Weighted voting for optimising Streamlined Blockchain Consensus Algorithms

Micloiu Diana[1]

Responsible Professor: Jérémie Decouchant[1]
Supervisor: Rowdy Chotkan [1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
May 24, 2024

Name of the student: Micloiu Diana
Final project course: CSE3000 Research Project
Thesis committee: Jérémie Decouchant, Rowdy Chotkan, Kaitai Liang

Abstract

TO DO

1 Introduction

Many distributed systems, such as state machine replication and blockchains, have a common core concept: *consensus*, which denotes the collective agreement among network participants. Distributed systems are known to be prone to hardware and software failures, which can compromise availability or even change the system's normal behaviour. Hence, consensus is needed as a mechanism for coordinating the system's critical actions and ensuring its functionality.

In the blockchain world, consensus algorithms lay at the basis of distributed ledger technologies. They are classified into permissioned and permissionless, distinguishing each other by either limiting participation to a predetermined set of nodes or allowing anyone to join. Out of the two, permissionless systems gained more popularity, with the seminal Nakamoto consensus relying on proof-of-work. However, its significant impact on energy consumption revealed the system's limitations and urged researchers to look for alternative consensus algorithms. In contrast, interest in permissioned systems grew as their efficiency in terms of throughput, latency, and finality was observed. Thus, the focus shifted towards finding ways to optimise their performance. Strategies such as system size reduction [1] and leader selection mechanisms [2] have been explored to enhance scalability and resilience. Notably, the first-generation Practical Byzantine Fault Tolerance algorithm (PBFT) [3] has been a focal point of research in permissioned systems.

PBFT is part of a more prominent family of protocols: Byzantine Fault Tolerant (BFT), which enables systems to tolerate arbitrary node failures ([4], [5], [6]). In particular, the protocol to withstand f failures requires $3f + 1$ nodes in the system. Hence, the capability of the system to resist failures comes with the cost of managing the demand for the increased number of nodes and higher communication complexity. Furthermore, in the efficient scenario, the leader is truthful. However, this is not always the case, and the protocols need to support intricate fallback strategies, which usually imply node synchronisation and state transfer.

The main disadvantages of BFT protocols, namely that they are slow and expensive to run, support the research of the streamlined and cluster-based algorithms. In this sense, researchers from VMware Research developed Hotstuff [7], a protocol that achieves partial synchrony by using leader rotation on each command to shift the communication burden from the leader. By using a star-type communication network, the protocol achieves linear message complexity and faster response times. Additionally, current research is being conducted to optimise the features of streamlined algorithms [8], such as Pili [9], Pala [10], Streamlet [11], Tendermint [12] and, previously mentioned, Hotstuff [7]. For instance, DAMYSUS [13] improves on top of Hotstuff by reducing the number of communication phases using trusted components, thus achieving better performance.

Over time, researchers discovered that the consensus phase could be a critical point of improvement. In this sense, the idea of using a weight metric as voting power gained popularity with proof-of-stake (PoS) and reputation-based [14] protocols. Building on top of this kind of mechanism, WHEAT [15] achieved higher performance for state machine replication in geographically distributed settings. Next, researchers put together an enhanced version of PBFT, namely BFT-SMaRt [16] and the weighted voting mechanism behind WHEAT to create AWARE [17], a deterministic, self-monitoring and self-optimising algorithm for optimising the latency of the blockchain.

So far, research on the benefits of weighted voting has only studied first-generation algorithms such as PBFT. The paper seeks to address this literature gap by investigating the impact of weighted voting on streamlined consensus algorithms, such as the Hotstuff [7] family ones. By extending the principles established by AWARE to newer generations of consensus mechanisms and evaluating their robustness in the face of node failures, we aim to contribute to the broader understanding of weighted voting's efficacy in accelerating consensus and fostering distributed trust in blockchain systems.

2 Related work

The literature related to the aforementioned scientific gap revolves around two key concepts: **weighted voting and streamlined algorithms**. Therefore, we provide an overview of each of the two research areas to gather a better understanding of the improvements that have been achieved over the years.

Weighted voting Inspired by the popularity of the Proof-of-stake protocols in permissionless blockchain systems, researchers have adapted this idea into a mechanism of using a weight metric as the voting power of nodes in permissioned systems. One of the first projects highlighting the advantages of weighted voting was the Cosmos Network, which used a Tendermint-based blockchain protocol and a stake-based voting approach for reaching consensus [18]. Next, the possibility of using weights for electing the leader has been researched in credit-based PBFT (CPFT) [19], a blockchain algorithm which tunes the weights based on their past behaviour such that the probability of electing a good leader increases. Three years later, new research on using vague sets and credit rating for optimising the consensus of credit-based PBFT blockchain algorithms appeared[20]. Later on, starting from the same idea of assigning credits to nodes, researchers developed CG-PBFT [21], a blockchain algorithm which uses a novel credit evaluation model together with a three-way quick sorting algorithm to group the nodes into three categories (master, consensus and observation node groups). In this way, CG-PBFT achieves around 50% increase in throughput. Moreover, current research has tackled the idea of a reward and punishment system based on node ranking in D-PBFT [22].

Streamlined algorithms The research area for streamlined BFT protocols gained interest with the introduction of Hotstuff [7], urging the study of possible optimisations performed on this second-generation protocol. Considering multiple points of improvement, researchers came up with variations of Hotstuff targeting a better overall performance of

the systems. Sync Hotstuff [23] represents one notable research, which shifts the focus from the partially synchronous model of Hotstuff to a fully synchronous one to highlight the trade-offs and impact on performance and security. Next, researchers targeted improving performance by introducing DAMYSUS [13], a blockchain protocol which enhances the Hotstuff one by using trusted components to increase resilience and decrease the number of communication phases. On the same path of decreasing the number of communication steps, Hotstuff-2 [24] represents a two-phase variant which explores the relation between leader responsiveness and liveness for achieving optimal results.

Weighted voting on streamlined algorithms Current research also explored the idea of applying the weighted voting scheme showcased in AWARE [17] to Hotstuff. FLASH-CONSENSUS [25] is a recently discovered approach of optimising AWARE by using an adaptive resilience threshold. The research also mentions experiments combining this new algorithm with weighted voting. Thus, the effectiveness of using weighted voting on Hotstuff has been proven. However, the research uses the FLASHCONSENSUS blockchain algorithm, not the original version of AWARE, and it also leaves out details regarding the implementation of weighted voting on the second-generation algorithm. Moreover, the impact on its Chained version remains unexplored.

3 Hotstuff: the streamlined approach

Inspired by the simplicity of the theoretical protocol Streamlet [11] and part of the BFT family, Hotstuff [7] is a blockchain protocol that sets itself apart by achieving linear (in the number of nodes) communication complexity. The protocol benefits from the mechanism of switching leaders in each successive round and requires at least $3f + 1$ nodes to tolerate $1/3$ Byzantine faults.

There are two protocol versions, namely the *Basic Hotstuff* and the *Chained Hotstuff*. The difference between the two comes from the latter's enhanced voting mechanism, which enables a pipelined approach of moving forward multiple blocks in only one round (*view*). For completely processing a block, Hotstuff uses five communication phases, with three phases being core: **prepare, pre-commit,** and **commit** and two additional phases, **new-view**, for sending the last prepared block at the beginning of the protocol and **decide** at the end to actually execute the blocks.

New-view In this first phase, the leader receives the last prepared block and its corresponding view number from each node. This step consists of gathering all the new-view messages triggered at the end of the precedent view.

Prepare This phase concerns finding the next proposal. The leader awaits for $2f + 1$ quorum of nodes with their block-view number information and chooses the block with the highest view number to be extended. Next, the leader sends to all replicas its proposal, and they each provide back a vote if the SAFENODE [7] condition is satisfied. In this sense, a replica will accept the proposal if the proposed block extends its latest locked block or at least a prepared block from a view higher than the one of the last locked block [13]. These checks will ensure the safety and liveness of the protocol, respectively.

Pre-commit The block proposed is marked as *prepared* as the leader gets $2f + 1$ votes from the replicas, forming a quorum certificate. The leader sends this certificate to all the replicas so that each can verify it, mark the proposed block as prepared and vote for it in the pre-commit phase.

Commit The leader collects again $2f + 1$ votes and forms a quorum certificate of the prepared block, which becomes *locked* at this step. Next, the leader sends this certificate to all replicas to lock the same block, followed by each sending back a vote.

Decide The leader gathers a quorum agreeing on the locked block and then *executes* it. All replicas perform the same action after they receive the certificate from the leader.

Locking mechanism The efficiency of communication complexity in the Hotstuff protocol comes with the cost of implementing one more step, namely locking. That is, the block is prepared and then locked in the commit phase to preserve the *liveness* of the blockchain consensus algorithm, and only afterwards is it considered *safe* to execute in the final phase.

4 WHEAT: the weighting mechanism

Based on the BFT-SMaRt protocol of State Machine Replication, WHEAT [15] solves the problem of optimising the system's latency in geo-replicated settings. The algorithm is able to attain better performance by introducing the concept of additional replicas. In the BFT family, quorums are formed by gathering a majority of replica responses. In contrast, in WHEAT, the size of a quorum is smaller or equal to that of the Byzantine majority.

Effectively, WHEAT uses weighted voting to achieve consensus faster by leveraging the heterogeneity of the wide-area network (WAN). That is, the protocol assigns higher weights to the replicas that yield the lowest end-to-end latency. In this way, they form smaller quorums but account for the majority needed to move forward. The rest of the replicas constitute a fallback strategy since they form a larger quorum that is in place if the faster replicas become idle. Furthermore, AWARE [17] enhances this technology by introducing mechanisms for self-monitoring (deterministic latency prediction) and self-optimisation (voting weights tuning and leader relocation), such that latency is decreased by giving more power to better performing replicas.

In the Byzantine Fault Tolerant world, a quorum system represents a collection of subsets of replicas which could possibly form a quorum, and any two subsets intersect by $f + 1$ replicas [26]. By adding Δ extra replicas and enforcing a quorum formation mechanism relying on weighted replication [17], WHEAT imposes the subsequent safe weight distribution scheme [15].

Consider a BFT system of n replicas withstanding a maximum of f faulty and including Δ additional ones. Hence, n can be expressed as follows:

$$n = 3f + 1 + \Delta \tag{1}$$

Furthermore, regarding consensus, each replica should wait for a quorum formation of Q_v weighted votes:

$$Q_v = 2(f + \Delta) + 1 \tag{2}$$

The weighted voting concepts take the form of a binary weight distribution over the replicas of WHEAT. Each node has value either $V_{\max}$ or $V_{\min}$, which are computed as follows:

$$V_{\max} = 1 + \frac{\Delta}{f} \tag{3}$$

$$V_{\min} = 1 \tag{4}$$

In the system, $2f$ replicas, which are best performing in terms of latency, are attributed weight $V_{\max}$, and all the others take $V_{\min}$. Consequently, the weighted quorum contains at most $n - f$ and at least $2f + 1$ replicas.

5 Weighted Hotstuff

6 Weighted Chained Hotstuff

7 Best Hotstuff: Optimal weighting scheme

8 Experimental Setup and Results

TO DO

9 Responsible Research

TO DO

10 Discussion

TO DO

11 Conclusions and Future Work

TO DO

References

[1] D. S. Silva, R. Graczyk, J. Decouchant, M. Völp, and P. Esteves-Verissimo, "Threat adaptive byzantine fault tolerant state-machine replication," in *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2021, pp. 78–87.

[2] Y. Amir, B. Coan, J. Kirsch, and J. Lane, "Prime: Byzantine replication under attack," *IEEE transactions on dependable and secure computing*, vol. 8, no. 4, pp. 564–577, 2010.

[3] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *OsDI*, vol. 99, no. 1999, 1999, pp. 173–186.

[4] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," in *Concurrency: the works of leslie lamport*, 2019, pp. 203–226.

[5] C. Cachin and M. Vukolić, "Blockchain consensus protocols in the wild," *arXiv preprint arXiv:1707.01873*, 2017.

[6] C. Natoli, J. Yu, V. Gramoli, and P. Esteves-Verissimo, "Deconstructing blockchains: A comprehensive survey on consensus, membership and structure," *arXiv preprint arXiv:1908.08316*, 2019.

[7] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, "Hotstuff: Bft consensus with linearity and responsiveness," in *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, 2019, pp. 347–356.

[8] E. Shi, "Streamlined blockchains: A simple and elegant approach (a tutorial and survey)," in *Advances in Cryptology–ASIACRYPT 2019: 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part I 25*. Springer, 2019, pp. 3–17.

[9] T. H. Chan, R. Pass, and E. Shi, "Pili: An extremely simple synchronous blockchain," *Cryptology ePrint Archive*, 2018.

[10] ——, "Pala: A simple partially synchronous blockchain," *Cryptology ePrint Archive*, 2018.

[11] B. Y. Chan and E. Shi, "Streamlet: Textbook streamlined blockchains," in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, 2020, pp. 1–11.

[12] E. Buchman, J. Kwon, and Z. Milosevic, "The latest gossip on bft consensus," *arXiv preprint arXiv:1807.04938*, 2018.

[13] J. Decouchant, D. Kozhaya, V. Rahli, and J. Yu, "Damysus: streamlined bft consensus leveraging trusted components," in *Proceedings of the Seventeenth European Conference on Computer Systems*, 2022, pp. 1–16.

[14] J. Yu, D. Kozhaya, J. Decouchant, and P. Esteves-Verissimo, "Repucoin: Your reputation is your power," *IEEE Transactions on Computers*, vol. 68, no. 8, pp. 1225–1237, 2019.

[15] J. Sousa and A. Bessani, "Separating the wheat from the chaff: An empirical design for geo-replicated state machines," in *2015 IEEE 34th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2015, pp. 146–155.

[16] A. Bessani, J. Sousa, and E. E. Alchieri, "State machine replication for the masses with bft-smart," in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 2014, pp. 355–362.

[17] C. Berger, H. P. Reiser, J. Sousa, and A. Bessani, "Aware: Adaptive wide-area replication for fast and resilient byzantine consensus," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 3, pp. 1605–1620, 2020.

[18] J. Kwon and E. Buchman, "The cosmos network: A primer on tendermint-based blockchains," Tendermint Inc. and Interchain GmbH, White Paper, 2016. [Online]. Available: https://github.com/cosmos/cosmos/blob/master/WHITEPAPER.md

[19] Y. Wang, Z. Song, and T. Cheng, "Improvement research of pbft consensus algorithm based on credit," in *Blockchain and Trustworthy Systems: First International Conference, BlockSys 2019, Guangzhou, China,*

December 7–8, 2019, Proceedings 1. Springer, 2020, pp. 47–59.

[20] Z. Zeng, B. Wen, W. Du, F. Zhang, and W. Zhou, "Pbft consensus algorithm optimization scheme based on vague sets and credit rating," in *2023 6th International Conference on Software Engineering and Computer Science (CSECS)*. IEEE, 2023, pp. 1–5.

[21] J. Liu, X. Deng, W. Li, and K. Li, "Cg-pbft: an efficient pbft algorithm based on credit grouping," *Journal of Cloud Computing*, vol. 13, no. 1, pp. 1–20, 2024.

[22] J. Wang, W. Feng, M. Huang, S. Feng, and D. Du, "Improvement of practical byzantine fault tolerance consensus algorithm based on diana in intellectual property environment transactions," *Electronics*, vol. 13, no. 9, p. 1634, 2024.

[23] I. Abraham, D. Malkhi, K. Nayak, L. Ren, and M. Yin, "Sync hotstuff: Simple and practical synchronous state machine replication," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 106–118.

[24] D. Malkhi and K. Nayak, "Hotstuff-2: Optimal two-phase responsive bft," *Cryptology ePrint Archive*, 2023.

[25] C. Berger, L. Rodrigues, H. P. Reiser, V. Cogo, and A. Bessani, "Chasing the speed of light: Low-latency planetary-scale adaptive byzantine consensus," *arXiv preprint arXiv:2305.15000*, 2023.

[26] D. M. M. Reiter *et al.*, "Byzantine quorum systems," *Distributed Computing*, vol. 11, no. 4, pp. 203–213, 1998.