

Development of an Effective Node Replication Based Consensus Algorithm for Reliable Small-Scale Blockchain Networks

Siwakorn Pasawang^{*}, Junghun Kim^{*}, Sang-Il Choi[°]

ABSTRACT

This paper explores blockchain as a decentralized and distributed storage technology where each device stores identical transaction data without reliance on a third party for verification or control. Consensus algorithms, particularly those based on voting systems, validate transactions and ensure tamper-proof records by achieving agreement among participants. However, existing voting-based algorithms face challenges such as high message transmission overhead, increased latency, and reliance on hardware evaluation to determine node reliability. To address these limitations, this study proposes a novel consensus algorithm designed for small-scale blockchain networks. The proposed scheme defines a fixed number of nodes within an internal group to perform consensus, significantly reducing transmission latency, improving throughput, and minimizing message exchanges. Simulators for PBFT, QPBFT, and the proposed scheme were developed and evaluated under varying conditions. Results indicate that the proposed scheme outperforms PBFT and QPBFT in terms of throughput and message verification success rates, while maintaining stable performance. This research provides valuable insights into the design of efficient consensus algorithms for reliable, small-scale blockchain systems and lays the groundwork for future implementation and evaluation in real-world networks.

Key Words : Blockchain, Consensus Algorithm, Effective Node Replication, Small-scale Blockchain Network, Performance Evaluation

I. Introduction

Blockchain is a decentralized, distributed storage system where all nodes store identical transaction data without the need for a central authority to validate or manage it. This structure ensures the integrity and immutability of stored data, as altering information in a single block would invalidate all subsequent blocks in the chain^[1-2]. As a result, blockchain has become a foundational technology in systems requiring secure, transparent, and tamper-proof data management.

A key component of blockchain technology is the

consensus algorithm, which ensures that all nodes in the network agree on the validity of transactions before new blocks are added. Consensus algorithms are generally categorized into two main types: attribute-based mechanisms and voting-based mechanisms. Attribute-based methods, such as Proof of Work (PoW) and Proof of Stake (PoS), rely on computational power or the quantity and age of held coins, respectively. These methods are widely adopted in public blockchain systems due to their strong security guarantees. However, they suffer from high energy consumption, low scalability, and potential centralization, especially when a small number of nodes consistently gain the right to generate new blocks^[3-4].

※ This work was supported by research grants from Daegu Catholic University in 2024

♦ First Author : Kasetsart University Department of Computer Science, siwakorn.pas@ku.th, 정희원

° Corresponding Author : Daegu Catholic University School of Computer Software, sangilchoi@cu.ac.kr, 종신회원

* Daegu Catholic University School of Computer Software, fainal2@cu.ac.kr

논문번호 : 202501-014-C-RN, Received January 11, 2025; Revised March 28, 2025; Accepted April 18, 2025

To improve on these limitations, variants such as Proof of Useful Work (PoUW) and Casper have been proposed. PoUW aims to channel computational efforts into solving socially valuable problems, while Casper combines PoS with Byzantine Fault Tolerance (BFT) principles to reduce energy waste and enhance security^[5-6]. Despite these advances, attribute-based approaches remain resource-intensive and are not always suitable for lightweight, small-scale blockchain networks.

Voting-based consensus algorithms, such as Practical Byzantine Fault Tolerance (PBFT), offer an alternative that emphasizes communication rather than computation. PBFT allows nodes to reach consensus through a structured message exchange process, ensuring system reliability even when a fraction of nodes behave maliciously^[7]. However, PBFT suffers from significant message overhead and latency due to its multi-phase communication rounds. As the number of nodes increases, these issues worsen, making PBFT impractical for networks with tight performance constraints.

To address some of PBFT's inefficiencies, Quantified-role PBFT (QPBFT) was introduced. QPBFT incorporates a node reliability scoring system to classify nodes into roles such as management, voting, and candidate nodes, thereby limiting communication to a smaller set of trusted participants^[8]. While this approach reduces message complexity compared to PBFT, it also introduces overhead from reliability evaluation and may reduce robustness when the number of high-reliability nodes is low or when scoring fluctuates frequently.

In the context of small-scale blockchain systems, such as enterprise networks, campus research environments, or closed IoT systems, efficiency, speed, and simplicity are more important than large-scale scalability. These systems require consensus mechanisms that operate reliably with fewer nodes and limited resources, while minimizing communication overhead and delay.

To meet these demands, this paper proposes a novel consensus algorithm tailored for small-scale, permissioned blockchain environments. The proposed scheme eliminates the need for ongoing reliability

evaluation or dynamic role assignment by predefining a fixed internal group of representative nodes. Within this group, a leader node is selected based on a pre-determined priority value. The consensus process follows a streamlined prepare-confirm sequence, allowing fast verification and block creation while keeping communication overhead low. By reducing the number of nodes involved in consensus and simplifying message exchange, the proposed approach improves performance without sacrificing fault tolerance.

Simulation results demonstrate that the proposed algorithm outperforms existing methods such as PBFT and QPBFT in key performance metrics, including transmission latency, message overhead, and success rate of message verification. These findings suggest that the proposed method offers a practical and efficient solution for small-scale blockchain networks, where lightweight operation, reliability, and quick consensus are critical.

II. Existing Schemes

2.1 Practical Byzantine Fault Tolerance Algorithm (PBFT)

PBFT (Practical Byzantine Fault Tolerance) is a consensus algorithm based on the BFT model, designed to improve efficiency by selecting a leader node—called the primary—to receive client requests. To ensure safety, PBFT assumes a system of $n = 3f + 1$ nodes, tolerating up to f faulty nodes^[9]. The consensus process consists of three phases: pre-prepare, prepare, and commit.

In the pre-prepare phase, the client sends a request to the primary node, which then creates a pre-prepare message using the request data and broadcasts it to the backup nodes. Each backup node verifies the message's integrity and, if valid, creates a prepare message and sends it to all other nodes, including the primary. In the prepare phase, nodes compare received messages to confirm consistency. Once a node receives $2f + 1$ matching prepare messages, it proceeds to the commit phase by broadcasting a commit message.

Each node, upon receiving at least $2f + 1$ valid commit messages, confirms the transaction and adds

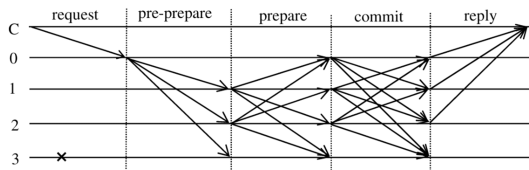


Fig. 1. Message Flow in the PBFT Consensus Algorithm

the block to its local blockchain. Finally, the node replies to the client, indicating that the consensus has been reached. Cryptographic techniques such as digital signatures or Message Authentication Codes (MAC) are used throughout to ensure data integrity^[10].

Despite its fault-tolerance and deterministic finality, PBFT suffers from performance limitations. The multi-phase message exchange process results in communication overhead, especially as the number of participating nodes increases. Moreover, its leader selection method using the formula $\text{primary node} = \text{view number} \bmod |R|$ can lead to temporary centralization and inefficiency, particularly during view changes. To mitigate these issues, implementations often limit the number of nodes in the consensus group^[11].

To address these limitations, particularly the communication overhead and the inefficiencies caused by static leader selection, an enhanced version called QPBFT (Quantified-role PBFT) was proposed. One notable improvement over PBFT is QPBFT, which enhances efficiency by assigning node roles based on a reliability scoring mechanism^[12].

2.2 Practical Byzantine Fault Tolerance Consensus Algorithm Based on Quantified-role (QPBF)

QPBFT (Quantified-role PBFT) improves upon the limitations of PBFT by reducing message overhead and time delays through a node reliability scoring mechanism. This mechanism evaluates each node based on two key attributes security and availability with multiple sub-metrics, as detailed in Table 1.

Each attribute, such as bit error rate, abnormal data ratio, identity fraud ratio, transmission success ratio, available storage capacity, and processor utilization, is normalized and weighted to compute a composite reliability score using the formula:

$$S(n_k) = \sum_{i=1}^6 w_i \times a_i^{n_k} \quad (1)$$

Nodes are then classified into four roles: management, voting, candidate, and ordinary nodes, as illustrated in Fig. 2. This classification is based on each node's calculated reliability score, which reflects its security and availability attributes. In normal operation, nodes with scores below 0.6 are filtered out as potentially malicious and are excluded from the consensus process to enhance overall network trustworthiness. Among the remaining reliable nodes, one is randomly selected to serve as the management node. This management node plays a central role by broadcasting the client's request to all voting nodes and coordinating the consensus process.

After verifying the integrity of the message, voting

Table 1. Node Reliability Evaluation System Table

First-level	Second-level	Scoring instructions	Normalized value	Weight
Security	Bit error rate	Ratio of error code to total code	a1	0.0675
	Abnormal data ratio	Ratio of abnormal data to original data	a2	0.2521
	Identity fraud ratio	Ratio of identity fraud to communication	a3	0.3743
Availability	Transmission success ratio	Ratio of successful transmission to communication	a4	0.0631
	Available storage capacity	Ratio of available capacity to total capacity	a5	0.0072
	Processor utilization	CPU resources used by running programs	a6	0.0197

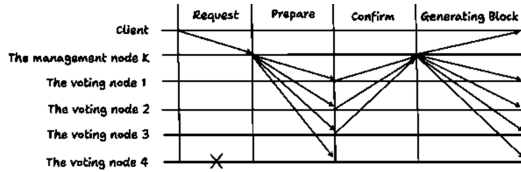


Fig. 2. Procedure of QPBFT Algorithm Procedure

nodes send confirm-messages back to the management node. If the management node receives at least $1 + n_v/2$ valid confirm-messages, it generates a new block within the designated cycle.

QPBFT effectively reduces the number of message exchanges by excluding low-reliability nodes from the consensus process. However, several challenges remain. As the total number of nodes increases, the number of exchanged messages still grows, even after filtering. Moreover, the success rate of message verification may decline due to the filtering mechanism. When only a few nodes pass the reliability threshold especially in cases of score imbalance faulty nodes may be included in the consensus group. To mitigate these issues, a representative group-based consensus structure can be adopted, where consensus nodes are pre-designated to form a fixed group^[13]. This approach minimizes node fluctuation and reduces communication load, as will be discussed in detail in the following sections.

III. Methodology

To address transmission latency and issues related to centralized systems, this paper proposes a new scheme. The concept of the proposed scheme involves creating an internal group to perform consensus^[14]. The design of this method reflects the fundamental principles of Byzantine Fault Tolerance (BFT), particularly the ability to maintain reliable consensus even in the presence of faulty or malicious nodes. The system assumes a permissioned blockchain environment with authenticated nodes and secure communication channels. Unlike other approaches, the proposed scheme does not define node roles based on credit values. Instead, it identifies the node with the highest priority value to serve as the leader node during normal operations. Fig. 3 shows the structure and mes-

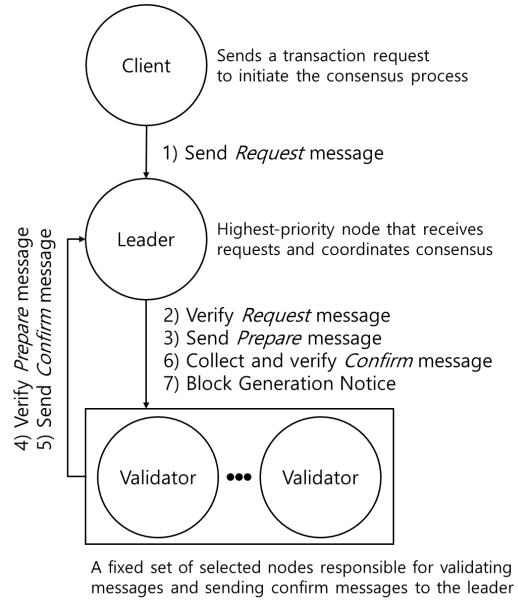


Fig. 3. Overview of the Proposed Consensus Scheme

sage flow of the proposed consensus scheme, including interactions between the client, leader, and representative nodes.

This paper explains the proposed scheme in terms of node attributes and normal operation, and it describes the development of simulators to compare the performance of PBFT, QPBFT, and the proposed scheme.

3.1 Node Attributes and Consensus Rights

In the proposed scheme, two attributes are used to define consensus for a node: the roles of nodes and the priority value. Nodes are categorized into two roles: master nodes and slave nodes. This reflects the use of a private blockchain system, which relies on a less computationally demanding consensus algorithm^[15]. According to the proposed scheme, master nodes can create new blocks within their own blockchain and actively participate in consensus. Slave nodes, on the other hand, cannot create new blocks but can still participate in the consensus process. Additionally, slave nodes are created by master nodes to assist in achieving consensus. This approach is designed to reduce the resources required for storing the blockchain and the overhead associated with generating new blocks. Each node's role is assigned a priority

value, which determines its eligibility to become the leader node responsible for handling client requests. The priority value is assigned to nodes when they are added to the blockchain system. In the simulation, this value is randomly generated between 0 and 100 for evaluation and comparison purposes.

3.2 Consensus Mechanism in Normal Operation

In the Request phase, the client sends a request with the message format $\langle m, \text{REQUEST}, -1 \rangle$ to the blockchain system. The consensus algorithm then selects n nodes to form a group of representatives for consensus, regardless of whether a node is faulty or not, as this helps address transmission delay issues. According to the QPBFT paper, transmission delay increases as the number of nodes in the blockchain system grows. After forming the group, the node with

the highest priority value, assigned during the initialization process, is selected as the leader node within the internal group. Next, the leader node receives the client request and generates a prepare-message in the format $\langle m, \text{PREPARE}, \text{node's ID}, D(m, \text{PREPARE}, \text{node's ID}) \rangle$. This message includes the request m , phase, node's ID, and a digest of the message log. The leader node broadcasts this prepare-message to the other nodes in the internal group, and each node stores it in its own message log. In the Prepare phase, when a node receives the prepare-message, it compares the digest in the message with a newly calculated digest, which is encrypted using the message, phase, and node's ID, to ensure the message has not been tampered with. This verification process utilizes a Message Authentication Code (MAC). If the verification is successful, the node generates a confirm-message in the format $\langle m, \text{CONFIRM}, \text{node's ID}, D(m, \text{CONFIRM}, \text{node's ID}) \rangle$ and forwards it to the leader node. If the node is faulty, it does not perform any action.

In the Confirm phase, when the leader node receives at least $n_{\text{internal}} \times (2/3)$ confirm-messages, which have not been tampered with, from other nodes in the internal group, the leader node notifies the other nodes, including external nodes, to generate a new block if they are master nodes. Otherwise, it takes no action. After generating a new block, each node in the internal group deletes the messages in its log and regenerates priority values randomly for the next round of consensus.

The proposed scheme is expected to solve existing problems for two main reasons. First, it helps reduce communication overhead between nodes by verifying that messages have not been tampered with, thereby preventing unauthorized modifications. Second, every node can participate in the consensus algorithm, either

Algorithm 1 Proposed Scheme Algorithm

1	The client sends a request $\langle m, \text{REQUEST}, -1 \rangle$.
2	A set of $\text{GR}[] \leftarrow$ Select the n nodes to be the group of representatives.
3	Leader-node \leftarrow the node which has the most priority value in the group of representatives and is not faulty node.
4	Leader-node receives a request in log.
5	Leader-node creates a prepare-message $\langle m, \text{PREPARE}, \text{node's ID}, D(m, \text{PREPARE}, \text{node's ID}) \rangle$
6	Leader-node broadcast a prepare-message to other nodes in the group of representatives.
7	LOOP FOR node from GR, where excludes Leader-node
8	The node receives a prepare-message in its log.
9	IF the node is not faulty node and verify a prepare-message which is not tampered.
10	The node creates a confirm-message $\langle m, \text{CONFIRM}, \text{node's ID}, D(m, \text{CONFIRM}, \text{node's ID}) \rangle$
11	The node sends a confirm-message to Leader-node.
12	END
13	IF Leader-node verifies confirm-messages and get at least $n_{\text{internal}} \times \frac{2}{3}$ messages
14	Leader-node notice other nodes to generate a new block for master nodes, otherwise they don't anything.
15	Clear log in each node.

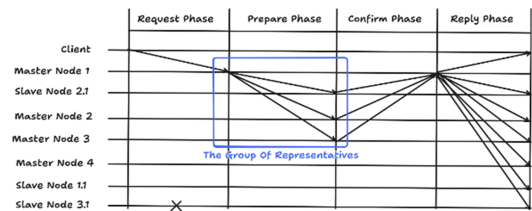


Fig. 4. Procedure of Proposed Algorithm

by serving as a leader node or by voting on the validity of messages. This approach enhances decentralization. Fig. 4 shows an overall message transmission flow and the consensus procedure.

IV. Simulation Environment

In the experiment on consensus algorithms, this paper developed simulators for PBFT, QPBFT, and the proposed scheme using Python 3.12.4. Since the simulators perform consensus on a local computer, certain node attributes and parts of the algorithms were modified or omitted to adapt to the simulation environment. When a client sends a request, all consensus algorithms process one request per round. As a result, PBFT does not define a sequence number in the transmitted message format, and PBFT and QPBFT use the same message format as the proposed scheme. For QPBFT, each node's attributes are generated within the range of 0 to 1 instead of being calculated normally, as proper evaluation is not feasible on a single computer.

V. Performance Analysis

Fig. 5 shows the environment for performance analysis. For the experimental setup to impact of the impact of the number of requests, each consensus algorithm is configured with 10 nodes, while the number of faulty nodes is a random value between 0 and one-third of the total number of nodes to simulate real-world conditions. In QPBFT, there are 6 management nodes and 4 voting nodes. In the proposed scheme, there are 6 master nodes, 4 slave nodes, and two group sizes (4 and 6 nodes) for comparison.

To evaluate transmission latency, the number of re-

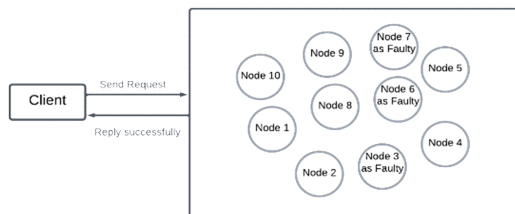


Fig. 5. Environment in Local Computer

quests is increased by 10 per iteration, ranging from 10 to 300 requests, to generate a graph. The probability of successfully proving messages is calculated using the formula: (2). This metric is used to assess the performance of the algorithms.

$$P(\text{Accepted Messages}) = \frac{n(\text{True Messages})}{n(\text{Requests})} \quad (2)$$

Based on the transmission latency results shown in Fig. 6, the efficiency of the proposed scheme and QPBFT remains stable, demonstrating satisfactory performance. The proposed scheme effectively reduces transmission latency. For 300 requests, the transmission latency of PBFT is 193.74 ms, while that of QPBFT is 31.77 ms. The proposed scheme shows a transmission latency of 37.55 ms when 4 nodes are selected as the group of representatives and 43.15 ms when 6 nodes are selected.

Based on the probability of successfully proving messages shown in Fig. 7, the proposed scheme demonstrates higher success rates than QPBFT while maintaining stability comparable to PBFT. The average probability for PBFT is 0.74, for QPBFT is 0.47, and for the proposed scheme is 0.91 when 4 nodes are selected as the group of representatives and 0.81 when 6 nodes are selected.

For the experimental setup to impact of the number of nodes, the number of nodes increases by 5 at each iteration, ranging from 10 to 100, to evaluate all simulators. Since QPBFT and the proposed scheme

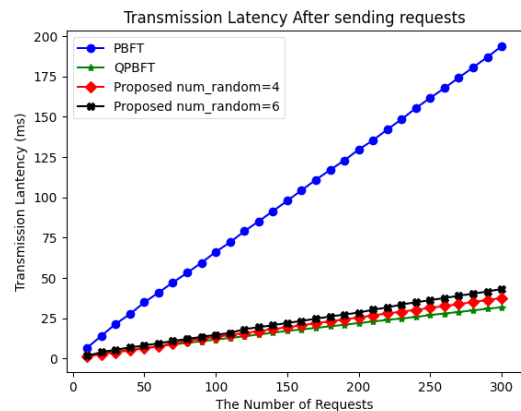


Fig. 6. Transmission Latency after Sending Requests

The Probability of Proving messages successfully after sending requests

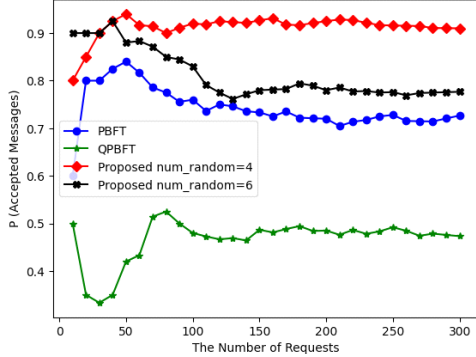


Fig. 7. The Probability of Proving Messages Successfully Graph after Sending Requests

define specific node roles, the QPBFT simulator randomly assigns the number of management nodes between 1 and half of the total nodes (using a seed) and calculates the number of voting nodes as $n(\text{voting nodes}) = n(\text{nodes}) - n(\text{management nodes})$. Similarly, the proposed scheme randomly assigns the number of master nodes between 1 and half of the total nodes (using a seed) and calculates the number of slave nodes as $n(\text{slave nodes}) = n(\text{nodes}) - n(\text{master nodes})$. The simulators send one request to measure the delay, which is then used to calculate throughput using the formula: (3) in transactions per second. To evaluate message exchanges, the simulators count the messages sent by each node.

$$\text{Throughput} = \frac{n(\text{request})}{\text{delay}} \quad (3)$$

Based on the throughput results shown in Fig. 8, the proposed scheme achieves higher throughput than other consensus algorithms. It can handle multiple requests to verify transactions as the number of nodes increases because it defines a specific group of representatives.

As shown in Fig. 9 and 10, due to the exponential increase in message exchanges in PBFT, the evaluation is divided into two graphs: one includes PBFT, and the other excludes it. The proposed scheme demonstrates stable message exchange rates for both 4-node and 6-node representative groups. In contrast, QPBFT shows an increase in message exchanges as

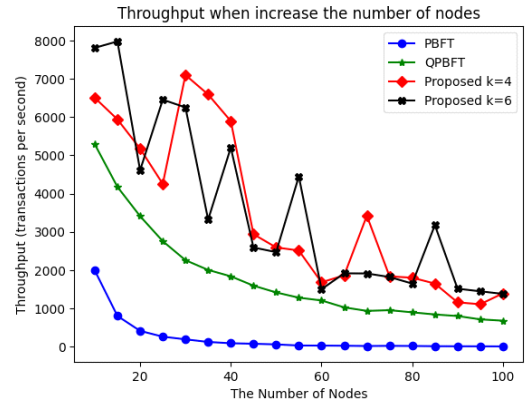


Fig. 8. Throughput When Increase the Number of Nodes

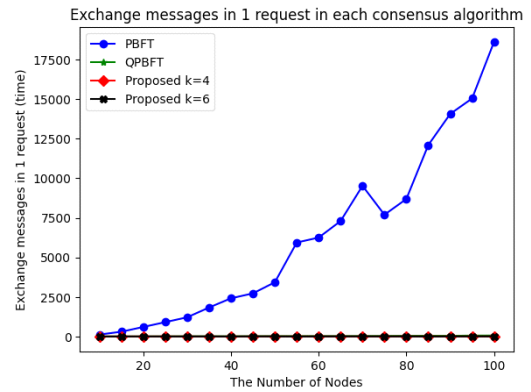


Fig. 9. Exchange Messages in 1 Request in Each Consensus Algorithm

Exchange messages in 1 request in each consensus algorithm without PBFT

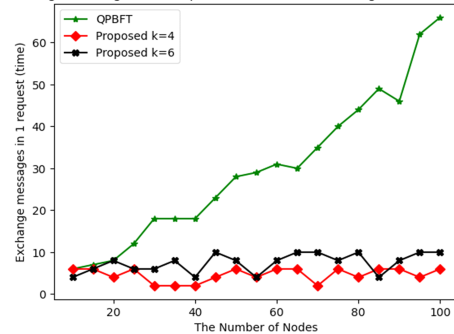


Fig. 10. Exchange Messages in 1 Request Graph without PBFT

the number of nodes increases, as illustrated in Fig. 10.

A summary of the experimental results is provided in the table below.

Table 2. Performance Comparison among Consensus Algorithms

Scheme	Transmission Latency	Success Rate of Message Verification	Message Overhead	Throughput (tx/sec)
PBFT	193.74	0.74	High	Low
QPBFT	31.77	0.47	Medium	Medium
Proposed Scheme (k=4)	37.55	0.91	Low	High
Proposed Scheme (k=6)	43.15	0.81	Low	High

As shown in Table 2, the proposed scheme demonstrates a significant improvement in both transmission latency and message overhead compared to PBFT and QPBFT. Particularly, it maintains a high success rate for message verification while achieving higher throughput, making it more suitable for small-scale blockchain networks where efficiency and reliability are critical.

VI. Conclusions

Since PBFT, QPBFT, and the proposed scheme were implemented as simulators for experiments on a local computer, testing on a real network was not possible. However, based on the evaluation of the workflows of all three consensus algorithms, the proposed scheme demonstrated the best results. Its performance, however, depends on the number of nodes selected for the internal group of representatives to perform consensus. If the internal group includes too many faulty nodes, this could lead to a decrease in the success rate of message verification, throughput, and other performance metrics.

Therefore, this study serves as a guideline for designing consensus algorithms for reliable small-scale blockchain systems. Finally, we hope to further develop consensus algorithms that can operate on real networks, allowing for real-world evaluations and implementations to facilitate training on the practical use of blockchain technology.

References

- [1] MD Masuduzzaman, A. Islam, J. W. Seo, and S. Y. Shin, "A Blockchain-enabled MEC-assisted CO₂ emission reduction scheme using internet of things," *J. KICS*, vol. 45, no. 11, pp. 2022-2028, 2020. (<https://doi.org/10.7840/kics.2020.45.11.2022>)
- [2] D. Arora and V. Wasson, "Blockchain based voting decentralized application," *2023 14th ICCNT*, pp. 2475-2674, Jul. 2023. (<https://doi.org/10.1109/ICCCNT56998.2023.10307528>)
- [3] P. T. Daely, J. M. Lee, and D. S. Kim, "Vehicle routing based mining for proof-of-useful-work blockchain," *J. KICS*, vol. 49, no. 1, pp. 124-133, 2024. (<https://doi.org/10.7840/kics.2024.49.1.124>)
- [4] C. Nguyen, D. T. Hoang, D. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, "Proof-of-stake consensus mechanisms for future blockchain networks: Fundamentals, applications and Opportunities," *IEEE Access*, vol. 7, pp. 1-19, Jun. 2019. (<https://doi.org/10.1109/ACCESS.2019.2925010>)
- [5] J. Xu, C. Wang, and X. Jia, "A survey of blockchain consensus protocols," *ACM Computer Surv.*, vol. 55, no. 13s, pp. 1-35, 2023. (<https://doi.org/10.1145/3579845>)
- [6] J. Huang, D. He, M. S. Obaidat, P. Vijayakumar, M. Luo, and K. K. R. Choo, "The application of the blockchain technology in voting systems: A review," *ACM Computing Surv.*, vol. 54, no. 3, pp. 1-28, 2021. (<https://doi.org/10.1145/3439725>)
- [7] Z. Khudoykulov, U. Tojiakbarova, S. Bozorov, and D. Ourbonalieva, "Blockchain based E-voting system: Open issues and challenges," *2021 ICISCT*, pp. 1-5, Nov. 2021. (<https://doi.org/10.1109/ICISCT52966.2021.9670245>)
- [8] G. Zhang, F. Pan, Y. Mao, S. Tijanic, M. Dang'Ana, S. Motepalli, S. Zhang, and H.-A. Jacobsen, "Reaching consensus in the

- byzantine empire: A comprehensive review of BFT consensus algorithms,” *ACM Computing Surv.*, vol. 56, no. 5, pp. 1-41, 2024.
(<https://doi.org/10.1145/3636553>)
- [9] M. Castro and B. Liskov, “Practical byzantine fault tolerance,” in *Proc. Third Symp. Operating Syst. Design and Implementation*, pp. 173-186, Feb. 1999.
- [10] Q. Zhang, Z. Qi, X. Liu, T. Sun, and K. Lei, “Research and application of BFT algorithms based on the hybrid fault model,” *2018 1st IEEE Int. Conf. HotICN*, pp. 114-120, Aug. 2018.
(<https://doi.org/10.1109/HOTICN.2018.8606021>)
- [11] X. Gu, T. Kang, A. Zhou, and L. Guo, “Consensus node group selection and adjustment algorithm based on dual random selection mechanism,” in *Proc. 2023 9th Int. Conf. Commun. and Inf. Process.*, pp. 407-414, Lingshui, China, 2023.
(<https://doi.org/10.1145/3638884.3638947>)
- [12] Z. Zhang, D. Zhu, and W. Fan, “QPBFT: Practical byzantine fault tolerance consensus algorithm based on quantified-role,” *2020 IEEE 19th Int. Conf. Trust, Security and Privacy in Comput. and Commun. (TrustCom)*, pp. 991-997, Jan. 2021.
(<https://doi.org/10.1109/TrustCom50675.2020.0132>)
- [13] G. Yuan, L. Feng, J. Ning, and X. Yang, “Improvement of practical byzantine fault tolerant consensus algorithm for blockchain,” *2021 IEEE 3rd ICFTIC*, pp. 182-187, Nov. 2021.
(<https://doi.org/10.1109/ICFTIC54370.2021.9647347>)
- [14] S. Liu, R. Zhang, C. Liu, C. Xu, and J. Wan, “An improved PBFT consensus algorithm based on grouping and credit grading,” *Scientific Reports*, vol. 13, no. 1, pp. 1-12, Aug. 2023.
(<https://doi.org/10.1038/s41598-023-28856-x>)

- [15] M. P. McBee and C. Wilcox, “Blockchain technology: Principles and applications in medical imaging,” *J. Imaging Inf. Medicine* vol. 33, no. 3, pp. 726-734, Aug. 2023.
(<https://doi.org/10.1007/s10278-019-00310-3>)

Siwakorn Pasawang



Jul. 2021~Current : B.S. degree student, Kasetsart University
<Research Interests> Blockchain, Generative AI.
[ORCID:0009-0007-9870-5373]

Junghun Kim



Feb. 2012 : M.S. degree, Kyungpook National University
Feb. 2018 : Ph.D. degree, Kyungpook National University
Dec. 2023~Current : Professor, Daegu Catholic University
<Research Interests> ICT technology, Biotechnology, Artificial Intelligence, Robotics
[ORCID:0000-0002-6118-1496]

Sang-Il Choi



Feb. 2012 : M.S. degree, Kyungpook National University
Feb. 2017 : Ph.D. degree, Kyungpook National University
Mar. 2024~Current : Professor, Daegu Catholic University
<Research Interests> ICT technology, Blockchain, Artificial Intelligence
[ORCID:0000-0003-3473-6131]