2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)

Difficulty Prediction for Proof-of-Work Based Blockchains

Kaiwen Zheng
School of Elect. and Computer Engr.
Georgia Institute of Technology
Atlanta, GA 30332 USA
kzheng71@gatech.edu

Shulai Zhang
Dept. of Electronic Engineering
Shanghai Jiao Tong University
Shanghai, P.R. China
zslzsl1998@sjtu.edu.cn

Xiaoli Ma
School of Elect. and Computer Engr.
Georgia Institute of Technology
Atlanta, GA 30332 USA
xiaoli@gatech.edu

***Abstract*—The fast growing of blockchain technologies has inspired an exceptionally broad set of new social applications due to its special features. The most two popular blockchains (Bitcoin and Ethereum) are Proof-of-Work (PoW) based. However, the randomness of the block produce time (BPT) greatly affects the performance, user experience, and even the security of a blockchain. Mining difficulty controls the stability of the BPT. However, the current difficulty control algorithm in Ethereum has two problems. First, the difficulty control algorithm cannot track the network hashrate fast enough. Thus, the BPT has a large variance and cannot converge to the target BPT on time. The second problem is that the target BPT cannot be flexibly set to any given value by adjusting the difficulty. To mitigate these two problems, we propose a linear predictor based difficulty control algorithm. Based on the relationship among the hashrate, the difficulty, and the BPT, the prediction based difficulty control algorithm has much better stability and flexibility on BPT. It is also shown that the proposed prediction based algorithm outperforms the existing Ethereum one in the real situation.**

***Index Terms*—Blockchain, difficulty control algorithm, Proof-of-Work**

I. Introduction

A blockchain is a sequence of blocks and each block contains some transaction data. Any blockchain promises the authority of transactions by consensus algorithms. Currently, Proof-of-Work (PoW) is a well adopted consensus algorithm [1], [2]. In a PoW-based blockchain, blocks are added to the growing chain by miners who participate in mining. This paper focuses on a significant problem in PoW-based blockchains, which is called difficulty control, also called difficulty adjustment or difficulty retargeting.

The mining process is described in a simple way here: miners repeatedly cryptographically hash the assembled block, each time with a different nonce. Whenever the hash from any miner falls below a target time t, whose maximum value is m, the corresponding block is mined and the miner then has the right to add this block to the blockchain and publish it. The difficulty to mine a block successfully is d, which is defined as m/t. Additional details of the mining process can be referred to [3].

The difficulty for each block directly affects a block produce time (BPT). The BPT is also influenced by the total computing power of all miners, which is called the hashrate. The hashrate is hard to measure, but its value is reflected by both the difficulty and the BPT. Although hashrate does not change rapidly, due to network propagation delay, the BPT is random. PoW-based blockchains must adjust the difficulty d to ensure that the expected BPT remains roughly constant around a desired value. Thus, an accurate estimate of the hashrate and a proper difficulty control are critical for the stability of a PoW-based blockchain.

Bitcoin [4] and Ethereum [5] are both PoW-based blockchains while their difficulty control algorithms are different. A lot of efforts have been devoted to design difficulty control algorithms. In [6], an improved difficulty control algorithm for Bitcoin is modeled as a nonhomogeneous Poisson process and it assumes that the hashrate is exponentially increasing. The work in [7] presents a stochastic model for the BPT and analyzes the marginal distribution of the BPT in Bitcoin. The algorithm proposed in [8] is specifically able to prevent coin-hopping attacks. A proactive difficulty control algorithm that collects hashrate commitments secured by bond from miners is proposed in [9]. However, these difficulty control algorithms do not provide flexibility to track the hashrate changes.

In this work, we propose a difficulty control algorithm from signal processing point of view. We point out that the adjustment of difficulty with linear predictor can have a better and more precise tracking of hashrate, thereby improving the stability of BPT.

II. Problem Formulation

Similar to [7], blocks are generated at the time instances $t_0 \le t_1 \le t_2 \le \cdots$ with $t_0 = 0$. The time between every two instances $X_n = t_n - t_{n-1}$ is called the block produce time (BPT) of the n_{th} block. Assume X_n is exponentially distributed with a rate λ, which is changing according to the ratio of hashrate and difficulty. By observing the previous BPT and adjusting the difficulty, the goal is to reach a constant target BPT X_{target}. Recall that expected value of an exponentially distributed random variable is equal to the inverse of its rate. Thus, $\mathbb{E}[X_n] = \frac{1}{\lambda}$. The BPT is proportional to the ratio of hashrate and difficulty.

In order to maintain a stable BPT for each block, the difficulty is controlled to trace the hashrate. In Ethereum, the difficulty of the n_{th} block d_n is updated according to the difficulty d_{n-1} and the BPT X_{n-1} of the previous block [5]:

$$d_n = (1 + \mu f(X_{n-1})) \times d_{n-1}, \tag{1}$$

$$f(X) = \begin{cases} 1 - \alpha X, & \text{if } 1 - \alpha X \ \geq -99 \\ -99, & \text{if } 1 - \alpha X < -99 \end{cases}, \tag{2}$$

where $\mu = \frac{1}{2048}$ is the step size, and $\alpha = \frac{1}{9}$ is the control factor to achieve $X_{\text{target}} = 13s$.

Although the current Ethereum difficulty control algorithm can maintain a relatively stable performance, there are still exisiting challenges. First, the current difficulty control algorithm is hard to be adjusted to reach a specific target BPT value. Second, because the difference between adjacent difficulties is limited, the difficulty can not be readjusted with the hashrate synchronously and adaptively.

To mitigate these problems, linear prediction is used in this paper to design new difficulty control algorithms. Let us define a PoW term as

$$PT_n = \frac{d_n}{r_n}, \tag{3}$$

where d_n is the difficulty and r_n is the hashrate at the n_{th} block.

According to the relationship between λ and the ratio of hashrate and difficulty and Eq. (3), PoW term is proportional to the expected value of BPT. Thus, the BPT will stay more stable around X_{target} if PT_n is closer to X_{target}. To achieve that, if the PoW term PT_n using the current difficulty control algorithm of Ethereum can be predicted at the n_{th} block, it is possible for us to design the prediction based difficulty d_n^P to adjust PoW term and improve the stability of BPT:

$$d_n^P = \frac{X_{\text{target}}}{PT_n^{\text{pred}}} d_n, \tag{4}$$

where PT_n^{pred} is the predicted value of PoW term of the n_{th} block. Then the prediction based PoW term using d_n^P can be expressed as

$$PT_n^P = \frac{d_n^P}{r_n} = \frac{X_{\text{target}}}{PT_n^{\text{pred}}} \frac{d_n}{r_n} = X_{\text{target}} \frac{PT_n}{PT_n^{\text{pred}}}. \tag{5}$$

In theory, if $PT_n^{\text{pred}} = PT_n$, the prediction based PoW term will be maintained at X_{target}.

III. Linear Prediction Based Difficulty Control Algorithm

A. Linear prediction

Linear prediction is a mathematical operation where future values of a discrete-time signal are estimated as a linear function of previous samples. The predictor coefficients are obtained by a large amount of training data. After that, if the desired PoW term of the n_{th} block PT_n is wanted, previous terms can be used to implement the prediction:

$$PT_n^{\text{pred}} = \mathbf{F}_n^T \boldsymbol{\theta}, \tag{6}$$

where $\mathbf{F}_n = [PT_{n-s}, \cdots, PT_{n-sl}]^T$, $\boldsymbol{\theta} = [\theta_1, \cdots, \theta_l]^T$, θ_i is the i_{th} predictor coefficient obtained by the training process, and l is the number of previous values, and s is the interval between every two previous values.

B. Prediction based difficulty algorithm

As aforementioned, if PT_n can be successfully predicted, we can improve the difficulty control algorithm to a stable and controllable one.

Using the linear predictor, an improved difficulty value d_n^P can be calculated from d_{n-1}, X_{n-1}, and PT_n^{pred}:

$$d_n^P = \frac{X_{\text{target}}}{PT_n^{\text{pred}}} (1 + \mu f(X_{n-1})) \times d_{n-1}. \tag{7}$$

However, this improved difficulty control algorithm cannot be used directly. The principle of linear prediction is to use the previous observed values to predict the current and future values, and in this case, only PoW terms can be used to predict PoW terms. In real applications, only the BPT can be observed. To satisfy the conditions of prediction, a PoW term calculation process is needed to complete the prediction based algorithm.

Since the BPT is random and even non-stationary, it is challenging to obtain the precise PoW term of the given block from observed data. Thus, two different methods are proposed to calculate the PoW term PT_n^{pred}.

The first method is to use the smoothed BPT

$$S_n = \frac{X_n + X_{n-1} + \cdots + X_{n-p_S+1}}{p_S}$$

with a smooth period p_S to replace PT_n. The prediction based difficulty control algorithm can be expressed as:

$$d_n^P = \frac{X_{\text{target}}}{S_n^{\text{pred}}} (1 + \mu f(X_{n-1})) \times d_{n-1}, \tag{8}$$

where $S_n^{\text{pred}} = [S_{n-s}, \cdots, S_{n-sl}]\boldsymbol{\theta}$, and $\boldsymbol{\theta} = [\theta_1, \cdots, \theta_l]^T$.

The second method is to use the integrated BPT

$$I_i = \frac{X_{(i-1)p_I+1} + X_{(i-1)p_I+2} + \cdots + X_{ip_I}}{p_I},$$

with an integration period p_I. Although the BPT observed at any given time is exponentially distributed and can be largely different from its PoW term, the average of a large amount of observed BPT will be closer to the average term in the period. With the increasing of the integration period p_I, the result of the calculated average term will be more accurate. The prediction of the integrated BPT I_i at the i_{th} period can be used to adjust the difficulty for all the blocks in the period:

$$d_n^P = \frac{X_{\text{target}}}{I_i^{\text{pred}}} (1 + \mu f(X_{n-1})) \times d_{n-1}, \tag{9}$$

where $I_i^{\text{pred}} = \mathbf{F}_i^{I\,T} \boldsymbol{\theta}$, $\mathbf{F}_i^I = [I_{i-s}, ..., I_{i-sl}]^T$, $i = \lfloor (n-1)/p_I \rfloor + 1$, and $\boldsymbol{\theta} = [\theta_1, \cdots, \theta_l]^T$.

However, the difficulty control algorithm in Eq. (7) still depends on the difficulty generated by the existing Ethereum algorithm. If we replace d with d^P since the n_{th} block, the new difficulty will start to diverge from the original difficulty because of the change of BPT, making the mining process unstable.

To further improve the stability of the proposed difficulty control algorithm, we can periodically update predictor coefficients to enhance the effect of prediction. After a given number of blocks L, we retrain our linear predictor with the latest observed values to maintain the prediction accuracy.

IV. Simulation Results

In the simulation, we use $800,000$ real BPT and difficulty from block number $7,300,001$ to $8,100,000$ published by Ethereum. To obtain the hashrate r_n for the simulation, we use the real difficulty d_n^{real} divided by the smoothed real BPT

$$S_n^{real} = \frac{X_n^{real} + X_{n-1}^{real} + ... + X_{n+1-P}^{real}}{P}$$

to estimate the hashrate, where the smoothed period $P = 1,000$. Since the hashrate may be changed suddenly due to the computational resources invested or withdrawn instantly by the miners, we also simulate this situation to compare the performance of the prediction based difficulty control algorithm and the current Ethereum algorithm. Parts of the Ethereum hashrate and the simulated hashrate are shown in Fig.1.

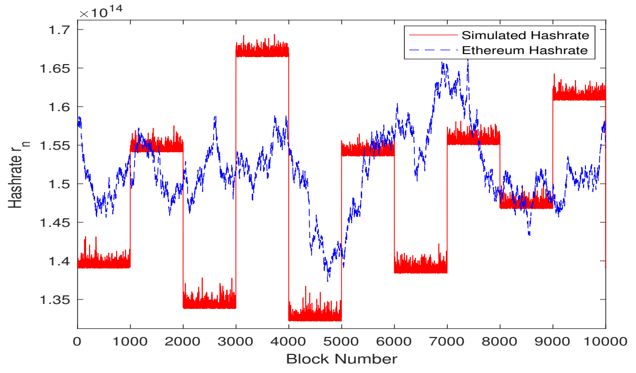

Fig. 1. Simulated hashrate vs. Ethereum hashrate ($l = 20, s = 1$)

A. Simulation results of linear prediction

As aforementioned, the proposed difficulty control algorithms predict the PoW term PT_n by the smoothed BPT S_n^{pred} and the integrated BPT I_i^{pred}. 50,000 observed values are used to obtain the predictor coefficients $\boldsymbol{\theta}$ in the PoW term and the smoothed BPT prediction. The results of the PoW term prediction are simulated and shown in Fig. 2.

From the simulation results, we can see that the prediction is precise, despite there still exists a one block delay compared with the real PoW term. The main reason is that the PoW term changes with the hashrate, which is determined by the miners and relatively irregular. Thus, the linear prediction can only use the term of the last block as the main frame and adjust it with the predicted trend. To further analyze the precision of prediction, we introduce the prediction mean square error (MSE) as:

$$\text{MSE} = \frac{1}{N}\sum_{n=1}^{N}(V_n - V_n^{\text{pred}})^2 \qquad (10)$$

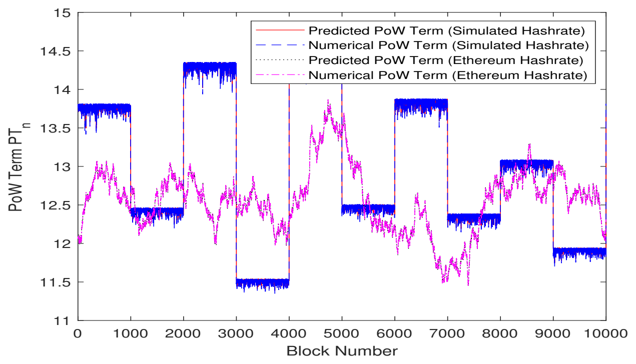

Fig. 2. Predicted PoW term using Ethtreum hashrate vs. Numerical PoW term ($l = 20, s = 1$)

where V_n is the desired value of the variable PT_n, S_n or I_i in the simulation, V_n^{pred} is the predicted value of the variable, which is PT_n^{pred}, S_n^{pred} or I_i^{pred}, and N is the number of predictions.

The prediction deviation for the PoW term using Ethereum hashrate is 3.1546×10^{-4}, which is quite low. Thus, even when the largest deviation occurs, the difference between PT_n and X_{target} is still low compared with the current design of Ethereum.

The simulations of the smoothed BPT are also implemented. The results are similar to the prediction of the PoW term. The predicted smoothed BPT is close to the numerical smoothed BPT and the prediction deviation is only 3.7144×10^{-4}.

The simulations of the integrated BPT I_i are also implemented. We use $p_I = 1,000$ at first to integrate the blocks and implement the prediction. The number of training data in this case is only 500 because of the reduction of the data size caused by integration and the prediction results are shown in Fig. 3.

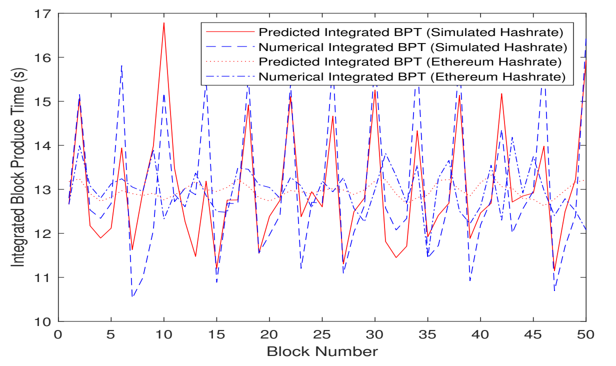

Fig. 3. Predicted integrated BPT vs. Numerical integrated BPT ($l = 3, s = 1$)

From the results we notice that the prediction of the integrated BPT is not as accurate as the other two predictions. However, the general trend can be predicted, which is important to correct the deviation. The prediction deviation is 0.2456, which is still acceptable. If we further increase the integration period to $5,000$, the prediction deviation is further decreased to 0.0241 even when the training number is decreased to 100. The larger the integration period is, the more precise the prediction can be.

B. Performance of the difficulty control algorithm

In order to verify the effects of the linear prediction for the PoW term, we first assume an ideal situation that the PoW term can be perfectly calculated from the observed BPT. In other words, we only consider the effect of prediction. The simulation result is shown in Fig. 4.

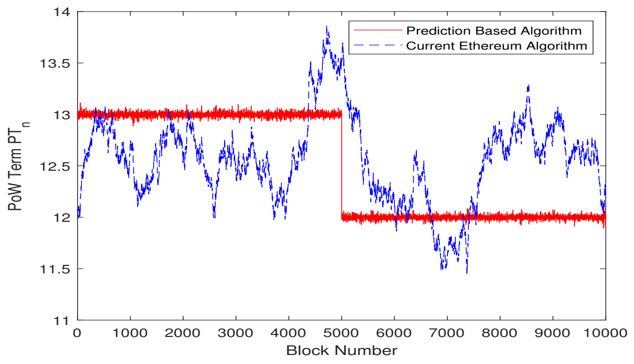

Fig. 4. PoW term of the prediction based algorithm vs. Current Ethereum algorithm

The prediction based difficulty control algorithm can largely improve the stability of the BPT. To further analyze the degree of improvement, we introduce the target deviation D_T to calculate the deviation from X_{target}:

$$D_T = \frac{\sum_{n=1}^{N}(PT_n - X_{\text{target}})^2}{N}. \quad (11)$$

The error D_T of prediction based algorithm is 3.98×10^{-4}, which is largely decreased from the 0.2420 of the original algorithm. Besides, the new proposed algorithm can be easily updated to achieve arbitrary targeted time only by changing the value of X_{target}. As shown in Fig. 4, if we change X_{target} to $12s$ from the $5,000_{th}$ block, the PT_n of the prediction based algorithm can converge to $12s$.

The simulation result of prediction based algorithm on generated hashrate is shown in Fig. 5. The improvement is even better than on historic hashrate. It shows that the prediction based algorithm can trace the sudden change of hashrate more quickly and precisely compared with the current Ethereum algorithm.

In conclusion, as long as the PoW term can be successfully obtained, a controllable and more stable difficulty control algorithm can be used to improve the performance of Ethereum.

C. Performance of the difficulty control algorithm using smoothed BPT

In Sec. IV-B, we have shown that as long as the PoW term can be perfectly observed, the prediction based algorithm can exert its effect. However, methods to calculate the term is

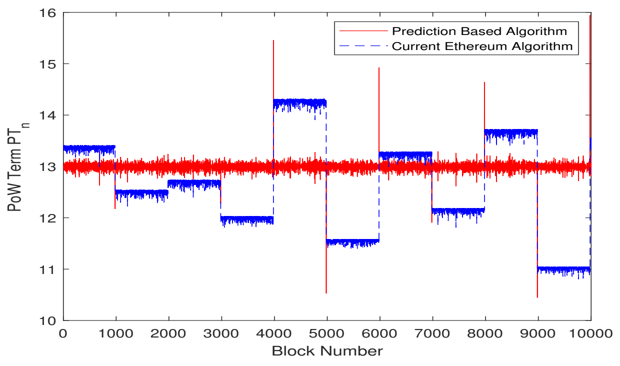

Fig. 5. PoW term of the prediction based algorithm vs. Current Ethereum algorithm

still needed. The simulation of the difficulty control algorithm using the smoothed BPT to represent the PoW term is shown in Fig. 6.

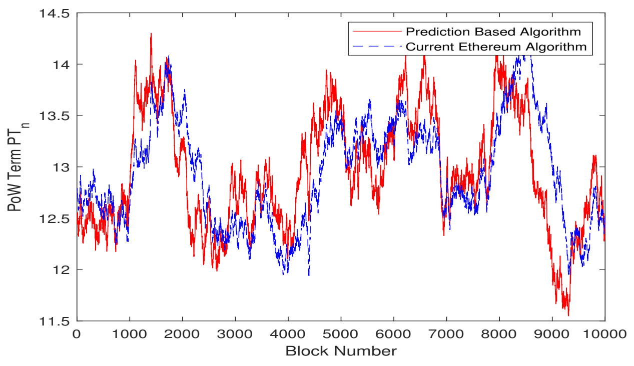

Fig. 6. PoW Term of the prediction based algorithm (Smoothed) vs. Current Ethereum algorithm

The result shows that the prediction based algorithm using smoothed BPT cannot improve the PoW terms. The deviation using prediction based algorithm is even larger than using the Ethereum algorithm. The main reason is that the smoothed BPT can be considered as the average PoW term of the current block and $p_S - 1$ previous blocks. Thus, if we use the average term to represent the term at the n_{th} block, there is a large difference.

Thus, the smoothed BPT is not a practical calculation method to implement the prediction based difficulty control algorithm.

D. Performance of the difficulty control algorithm using integrated BPT

Different from the smoothed BPT, the integrated BPT calculates the average of p_I PoW terms, which is much closer to the true value. Using the integrated BPT to represent the average term, we can reconstruct the average PoW term within an integration period. The simulation result is shown in Fig. 7.

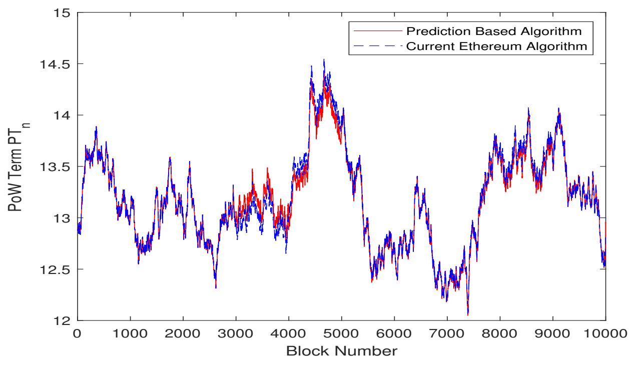

Fig. 7. PoW term of the prediction based algorithm (Integrated) vs. Current Ethereum algorithm

Although the new rate calculated by the prediction based difficulty control algorithm is very close to the original one, the improvement is still visible. D_T is improved from 0.24 to 0.2208. The main reason why the improvement using the integrated BPT is not as good as the improvement in the ideal situation is that the integration mitigate the deviation. The longer the integration period is, the more powerful the mitigation effect is. However, short integration period causes the difference between the integrated BPT and the average rate because of the large impact of outliers. As a result, the performance with shorter integration period becomes worse. The simulation using the generated hashrate is also implemented and the result is shown in Fig. 8. It shows that the prediction based algorithm using integrated BPT is more powerful in sudden change situations. The D_T is decreased from 1.0923 to 0.4409.

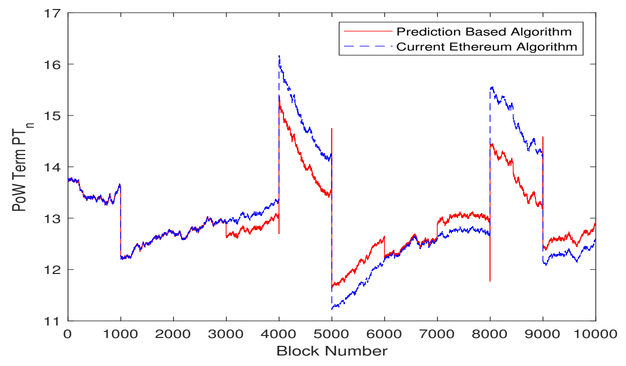

Fig. 8. PoW term of the prediction based algorithm (Integrated) vs. Current Ethereum algorithm ($X_{\text{target}} = 13s$)

In conclusion, although the improvement from the prediction is reduced by the integration, the integrated BPT can be used to reconstruct the PoW term in a given period.

V. Conclusion

Difficulty control is an important part of the Ethereum blockchain. Although the current Ethereum algorithm can achieve around an average $13s$ BPT, the algorithm is not flexible and adaptive to track the hashrate changes. Based on linear prediction, we propose a prediction based difficulty control algorithm to mitigate these two problems. The prediction can reach high accuracy with small MSE. The prediction based algorithm can perfectly work if the PoW term can be observed from past values. The smoothed BPT S_n and the integrated BPT I_n are used to calculate the PoW term from the observed BPT. Although the smoothed BPT cannot achieve our goal, the integrated BPT can be used to implement the difficulty control algorithm in a real situation. Although there are some improvements achieved now, the performance of the prediction based difficulty control algorithm can be further improved.

References

[1] C. Dwork and M. Naor, “Pricing via processing or combatting junk mail,” in *Proc. Annual International Cryptology Conference*. Springer, 1992, pp. 139–147.

[2] A. Back, “Hashcash-a denial of service counter-measure,” 2002. [Online]. Available: http://www.hashcash.org/papers/amortizable.pdf

[3] F. Tschorsch and B. Scheuermann, “Bitcoin and beyond: A technical survey on decentralized digital currencies,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2084–2123, 2016.

[4] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[5] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[6] D. Kraft, “Difficulty control for blockchain-based consensus systems,” *Peer-to-Peer Networking and Applications*, vol. 9, no. 2, pp. 397–413, 2016.

[7] D. Fullmer and A. S. Morse, “Analysis of difficulty control in bitcoin and proof-of-work blockchains,” in *Proc. IEEE Conference on Decision and Control (CDC)*, 2018, pp. 5988–5992.

[8] D. Meshkov, A. Chepurnoy, and M. Jansen, “Short paper: Revisiting difficulty control for blockchain systems,” in *Proc. Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 2017, pp. 429–436.

[9] G. Bissias, D. Thibodeau, and B. N. Levine, “Bonded mining: Difficulty adjustment by miner commitment,” in *Proc. Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 2019, pp. 372–390.