Enhancing Ethereum Proof-of-Authority Networks with DAG-based Consensus

Yongrae Jo
memex@postech.ac.kr
Pohang University of Science and Technology
Pohang, Republic of Korea

Chanik Park
cipark@postech.ac.kr
Pohang University of Science and Technology
Pohang, Republic of Korea

Abstract

Ethereum's Proof-of-Authority (PoA) is one of the popular Byzantine fault-tolerant (BFT) consensus protocols for permissioned blockchains, owing to its high performance and resource efficiency compared to Proof-of-Work (PoW). However, PoA still faces performance issues due to limited block generation because only one block, mostly proposed by a leader, is accepted in a round. This also leads to resource inefficiency in the system, as blocks proposed by non-leaders within the same round are discarded. Meanwhile, directed acyclic graph (DAG)-based BFT has become known as a promising solution for a blockchain consensus engine due to its high performance. In this paper, we introduce DaPoA, a novel design proposal that enhances the Ethereum PoA network (i.e., Clique) with a DAG-based consensus to support high performance solidity smart contract execution. A key idea behind DaPoA is the enabling of multiple authorities to receive transactions from clients and propose their blocks concurrently in a round, and these blocks, without discarding non-leader's proposals, are incorporated into a reliable sub-DAG using the DAG-based consensus. Further, we propose an optimization technique called asynchronous block proposals, enabling proposing multiple blocks across multiple rounds, which is useful under high transaction workload.

CCS Concepts: • **Computer systems organization → Distributed architectures**; • **Security and privacy → Distributed systems security**.

Keywords: Proof-of-Authority, Ethereum, DAG-based Consensus

ACM Reference Format:
Yongrae Jo and Chanik Park. 2018. Enhancing Ethereum Proof-of-Authority Networks with DAG-based Consensus. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

In blockchain technology, consensus protocols enable trustworthy transactions among untrusted participants with properties such as immutability, transparency, and verifiability. Representatively, cryptocurrencies such as Bitcoin and Ethereum uses the Proof-of-Work (PoW) [13, 16]. While effective, PoW suffers from low performance and demands significant and unnecessary computational power to solve complex mathematical problems based on cryptographic hash puzzles for ensuring the properties. These issues are a major obstacle preventing the widespread adoption of blockchain-based services.

A promising approach to address these shortcomings is the introduction of Proof-of-Authority (PoA), a Byzantine fault-tolerant (BFT) style of consensus protocol for permissioned (or enterprise) blockchains, firstly proposed in Ethereum community. It operates through a predetermined set of authority nodes (or sealers) to finalize (or seal) blocks. PoA offers relatively fast performance in terms of transaction throughput, confirmation latency, energy efficiency, and low probability of being forked. For these reasons, PoA is being popular in many blockchain networks and there are production-ready open-source implementations such as Clique [2], Aura [1], and Hyperledger Besu [10]. As a result, PoA is being actively utilized in various blockchain-based services such as IoT [9], Smart Grid [11], MicroGrid [18], and Smart Home [14].

However, PoA has several issues. First, it is inefficient due to a limited block generation in a round. In the Ethereum Clique PoA network, only one block by a sealer is allowed to be accepted in a round. This leads to the rest of the sealers doing nothing or making unnecessary proposals, resulting in resource waste. Second, PoA essentially centralizes the block generation authority to a specific leader in a round, creating a potential single-point-of-failure.

To tackle these issues, we focus on the existing Ethereum Clique PoA network and propose an extension by integrating a directed acyclic graph (DAG)-based Byzantine fault-tolerant consensus. Recent research has shown that DAG-based consensus is highly effective in enhancing blockchain performance [12, 15]. This approach organizes the blocks of a blockchain into a DAG structure, allowing multiple authority nodes to propose blocks concurrently in a decentralized manner. These proposed blocks, if valid, are integrated into the DAG without wasting the resources of the participating authorities. Therefore, by supporting DAG-based consensus in PoA, we can improve PoA's performance and simultaneously alleviate its centralization issue.

In this paper, we propose DaPoA, a novel design proposal, by extending Ethereum PoA Clique Networks with DAG-based consensus to support high performance in permissioned blockchain systems. DaPoA enables multiple authorities to propose their blocks concurrently in a round, and these blocks are incorporated into a reliable DAG using the DAG-based consensus. With this, DaPoA reduce a waste of resource because the non-leader's proposals in a round are not discarded, which is a key difference from the existing Clique algorithm. Additionally, we develop an optimization technique called asynchronous block proposals to further enhance performance. We have implemented a prototype of DaPoA on top of the Ethereum Clique, by integrating a state-of-the-art DAG BFT consensus protocol with minimal modifications.

2 Background and Related Work

In this section, we describe the background knowledge related to the Ethereum PoA algorithm and DAG-based consensus, along with relevant research.

2.1 Proof-of-Authority

A PoA network consists of n fixed authority nodes, called sealers. Each sealer is equipped with a private/public key pair, which is used to sign messages and verify signatures. PoA proceeds in consecutive rounds, with one predetermined sealer in each round, i.e., leader, proposing a block in a rotating manner. This block proposal contains a batch of transactions and is signed by the leader, then broadcasted to other sealers, determining the finality of the block. Note that non-leader sealers can also submit block proposals, but with a random delay calculated by the formula of $\frac{2}{n+1} \times \text{wiggleTime}$, where the default value of wiggleTime is 500. However, the leader's block is highly likely to be accepted first under normal circumstances, based on a higher difficulty value in the block header because the value is 2 for a leader proposal, while it is 1 for non-leader proposals. Consequently, in a single round, blocks proposed by non-leader sealers usually do not get contained in the finalized block, leading to unnecessary resource wastage by non-leaders in normal case operation.

CoPoA [17] represents a recent effort to enhance PoA performance by supporting concurrent block generation in the PoA network. In CoPoA, multiple authorities are allowed to propose blocks concurrently in a round, which are then incorporated into a super block made by round leader. While the goal of exploiting idle resources in non-authority nodes to improve block concurrency in the PoA networks is similar, the key difference between our proposal and CoPoA lies in the usage of DAG BFT as the backend consensus protocol to achieve high performance. Moreover, in CoPoA, although the blocks themselves are proposed concurrently, ultimately it is the single leader's responsibility to combine them into a super block. This may potentially lead to a performance bottleneck in terms of centralization, especially under malicious leaders. On the other hand, DAG BFT does not essentially require a leader node, and is therefore free from the potential problems.

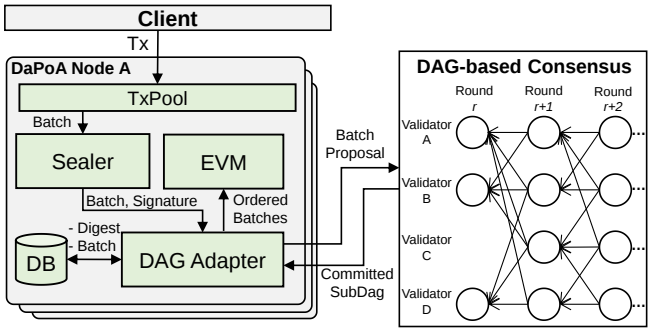

Figure 1. Design of DaPoA.

2.2 DAG-based Consensus

DAG-based BFT consensus is a recent breakthrough for high performance in blockchain community [12, 15]. The main idea of DAG BFT is to separate the reliable dissemination of transaction block data, which occupies a high traffic volume on the network, from the ordering logic that determines their sequence, with zero message overhead. The system supports Byzantine fault tolerance for f failures among N nodes in a partially asynchronous network.

We briefly describe its operation. The protocol proceeds in consecutive rounds, and in a round, each node called validator proposes a block that contains transactions received from clients and corresponds to a vertex in the DAG, and reliably broadcast it to other nodes to achieve the integrity and availability of the proposed block. Then, each validator replies with an acknowledgment containing the block digest, round number, and creator's identity, along with a signature if the block is valid. If $2f + 1$ distinct acknowledgments are obtained, each validator combines them to create a certificate of block availability and broadcasts it to other nodes. This is then included in the block of the next round, resulting in the block having references, i.e., edges, to previous blocks, leading to a local view of the DAG in each validator. Finally, by deterministically interpreting their local DAG, they achieve consensus by locally calculating the total order of the blocks. Note that DAG blocks have a property of causal history due to references, and this enables a single certificate for multiple DAG blocks, thereby making high-throughput consensus possible.

3 Design and Implementation

In this section, we describe the design and implementation of DaPoA. Each DaPoA node corresponds to an authority

node (or sealer) in Ethereum PoA and is based on Geth. We reused modules within Geth, such as Clique, TxPool, and the Ethereum Virtual Machine (EVM), enabling minimal modifications for our purposes. Additionally, we have introduced a DAG Adapter module for the integration of DAG consensus. For DAG BFT, we utilized Narwhal from Sui Blockchain [8].

The operation of DaPoA is presented in Fig. 1. DaPoA nodes continuously receive transactions from nearby clients and store them in the local TxPool. Then, each node extracts transactions from the TxPool to form a batch and passes it to the Sealer module at the beginning of each round. The Sealer calculates the digest of the given batch and creates a corresponding signature. Next, the batch and signature are transmitted to the DAG adapter for consensus. Each DAG adapter in DaPoA nodes is a key component for supporting DAG-based consensus, corresponding to one validator in the consensus network and communicating necessary data for protocol operations. Specifically, the DAG adapter builds a batch proposal based on the batch and signature received from the Sealer and sends it to the corresponding validator. The DAG adapter also maintains a local key/value DB using the digest as a key and the batch as a value, adding an entry to this DB with every batch proposal. Once a unit of consensus is completed in DAG BFT, the DAG Adapter receives a CommittedSubDag from its validator, containing committed blocks over the DAG from multiple rounds, their order, and proof, i.e., quorum certificates for the sub-dag blocks. Finally, the DAG Adapter builds ordered batches by reading batches of transaction payload in order from the local DB and passes them to the EVM to execute the committed transactions. Through this process, DaPoA integrates DAG BFT consensus with Ethereum PoA, allowing concurrent block proposals without discarding non-leader's proposals in the DAG, thereby improving the performance of Ethereum PoA.

We also introduce asynchronous block proposals, a technique to boost the performance of DaPoA. A synchronous operation in which each DaPoA node proposes a block and waits for it to be committed on the DAG would be inefficient, especially under high transaction workloads, as it may lead to increased idle time of resources. Therefore, in DaPoA, multiple blocks are allowed to be proposed asynchronously across rounds before the previous rounds are fully completed. As a result, many blocks can be committed on the DAG in a unit of consensus, enabling a higher throughput. Additionally, this provides opportunities to process numerous transactions in a CommittedSubDag, further enhancing the transaction execution efficiency.

We elaborate on the implementation details of the DAG Adapter, which provides abstraction APIs for hiding technical details of DAG BFT to the Geth. With this, it makes the Geth easily utilize external modules of DAG BFT while minimizing the implementation effort in Geth. Another functionality of the DAG Adapter is that it hides the differences between two different languages. Narwhal is written in Rust, while Geth is written in Golang. This necessitates consistent conversions between the data types of different languages. This conversion is handled by the DAG Adapter, and to achieve this, we utilized ProtocolBuffers over gRPC [3, 6, 7]. For consistent signature creation and verification, we employed blst [5] for BLS12-381, enabling consistent signature generation and verification across both Rust and Golang.

4 Conclusion and Future Work

In this paper, we present DaPoA, a novel design proposal that enhances existing Ethereum PoA networks through the integration of a DAG-based consensus. DaPoA addresses a problem of limited block generation in PoA networks by supporting concurrent block generation in a decentralized manner using DAG-based consensus. The implementation of DaPoA involves minimal modifications to the existing Ethereum Geth. As our current prototype is still in its initial stages, it is necessary to explore several areas for future work. We plan to perform a systematic evaluation of DaPoA using Hyperledger Caliper [4], applying a realistic workload to analyze the throughput and latency of the prototype. Additionally, we plan to identify performance bottleneck and further design optimization points based on this analysis. Lastly, we are particularly interested in methods to accelerate transaction execution within DAG BFT.

Acknowledgments

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2021-0-00484, Core Technologies for Hybrid P2P Network-based Blockchain Services)

References

[1] 2023. *aura.* https://github.com/openethereum/parity-ethereum/
[2] 2023. *clique.* https://github.com/ethereum/go-ethereum/tree/master/consensus/clique/
[3] 2023. *A high performance, open source universal RPC framework.* https://grpc.io//
[4] 2023. *Hyperledger Caliper.* https://hyperledger.github.io/caliper/
[5] 2023. *Multilingual BLS12-381 signature library.* https://github.com/supranational/blst/
[6] 2023. *A native gRPC client & server implementation with async/await support.* https://github.com/hyperium/tonic
[7] 2023. *Protocol Buffers. language-neutral, platform-neutral extensible mechanisms for serializing structured data.* https://protobuf.dev/
[8] 2023. *Sui Narwhal Implementation.* https://github.com/MystenLabs/sui/tree/main/narwhal
[9] Subhi Alrubei, Edward Ball, and Jonathan Rigelsford. 2021. Securing IoT-Blockchain Applications Through Honesty-Based Distributed Proof of Authority Consensus Algorithm. In *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA).* 1–7. https://doi.org/10.1109/CyberSA52016.2021.9478257
[10] Hyperledger Besu. 2023. *Proof of authority consensus.* Retrieved Aug. 8, 2023 from https://besu.hyperledger.org/private-networks/concepts/poa

[11] Ugonna Chikezie, Tutku Karacolak, and Josue Campos Do Prado. 2021. Examining the Applicability of Blockchain to the Smart Grid Using Proof-of-Authority Consensus. In *2021 IEEE 9th International Conference on Smart Energy Grid Engineering (SEGE)*. 19–25. https://doi.org/10.1109/SEGE52446.2021.9534994

[12] George Danezis, Lefteris Kokoris-Kogias, Alberto Sonnino, and Alexander Spiegelman. 2022. Narwhal and Tusk: A DAG-Based Mempool and Efficient BFT Consensus. In *Proceedings of the Seventeenth European Conference on Computer Systems* (Rennes, France) *(EuroSys '22)*. Association for Computing Machinery, New York, NY, USA, 34–50. https://doi.org/10.1145/3492321.3519594

[13] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system.

[14] Pranav Kumar Singh, Roshan Singh, Sunit Kumar Nandi, and Sukumar Nandi. 2019. Managing Smart Home Appliances with Proof of Authority and Blockchain. In *Innovations for Community Services*, Karl-Heinz Lüke, Gerald Eichler, Christian Erfurth, and Günter Fahrnberger (Eds.). Springer International Publishing, Cham, 221–232.

[15] Alexander Spiegelman, Neil Giridharan, Alberto Sonnino, and Lefteris Kokoris-Kogias. 2022. Bullshark: DAG BFT Protocols Made Practical. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* (Los Angeles, CA, USA) *(CCS '22)*. Association for Computing Machinery, New York, NY, USA, 2705–2718. https://doi.org/10.1145/3548606.3559361

[16] Gavin Wood et al. 2014. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper* 151, 2014 (2014), 1–32.

[17] Xiaohua Wu, Jinpeng Chang, Hongji Ling, and Xueqi Feng. 2022. Scaling proof-of-authority protocol to improve performance and security. *Peer-to-Peer Networking and Applications* 15, 6 (2022), 2633–2649.

[18] Jiawei Yang, Jiahong Dai, Hoay Beng Gooi, Hung Dinh Nguyen, and Amrit Paudel. 2022. A Proof-of-Authority Blockchain-Based Distributed Control System for Islanded Microgrids. *IEEE Transactions on Industrial Informatics* 18, 11 (2022), 8287–8297. https://doi.org/10.1109/TII.2022.3142755

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009