Evaluating the network traffic on an improved version of the Committeeless Proof-of-Stake blockchain consensus algorithm

George Gigilas Junior[1], Filipe Franco Ferreira, Marco A. A. Henriques[1]

[1]Faculdade de Engenharia Elétrica e de Computação
Universidade Estadual de Campinas (Unicamp) – Campinas, SP - Brasil

{g216741,f251027}@dac.unicamp.br, maah@unicamp.br

Abstract.

Blockchain is a powerful way to store and process data in a decentralized way. Among its consensus algorithms, Committeeless Proof-of-Stake (CPoS) is a promising alternative to the better-known Proof-of-Work and Proof-of-Stake, with its reduced power consumption and more straightforward design without validation committees. However, CPoS is still an emerging algorithm and requires more testing to validate its correctness and efficiency. One of the problems it has to deal with is the higher network traffic compared to similar approaches due to the larger number of messages sent. This article aims to modify CPoS in order to reduce the data traffic and improve its performance.

1. Introduction

Blockchain can be defined as a distributed ledger replicated between the member nodes that form the network, in which they produce blocks to be added to the chain. Distributed consensus algorithms, such as Proof-of-Work (PoW) [1] and Proof-of-Stake (PoS) [2], help decide what block will be appended next. While the first is used in Bitcoin [1], the most popular cryptocurrency [3], it requires excessive computational operations, which wastes a lot of energy and, in turn, makes the Bitcoin blockchain one of the most power-hungry systems on Earth [4]. The latter, on the other hand, is more sustainable and efficient [2], as it is based on sortitions made at regular intervals. The chance of a node being chosen is proportional to its stake. This algorithm became more popular with its adoption by Ethereum [5], the next most popular cryptocurrency [3]. Despite its efficiency, PoS centralizes the block validation process on the validation committee nodes. Since decentralization is among the most important blockchain properties, this is a concern. Also, it adds extra complexity to the scheme and can be exploited as seen in M. Neuder, et. al [6].

To solve this problem, D. Martins [7] proposed a PoS consensus algorithm without a validation committee: the Committeeless Proof-of-Stake (CPoS). It has shown satisfactory results but can be improved by reducing the network traffic. The generated traffic has high latency implications that can limit CPoS' scalability. On the other hand, due to the focus of previous implementations of CPoS on the consensus mechanism, the blocks sent through the network were smaller than they should be, as they were not fully loaded with transactions.

In this context, this work completes the blocks with random bytes to simulate the overhead of heavier blocks and decreases the traffic data by reducing communication between peers using an improved protocol. With this new version, we performed new experiments, searching for the scheme's best parameter settings. As a result, we got new and positive results, although some instabilities still remain and need to be addressed.

2. Committeeless Proof-of-Stake consensus scheme

In Committeeless Proof-of-Stake (CPoS) [7], instead of having a committee for block validation, all participant nodes can verify the sortitions and confirm blocks locally. Also, there can be many blocks produced per round, but only one of them will be confirmed on the blockchain. If W is the total global amount of stake in a round, p the probability of a node being selected in one sortition, and τ the expected total number of blocks produced per round, we have that $\tau = W \times p$. The probability p can be adjusted by proper configuration of parameters τ and W.

The sortition happens every round and consists of using the hash of the previous blockchain block, along with other node and block data, to get a verifiable pseudorandom number: the proof hash. This number, based on the probability p, gives each node a number of sortition selected tickets that can be 0 or higher. Although each selected ticket can be associated with a new block proposal, after this process, the nodes share only the block with the smallest proof hash. Then, from all shared blocks, the one with the smallest proof hash is added to the blockchain in consensus.

This is a delayed consensus algorithm; so it takes a few rounds to confirm the winning block (and irreversibly add it to the chain). During this time, the nodes can receive a new block with an even smaller proof hash through a gossip-based communication protocol. In this case, such nodes will remove the block included temporarily in their local chains and replace it with the newly received block. Since the selection criteria are unambiguous, the mechanism will probabilistically converge.

3. Related Work

In the initial proposal of CPoS, its properties, and fundamental relations controlling its parameters were shown [7]. It was also shown that CPoS is more secure than PoW regarding conflicting confirmations or blockchain reversions. Also, CPoS has lower power consumption and requires less computational power than PoW, as expected. Compared to Casper (Ethereum's PoS) [9], CPoS confirms blocks in a simpler way, and the confirmation latency is constant and lower for each produced block.

However, because it is sensitive to its parameters, CPoS has to be fine-tuned to reach the best possible performance. While previous work explored parameter tradeoffs, it is still a challenge to determine optimized settings [7,8]. For instance, a higher value of τ makes the scheme more reliable and fault-tolerant, but significantly increases the network traffic due to the large amount of blocks shared each round. This can be a problem because network delays can impact the synchronization and invalidate blocks that arrive too late.

4. Method

We made additions to the code and performed new experiments to evaluate the network traffic produced by the scheme while varying the configurable parameters, as the traffic is one of the most crucial characteristics of CPoS. The main addition was the insertion of pseudo-transactions in the blocks, which increases their size and impacts the communications. Because there is no actual application generating transactions yet, we randomly generated them with normally distributed sizes, with a mean based on the size of Ethereum's transactions at the time of the study (around 600 bytes).

Another important modification was in the number of peers for each node. A more restrictive limit can reduce communication costs without interfering with the consensus. Then, we refactored the network topology by reducing the number of random peers that the beacons (the initial points of contact with CPoS for a new node) share each time a new node joins the scheme. Also, to prevent a node's peer list from growing or shrinking too much, we added a management system that randomly forgets peers or requests the beacon for more, depending on the size of the list. The number of peers shared by the beacon and the limits of the peer list size are new configurable parameters. Finally, we refactored the resynchronization routines to request no more than one block at a time (when a node finds a fork condition), which further reduced the data volume sent by the nodes.

For our experiments, we used 25 nodes distributed in Docker containers among 18 hosts, Intel Core i5-2310 CPUs running Ubuntu 22.04 at 2.90GHz with 8GiB RAM and 1Gbps LAN. There was only one beacon running Ubuntu 24.04 in a VM with two cores and 8GB of RAM, connected to a different, but close network segment (just two switches hops). As for the parameters, we used different values of τ , round time (to accommodate round computation and block transmissions) and number of peers P defined by the beacon. The minimum size of the peer list was set to P-1 and the maximum number to P+2.

5. Results and Discussion

The experiments were executed on the CPoS code, available on Github[1], with the aforementioned additions. To explore different parameter sets, we used combinations of the following values: $\tau \in (3, 5, 10, 20)$, round time $\in (5, 1, 15)$ seconds, and number of initial peers $P \in (3, 5, 10)$. For all tests, we executed the scheme with 25 nodes, at most 2 nodes per host, for 30 rounds, and extracted the mean of 3 runs for each combination.

As seen in Table 1, the results were sensitive to the configurable parameters, mainly to the value of τ. For $\tau = 3$ and 5, there was a high amount of confirmed blocks and, consequently, blocks confirmed per minute. This indicates that the system worked well for these parameters, even though each peer had an initial peer list of three peers. It is important to note that, as the round time gets bigger, the throughput (blocks confirmed per minute) gets lower because there are fewer rounds (and fewer blocks produced) each minute.

The system gets unstable for $\tau = 10$ and 20, as the throughput is lower. As can be seen, the confirmation delay surpassed 30 rounds, which indicates there was no consensus for the majority of blocks produced. This is due to the increase of blocks produced per round, which causes the nodes to trigger more resynchronization requests, as stated in CPoS' original proposal [7]. With more blocks per round, receiving a late block is more likely, which leads to a fork. When the node detects a fork, it asks its

[1]https://github.com/regras/cpos_v2

peers for blocks to work towards converging into a single global blockchain vision. The increase in resynchronization requests indicates the nodes' visions are not converging and is likely why the throughput is so low for these cases which, in turn, explains the high confirmation delays. The run with $\tau = 10$ and round time 5s seems to be at the frontier of stable behavior.

Table 1. Results of the experiments ran with three peers per node

tau	Round Time	Blocks confirmed/min	Confirmation delay (rounds)	Number of messages	Data sent (MiB)
3	5s	8.66	5.3	3,469	265
3	10s	4.61	5.0	3,872	303
3	15s	2.72	4.5	3,886	306
5	5s	7.12	5.3	5,018	371
5	10s	4.95	5.0	5,304	419
5	15s	2.79	5.9	5,112	395
10	5s	4.60	6.6	6,055	401
10	10s	0.55	> 30	6,482	405
10	15s	0.64	> 30	6,119	370
20	5s	0.81	> 30	7,400	454
20	10s	0.34	> 30	7,079	444
20	15s	0.31	> 30	7,168	443

In Table 2, we have the results from the experiments for an initial peer list of 5 peers. For $\tau = 3$ and $\tau = 5$, the throughput results got even better than those in Table 1, but the total data transferred (network traffic) was higher, due to the increased number of peers. Also, for higher values of τ, the results were better, especially for $\tau = 10$, but still worse than the others. Nevertheless, we confirmed that a higher initial peer list is a step towards stabilization.

Table 2. Results of the experiments ran with five peers per node

tau	Round Time	Blocks confirmed/min	Confirmation delay (rounds)	Number of messages	Data sent (MiB)
3	5s	9.37	4.1	6,001	467
3	10s	4.33	5.0	6,154	479
3	15s	3.31	4.1	7,158	569
5	5s	9.82	4.6	8,180	631
5	10s	5.16	4.5	8,577	683
5	15s	3.52	4.5	8,825	707
10	5s	6.79	6.3	9,408	678
10	10s	3.27	6.1	9,507	673
10	15s	1.83	9.6	9,758	707
20	5s	4.48	6.9	10,793	735
20	10s	2.08	> 30	11,528	793
20	15s	0.68	> 30	11,681	772

Another set of experiments can be seen in Table 3, with 10 initial peers for every node. Now, the system showed good results for all values of τ, which shows that a higher number of peers reduces the number of resynchronizations since blocks tend to arrive quicker on all nodes. Regarding network traffic, comparing the total data with Tables 1 and 2, we can clearly see that increasing the number of initial peers on each peer list increases the number of messages and the total data volume. The more peers a node has, the more peers it will send its blocks to. This points towards a tradeoff between stability and network traffic.

Table 3. Results of the experiments ran with ten peers per node

tau	Round Time	Blocks confirmed/min	Confirmation delay (rounds)	Number of messages	Data sent (MiB)
3	5s	9.73	4.9	12,487	976
3	10s	4.92	4.0	12,575	991
3	15s	3.48	3.8	12,714	1,000
5	5s	9.76	4.3	14,921	1,160
5	10s	5.09	5.4	16,561	1,315
5	15s	3.52	4.6	16,785	1,339
10	5s	9.51	4.5	19,831	1,556
10	10s	5.30	4.1	21,703	1,735
10	15s	3.61	4.1	22,290	1,780
20	5s	3.46	5.8	17,014	1,185
20	10s	1.70	12.5	18,787	1,382
20	15s	1.11	8.7	18,586	1,350

6. Conclusions and Future Work

Committeeless Proof-of-Stake seems to be a promising blockchain consensus algorithm compared to the traditional PoW and PoS algorithms. It surpasses their advantages by having a lower power consumption than PoW and by not needing a validation committee as conventional PoS. However, CPoS is still new and needs continuous improvement. In this ongoing work, we improved the CPoS code by supporting pseudo-transactions in the blocks, refactoring the network topology and the resynchronization algorithm, as well as removing some bugs. We performed new tests to evaluate the network traffic, since its high data traffic is one of the main characteristics that affect the system's overall performance, while also fine-tuning its configurable parameters. As a result, we understood that the system converges well into a consensus and confirms a reasonable number of blocks if the values of τ and round time are kept within secure margins. This also helps reduce the network traffic, compared to other settings. This traffic is still high and new ways of dealing with it must be found.

Future work includes the search for new ways of reducing the network traffic. This can be done with a change in the logical topology and resynchronization mechanism. The data collected indicates that the alternative block-sharing mechanism as proposed in ref. [7] can be indeed a good solution. With this, initially, only the block headers are shared, until there is a consensus about the winner. Only after this, the node that authored the block will share its entire content. This can be effective and considerably reduce the traffic volume. Finally, it is important to perform new experiments with a larger number of well-distributed nodes, and with more rounds. Tests involving dishonest nodes are also needed to evaluate their impact on the system's security, resilience, and stability.

Bibliographic References

[1] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system. [Online] Whitepaper (2009). Available at: https://bitcoin.org/bitcoin.pdf (last accessed on July 5th 2024).

[2] C. T. Nguyen, et al., Proof-of-Stake Consensus Mechanisms for FutureBlockchain Networks: Fundamentals, Applications and Opportunities. IEEE Access, v. 7, p. 85727–85745 (2019). ISSN 21693536. Available at: https://ieeexplore.ieee.org/document/8746079 (last accessed on July 5th 2024).

[3] J. Royal, B. Baker, 12 most popular types of cryptocurrency (2024). Bankrate. Available at: https://www.bankrate.com/investing/types-of-cryptocurrency/ (last accessed on August 18th 2024).

[4] M. Wendl, M. H. Doan, R. Sassen, The environmental impact of cryptocurrencies using proof of work and proof of stake consensus algorithms: A systematic review. Journal of Environmental Management (2023), 326:116530.

[5] G. Wood, Ethereum: A secure decentralized generalized transaction ledger. Ethereum Project Yellow Paper (2014).

[6] M. Neuder, D. J. Moroz, R. Rao, D. Parkes, Low-cost attacks on Ethereum 2.0 by sub1/3 stakeholders. (2021). Available at: https://arxiv.org/pdf/2102.02247.pdf (last accessed on July 5th 2024).

[7] D. F. G. Martins, Um novo mecanismo de consenso probabilístico para blockchains públicas. Dissertação de mestrado (2021). Available at: https://repositorio.unicamp.br/Busca/Download?codigoArquivo=507683 (last accessed on July 5th 2024).

[8] V. Peixoto, M. A. A. Henriques, Analysis of Committeeless Proof-of-Stake protocol: Searching for a better point of operation. XXII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (2023). Juiz de Fora, MG, Brasil: Sociedade Brasileira de Computação.

[9] V. Buterin, V. Griffith, Casper the Friendly Finality Gadget (2019). Available at: https://arxiv.org/pdf/1710.09437 (last accessed on August 18th 2024).