Block Interval Adjustment Based on Block Propagation Time in a Blockchain

Masumi Arakawa[†] and Kazuyuki Shudo[†‡]
[†]*Tokyo Institute of Technology, Tokyo, Japan*
[‡]*Kyoto University, Kyoto, Japan*

***Abstract*—Transaction confirmation throughput is one of the most important performance metrics of public blockchains. Throughput can be improved by shortening the block generation interval, but that approach could sacrifice security because shorter intervals increase the fork rate. Our proposed method improves the throughput by adjusting the block generation interval to be as short as possible. Part of the nodes observe the block propagation times over the blockchain network, and the interval is determined so as not to increase the fork rate. Our simulation showed the proposed method works as it is supposed to.**

***Index Terms*—blockchain, throughput, block generation interval, block propagation time**

I. Introduction

Blockchain is a distributed ledger technology proposed by Satoshi Nakamoto in 2008 as one of the technologies that make up the cryptocurrency Bitcoin [1]. Blockchain has been developed and is used in various fields, including cryptocurrencies such as Bitcoin and Ethereum [2], supply chain management and electronic voting. Blockchain makes it possible to securely manage information without using a centralized system, even when multiple malicious nodes are participating in the network. Despite this advantage, blockchain faces some challenges, including low transaction throughput and long confirmation time. For example, Bitcoin's throughput was initially approximately 7 transactions per second (TPS) and is now reported to be up to about 27 TPS [3], which is significantly lower than the average throughputs of typical non-decentralized payment systems such as Visa and PayPal, which have throughputs of approximately 1,700 TPS [4] and 600 TPS [5], respectively. This low transaction throughput of blockchain causes significant delays in transaction confirmation and a significant increase in transaction fees to have the transaction processed when demand exceeds the blockchain's capacity. This is a major problem for blockchain as a currency or payment system and limits the wide range of its applications.

These problems can be resolved by shortening the block generation interval, which is currently fixed in a blockchain, or by increasing the block size to increase the number of transactions that can be contained in a single block. However, if the block generation interval is shortened so that the block propagation time is no longer sufficiently shorter than the block generation interval, the fork rate will increase. Similarly, increasing block size increases block propagation time and consequently the fork rate. An increase in the fork rate is undesirable because it worsens security. The fork rate is affected by the block propagation time and the block generation interval; the shorter the block generation interval and the longer the block propagation time, the higher the fork rate is [6]. Block propagation times change due to changes in topology, the introduction of new protocols in blockchain networks, and changes in Internet bandwidth. For example, according to a website that monitors the Bitcoin network, block propagation times have been reduced in recent years [7]. In this situation of shorter block propagation time, it is possible to increase the throughput without changing the fork rate.

Therefore, we propose a method to adjust the block generation interval based on the block propagation time. The proposed method adjusts the block generation interval to maintain the set fork rate, which allows the block generation interval to be shorter than it currently is, thereby achieving good throughput while simultaneously maintaining security by keeping the fork rate within an acceptable range. Moreover, the block generation interval can quickly adjust to changes in the block propagation time.

Section 2 shows the background and related work of this paper. Section 3 describes block interval adjustment based on block propagation time, the proposed method. Section 4 shows the evaluation results. Finally, Section 5 summarizes the paper.

II. Background

This section provides an overview of block propagation, forking, throughput, and the block generation interval in a blockchain; we also describe related research.

A. Block propagation time and forking

In blockchains that use proof-of-work as the consensus algorithm, new blocks are generated by mining, which entails competition between nodes, called miners, who try to generate new blocks. In some blockchains that employ proof-of-stake, new blocks are generated stochastically through competition among nodes. In a blockchain, which is a distributed system, data are shared in the system by means of repeated transfers between nodes, so it takes time from when a new block is generated by a node to when it reaches all the nodes through the network, and that time is called the block propagation time.

During block propagation, a node that has not yet received the block continues mining because it does not know that the

This work was supported by JSPS KAKENHI Grant Number JP21H04872.

block has been generated, and it may generate a different block of the same height. When multiple blocks of the same height are generated, the chain splits into multiple branches, which is called a fork. The occurrence of a fork is undesirable from a security standpoint [8] because it distributes the computational power of miners, making the blockchain more vulnerable to 51% attack. The fork rate and block propagation time are closely related, and a study by Decker et al. [6] showed that with a fixed block generation interval, the fork rate decreases as the propagation time decreases. Therefore, a shorter block propagation time is preferred.

Various research has been conducted with the aim of reducing the block propagation time, and various proposals have been made, such as Graphene [9], which significantly reduces block size, and mechanisms for selecting neighboring nodes [10] [11], etc. Real-world blockchains such as Bitcoin have incorporated research results, and the propagation time is decreasing [7]. Furthermore, this decrease is due to the introduction of the compact block relay (CBR) [12] and relay networks [13] [14]. Changes in Internet bandwidth and the regional proportions of nodes may also affect the block propagation time.

B. Transaction confirmation throughput

A transaction is "confirmed" when it is included in a new block. A transaction is further confirmed when a new block is added after the block to which it belongs. When there are a certain number of confirmations (e.g., six for Bitcoin), the transaction is considered almost certainly secure. The throughput of such confermations is measured in transactions per second (TPS) and is one of the performance indicators of the blockchain. Simply put, throughput is calculated by dividing the number of transactions that fit in a single block by the block generation interval. In practice, the average throughput is often considered because there are many different types of transactions of different sizes and the number of transactions in a single block varies.

While there have been many studies, proposals, and implementations of methods to increase throughput, the simplest method is to increase the block size and shorten the block generation interval. However, increasing the block size increases the time it takes to propagate the block, which increases the fork rate unless the propagation speed is increased. Large blocks may also lead to centralization because individual users in the network will not be able to propagate blocks efficiently and it will be difficult to validate a large number of transactions within a given time period. There is also a problem with the method of shortening the generation interval, as described below.

C. Block generation interval

Shortening the block generation interval improves TPS. However, some studies have shown that shortening the block generation interval is problematic [8] [15]. The block generation interval and the fork rate are related, and Gervais et al. showed, using a simulator, that shortening the block generation interval increases the fork rate of the blockchain [15]. To shorten the block generation interval without increasing the fork rate and to improve scalability, the block propagation time must be reduced. This issue also motivates the shortening of the block propagation time. In practice,the average interval between block generation is fixed in most existing blockchains. In Bitcoin, for example, the average block generation interval is adjusted to 10 minutes by changing the mining difficulty according to the total computing power of the miners. Therefore, even if the block propagation time is reduced, as is actually happening with Bitcoin, it does not contribute to improved scalability.

D. Related work

For long-term use of the blockchain, it is desirable to have a mechanism to adjust the block generation interval or block size from a scalability perspective. In their study focusing on fairness among miners, Kanda et al. proposed a method to adjust the block generation interval so that the fork rate reaches a set value based on the target fork rate and the actual observed fork rate using the following formula [16].

$$newBlockInterval = oldBlockInterval \frac{ForkRate}{targetForkRate} \tag{1}$$

Kanda's method targets long-term changes in block propagation time, and their simulation experiments confirmed its long-term effectiveness. However, since a long period of time is required to observe the fork rate with high accuracy, their method has the disadvantage that when propagation time changes quickly, it takes time for the change to be reflected in the block generation interval. Shortening the time period to accelerate the response results in large errors and unintended fluctuations in the block generation interval.

III. Proposed method

This section describes the block generation interval adjustment method based on block propagation time proposed in this research.

A. Adjustment method

Let X_b be a random variable of the time between the generation of a block and the generation of the next block, and let t_b be the block generation interval. The probability P_b of a block being generated in the entire Blockchain network at any given time is kept constant by adjusting the difficulty of block generation, and it can be expressed using t_b as follows.

$$P_b = Pr[X_b < t + 1 | X_b > t] \approx \frac{1}{t_b} \tag{2}$$

When the block generation interval is fixed, changes in propagation time affect the fork rate [6]. Let $f(t)$ be the expected value of the fraction of nodes that have received the block at time t with 0 as the time when the block was generated; the fork rate can be expressed by the following formula using P_b and $f(t)$ [6].

$$Pr[F \geq 1] = 1 - (1 - P_b)^{\int_0^\infty (1-f(t))dt} \tag{3}$$

The equation can be transformed by setting the target fork rate as P_{fork}.

$$P_{\text{fork}} = 1 - (1 - \frac{1}{t_b})^{\int_0^\infty (1-f(t))dt} \quad (4)$$

$$(1 - \frac{1}{t_b}) = (1 - P_{\text{fork}})^{\frac{1}{\int_0^\infty (1-f(t))dt}} \quad (5)$$

$$t_b = \frac{1}{1 - (1 - P_{\text{fork}})^{\frac{1}{\int_0^\infty (1-f(t))dt}}} \quad (6)$$

On the basis of the above equation, once $\int_0^\infty (1 - f(t))dt$ is known, the block generation interval can be determined to match the set target fork rate.

B. Propagation time measurement

To measure $\int_0^\infty (1 - f(t))dt$, information on the time at which each participating node received the block is required. For this purpose, all nodes must have common clock information and share the time of block reception. The common clock can be based on real time and synchronized at each node using NTP with an external time server as the source, by using GPS time information, or by introducing a Byzantine fault-tolerant time synchronization protocol that operates on the blockchain network. The sharing of receive time information is a new type of communication, and there is concern that if all nodes share information, the amount of communication for each node will increase as the number of nodes increases and the number of blocks per unit time increases. The additional requirement to synchronize with a common clock and increased communication costs may increase the difficulty of participating in a blockchain network, which is undesirable.

Thus, we introduce a method in which a certain number of nodes with common time information take measurements and share the result with all nodes. With this method, the difficulty of participation is solved because the increase in communication volume is small for nodes that do not participate in the measurement. The measurement results are shared by all nodes, which requires communication, but methods to improve efficiency, such as building a distribution tree, can be considered.

In both method, an attack such as a malicious node sending false time information may occur. If this occurs, it will affect the measurement results and cause the fork rate to deviate from the set target value. Possible mitigation measures include excluding time outliers and scoring the nodes participating in the measurement to eliminate suspicious nodes, but it is difficult to prevent completely.

IV. EVALUATION

To evaluate the effectiveness of the proposed method, experiments were conducted using a public blockchain simulator, SimBlock [17]. The main parameters used in the experiments are shown in Table I. The initial values for internode latency and bandwidth were the six regional values measured by Nagayama et al. [18]. The Bitcoin protocol is designed to relay isolated blocks (not main chain blocks) under certain circumstances, and SimBlock follows suit. Therefore, in the implementation of Kanda's method for observing and comparing fork rates, we modified SimBlock to relay blocks outside of the main chain under any circumstances to share the fork rate among the nodes. In the experiment, only a limited number of nodes measured the time of block reception.The nodes participating in the measurement were selected at random.

TABLE I
MAIN PARAMETERS USED IN THE EXPERIMENTS.

Number of nodes	5,000
Block generation interval	Variable
Block size	534 KB
Compact block size	18 KB
Percentage of CBR nodes	96.8 %
Bandwidth	Variable
Latency	Values by Nagayama et al. [18]

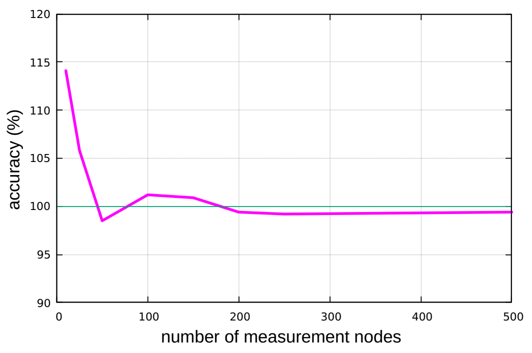

Fig. 1. Effect of different numbers of measuring nodes.

A. Effect of different numbers of measuring nodes

To see how much the number of measuring nodes affects the results of $\int_0^\infty (1 - f(t))dt$ when not all nodes participate in the measurement, we varied the number of measuring nodes and observed the change in error. The ratio of the average value calculated from the information of all nodes to the average value calculated from the information of some randomly selected measuring nodes is shown in Figure 1. For 10 nodes, or 0.2 % of the total number of nodes measured, the difference from the measurement results at all nodes is 14.1 %, resulting in a large error. Increasing the number of measuring nodes to 50 improves the difference to 2.2 %, and at 100 nodes, the difference is only 1.2 %. It can be seen that even 100 nodes, which is 2% of the total, gives a result close to the correct value. In each subsequent experiment, the number of measuring nodes was set to 100.

B. Constant bandwidth

Simulations were performed by applying the proposed method. First, we assessed how the fork rate and block generation interval change for the proposed method when the bandwidth between nodes is constant. To clarify the effect, the initial block generation interval was set shorter so that the fork rate would be higher. The target fork rate was set to 5.00 %, and the interval was first adjusted after 4,000 blocks

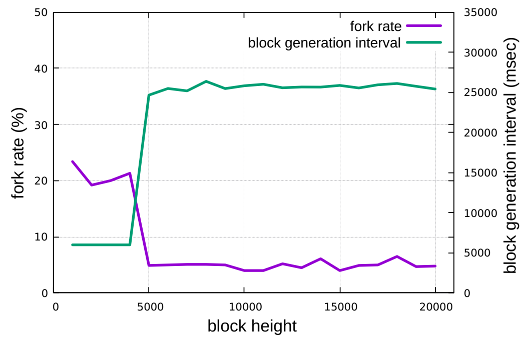

Fig. 2. Fork rate and block generation interval with constant bandwidth.

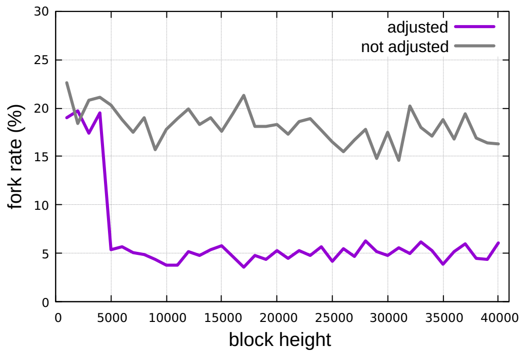

Fig. 3. Fork rate with increasing bandwidth (with CBR).

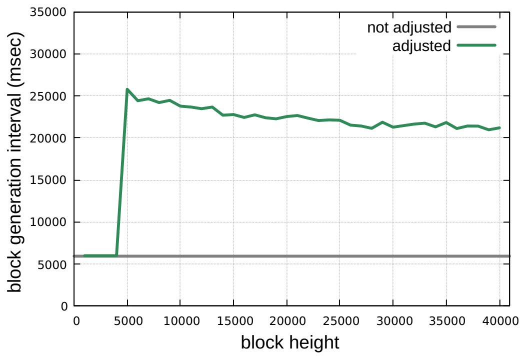

Fig. 4. Block generation interval with increasing bandwidth (with CBR).

and then adjusted every 1,000 blocks thereafter. In subsequent experiments, the first adjustment was made at block height 4,000, which is why there are large fluctuations at block height 4,000. Experiments were conducted up to 20,000 blocks, and the fork rate and block generation interval for every 1,000 blocks are shown in Figure 2. The fork rate was high in the first 4,000 blocks before the adjustment began, but after the interval adjustment was made, the rate remained around the set 5.00 %. The average fork rate for the 16,000 blocks after the start of the adjustment is 4.90 %, which is close to the set value. The block generation interval was maintained at around 26,000 milliseconds, although there are some small fluctuations.

C. Increasing bandwidth

In the case of increasing bandwidth, we observed the same changes in the fork rate and block generation interval as in our previous experiments. Experiments were conducted with and without CBR, with the bandwidth set to increase by a factor of 1.4 every 4,000 blocks. As in the constant bandwidth case, the block generation interval was adjusted with proposed method, and simulations were performed up to 40,000 blocks. The results were then compared to those in the case with no adjustment. Likewise, the target fork ratio was set to 5.00%, with the first adjustment made at a block height of 4,000 and subsequent adjustments made every 1,000 blocks thereafter. As the bandwidth increases every 4,000 blocks, the block propagation time decreases every 4,000 blocks; therefore, the block generation interval becomes shorter.

The results are shown in Figures 3, 4, 5, and 6. Without interval adjustment, the fork rate gradually decreases with increasing bandwidth. On the other hand, when the proposed method is used to adjust the spacing, the fork rate is generally maintained approximately at the set level of 5.00 %. The average fork rate for the 36,000 blocks after the start of the adjustment is 5.01% with CBR and 4.86% without CBR. The block generation interval is adjusted to become shorter as the bandwidth increases and the propagation time decreases. In the case without CBR, the size of blocks transferred between nodes is much larger than that in the case with CBR, and the effect of the bandwidth on the propagation time is greater. Therefore, the effectiveness of the proposed method is well demonstrated, and the difference in fork rate between the case with and without interval adjustment is large.

D. Decreasing bandwidth

In the case of decreasing bandwidth, we also observed the same changes as in the previous experiments: we set the bandwidth to decrease by a factor of 0.7 every 4,000 blocks and conducted experiments with and without CBR. The initial block generation interval was set longer than that in other experiments to reduce the initial fork rate. As before, the block generation interval was adjusted with the proposed method, and simulations were run up to 40,000 blocks and compared to the case with no adjustment. The target fork rate and the first adjustment are the same as before. As the bandwidth decreases, the block propagation time increases every 4,000 blocks; therefore, the block generation interval becomes longer.

The results are shown in Figures 7, 8, 9, and 10. Without interval adjustment, the fork rate gradually increases with decreasing bandwidth. On the other hand, when the proposed method is used to adjust the interval, the fork rate is generally maintained at approximately the set level of 5.00 %. The average fork rate for the 36,000 blocks after the start of the adjustment was 5.25% with CBR and 5.45% without CBR, slightly higher than the set rate of 5.0%. In the proposed method, for each 1,000 blocks between the bandwidth change and the next adjustment, the block generation interval is

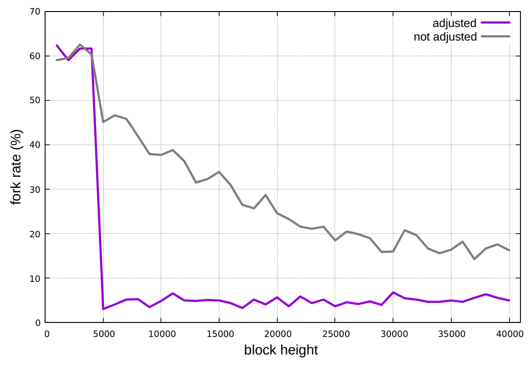

Fig. 5. Fork rate with increasing bandwidth (without CBR).

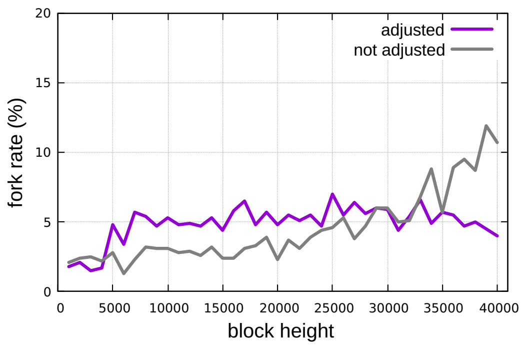

Fig. 7. Fork rate with decreasing bandwidth (with CBR).

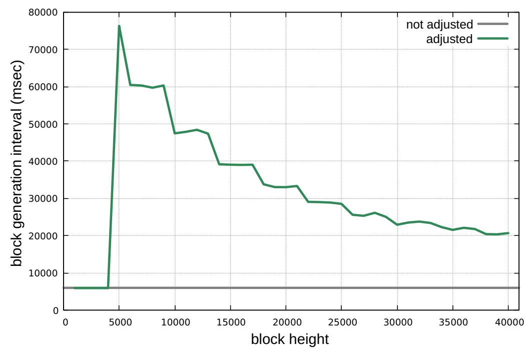

Fig. 6. Block generation interval with increasing bandwidth (without CBR).

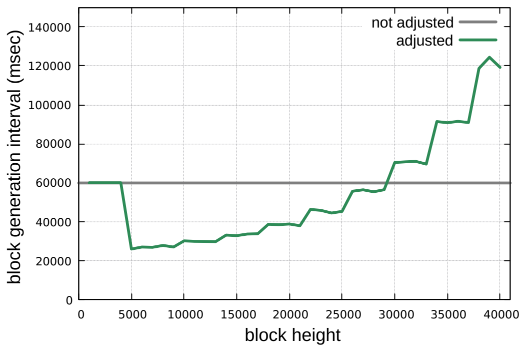

Fig. 8. Block generation interval with decreasing bandwidth (with CBR).

adjusted based on the propagation time before the bandwidth change, which sets the generation interval shorter than the ideal value. Therefore, in every 1,000 blocks after the bandwidth change, the fork rate exceeds the set value, which is the cause of the deviation from the set value. The graphs show that after the bandwidth change, the fork rate is higher than 5.00 % for, e.g., blocks 16,000-17,000, 24,000-25,000, and 32,000-33,000. The reason for the larger deviation from the target fork rate in the case without CBR is that the effect of bandwidth changes on the fork rate is larger than that in the case with CBR. As the incremental bandwidth decreases and the propagation time increases, the block generation interval is adjusted to increase every 4,000 blocks.

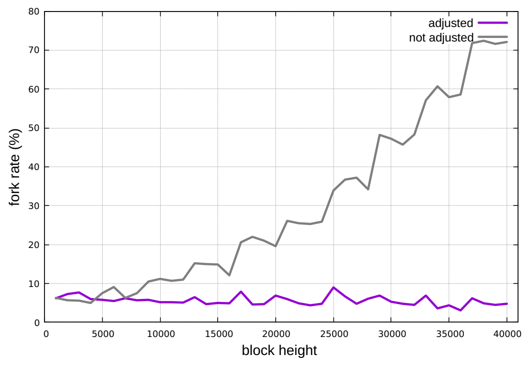

Fig. 9. Fork rate with decreasing bandwidth (without CBR).

E. Comparison with existing method

A comparison was made with the method of adjusting the block generation interval based on the fork rate by Kanda et al [16]. Experiments were conducted up to a block height of 24,000, with the bandwidth set to decrease by a factor of 0.1 at a block height of 8,000 and then return to a factor of 1.0 at a block height of 16,000. CBR was not used in this experiment, and the target fork rate and the first adjustment are the same as before, 5.00 % and 4,000 blocks, respectively. The adjustment interval was set to 500 blocks for the proposed method and 500, 1,000, and 2,000 blocks for the method of Kanda et al. The results are shown in Figures reffig:compare and reffig:compare. When the bandwidth decreases at a block height of 8,000, there is a temporary increase in the fork rate

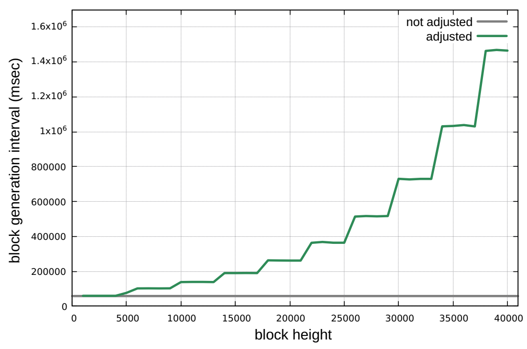

Fig. 10. Block generation interval with decreasing bandwidth (without CBR).

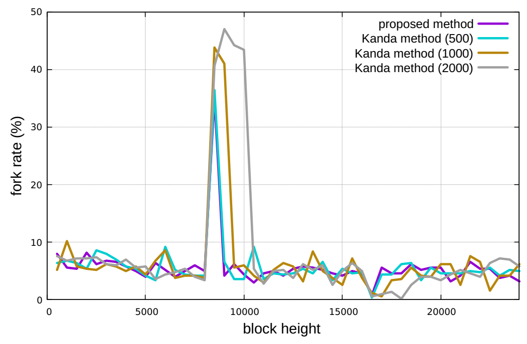

Fig. 11. Comparison of fork rate when the bandwidth changes.

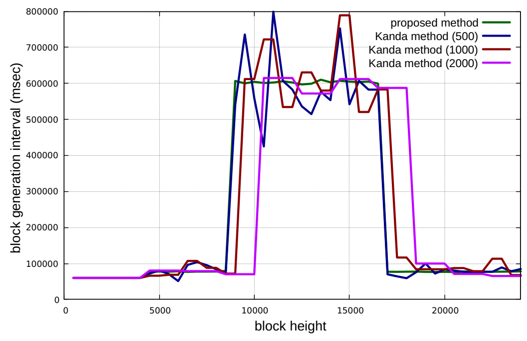

Fig. 12. Comparison of block generation intervals when the bandwidth changes.

for all methods, but the shorter the adjustment interval is, the more quickly the fork rate is adjusted to approach the set rate of 5.00 %. When the bandwidth is restored at a block height of 16,000, there is a similar temporary decrease in the fork rate, but again, the shorter the adjustment interval is, the more quickly it returns to normal. Moreover, for the block generation interval, the shorter the adjustment interval is, the shorter the time between the bandwidth change and the corresponding adjustment of the generation interval. Shorter adjustment intervals are better at following changes in propagation time. On the other hand, looking at the fluctuations in the block generation interval, in the proposed method, the generation interval remains constant when the bandwidth is constant. By contrast, Kanda et al.'s method inadvertently raises or lowers the generation interval when the adjusting interval is short. As a result, the variability of the fork rate also increases.

V. Conclusion

Scalability is one of the major challenges in blockchain. This paper proposes a method to adjust the block generation interval based on the block propagation time. By focusing on the close relationship between the block generation interval, fork generation rate, and block propagation time, the proposed method makes it possible to adjust the block generation interval to maintain the fork rate. This prevents the deterioration of security caused by an increase in the fork rate, while maximizing the improvement in scalability. The effectiveness of the proposed method was confirmed and compared with that of existing methods using a simulator.

Future work will include improving the detailed method for measuring propagation time and investigating methods for addressing malicious nodes.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[2] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger EIP-150 REVISION."
[3] E. Georgiadis, "How many transactions per second can bitcoin really handle ? theoretically," *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 416, 2019.
[4] Visa, https://usa.visa.com.
[5] "PayPal, Inc — PayPal Q3-21 Investor Update," https://investor.pypl.com/.
[6] C. Decker and R. Wattenhofer, "Information propagation in the Bitcoin network," in *IEEE P2P 2013 Proceedings*, 2013, pp. 1–10.
[7] "Bitcoin Network Monitor - DSN Research Group, KASTEL @ KIT," https://dsn.tm.kit.edu/bitcoin/, Accessed: Dec. 26. 2021.
[8] Y. Sompolinsky and A. Zohar, "Secure High-Rate Transaction Processing in Bitcoin," in *Financial Cryptography and Data Security 2015 (FC15)*, R. Böhme and T. Okamoto, Eds., 2015, pp. 507–527.
[9] A. P. Ozisik, G. Andresen, B. N. Levine, D. Tapp, G. Bissias, and S. Katkuri, "Graphene: efficient interactive set reconciliation applied to blockchain propagation," in *Proceedings of the ACM Special Interest Group on Data Communication*, 2019, pp. 303–317.
[10] Y. Aoki and K. Shudo, "Proximity Neighbor Selection in Blockchain Networks," in *Proc. 2nd IEEE International Conference on Blockchain (IEEE Blockchain 2019),*, 2019, pp. 52–58.
[11] H. Matsuura, Y. Goto, and H. Sao, "Region-based Neighbor Selection in Blockchain Networks," in *Proc. 2021 IEEE International Conference on Blockchain (IEEE Blockchain 2021)*, 2021, pp. 21–28.
[12] M. Corallo, "Compact Block Relay (BIP-152)," https://github.com/bitcoin/bips/blob/master/bip-0152.mediawiki.
[13] R. Nagayama, R. Banno, and K. Shudo, "Identifying impacts of protocol and internet development on the bitcoin network," in *2020 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2020, pp. 1–6.
[14] K. Otsuki, R. Banno, and K. Shudo, "Quantitatively Analyzing Relay Networks in Bitcoin," in *2020 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2020, pp. 214–220.
[15] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the Security and Performance of Proof of Work Blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 3–16. [Online]. Available: https://doi.org/10.1145/2976749.2978341
[16] R. Kanda and K. Shudo, "Block Interval Adjustment Toward Fair Proof-of-Work Blockchains," in *2020 IEEE 36th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 2020, pp. 1–6.
[17] Y. Aoki, K. Otsuki, T. Kaneko, R. Banno, and K. Shudo, "SimBlock: A Blockchain Network Simulator," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 325–329.
[18] R. Nagayama, K. Shudo, and R. Banno, "Simulation of the Bitcoin Network Considering Compact Block Relay and Internet Improvements," *IEICE Technical Report; IEICE Tech. Rep.*, vol. 119, no. 460, pp. 179–183, 2020.