Comparison of Consensus Algorithms in Blockchain for the Implementation of Central Bank Digital Currencies

Carlos G. V. N. Soares
Department of Electrical Engineering
University of Brasília (UnB)
Brasília, Federal District, Brazil
carlosgabriel1999@gmail.com

Carlo K. da S. Rodrigues
Center for Mathematics, Computing and Cognition
Federal University of ABC (UFABC)
Santo André, São Paulo, Brazil
carlo.kleber@ufabc.edu.br

William F. Giozza
Department of Electrical Engineering
University of Brasília (UnB)Brasília,
Federal District, Brazil
giozza@unb.br

ABSTRACT

The choice of the consensus mechanism in the implementation of a Central Bank Digital Currency (CBDC) with Blockchain technology is critical, as it impacts the efficiency, scalability, and security of the system. In this context, this work analyzes three consensus algorithms: Practical Byzantine Fault Tolerance (pBFT), HotStuff, and Raft. Simulations are performed in the NS-3 environment to measure consensus times in networks with different numbers of nodes, message sizes, and latencies. The results show the superiority of the Raft algorithm, as it is the most scalable and optimizes consensus time by up to 134% and 277% compared to the pBFT and HotStuff algorithms, respectively. The paper concludes with a discussion of findings and suggestions for future work.

CCS Concepts

Computer systems organization → Distributed ledgers; Consensus mechanisms; Applied computing → Financial applications; Software and its engineering → Performance prediction and evaluation.

Keywords

Central Bank Digital Currency (CBDC); Distributed Ledger Technology (DLT); Consensus Algorithms; Blockchain; Performance Analysis.

1. INTRODUCTION

Central Bank Digital Currencies (CBDCs) are digital representations of fiat money issued and regulated by monetary authorities, with projects underway in Brazil and other countries [22]. Implemented on Distributed Ledger Technologies (DLT) such as Blockchain [27], CBDCs aim to enhance the security and transparency of the financial system through asset tokenization [4]. DLT ensures data immutability [15] in private, permissioned networks, which differentiates them from public cryptocurrencies and guarantees regulatory oversight by the central bank [30].

This centralized management allows for greater auditability and the application of monetary policies [26], but integration with existing financial systems critically depends on the chosen Blockchain platform (e.g., Hyperledger, Corda) [3, 25]. Within these platforms, the choice of consensus algorithm is a decisive factor, as it directly impacts the scalability, efficiency, and security of the entire operation[19, 21].

Therefore, this paper compares three consensus algorithms as candidates for application in CBDCs: pBFT (Practical Byzantine Fault Tolerance) and HotStuff, which are Byzantine fault-tolerant, and Raft, which is crash fault-tolerant. To this end, following a literature review, simulations are conducted in the NS-3 environment to measure the consensus time under different network conditions (number of nodes, message size, and latency). The analysis of variants such as IBFT (Istanbul Byzantine Fault Tolerance) [20] and QBFT (Quorum Byzantine Fault Tolerance) [24], as well as an in-depth study of systemic security, are left as future work. The contribution of this research is the evaluation of the time to reach consensus (efficiency) and the impact of the number of nodes (scalability) on these algorithms, thereby advancing the body of knowledge for CBDC development.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. Section 3 reviews the consensus algorithms analyzed in this research. Section 4 compares these consensus algorithms, including numerical results and their corresponding analyses. Finally, Section 5 presents the conclusions and suggestions for future work.

2. RELATED WORK

Subsection 2.1 presents three key papers that address consensus algorithms applied to the implementation of CBDC systems, while Subsection 2.2 discusses five papers comparing consensus algorithms, highlighting potential candidates also suitable for use in Blockchain-based CBDC implementations.

2.1 Consensus Algorithms applied to CBDCs

In [10] conduct an analysis of the components of a CBDC, presenting the fundamental elements of its structure: types of DLT, smart contracts, virtual machines, and consensus algorithms. Specifically regarding consensus algorithms, the suitability of the following for CBDC applications is discussed: PoW (Proof of Work), PoS (Proof of Stake), DAG (Directed Acyclic Graph), DPoS (Delegated Proof of Stake), PoA (Proof of Authority), pBFT, and HotStuff. The study concludes that the pBFT and HotStuff algorithms, both from the BFT category, are the most appropriate for CBDCs. This recommendation is based on the ability of these algorithms to achieve consensus even in the presence of faults in processing nodes, a characteristic not found in the other evaluated algorithms, making them robust and reliable.

Focusing on security, [5] performs a literature review on available offline payment solutions for CBDCs. Their study establishes essential criteria for network security, such as double-spending prevention, unforgeability, non-repudiation, and verifiability. Based on these parameters, the authors conclude that PoW type algorithms are not suitable for such an application. However, the study does not indicate which algorithms might be more appropriate and points to a lack of clear technical guidelines on which algorithms would be most suitable.

A different perspective is offered by [16], who investigate the environmental impact and energy consumption of CBDCs. Their

study compares the use of CBDCs with traditional payment methods, as well as analyzing Blockchain-based payment systems. The comparison indicates that current financial methods have the lowest energy consumption, followed by the Raft algorithm, then the PoA and pBFT algorithms, and lastly, PoW. Furthermore, the authors highlight that the choice of consensus algorithm in a system that adopts CBDCs has a significant impact not only on energy consumption but also on the overall functioning of the system. For example, the cost associated with the pBFT consensus algorithm can increase exponentially with the number of participants, while at the same time, limiting the number of participants can compromise the system's security.

2.2 Comparison of Consensus Algorithms

[29] compares the PoW, PoS, DPoS, pBFT, and Ripple consensus algorithms through simulations. Their evaluation focuses on requirements such as fault tolerance, energy efficiency, and scalability, and also identifies the type of network to which each algorithm is applied. Ripple and pBFT are applicable to permissioned networks, while the others are used in public networks. PoW exhibits the highest scalability, followed by PoS, DPoS, pBFT, and Ripple. Regarding energy efficiency, Ripple and pBFT have the best performance, followed by DPoS, PoS, and PoW. As for fault tolerance, PoW, PoS, and DPoS are the most robust, followed by pBFT and Ripple.

A performance comparison between IBFT, Clique, PoW, and Raft is presented by [1]. The algorithms are compared in terms of latency, throughput, and failure rate using a system called UBETA for P2P (Peer-to-Peer) energy trading. The IBFT algorithm has the lowest latency in most test cases, followed by the Raft, PoW, and Clique algorithms. Regarding throughput, IBFT and Raft showed similar values, followed by PoW and Clique. In addition, the IBFT algorithm exhibits the lowest failure rate as the number of nodes increases, followed by Clique, PoW, and Raft.

The work by [18] evaluates the IBFT, QBFT, Raft, and Clique algorithms on the GoQuorum platform using the Hyperledger Caliper tool. The analysis included transactions per second, transaction latency, and RAM consumption across both private and public networks with varying numbers of nodes. In all tests, the Raft algorithm was the best performer. Following Raft, the IBFT and QBFT algorithms performed similarly, while the Clique algorithm had the worst performance among them across all metrics.

Focusing on the Hyperledger Besu platform, [6] analyzes the Clique, IBFT 2.0, and QBFT consensus algorithms. The analyzed metrics include transaction rates, latency, resource consumption, and scalability. The results indicate that Clique and QBFT exhibit similar throughput rates, with IBFT 2.0 having the lowest rate among them. The same applies to latency: Clique and QBFT show similar results, while IBFT 2.0 has the highest latency. Regarding scalability, QBFT demonstrates the greatest scalability, with IBFT 2.0 and Clique following.

Using the NS-3 simulator, [11] evaluates Paxos, Raft, and pBFT by varying the number of nodes, message size, and delays. Their findings demonstrate that pBFT is superior in both robustness and consensus speed, consistently outperforming Raft and Paxos under the tested conditions.

It can be concluded that PoW and PoS type algorithms are not suitable for CBDCs, as they do not feature a structure centralized under a single entity. In contrast, BFT based algorithms, such as pBFT and HotStuff, are more indicated for CBDCs, offering greater efficiency and scalability. Although the Ripple and Raft algorithms have also shown good performance, Ripple's proprietary nature limits its adoption in financial systems. Hence, the contribution of this research is, specifically, to perform a novel comparison of potential candidate algorithms (i.e., pBFT, HotStuff, and Raft) for use in Blockchain based CBDCs. It should be noted that the available literature on the topic of this research is still incipient and therefore characterized by a limited number of works.

3. EVALUATED ALGORITHMS

This section provides an overview of the following three algorithms: pBFT, HotStuff, and Raft. These three algorithms were chosen for analysis in this research because, according to the literature (see Section 2), they exhibit good overall performance and lack the more critical limitations found in other algorithms, such as the requirement for complete decentralization. Furthermore, this work aims to complement previous studies by specifically comparing pBFT, HotStuff, and Raft in simulations within the context of CBDCs, thereby identifying a gap in scientific literature.

3.1 pBFT

The pBFT algorithm is one of the most traditional consensus mechanisms, capable of tolerating up to f faults in a system composed of $n = 3f + 1$ replicas and requiring a fully interconnected network. The process operates in three phases: it begins with the leader proposing a new block (Pre-prepare); continues as other validators verify and broadcast their acceptance (Prepare); and concludes with the confirmation of the block (Commit). To advance between phases and finalize the block, an agreement from at least two-thirds of the network is required.

3.2 Raft

The Raft algorithm, a Crash Fault Tolerant (CFT) type, was designed for simple implementation, using a leader-follower model to coordinate log replication [7, 12, 23]). Its operation is based on the election of a leader through voting, who maintains authority by sending periodic heartbeats to follower nodes; the leader's failure triggers a new election. This leader is solely responsible for replicating the command logs that update the followers' state machines, overwriting divergent logs to maintain consistency.

3.3 HotStuff

The HotStuff algorithm [28] represents an evolution of BFT type consensus mechanisms, designed to optimize communication with linear message complexity and without the need for full interconnection between nodes. Its consensus mechanism operates in three voting phases. In the Prepare phase, the leader proposes a new block, and validators respond with votes. Next, in the Pre-commit phase, the leader aggregates the votes into a Quorum Certificate (QC) upon receiving $(n - f)$ messages, where n is the total number of validators and f is the maximum number of faults tolerated. This QC is then broadcast to the network. Finally, in the Commit phase, validators cast their final votes, which are again aggregated by the leader into a new QC to finalize the consensus.

4. Performance Evaluation

4.1 Simulation Setup

The experiments are modeled in the NS-3 environment, an open-source network simulator that is widely adopted by the academic community and allows for the modeling of various protocols and algorithms. In the context of CBDCs, it enables the simulation of Blockchain networks with different consensus algorithms[2, 14]. The algorithms were implemented using C++, and the simulations were conducted by adapting a pre-existing simulator. The implementation code for each algorithm is available on GitHub.

4.2 Experimental Setup

Table 1. Parameters Used in the Simulation

Parameters	Values
Nodes (*n*)	8, 16, 32, 64
Size (bytes)	256, 1024
Latency (*ms*)	10, 100

The performance metric evaluated in the experiments is the consensus time, i.e., the average time required for the network nodes to validate transactions. The experiments were conducted by varying three independent variables: the number of participating nodes in the network (*n*), the size of the transmitted messages (in bytes), and the communication latency between nodes (in *ms*), as shown in Table 1.

High-performance networks were simulated with a latency of 10*ms*, while scenarios more representative of real-world implementations were modeled with a latency of 100*ms [9]*. The selected message sizes of 256 bytes and 1024 bytes [8, 13] represent approximations of transactions in public Blockchain and CBDC networks, respectively. The simulations were conducted on network topologies with 8, 16, 32, and 64 nodes. The selection of these values is based on the permissioned architecture of CBDC networks, which are generally composed of a restricted set of validator nodes, such as the Central Bank and government-designated financial institutions [17].

Table 2. Computational Environment Used in the Simulation

Resource	Specification
Operating System	Ubuntu 24.04.1 LTS
Processor	Intel® Core™ i9-11900H
Memory	12GB DDR4 3200MHz
Simulator	NS-3.32

To ensure the statistical validity and reliability of the results, the average values reported in the experiments are obtained from 30 executions (runs). This approach allowed for the analysis of the data with a 95% confidence level, minimizing the impact of random fluctuations and providing a robust basis for the inferences. Finally, the computational resources used to perform the CBDC network simulation are detailed in Table 2.

4.3 Network Topology

NS-3 offers complete flexibility in the topological configuration of the network, allowing for arbitrary connections between system nodes. In the simulations performed, a fully interconnected network topology was implemented to ensure a consistent comparison between the algorithms. This configuration was adopted as it is a fundamental requirement of the pBFT algorithm, although it may potentially impact the performance of the other analyzed algorithms. In this topology, considering *n* nodes, the total number of connections is determined by the combination *C(n, 2)*, mathematically expressed as:

$$Number\ of\ connections\ =\ n \cdot \frac{(n-1)}{2}$$

Thus, for the values of *n* indicated in Table 2, a total of 28, 120, 496, and 2016 connections are obtained for 8, 16, 32, and 64 nodes, respectively.

4.4 Analysis of Results

This subsection presents the experimental results obtained through simulations conducted in the NS-3 environment, focusing on the comparative analysis of the pBFT, Raft, and HotStuff consensus algorithms. The performance metric, as previously mentioned, is the consensus time, i.e., the time required to establish consensus among the participating network nodes. The data are represented on a logarithmic scale to provide an appropriate visualization of the variations observed in the experimental results.

4.4.1 Network with 10ms Latency

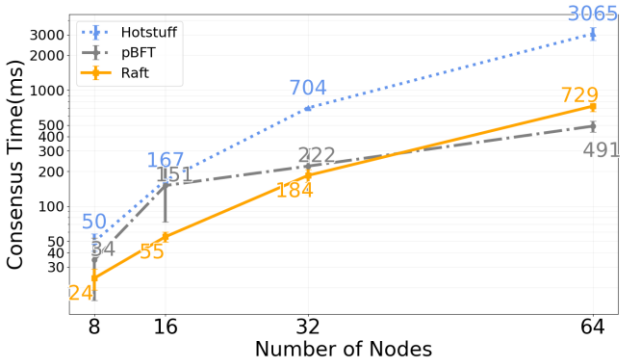

Figure 1: Consensus time for 10ms latency and 256-byte messages

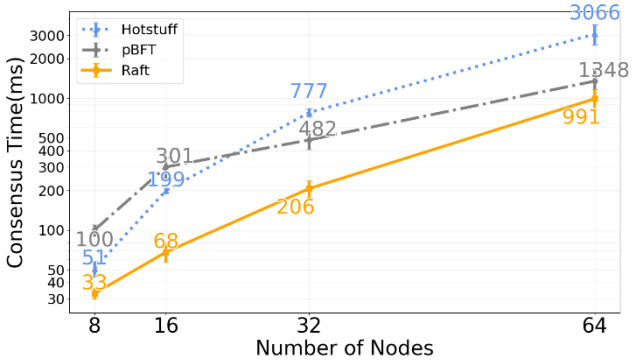

Figure 2: Consensus time for 10ms latency and 1024-byte messages

In the scenario with 256-byte messages, depicted in Figure 1, the RAFT protocol exhibits the best performance in configurations of 8, 16, and 32 nodes, with the lowest response times (24ms, 55ms,

and 184ms, respectively). The HotStuff protocol, although starting with performance comparable to pBFT at 8 nodes (both at 50ms), shows the most pronounced performance degradation as the network scales, reaching the highest value at 64 nodes (3065ms). In contrast, pBFT, despite being surpassed by RAFT in smaller networks, demonstrates better scalability, taking the lead as the most efficient protocol in the 64-node configuration (491ms).

In the configuration involving 1024-byte messages, detailed in Figure 2, the RAFT protocol shows consistent superiority, delivering the best performance across all tested node configurations (8, 16, 32, and 64). pBFT, on the other hand, starts as the least efficient protocol in an 8-node network and, although it scales better than HotStuff, it remains consistently behind RAFT. The HotStuff algorithm again demonstrates the poorest scalability, starting with intermediate performance at 8 nodes but recording the longest response times in scenarios with a higher number of nodes.

In conclusion, Raft proved to be the most efficient protocol, a direct consequence of its leader-based model with linear communication (O(n)), which remained robust to increases in both node count and load. pBFT, in turn, showed mixed behavior; its high quadratic communication complexity (O(n^2)), required for Byzantine fault tolerance, explains its loss of efficiency as the network and message sizes grew. Finally, HotStuff exhibited the lowest scalability, indicating that despite its theoretically linear architecture, its performance was severely compromised as the network grew; it is theorized that this behavior is due to the computational cost of validating Quorum Certificate.

4.4.2 Network with 100ms Latency

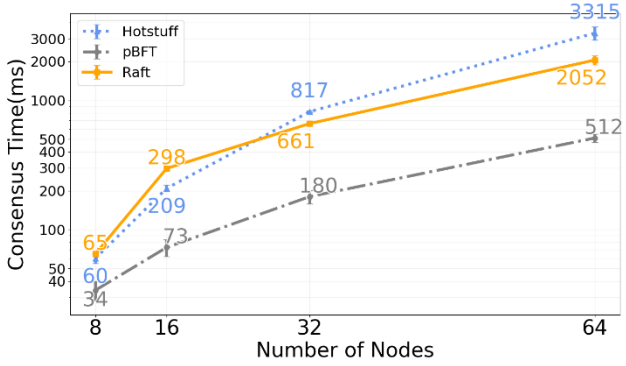

Figure 3: Consensus time for 100ms latency and 256-byte messages

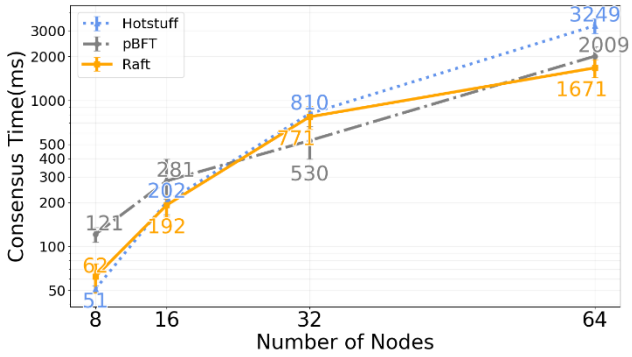

Figure 4: Consensus time for 100ms latency and 1024-byte messages

In the scenario with 100ms latency and 256-byte messages, depicted in Figure 3, the pBFT protocol demonstrates notable performance superiority across all node configurations (8, 16, 32, and 64). Its performance scales more efficiently compared to the other algorithms, maintaining the lowest response times. The Raft and HotStuff protocols show similar initial performance at 8 nodes, but both exhibit a significantly more pronounced degradation as the network grows, becoming slower than pBFT.

In the configuration with larger, 1024-byte messages, shown in Figure 4 with the same 100ms latency, HotStuff starts with the best performance in an 8-node network, but its scalability proves inferior, ending as the slowest protocol at 64 nodes. Raft, in turn, takes the lead in 16 and 64-node networks. pBFT, despite a less impressive start, improves its performance to tie with Raft in the 32-node configuration but is once again surpassed in the larger network.

Compared to the 10ms latency scenario, the pBFT algorithm demonstrated consistent performance, remaining efficient under both 10ms and 100ms latencies for both data packet sizes, with the exception noted in the 64-node topology with 1024-byte messages. In contrast, the Raft protocol proved to be highly sensitive to the increase in latency, suffering from performance degradation. HotStuff, meanwhile, exhibited the lowest scalability, showing the most pronounced performance deterioration among the evaluated algorithms in response to increased latency, regardless of the network configuration.

4.5 Critical evaluation

From the analysis of the numerical results presented individually in the previous subsections, the following general findings can be made.

a) The performance of the Raft algorithm is characterized as follows. In low-latency scenarios, it proved to be the most efficient and robust protocol, consistently delivering the best performance due to its linear communication model. In high-latency scenarios, its performance drops significantly, revealing its high sensitivity to increased network latency. Therefore, the Raft algorithm is the ideal choice for networks with guaranteed low latency, but its performance is considerably hindered in less ideal network environments.

b) The performance of the pBFT algorithm is characterized as follows. In low-latency scenarios, its behavior is mixed, often being surpassed by Raft in smaller networks but demonstrating better scalability and potentially taking the lead in larger network configurations. In high-latency scenarios, it demonstrates consistency, which is the most efficient algorithm in most tested configurations. Therefore, the pBFT algorithm stands out for its robustness and consistent performance, being particularly effective and superior to the others in high-latency environments.

c) The performance of the HotStuff algorithm is characterized as follows. In low-latency scenarios, although its initial performance may be comparable, it exhibits the poorest scalability and the most pronounced performance degradation as the network grows. In high-latency scenarios, this trend intensifies, and despite good initial performance in very small networks, its scalability proves inferior, ending as the slowest protocol in larger configurations. Therefore, the HotStuff algorithm demonstrated the lowest scalability in the evaluated scenarios, with its performance being severely compromised by increases in network size and latency.

In summary, considering the previous discussion, it is concluded that in the context of CBDCs where low latency is expected, the Raft algorithm is the most suitable choice.

5. Final Conclusions and Future Work

This work conducted a comparative analysis of three consensus algorithms: pBFT, HotStuff, and Raft, evaluating their applicability for the implementation of Blockchain-based CBDCs. The analysis methodology involved simulations in the NS-3 environment to measure the consensus time under different network configurations, considering message size, latency, and the number of nodes. In general, the experimental results showed that the Raft algorithm is the most efficient and scalable for implementing CBDCs, followed by pBFT and HotStuff. Furthermore, it was also noted that pBFT's performance was significantly affected by message size, whereas HotStuff was minimally impacted.

As future work, we suggest the following directions: (i) analyzing the impact of the connection topology between network nodes on efficiency, scalability, and systemic security; (ii) comparing other consensus algorithms, as specified in Section 3; and (iii) creating new algorithms to optimize the performance of existing ones.

6. REFERENCES

[1] Abdella, J. et al. An Architecture and Performance Evaluation of Blockchain-Based Peer-to-Peer Energy Trading. PP, 1. https://doi.org/10.1109/TSG.2021.3056147.

[2] Campanile, L. et al. Computer network simulation with ns-3: A systematic literature review. 9, 2, 272.

[3] Capocasale, V. et al. Comparative analysis of permissioned blockchain frameworks for industrial applications. 4, 1, 100113. https://doi.org/https://doi.org/10.1016/j.bcra.2022.100113 .

[4] Chen, Y. Blockchain tokens and the potential democratization of entrepreneurship and innovation. 61, 4, 567–575.

[5] Chu, Y. et al. Review of offline payment function of CBDC considering security requirements. 12, 9, 4488.

[6] Fan, C. et al. Performance Analysis of Hyperledger Besu in Private Blockchain. *2022 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)* 64–73.

[7] Fu, W. et al. An improved blockchain consensus algorithm based on raft. 46, 9, 8137–8149.

[8] Gelashvili, R. et al. Jolteon and ditto: Network-adaptive efficient consensus with asynchronous fallback. *International conference on financial cryptography and data security* 296–315.

[9] Gueta, G.G. et al. SBFT: A scalable and decentralized trust infrastructure. *2019 49th Annual IEEE/IFIP international conference on dependable systems and networks (DSN)* 568–580.

[10] Guo, S. et al. DLT Options for CBDC. 13, 1, 57–88. https://doi.org/doi:10.2478/jcbtp-2024-0004.

[11] Hidayat, S.A. et al. Performance comparison and analysis of paxos, raft and pbft using ns3. *2022 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS)* 304–310.

[12] Huang, D. et al. Performance analysis of the raft consensus algorithm for private blockchains. 50, 1, 172–181.

[13] Islam, M.M. A privacy-preserving transparent central bank digital currency system based on consortium blockchain and unspent transaction outputs. 16, 4, 2372–2386.

[14] Jin, K. et al. Distributed node consensus protocol: Analysis, evaluation and performance. *2016 IEEE International Conference on Communications (ICC)* 1–6.

[15] Johar, S. et al. Research and Applied Perspective to Blockchain Technology: A Comprehensive Survey.

[16] Lee, S. and Park, J. Environmental implications of a central bank digital currency (CBDC).

[17] Li, D. et al. Design principles and best practices of central bank digital currency. 17, 5, 411.

[18] Lopes, J. Exploring consensus algorithms in Quorum.

[19] Mingxiao, D. et al. A review on consensus algorithm of blockchain. *2017 IEEE international conference on systems, man, and cybernetics (SMC)* 2567–2572.

[20] Moniz, H. The Istanbul BFT consensus algorithm.

[21] Nguyen, G.-T. and Kim, K. A survey about consensus algorithms used in blockchain. 14, 1.

[22] Nóbrega, L.P. et al. The Efficiency of the Brazilian Payment System as per the Deployment of the Digital Currency based on DLT: Challenges and Opportunities. 13, 2.

[23] Ongaro, D. and Ousterhout, J. In search of an understandable consensus algorithm. *2014 USENIX annual technical conference (USENIX ATC 14)* 305–319.

[24] Tkachuk, R.-V. et al. 2023. On the Performance of Consensus Mechanisms in Privacy-Enabled Decentralized Peer-to-Peer Renewable Energy Marketplace. *2023 26th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)* (2023), 179–186.

[25] Ucbas, Y. et al. Performance and Scalability Analysis of Ethereum and Hyperledger Fabric. 11, 67156–67167. https://doi.org/10.1109/ACCESS.2023.3291618.

[26] Wang, G. and Hausken, K. A game between central banks and households involving central bank digital currencies, other digital currencies and negative intere st rates. 10, 1, 2114178.

[27] Ward, O. and Rochemont, S. Understanding central bank digital currencies (CBDC). 13, 2, 263–268.

[28] Yin, M. et al. HotStuff: BFT consensus with linearity and responsiveness. *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing* 347–356.

[29] Zhang, S. and Lee, J.-H. Analysis of the main consensus protocols of blockchain. 6, 2, 93–97.

[30] Zheng, Z. et al. Blockchain challenges and opportunities: A survey. 14, 4, 352–375.