DABFT: A Novel Adaptive Byzantine Fault Tolerance Framework for High-Performance Blockchain Consensus

Veeramma Yatnalli
JSS Academy of Technical Education, Bengaluru, India
veerammayatnalli@jssateb.ac.in

Saroja S. Bhusare
JSS Academy of Technical Education, Bengaluru, India
sarojasbhusare@jssateb.ac.in (corresponding author)

Praveen M. Dhulavvagol
School of Computer Science and Engineering, KLE Technological University, Hubli, India
praveen.md@kletech.ac.in

Guruprasad Konnurmath
School of Computer Science and Engineering, KLE Technological University, Hubli, India
guruprasad.konnurmath@kletech.ac.in

Rajani Shetty
SJB Institute of Technology, Bengaluru, India
rajani@sjbit.edu.in

Received: 7 May 2025 | Revised: 28 May 2025 | Accepted: 6 June 2025

 DOI: https://doi.org/10.48084/etasr.11970

ABSTRACT

Byzantine Fault Tolerance (BFT) consensus mechanisms are vital for ensuring reliability in permissioned blockchain networks. However, existing BFT implementations such as Quorum-based BFT (QBFT), Istanbul BFT (IBFT) 2.0, and CLIQUE within the Hyperledger Besu framework face significant challenges, including high resource consumption, limited throughput, and scalability bottlenecks, particularly under complex and high-volume smart contract workloads. In this paper, we propose Dynamic Adaptive Byzantine Fault Tolerance (DABFT), a novel consensus framework designed to address these limitations through Adaptive Quorum Selection (AQS) and Optimized Transaction Batching (OTB). We evaluate DABFT using Hyperledger Caliper, benchmarking its performance against established BFT protocols under diverse transaction loads and operational conditions. The results demonstrate that DABFT achieves up to 30% higher throughput, 25% lower latency, and a 15% reduction in resource usage compared to QBFT. These improvements highlight DABFT's potential to significantly enhance the performance, scalability, and resource efficiency of enterprise-grade blockchain deployments.

Keywords-Byzantine Fault Tolerance (BFT); blockchain; Quorum-based BFT (QBFT); CLIQUE; ERC-20; hyperledger caliper

I. INTRODUCTION

Blockchain technology has emerged as a key infrastructure for enabling secure, transparent, and decentralized data handling across diverse sectors such as finance, supply chain management, healthcare, and public services. In permissioned blockchain systems, where network participants are known and authenticated, Byzantine Fault Tolerance (BFT) consensus mechanisms play a pivotal role in maintaining system integrity and consistency, even in the presence of faulty or malicious nodes [1]. Widely adopted BFT protocols, including Quorum-based BFT (QBFT), Istanbul BFT (IBFT) 2.0, and CLIQUE within the Hyperledger Besu framework, offer deterministic consensus finality and resilience to Byzantine failures [2]. However, these protocols face significant limitations that

hinder their deployment in high-performance, real-world enterprise applications.

One of the most pressing challenges is limited scalability. As the validator count increases, conventional BFT protocols experience steep declines in performance, primarily due to their quadratic communication overhead required to reach agreement [3]. Additionally, these protocols often suffer from inefficient resource utilization. For example, IBFT 2.0 employs frequent message exchanges, multiple voting phases, and repeated block proposals under certain fault scenarios, which collectively lead to high Central Processing Unit (CPU), memory, and network overhead [4]. Another concern is the rigidity of current BFT models: most rely on static quorum thresholds, making them poorly suited to dynamic network environments where node participation and reliability may vary. These issues are magnified under heavy workloads, such as intensive transaction processing or smart contract execution, ultimately resulting in greater latency, reduced throughput, and degraded user experience [5].

These challenges underscore the difficulty of applying existing BFT protocols in enterprise contexts where performance, responsiveness, and efficient use of resources are paramount. This study is motivated by the need to develop a more scalable and adaptive BFT solution tailored to the demands of dynamic enterprise blockchain networks [6, 7]. In response, we propose Dynamic Adaptive Byzantine Fault Tolerance (DABFT), a novel consensus framework that introduces two key innovations: Adaptive Quorum Selection (AQS) and Optimized Transaction Batching (OTB). Through these mechanisms, DABFT delivers marked improvements in throughput and latency while reducing system overhead, particularly under transaction loads ranging from 100 to 500 transactions per second (Tps).

II. PROPOSED DABFT FRAMEWORK: SYSTEM DESIGN AND ARCHITECTURE

The DABFT centers on two key components: AQS and OTB. These mechanisms work in tandem to enable real-time adaptation to fluctuating network conditions and transaction loads, all while maintaining the core safety and liveness guarantees fundamental to BFT systems [8]. In contrast to protocols that rely on static quorums, DABFT continuously evaluates the operational status of nodes and network performance, dynamically adjusting quorum composition to enhance consensus efficiency, especially during periods of network congestion, partial node outages, or high transaction throughput [9].

The consensus process begins with the selection of a leader (proposer) for each round, using either a round-robin approach or performance-based criteria. The elected leader then compiles a block by batching transactions, with batch size optimized in real time based on factors such as transaction pool size, network latency, and target throughput. In the quorum formation stage, validator nodes are selected using live performance metrics including availability, response time, and historical trustworthiness. This dynamic selection allows quorum size and membership to vary from round to round, which reduces communication overhead and improves fault resilience. During the voting and agreement phase, validators verify the proposed block and cast their votes. The flexibility of quorum composition expedites consensus without compromising BFT. Once a sufficient number of valid votes is collected, the protocol deterministically commits the block to the ledger. DABFT adheres to classical BFT fault tolerance thresholds, tolerating up to f faulty nodes in a network of $3f + 1$ total nodes, thereby ensuring consensus safety and finality.

A. Adaptive Quorum Selection (AQS) & Optimized Transaction Batching (OTB)

Traditional BFT protocols operating with static quorum sizes, while effective in small-scale or stable network environments, become increasingly inefficient in dynamic or large-scale systems, where node availability and performance can vary significantly. The DABFT protocol addresses this limitation through its AQS module. By continuously evaluating and prioritizing nodes based on performance and reliability metrics, AQS reduces consensus latency and minimizes communication overhead.

Quorum composition is recalculated at regular intervals or triggered by signs of system degradation, thereby maintaining protocol robustness without incurring excessive resource consumption [9]. The adaptive quorum size is defined as:

$$Q_{adaptive} = \left\lceil \frac{2N+1}{3} \right\rceil + \delta \quad (1)$$

where Nrepresents the total number of validator nodes, and δ is a dynamic adjustment factor influenced by network congestion and transaction backlog.

In parallel, the OTB module dynamically adjusts transaction batches based on current network load and block propagation conditions. This strategy reduces the frequency of consensus rounds and increases throughput by improving overall block proposal efficiency. During periods of high network activity, OTB helps alleviate congestion, while under lighter loads, it accelerates consensus by reducing batching delays. The batch size B is determined by:

$$B = \min\left(B_{max}, \frac{T_{pending}}{N_{active}}\right) \quad (2)$$

where $T_{pending}$ is the number of pending transactions and N_{active} is the number of active validator nodes. The implementation of the AQS and OTB algorithms is presented below:

Algorithm: AQS

```
function selectAdaptiveQuorum(validators, f):
    metrics = collectNodeMetrics(validators) #
uptime, latency, trust score
    sorted_validators =
sortByPerformance(metrics)
    quorum_size = 2 * f + 1
    quorum = selectTopK(sorted_validators,
quorum_size)
    return quorum

function collectNodeMetrics(validators):
    metrics = {}
    for node in validators:
```

```
        metrics[node] = {
            'uptime': getUptime(node),
            'latency': getNetworkLatency(node),
            'trust': getTrustScore(node)
        }
    return metrics
```

Algorithm: OTB

```
function createTransactionBatch(txPool,
maxBlockSize, networkLoad):
    baseBatchSize =
determineBaseBatchSize(networkLoad)
    txBatch = []

    for tx in txPool:
        if len(txBatch) < baseBatchSize and
estimateSize(txBatch + [tx]) < maxBlockSize:
            txBatch.append(tx)
        else:
            break
    return txBatch

function determineBaseBatchSize(networkLoad):
    if networkLoad == 'high':
        return 200
    elif networkLoad == 'moderate':
        return 100
    else:
        return 50
```

B. System Architecture

The DABFT framework is implemented as a modular extension within the Hyperledger Besu environment and comprises the key components illustrated in Figure 1.

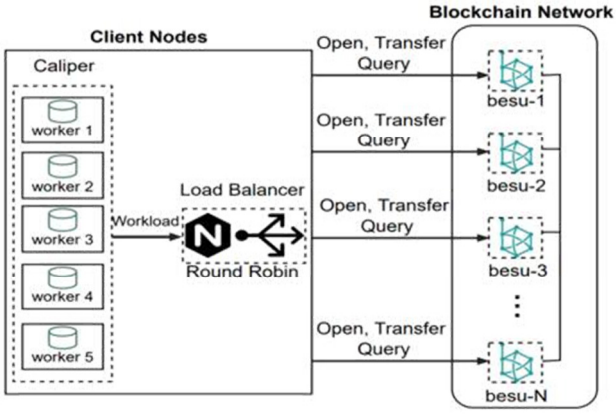

Fig. 1. Proposed system architecture.

The architecture is specifically designed to support the evaluation of existing consensus protocols, including QBFT, IBFT 2.0, and CLIQUE, within the Hyperledger Besu framework. At the heart of the performance testing setup is Hyperledger Caliper 0.5.x, a benchmarking tool tailored for blockchain networks, which orchestrates workload generation across multiple client nodes (workers). Each Caliper worker node is responsible for issuing operations, such as account creation, value transfer, and transaction queries, under a controlled testing scenario. To evenly distribute these operations across the blockchain network, a Load Balancer is employed. As depicted in Figure 1, the Load Balancer uses a round-robin strategy to forward requests to validator nodes (besu-1 through besu-N), ensuring balanced workload distribution and minimizing potential performance bottlenecks [10-12].

III. EXPERIMENTAL SETUP AND METHODOLOGY

A. Experimental Environment

The experimental environment consisted of six virtual machines, each running Ubuntu 22.04 LTS and configured with 4 virtual CPUs and 8 GB of RAM. These virtual machines were connected via a 1 Gbps internal network and collectively supported a permissioned blockchain deployment comprising four validator nodes and two client nodes.

To simulate realistic enterprise blockchain scenarios, smart contracts based on the Ethereum ERC-20 token standard and asset transfer logic were deployed across the network. Performance benchmarking was conducted under three defined transaction throughput conditions: low (50 Tps), moderate (150 Tps), and high (300+ Tps), enabling the evaluation of scalability and responsiveness under varying load levels.

B. Dataset Description

To evaluate the performance of the DABFT consensus mechanism, real-world smart contract transaction data and execution traces were obtained from the Ethereum Goerli [13] and Ropsten [14] public test networks. These testnets offer representative blockchain activity, including contract deployment, invocation, and transactional behavior. Additionally, the evaluation incorporated the Ethereum Smart Contract Dataset (ESCD) [15], which includes verified smart contracts annotated with metadata such as gas consumption, invocation frequency, and transaction success rates. This dataset facilitated the execution of diverse contract logic and transaction types, further enriching the testing scenarios and ensuring relevance to real-world enterprise applications.

IV. EXPERIMENTAL RESULTS AND COMPARATIVE ANALYSIS

Deploying the collected datasets on a private Hyperledger Besu network facilitated a comparative performance evaluation of the proposed DABFT protocol against existing consensus mechanisms, namely QBFT, IBFT 2.0, and CLIQUE. The comparison focused key performance metrics such as throughput, latency, and resource utilization under realistic workloads and varying network conditions [16].

Table I presents the transaction throughput results. DABFT consistently outperforms the other consensus protocols across all tested load levels (100 to 500 Tps), achieving an average throughput of 123.9 Tps. In comparison, QBFT and IBFT 2.0 average 93.1 Tps and 91.7 Tps, respectively, while CLIQUE lags significantly with an average of 58.9 Tps. Thus, the proposed DABFT delivers approximately 30% higher throughput than both QBFT and IBFT 2.0, and more than double the throughput of CLIQUE, which is hindered by its

less efficient leader-based design and block propagation mechanics.

Table II summarizes the transaction latency, defined as the time elapsed from transaction submission to final confirmation. DABFT records the lowest average latency at 2.4 seconds, outperforming QBFT (3.4 s), IBFT 2.0 (3.3 s), and CLIQUE (4.3 s). This latency advantage is primarily attributed to DABFT's adaptive quorum selection and optimized batching mechanisms, which accelerate the consensus process. On average, DABFT reduces latency by approximately 25% compared to QBFT and IBFT 2.0, and by up to 35% relative to CLIQUE.

Table III highlights the comparative resource utilization across the consensus mechanisms. DABFT demonstrates substantial gains in efficiency, utilizing only 57.6% CPU, which reflects a 15–20% reduction compared to the other protocols. Its memory usage stands at 358 MB, approximately 18% lower than that of QBFT and IBFT 2.0. Furthermore, DABFT's network bandwidth consumption is 10.2 MB/s, representing an average reduction of around 20%. These improvements underline DABFT's suitability for resource-constrained environments and its potential for high-performance enterprise deployments.

TABLE I. TRANSACTION THROUGHPUT ANALYSIS

Consensus Mechanism	100 Tps	200 Tps	300 Tps	400 Tps	500 Tps
QBFT [17]	96.2	94.5	93.1	91.8	89.7
IBFT 2.0 [11]	94.1	93.3	92.0	90.6	88.5
CLIQUE [16]	61.8	60.2	58.9	57.5	56.2
DABFT (Proposed)	126.5	125.1	124.3	122.8	120.9

TABLE II. TRANSACTION LATENCY ANALYSIS

Consensus Mechanism	100 Tps	200 Tps	300 Tps	400 Tps	500 Tps
QBFT [17]	2.8 s	3.1 s	3.4 s	3.7 s	4.0 s
IBFT 2.0 [11]	2.6 s	2.9 s	3.3 s	3.6 s	3.9 s
CLIQUE [16]	3.8 s	4.0 s	4.3 s	4.5 s	4.8 s
DABFT (Proposed)	1.9 s	2.1 s	2.4 s	2.6 s	2.9 s

TABLE III. RESOURCE UTILIZATION COMPARISON ACROSS CONSENSUS MECHANISMS

Consensus Mechanism	CPU Usage (%)	Memory Usage (MB)	Network Bandwidth (MB/s)
QBFT [17]	68.4	420	12.5
IBFT 2.0 [11]	70.2	440	13.1
CLIQUE [16]	64.9	405	11.8
DABFT (Proposed)	57.6	358	10.2

V. CONCLUSION

The proposed Dynamic Adaptive Byzantine Fault Tolerance (DABFT) consensus mechanism addresses key limitations of conventional Byzantine Fault Tolerance (BFT) protocols, such as Quorum-based BFT (QBFT), Istanbul BFT (IBFT) 2.0, and CLIQUE, by tackling challenges related to limited scalability, high latency, and excessive resource consumption in permissioned blockchain networks. By incorporating Adaptive Quorum Selection (AQS) and Optimized Transaction Batching (OTB), DABFT dynamically adjusts to network conditions, enabling consistent performance across varying transaction loads. Experimental results demonstrate that DABFT achieves up to 30% higher throughput, 25–30% lower latency, and 15–20% reduced Central Processing Unit (CPU), memory, and bandwidth usage compared to traditional protocols. These findings affirm DABFT's suitability as a scalable, efficient, and high-performance consensus solution for enterprise blockchain applications. Future enhancements will focus on refining quorum adaptation strategies and integrating with sharding and cross-chain mechanisms to further improve scalability and interoperability.

REFERENCES

[1] A. T. Sherman, F. Javani, H. Zhang, and E. Golaszewski, "On the Origins and Variations of Blockchain Technologies," *IEEE Security & Privacy*, vol. 17, no. 1, pp. 72–77, Jan. 2019, https://doi.org/10.1109/MSEC.2019.2893730.

[2] P. M. Dhulavvagol, P. M. R, N. C. Kundur, J. N, and S. G. Totad, "Scalable Blockchain Architecture: Leveraging Hybrid Shard Generation and Data Partitioning," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 8, 2023, https://doi.org/10.14569/IJACSA.2023.0140839.

[3] J. Yin *et al.*, "SmartDID: A Novel Privacy-Preserving Identity Based on Blockchain for IoT," *IEEE Internet of Things Journal*, vol. 10, no. 8, pp. 6718–6732, Apr. 2023, https://doi.org/10.1109/JIOT.2022.3145089.

[4] W. Liang *et al.*, "PDPChain: A Consortium Blockchain-Based Privacy Protection Scheme for Personal Data," *IEEE Transactions on Reliability*, vol. 72, no. 2, pp. 586–598, Jun. 2023, https://doi.org/10.1109/TR.2022.3190932.

[5] G. Venkatadri *et al.*, "Privacy Risks with Facebook's PII-Based Targeting: Auditing a Data Broker's Advertising Interface," in *2018 IEEE Symposium on Security and Privacy (SP)*, San Francisco, CA, May 2018, pp. 89–107, https://doi.org/10.1109/SP.2018.00014.

[6] R. Yousuf, Z. Jeelani, D. A. Khan, O. Bhat, and T. A. Teli, "Consensus Algorithms in Blockchain-Based Cryptocurrencies," in *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, Bhilai, India, Feb. 2021, pp. 1–6, https://doi.org/10.1109/ICAECT49130.2021.9392489.

[7] Y. Liu *et al.*, "Building blocks of sharding blockchain systems: Concepts, approaches, and open problems," *Computer Science Review*, vol. 46, Nov. 2022, Art. no. 100513, https://doi.org/10.1016/j.cosrev.2022.100513.

[8] S. Islam, M. J. Islam, M. Hossain, S. Noor, K.-S. Kwak, and S. M. R. Islam, "A Survey on Consensus Algorithms in Blockchain-Based Applications: Architecture, Taxonomy, and Operational Issues," *IEEE Access*, vol. 11, pp. 39066–39082, 2023, https://doi.org/10.1109/ACCESS.2023.3267047.

[9] S. Sharma, O. Sharma, and J. Arora, "Consensus Mechanisms Analysis: A Remedy for the Byzantine Generals Problem," in *2023 3rd International Conference on Technological Advancements in Computational Sciences (ICTACS)*, Tashkent, Uzbekistan, Nov. 2023, pp. 674–678, https://doi.org/10.1109/ICTACS59847.2023.10390006.

[10] Q. Zhang, J. Su, Z. Ma, Y. Zhang, J. Yang, and J. Zhan, "Blockchain Model Testing and Implementation Based on Improved PBFT Consensus," in *2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, Cracow, Poland, Sep. 2021, pp. 1010–1015, https://doi.org/10.1109/IDAACS53288.2021.9660959.

[11] A. Baliga, I. Subhod, P. Kamat, and S. Chatterjee, "Performance Evaluation of the Quorum Blockchain Platform," arXiv, 2018, https://doi.org/10.48550/ARXIV.1809.03421.

[12] P. M. Dhulavvagol, S. G. Totad, and S. Sourabh, "Performance Analysis of Job Scheduling Algorithms on Hadoop Multi-cluster Environment," in *Emerging Research in Electronics, Computer Science and*

Technology, Singapore, 2019, vol. 545, pp. 457–470, https://doi.org/10.1007/978-981-13-5802-9_42.

[13] K. Li, Y. Tang, J. Chen, Y. Wang, and X. Liu, "TopoShot: Uncovering Ethereum's Network Topology Leveraging Replacement Transactions," arXiv, 2021, https://doi.org/10.48550/ARXIV.2109.14794.

[14] *Ropsten Testnet.* (2022), Ethereum. [Online]. Available: https://github.com/ethereum/ropsten.

[15] T. Durieux, J. F. Ferreira, R. Abreu, and P. Cruz, "Empirical review of automated analysis tools on 47,587 Ethereum smart contracts," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, Seoul, South Korea, Jun. 2020, pp. 530–541, https://doi.org/10.1145/3377811.3380364.

[16] C. Fan, C. Lin, H. Khazaei, and P. Musilek, "Performance Analysis of Hyperledger Besu in Private Blockchain," in 2022 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS), Newark, CA, USA, Aug. 2022, pp. 64–73, https://doi.org/10.1109/DAPPS55202.2022.00016.

[17] P. Szilágyi, "EIP-225: Clique proof-of-authority consensus protocol," Ethereum Improvement Proposals, Ethereum Foundation, Mar. 2017. [Online]. Available: https://eips.ethereum.org/EIPS/eip-225.