increase the system's vulnerability.

Considerable work has also been done to discourage the creation of large Bitcoin mining pools. A series of studies attempted to disincentivize large pools, making it difficult to delegate the PoW. Eyal and Sirer (2014) proposed the *Two-Phase PoW* comprising two cryptopuzzles. The first one is identical to the one already existing on Bitcoin. Therefore, the miner must find a nonce such that the block header's hash is less than a set target. In the second part, the block header must be signed with the coinbase transaction's private key, and the hash of this digital signature must be below the second difficulty parameter. Miller et al. (2015) proposed to disable mining pool enforcement mechanisms cryptographically strongly through *strongly nonoutsourceable puzzles*.

Shi (2016) proposed a very interesting proposal, even if only theoretical. The idea is to find a problem with several solutions rather than a single solution. The first miner (or pool) that finds all the solutions will cause the block to be inserted. The block will be inserted by one of the miners randomly among those who have found at least one solution.

This idea is similar to the line taken by the Bobtail algorithm, suggested by Bissias and Levine (2020). Bobtail is a possible concrete realization of Shi's model, but it is rather complex and difficult to manage. The previous study proposed maintaining a public ranking of the lowest k hashes in the network. The block is inserted by whoever is at the top of the ranking (i.e., the one who has found the lowest hash) when the average of these k values is less than a certain target t_k. One of the advantages of this idea is that they have forks comparable to the current Bitcoin protocol, even if $k > 1$. Conversely, the biggest drawback is that identifying a single miner responsible for entering the next block risks blocking the system in case the miner identified by the protocol is malicious and refuses to proceed.

Finally, check Kou et al. (2022) and Xu et al. (2019) for more information regarding how blockchain works, including their reliability and an overview of active research strands. If the reader is instead more interested in economic and financial aspects, such as cryptocurrency trading or price prediction using machine learning algorithms, see Fang et al. (2022) and Sebastião and Godinho (2021).

We will see in the next section how our model is similar to Bobtail. However, our model is simpler, the amount of communication is smaller, and it does not suffer from the problem of system blocking. Considering the greater simplicity of our new protocol, we can prove mathematically that the expected value of miners' profit remains unchanged compared with that of Bitcoin. Moreover, our model's variance decreases

and decreases more than Bobtail does, as we prove in "New Proof-of-Work systems" and "Analysis of the new systems" sections.

From another aspect, in our protocol, the probability of having forks increases. Luckily, this case does not create any issue, as we prove in "Blockchain forks" section.

New proof-of-work systems

Currently, the miner (or pool) that first finds a nonce that returns a hash of the block header less than a certain target when inserted into the block has the right to insert a new block into the Bitcoin blockchain and thus receive the reward. Our idea is to modify Bitcoin's PoW protocol by searching for multiple nonces, each of which returns a hash of the header less than the target when inserted into the producer block. Each miner can find multiple nonces with rewards in proportion to the number of nonces found.

Specifically, one can set an integer parameter $j \geq 1$ and establish that a new block B is inserted when the entire network finds j nonces. Notably, the case $j = 1$ is essentially the current Bitcoin protocol but with a small difference. That is, considering that rewarding multiple miners is generally necessary, doing so with the coinbase transaction of the block B while proceeding with the PoW is not possible. Which miners will deserve a fraction of the block reward is not yet known. Every nonce found must be published and verified by the network and inextricably linked to the Bitcoin address of the miner that found them. The coinbase transaction of block $B + 1$ will then redistribute the rewards among the miners who found the nonces during the creation of block B.

We switch the current Bitcoin game to a repeated game that is j times easier to maintain the desired block speed. However, the block is inserted when the network is successful j times.

We can use a hash function h, for example, SHA256, already used by Bitcoin, to link a nonce with the address of the miner who found it. The miners will look for a particular value nonce_1, to calculate another value, nonce_2, with the following formula:

$$\text{nonce}_2 = h(\text{miner address}||\text{nonce}_1).$$

nonce_2 is the value that will return a digest below the set target when inserted into the block header.

When a miner finds a suitable nonce_2, the miner will publish their block, the value nonce_1, and their Bitcoin address. The network can then verify the PoW's correct execution.

Considering that every hash function is collision-resistant, we can reasonably ensure that another miner cannot find another nonce_1, which will return the same value nonce_2, concatenated with their address.

Our model can be summarized as follows:

1. When a miner finds a correct nonce_2, they publish the value nonce_1, the block header, and the Bitcoin address where they will receive the reward. The network can easily verify the correctness of the published nonces;
2. When the number of nonces published by the entire network equals j, the nonces are sorted, and the miner who owns the smallest nonce_2 gets the right to insert the

new block. The miner must use their digital signature to prove the possession of the address. The rewards will be distributed in the next block;

3. The network starts to work for the insertion of a new block. The miners can insert the transactions they prefer, but the protocol will fix the coinbase transaction. Moreover, every miner is required to use the same. This shared coinbase distributes the reward plus the fees in proportion to the number of nonces found in the mining of the last inserted block;
4. The process starts again from step 1.

The easiest way to switch from the current Bitcoin protocol to our new protocol is to establish that a single block is entered without the coinbase transaction, postponing the reward distribution to the next block.

In our protocol, we select a single miner to be the only one able to insert a block, opening the way for the Denial of Service attack, already introduced in Bobtail (Bissias and Levine 2020).

However, this case is not an issue. If the miner who has the right to insert the next block, the one with the smallest nonce_2, does not proceed, then the network continues to perform calculations looking for a new nonce_2. A few moments later (on average, $\frac{10}{j}$ minutes), the network will find a new nonce. At that point, two miners will be able to insert the block. There will be $j+1$ nonce_2 that are valid, and thus, the two miners who found the two smaller nonces have the right to insert the new block. Any malicious miner who tries to stop the system by not inserting the block when they are the only one with this right will be at risk of losing their reward shortly after that once a second miner acquires this right.

For each choice of j, we have a system with the same average block insertion time rate as the current Bitcoin protocol. However, the variance of the insertion time is smaller and decreases as j increases. In addition, a block will more likely be inserted close to the average time (see "Analysis of the new systems" section for proof of these facts).

The second advantage of this modified PoW model is that miners who do not insert a block but can still show that they worked for the system are rewarded accordingly. The miners receive rewards for each block in proportion to the number of non-nonces they have found.

A possible drawback of this new model is the forks. If two miners find the j-th nonce almost simultaneously, then two users (the ones with the two smallest nonces) can propose inserting their block into the network. The network has two different last blocks (a fork). Each miner then resumes mining by choosing which branch of the network to try and attach a block to. One of the two branches prevails over the other in a short time, and the blockchain resolves the fork. The value of j must therefore be optimized to reward as many miners as possible without disproportionately increasing the number of communications and forks.

In any case, the number of communications in our protocol is less than that in Bobtail. The reason is that, in our protocol, any suitable nonce is immediately accepted and transmitted to the network. By contrast, in Bobtail, nonces are repeatedly substituted, and before identifying the winning ones, the protocol needs many communications.

We followed Shi's smart idea but made an essential change to build our new system. Shi proposed to insert the new block once a single miner has reached a fixed number of solutions to the problem.

By doing so, however, large miners would have an unfair advantage and tend to have a reward greater than their weight. To avoid this issue and share the reward more fairly, we chose to insert the block when the entire network finds a fixed number j of solutions to the problem and not when a single miner does allow us.

While benefiting small miners with an extra reward over large ones would be nice, it is impossible because miners are pseudo-anonymous. Therefore, a large miner can present himself to the network as many small miners and get an extra reward.

Our protocol allows for the maximum possible: leaving the average reward of miners unchanged but decreasing the variance, thereby making it less likely that a miner will get a reward much smaller than the expected average reward.

The larger the j, the more likely each miner will earn the amount of Bitcoin they are entitled to based on their computing power, thereby significantly reducing business risk. See "Analysis of the new systems" section for proof of this fact.

Hence, our protocol would give small miners an incentive to mine and thus safeguard decentralization.

Analysis of the new systems

We can use a Bernoulli process $\{X_n\}_{n\in\mathbb{N}}$ to model the Bitcoin protocol, where X_n are independent and identically distributed Bernoulli random variables of parameter $p = \frac{1}{600H}$ and H is the total hashrate of Bitcoin. The parameter p has been chosen because, on average, the Bitcoin blockchain adds a new block every 10 min. Hence, we can model the repeated tests that the entire network performs every second to find a block with the hash of the header less than the target, where $X_n = 1$ means that the network was successful in mining the block in the nth second. The hash function involved in the Bitcoin consensus protocol is not strictly random. Still, it is so unpredictable that approximating it with a random variable seems to be the most reasonable solution.

Consider now the random variable S, which models the time of the first success of a Bernoulli process:

$$S = \min\{n \in \mathbb{N} : X_n = 1\}.$$

S is a geometric random variable of parameter p, and its expected value is $\mathrm{E}(S) = \frac{1}{p} = 600H$, the average number of hashes required to insert a new block into the Bitcoin blockchain. Considering that the network processes H hashes in one second, the average waiting time is approximately 10 min. The variance of S is also straightforward, which is equal to.

$$V(S) = \frac{(1-p)}{p^2} \sim \frac{1}{p^2},$$

The standard deviation of S is of the same order as its expected value.

Similarly, we can use a Bernoulli process $\{Y_n\}_{n\in\mathbb{N}}$ to model our new protocol, where Y_n are independent and identically distributed Bernoulli random variables of parameter jp

and $j < 600H$. T_j is the random variable that models the time of jth success of the Bernoulli process $\{Y_n\}_{n\in\mathbb{N}}$:

$$T_j = \min\left\{n \in \mathbb{N} : Y_1 + \cdots + Y_n = j\right\}.$$

Notably, T_j is distributed as a negative binomial random variable with parameters (j, jp), where the first parameter models the required number of successes, whereas the second parameter models the success probability in each Bernoulli trial.

A closed formula to compute its expected value and its variance is as follows:

$$E(T_j) = j\frac{1}{jp} = 600H, V(T_j) = j\frac{(1-jp)}{(jp)^2} = \frac{(600^2H^2)}{j} - 600H.$$

Therefore, the new protocol has an average waiting time for inserting a block identical to the current Bitcoin protocol. In contrast, the variance of this waiting time is smaller for each $j \geq 2$ and decreases as j increases.

Specifically, we can compute the probability of inserting a block between 9 and 11 min (540 and 660 H hashes, respectively) in the actual Bitcoin protocol. Then, we can compare it with the same probability of our models with some fixed j values, such as $j = 10$ or $j = 100$:

$$P[540H < S < 660H] \sim 7.31\%,$$
$$P[540H < T_{10} < 660H] \sim 24.69\%,$$
$$P[540H < T_{100} < 660H] \sim 72.36\%.$$

Therefore, switching to the generalized protocol would make it rarer for a block to be inserted in a much shorter or much longer time than the expected average time of 10 min. This considerable reduction in the variance of the block insertion time would significantly improve the system's security, given that the high variance of block insertion time is well known to be at the root of critical attacks on the PoW blockchain (Bissias and Levine 2020).

Let us now calculate how the profit is distributed among the miners in the new protocol compared with the current Bitcoin protocol.

Let q be the probability of a generic Bitcoin miner finding a suitable nonce such that the hash of the block header is less than the network target. Notably, this probability does not depend on the target but only on the miner's weight (i.e., the ratio of his/her hashrate to the network's hashrate). We can get a reasonable estimate of this probability from past block insertion statistics. Let $M = 52,560$ be the average number of blocks mined in one year. We now define the Bernoulli process $\{B_k\}_{k\in\mathbb{N}}$, where B_k are independent and identically distributed Bernoulli random variables of parameter q, which model the probability of adding a block for the considered miner.

Let $U = 6.25\sum_{k=1}^{M} B_k$ be the profit of the miner in one year.

Evidently,

$$E(B_k) = q, \quad V(B_k) = q(1-q),$$
$$E(U) = 6.25Mq, \quad V(U) = (6.25)^2Mq(1-q).$$

From another aspect, we define a Bernoulli process $\{C_i\}_{i\in\mathbb{N}}$ to model our protocol, where C_i are independent and identically distributed Bernoulli random variables of parameter q and a Binomial random variable $N_k = \sum_{i=1}^{j} C_i$ that counts the number of nonces found for the block k by the considered miner.

Finally, let V be the random variable defined as $V = 6.25 \sum_{k=1}^{M} \frac{N_k}{j}$ that models the miner's profit in a year in our new protocol.

Thus, we have the following:

$$E(N_k) = jq, \quad V(N_k) = jq(1-q)$$

$$E(V) = 6.25Mq, \quad V(V) = (6.25)^2 M \frac{q(1-q)}{j}.$$

We have obtained that, in the new protocol, the profit of each miner has the same expected value as in the current Bitcoin protocol. Moreover, the variance of this profit is always lower and decreases by a factor of $\frac{1}{j}$.

Notably, in the Bobtail protocol, the decrease in variance occurs through a multiplicative factor close to $\frac{4}{3k}$. Considering that their parameter k is the analog of our parameter j, the decrease in variance in our protocol is, therefore, always greater for all values of the parameter.

Estimating the probability for a miner of weight p may be convenient to obtain an annual profit not too much lower than the expected value to further evaluate the improvement with the new system. For example, in Bitcoin, a miner has more than a 95% chance of making an annual profit U greater than 70% of the expected profit if it holds a weight equal to 0.0005304, that is

$$P[U > 0.7E(U)] \geq 0.95, p = 0.0005304.$$

We can repeat the same estimation for the new model. For example, if $j = 10$, then to obtain the same enterprise risk, it is sufficient for a miner to have a weight of $p = 0.00005305$, that is,

$$P[V > 0.7E(V)] \geq 0.95, p = 0.00005305.$$

More generally, the weight a miner has on Bitcoin can be divided j on the new protocol to maintain the same probability of making a suitably high profit.

Thus, in the new system, one could create a mining company with the same business risk but with a lower initial investment by one order of magnitude, in the case $j = 10$, or by two orders of magnitude, in the case $j = 100$.

Finally, our new protocol makes minimal modifications to Bitcoin, leaving the block insertion on average every 10 min and keeping the reward proportional to the miners' computing power. Therefore, our protocol has the same transactions per second as the current Bitcoin network and significantly reduces the risk of a selfish-mining attack or 51% attack (Azimy et al. 2022; Eyal and Sirer 2018), owing to the reduced variance of the average block insertion time. The only flaw is the greater number of blockchain forks than Bitcoin, addressed in the next section.

Blockchain forks

Decker and Wattenhofer (2013) conducted an excellent study on the number of forks in Bitcoin. They estimated the number of forks through a mathematical model and verified it with the observed blockchain fork rate. To better match their model with past observed data, they estimated the probability of having a fork in Bitcoin with the following:

$$p_1 = P[\textit{have a fork}] = 1 - \left(1 - \frac{1}{633.68}\right)^{11.37} \approx 1.78\%,$$

justifying the presence of 633.68 instead of the expected value of 600 through the decrease in the network's computational power during the period under consideration. For a better fit of the model to the average situation, we prefer to use 600.

The size of the two parts of the network that work in the competition should be considered to attach the blocks to the two different ends of the fork to estimate the average probability of having a fork of length two, a fork with two blocks applied in both branches.

Let x and $1 - x$ be the network rate that attaches a block to the first and second end of the fork, respectively.

When the network produces a second fork following the first, there are three possible cases:

- *Case 1* the two blocks of the new fork are attached to the first end of the previous fork;
- *Case 2* the two blocks of the new fork are attached to the second end of the previous fork;
- *Case 3* one block from the new fork is attached to the first end, whereas the other is attached to the second end.

A fork of length 2 is created only in the third case. In the first two cases, the fork remains of length 1. For the Bitcoin protocol, these three probabilities are as follows:

$$P[\textit{case } 1] = x^2 p_1,$$
$$P[\textit{case } 2] = (1 - x)^2 p_1,$$
$$P[\textit{case } 3] = 2x(1 - x)p_1.$$

Notably, the above probability estimates assume that the probability of a node finding a block is uniformly distributed at random among all nodes, as assumed by Decker and Wattenhofer (2013).

Then, the probability of having in Bitcoin a fork of length 2 is as follows:

$$P[\textit{for } k \textit{ of length } 2] = \int_0^1 p_1 2x(1 - x)p_1 dx = \frac{p_1^2}{3}.$$

In the same way, on average, we have the following:

$$P[for\ k\ of\ length\ k] = \frac{p_1^k}{3^{k-1}}.$$

In particular, we obtain the following:

$$P[fork\ of\ length\ 6] = \frac{p_1^6}{3^5} \approx 1.8 \cdot 10^{-13}.$$

A Bitcoin transaction is generally considered secure when it is on a block with six confirmations, a block to which six other blocks on the blockchain have been attached. There is a negligible probability of a voided transaction due to a fork.

Therefore, following the same strategy, we can prove that the probability of having a fork in our protocol may be estimated as follows:

$$p_j = P[have\ a\ for\ k] = 1 - \left(1 - \frac{j}{600}\right)^{11.37},$$

and then

$$p_j = P[for\ k\ of\ length\ k] = \frac{p_j^k}{3^{k-1}},$$

as the parameter j varies, it follows that

$$p_j = P[for\ k\ of\ length\ 11\ and\ j = 10] = \frac{p_{10}^{11}}{3^{10}} \approx 7.4 \cdot 10^{-14},$$

implying that if we take $j = 10$ and wait for 11 confirmations, our system is safer than the classic Bitcoin with six confirmations. In the same way,

$$p_j = P[for\ k\ of\ length\ 24\ and\ j = 100] = \frac{p_{100}^{24}}{3^{23}} \approx 1.2 \cdot 10^{-13},$$

implying that if we take $j = 100$ and we wait for 24 confirmations, our system is safer than the classic Bitcoin with six confirmations.

This result shows that our new protocol's higher average fork probability is not a problem because it is unsuitable for real-time payments, but neither is Bitcoin. Moreover, for payments where it is sufficient to fix the day of the transaction, it does not matter if you have a 4-h delay instead of a 1-h delay.

Notably, we have not considered forks involving three or more blocks competing to be the new block of the blockchain because this eventuality has negligible probability compared with the probability of the forks with two competing blocks. For further details, see Fig. 4 in the next section.

Numerical simulations

In this section, we show some plots that confirm the results obtained and described in theory. We simulated our protocol for different values of j (in particular, we have chosen $j = 1, 5, 10, 25, 50$, and 100). Moreover, we compared the obtained data with

Mining pool	Number of blocks mined in 2021	% of blocks mined in 2021
F2Pool	7950	15.037
AntPool	7669	14.506
Poolin	6284	11.886
ViaBTC	5657	10.7
Binance Pool	5421	10.254
BTC.com	4949	9.361
Foundry USA	3617	6.842
SlushPool	2192	4.146
Huobi.pool	2151	4.069
unknown	2011	3.804
1THash	1108	2.096
SBI Crypto	685	1.296
BTC.TOP	549	1.038
MARA Pool	395	0.747
Rawpool	371	0.702
EMCDPool	360	0.681
OKExPool	324	0.613

Mining pool	Number of blocks mined in 2021	% of blocks mined in 2021
Lubian.com	287	0.543
SpiderPool	158	0.299
WAYI.CN	140	0.265
OKKONG	130	0.246
Sigmapool.com	119	0.225
Luxor	114	0.216
TMSPool	63	0.119
ArkPool	61	0.115
NovaBlock	54	0.102
KuCoinPool	21	0.04
Relayed By	11	0.021
Bitcoin.com	5	0.009
Minerium	4	0.008
mmpool	3	0.006
Solo CK	3	0.006
EclipseMC	1	0.002
KanoPool	1	0.002

Fig. 1 Mining pools active during the year 2021

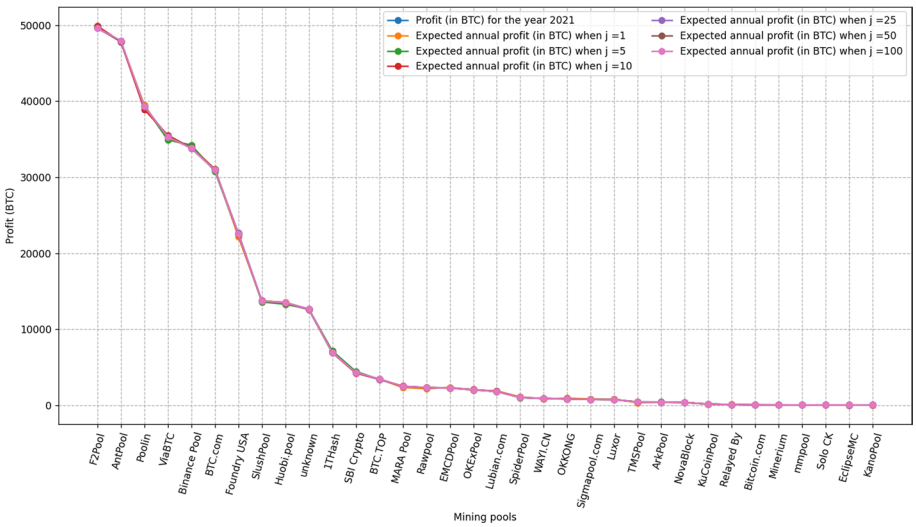

Fig. 2 Expected annual profit for different values of j

the real data of the Bitcoin blockchain in 2021. Our model with $j = 1$ is equivalent to the current Bitcoin protocol.

We have checked who mined all the blocks in 2021 to simulate the data. Then, we assigned the corresponding computing power to each mining pool or single miner. That is, each miner i has a chance to insert a new block equal to the following:

$$p_i = \frac{number\ of\ blocks\ created\ by\ i\ in\ 2021}{total\ number\ of\ blocks\ created\ in\ 2021}.$$

A total of 52,868 blocks were mined by 34 different mining pools or single miners, with the distribution depicted in Fig. 1.

First, we simulated the expected annual profit for each mining pool.

From Fig. 2, whatever j is, we have the same expected profit.

In our second simulation, the aim is to show that the annual profit of each miner is closer to the expected profit if j increases. That is, the variance of the profit becomes lower if j grows. The profit for each miner during the year 2021 is denoted as $\{x_1, \ldots, x_{34}\}$. For each chosen j value, we have simulated the network 10 times, obtaining each simulation profit values $\{\widehat{x_1}, \ldots, \widehat{x_{34}}\}$. Then, we have computed, for each miner or mining pool, the *discrepancy* MSD, that is, the mean of the square of the differences between the real and the simulated data, as follows:

$$MSD_i = \sqrt{\left(\frac{1}{34}\sum_{k=1}^{34}\left(x_k - \widehat{x_k}\right)^2\right)} \forall i \in \{1, 10\}.$$

Finally, we have used these numbers to draw the boxplots for each value of j (Fig. 3). A boxplot is a standard method for graphically showing the spread of numerical data through their quartiles. It is represented by a box divided into two parts, where two segments emerge. The box is bounded by the first and third quartiles and divided within it by the median. The minimum and maximum values instead bound the segments.

Figure 3 shows that the estimated profit is closer to the expected average profit if parameter j increases, as predicted by our model.

Finally, the last simulation (Fig. 4) shows our protocol's expected number of forks. Notably, the number of forks involving three or more miners is negligible up to $j = 25$. Therefore, even if the number of forks is evidently increasing, our assumptions and analysis in "Blockchain forks" section remain valid.

From the plot, this increase is proportional to j. However, one must keep in mind that, for the current Bitcoin protocol, the actual number of forks is still lower than indicated by the model because of network effects that a mathematical model does not easily capture. If our protocol were implemented in practice, we could expect the same thing to occur, and thus, the forks would be much less than the model's estimate.

Another technique for performing numerical and optimization experiments can be found in Kou et al. (2021).

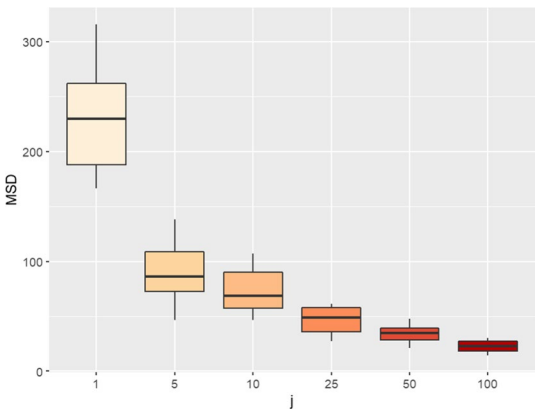

Fig. 3 Decrease in variance for different values of j

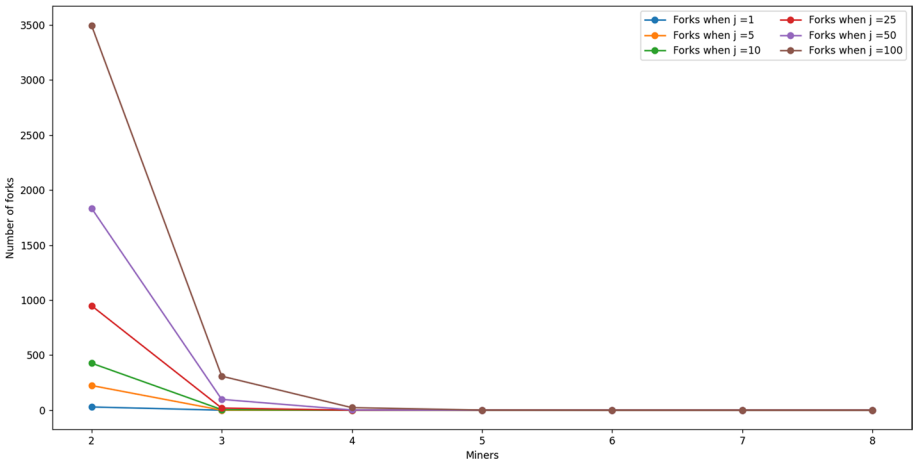

Fig. 4 Number of forks involving two or more miners for different values of j

Conclusions

The new PoW system that we presented would regularize the variance of block entry, and as we have seen, this fact would improve the system's security. A second result has the notable advantage of distributing the rewards more widely among the miners, which would greatly reduce the variance of profit distribution.

Bitcoin's essential characteristics are all maintained, allowing us to avoid new ways of attacking the system. Adopting this new protocol would therefore allow for a significant reduction in the business risk associated with mining. This new protocol would also allow for the start of this activity with a smaller investment in computing resources than the current Bitcoin.

The level of business risk reduction is proportional to parameter j, the number of nonces the network must find for a block insertion. The larger the j, the smaller the variance of the profit.

The only slightly negative consequence is that increasing j also increases the likelihood that some miners will almost simultaneously propose a new block for inclusion in the blockchain. Fortunately, Nakamoto (2008) devised an ingenious system to handle these critical situations. This system is applied in our new system and works so well that we obtain a system as functional as the original Bitcoin system by waiting a little longer to consider a transaction safe.

Abbreviation

PoW Proof of Work

Acknowledgements

Both authors are members of GNSAGA of INdAM and of CrypTO, the group of Cryptography and Number Theory of Politecnico di Torino. The authors would like to thank Giulia Della Croce Di Dojola for helpful discussions.

Author contributions

The author read and approved the final manuscript

Funding

This research received no external funding.

Availability of data and materials

All the information we have used to write this article can be found in the References section. We have used results from other research papers and public data available on web sites.

Declarations

Competing interests
The authors declare no competing interests.

Received: 3 August 2022 Accepted: 25 April 2023
Published online: 03 May 2023

References

Antonopoulos AM (2017) Mastering bitcoin: programming the open blockchain. O'Reilly Media Inc
Azimy H, Ghorbani AA, Bagheri E (2022) Preventing proof- of-work mining attacks. Inf Sci 608:1503–1523
Back A et al (2002) Hashcash-a denial of service counter-measure
Bissias G, Levine BN (2020) bobtail: improved blockchain security with low-variance mining. In: NDSS
D'Arco P, Ansaroudi ZE, Mogavero F (2020) Multi-stage proof-of-works: properties and vulnerabilities. Cryptology ePrint Archive
Decker C, Wattenhofer R (2013) Information propagation in the bitcoin network. In: IEEE P2P 2013 proceedings. IEEE, pp 1–10
Dwork C, Naor M (1992) Pricing via processing or combatting junk mail. In: Annual international cryptology conference. Springer, pp 139–147
Eyal I, Sirer EG (2014) How to disincentivize large bitcoin mining pools. Blog post. http://hackingdistributed.com/2014/06/18/how-to-disincentivize-large-bitcoin-mining-pools
Eyal I, Sirer EG (2018) Majority is not enough: bitcoin mining is vulnerable. Commun ACM 61(7):95–102
Fang F, Ventre C, Basios M, Kanthan L, Martinez-Rego D, Wu F, Li L (2022) Cryptocurrency trading: a com-prehensive survey. Financ Innov 8(1):1–59
Jakobsson M, Juels A (1999) Proofs of work and bread pudding protocols. In: Secure information networks. Springer, pp 258–272
Kou G, Xiao H, Cao M, Lee LH (2021) Optimal computing budget allocation for the vector evaluated genetic algorithm in multi-objective simulation optimization. Automatica 129:109599
Kou G, Yi K, Xiao H, Peng R (2022) Reliability of a distributed data storage system considering the external impacts. IEEE Trans Reliab
Meneghetti A, Sala M, Taufer D (2020) A survey on pow-based consensus. In: Annals of emerging technologies in computing (AETiC), Print ISSN, pp 2516–0281
Miller A, Kosba A, Katz J, Shi E (2015) Nonoutsourceable scratch-off puzzles to discourage bitcoin mining coalitions. In: Proceedings of the 22nd ACM SIGSAC conference on computer and com munications security, pp 680–691
Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. Decent Bus Rev 21260
Rani P, Bhambay R (2023) A comparative survey of consensus algorithms based on proof of work. In: Emerging technologies in data mining and information security. Springer, pp 261–268
Sarkar P (2019) A new blockchain proposal supporting multi-stage proof-of-work. Cryptology ePrint Archive
Sebastião H, Godinho P (2021) Forecasting and trading cryptocurrencies with machine learning under changing market conditions. Financial Innovation 7(1):1–30
Shi N (2016) A new proof-of-work mechanism for bitcoin. Financial Innovation 2(1):1–8
Xu M, Chen X, Kou G (2019) A systematic review of blockchain. Financial Innovation 5(1):1–14

Publisher's Note