Pack and Propagate: an Improved Blockchain Consensus Algorithm

Faisal N. Abu-Khzam
Lebanese American University, Lebanon
`faisal.abukhzam@lau.edu.lb`

Khaleel Mershad
Lebanese American University, Lebanon
`khaleel.mershad@lau.edu.lb`

Elio Rahi
Lebanese American University, Lebanon
`elio.rahi@lau.edu`

Jad El Masri
Lebanese American University, Lebanon
`jad.elmasri@lau.edu`

January 2024

Abstract

Consensus algorithms play a central role in the operation of blockchain systems. The main objective is to ensure agreement among network participants on the validity of transactions and the state of the distributed ledger. The distributed model can sometimes suffer from transaction confirmation latency that is due to how the consensus process propagates throughout the network. With this in mind, a new consensus protocol is proposed that can avoid long chains of successive transactions by first computing a d-packing network set: a group of nodes any two of which are at least $d + 1$ hops away from each other. The main idea is to start the consensus propagation process from each of the d-packing nodes since any other node in the network is within d hops from the packing nodes. This approach is implemented and tested across networks of varying sizes, ranging from small to very large. Thorough experimental evaluation showed, empirically, that consensus is almost always achieved at a faster pace on medium-to-large networks, and can result in saving 5.8% of the time and up to 4% reduced variability as well as 80% decrease in worst-case max propagation time.

Keywords: Blockchain systems, d-packing, consensus algorithm, propagation model, consensus delay, Ethereum.

1 Introduction

Blockchain systems are a groundbreaking technology enabling secure and transparent transactions in a decentralized environment. The main approach depends on the utilization of a distributed transaction confirmation system to avoid any possible third-party intervention. The idea of blockchain consensus is based on the original work of Lamport et al. on the Byzantine problem, which simply asks how to ensure a "trustworthy communication in an adversarial network environment" [8]. Blockchain systems started to gain popularity after the 2008 work of Satoshi Nakamoto [10], who presented the Bitcoin system. The proposed system, which gave birth to the first generation of blockchain networks, used Merkle trees along with a bookkeeping mechanism to efficiently record data [11].

At the heart of blockchain systems is the concept of a consensus algorithm, enabling participants in the network to agree on whether a transaction is valid or not and the current state of the blockchain. The consensus algorithm establishes trust in an untrusted environment and ensures each copy of the distributed ledger is valid and identical.

Consensus algorithms comprise the rules and processes that determine how consensus should be securely reached in a blockchain network. They are critical for maintaining the integrity, security, and authenticity of the blockchain; defending against cyber attacks, and ensuring transactions are confirmed and recorded in a timely manner.

Several popular consensus algorithms have been proposed, each with its own strengths and specific applications. Among the popular ones are proof of work (PoW), proof of stake (PoS), practical byzantine fault tolerance (PBFT), proof of authority (PoA), and proof of elapsed time (PoET). A detailed description of the most used consensus algorithms is presented in the next section.

Efficiency in consensus algorithms is crucial, as it directly impacts the scalability and performance of the whole blockchain system. The main objective of this work is to achieve consensus with minimal computational effort and time, without compromising the security and decentralization of the network.

While traditional consensus algorithms such as PoW execute the consensus processes among all consensus nodes, using a smaller set of nodes for faster consensus is an appealing, potential solution to improve the overall performance. By limiting the process to a subset of nodes, networks can achieve agreement in a more efficient manner, potentially with a higher transaction throughput. An example of such an algorithm is PoS, in which a set of 128 validators are selected to form a consensus committee for each epoch. However, the proposed solution should not affect the system's degree of decentralization. An efficient consensus algorithm should be able to securely reach an agreement among all consensus nodes in the least possible time.

The notion of a d-packing plays a central role in our proposed approach. Simply, a d-packing in a network (or graph) is a set S of nodes such that any two nodes in S are within a distance of at least $d + 1$ from each other. As such, an independent set is nothing but a 1-packing. While computing a maximum-size d-packing is NP-hard, greedily computing any d-packing can be done efficiently. In our work, the smaller the d-packing the better, thus a greedy approach that favors maximum-degree nodes should be good enough, at least from a computational standpoint. Furthermore, and to compute the smallest possible d-packing, our greedy approach favors nodes of higher connectivity, or influence, simply by making sure a large number of nodes can be reached in at most d-hops from each of the selected (packing) nodes. This helps us achieve our objective, which is faster transaction propagation.

Our main contribution consists of introducing the d-packing approach and exploring its potential application in the blockchain consensus process, proving how this algorithm reduces the latency and improves the overall performance of the blockchain system. By carefully selecting a subset of nodes based on their connectivity in the network, we demonstrate a novel approach that can initiate further work in this area, potentially further enhancing consensus efficiency in blockchain systems.

2 Preliminaries

A blockchain system functions as a decentralized ledger, maintaining records of transactions. Its pivotal feature lies in being an append-only database, meaning new transactions can be added but existing ones cannot be altered or removed. Transactions are grouped into blocks, with each two consecutive blocks interconnected through cryptographic hashes to create the blockchain. Additionally, the blockchain is typically duplicated across all nodes within the blockchain network. These attributes establish blockchain as a reliable and stable record-keeping system.

Figure 1 illustrates the general structure of blockchain, which consists of a group of connected blocks. Each block comprises two main components: the block header and the block body. The block header contains metadata like the block version, nonce, timestamp (representing the block's creation time), height (indicating its position in the chain), Merkle tree root hash (calculated by combining all transactions), parent block hash, and current block hash. On the other hand, the block body holds a collection of transactions along with a transaction counter.

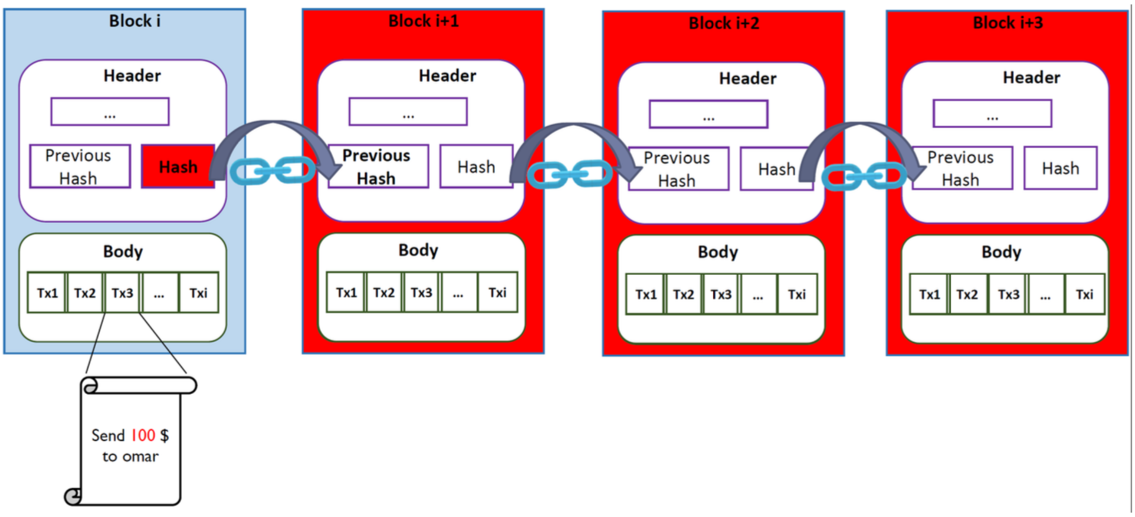


Figure 1: Blockchain general structure illustrating blocks, block header/body, and transactions.

For every block, a hash is calculated and included in the following block, enabling cryptographic linkage between blocks and ensuring the immutability of the blockchain. Should an attacker attempt to alter a transaction value within block i, the hash of that block becomes invalid, thereby invalidating all subsequent blocks, as they rely on this hash for their own integrity. The Merkle tree root facilitates rapid transaction searches. The nonce, a random number determined by the block miner (which is the blockchain node responsible for creating new blocks), must generate a hash below a predetermined target to satisfy the mining condition.

Both industry and academia have shown significant interest in blockchain owing to its attributes of being distributed, decentralized, immutable, and transparent.

- **Distributed:** In blockchain, data storage occurs in a distributed manner, with the whole blockchain stored across various nodes within the blockchain network. Furthermore, decisions regarding the addition of new data are made through a consensus protocol rather than by a central authority.
- **Decentralization:** The blockchain system operates in a peer-to-peer (P2P) manner, eliminating the need for a centralized third party. Transactions are endorsed in a decentralized fashion by the peer-to-peer network with each node having equal voting rights to all other nodes.
- **Immutability:** As discussed before, the blockchain consists of interconnected blocks secured through cryptographic hashes. Additionally, transactions are authenticated through digital signatures.
- **Transparency:** Transactions undergo validation by participant nodes before being added to the ledger, enabling all blockchain nodes to monitor and observe changes on the blockchain.

2.1 Consensus Algorithms

At the heart of the blockchain is the consensus algorithm, which establishes trust and fairness within the blockchain network and enables the blockchain nodes to add new transactions to the blockchain, ensuring that the transactions will be added correctly, without the need to trust any other nodes in the network. Consensus algorithms comprise the rules and processes that determine how consensus should be securely reached in a blockchain network. They are critical for maintaining the integrity, security, and authenticity of the blockchain; defending against cyber attacks, and ensuring transactions are confirmed and recorded in a timely manner. However, as blockchain networks grow in size, scalability challenges can lead to increased latency and reduced transaction throughput [4].

At its core, the consensus algorithm is a distributed program that is executed in parallel by the consensus nodes. The consensus algorithm includes mechanisms to secure the blockchain content against malicious actors who try to manipulate the blockchain by adding invalid transactions. A large number of consensus algorithms have been proposed in the literature for different blockchain types and environments, the most popular among which are:

- **Proof of Work (PoW):** The first and most used consensus algorithm in blockchain platforms. Miners compete to solve a difficult puzzle and get rewarded for it. PoW is very energy-consuming and, in most cases, provides advantages for high-processing computers to solve the puzzle first.

- **Proof of Stake (PoS):** Participants validate blocks based on their stake in the blockchain cryptocurrency and long-term wealth, promoting energy efficiency and incentivizing saving over spending. Several variants of PoS exist, such as Delegated Proof of Stake (DPoS), Leased Proof of Stake (LPoS), Proof of Importance (PoI), and Proof of Validation (PoV).

- **Practical Byzantine Fault Tolerance (PBFT):** Employed in scenarios where members partially trust each other. The main condition in PBFT is that the number of malicious nodes must not surpass one-third of the total participants. PBFT comprises three stages: pre-prepare, prepare, and commit. At each stage, a node proceeds to the next stage only upon receiving consistent responses from two-thirds of all nodes in the network. This capability enables PBFT to function effectively in the presence of some Byzantine replicas. However, PBFT has scalability challenges due to the exponential increase in the number of messages as new nodes are added.

- **Proof of Authority (PoA):** Validators are selected based on their identities rather than their stakes. Nodes with the most reputable identities are chosen to validate the next block. Validators are incentivized to act honestly because their reputation is on the line.

- **Proof of Capacity (PoC):** Similar to PoW in that participants must solve a computational puzzle to create new blocks. However, PoC differs by leveraging computer storage rather than processing power. In a PoC system, participants dedicate part of their hard drive space as a "plot," a pre-computed segment of data that helps generate solutions to the puzzle.

- **Tendermint:** Combines Byzantine Fault Tolerance with the decentralized environment of the blockchain, and is most suitable for consortium blockchains due to its speed and high security.

The scalability and performance of blockchain networks are greatly impacted by efficient block propagation, which is essential for reducing latency and optimizing resource usage. Many transmission methods and consensus algorithms have been developed to overcome the challenges faced by large-scale decentralized networks. Compact Block Relay (CBR), which was part of Bitcoin Improvement Proposal 152, is a well-known example. It assumes that peers already know the majority of transactions and just sends the block header and a list of short transaction identifiers. This technique significantly reduces bandwidth use and accelerates transmission, making it particularly helpful in high-throughput scenarios where network congestion is an issue [3].

The importance of efficient consensus mechanisms has driven significant research attention, particularly in exploring novel approaches that enhance blockchain scalability and reduce computational overhead. A comprehensive survey by Xiao et al. [17] categorized blockchain consensus protocols, highlighting their scalability challenges and applications. Similarly, Ferdous et al. [5] provided a broad overview of consensus mechanisms, emphasizing the trade-offs between decentralization, security, and performance in blockchain networks. A survey by Wang et al. [14] explored mining strategies and consensus mechanisms, outlining the key challenges associated with traditional protocols like PoW and PoS. Their work underscores the growing need for consensus designs that optimize resource usage while maintaining decentralization. Bashir [2] further detailed the evolution of blockchain technology, including an in-depth discussion of consensus protocols and their practical applications in various sectors.

Graph-theoretical approaches have also been creatively utilized to tackle block propagation challenges. Tan and Yap [13] introduced DAP-CBR, a dynamic pre-filling technique leveraging network graph properties to enhance propagation efficiency. Likewise, Fu et al. [6] examined graph-based optimizations for blockchain consensus, proposing mechanisms to minimize redundant transaction requests and improve scalability. Another significant area of research focuses on timeliness metrics for block propagation. Wen et al. [16] introduced an architecture that reduces the Age of Information (AoI) metric, ensuring timely and consistent block propagation. Their work contributes to maintaining ledger integrity and synchronization across distributed systems, which are critical for blockchain scalability.

Incentive mechanisms have also emerged as a key factor in reducing block propagation delays. Wen et al. [15] proposed BPIM, an evolutionary game-based incentive model tailored for 6G wireless blockchain networks. This approach classifies consensus nodes into stages and integrates individual incentives with global network objectives, achieving significant latency reductions. The study of practical Byzantine fault tolerance (PBFT) and its variations continues to be a cornerstone of consensus research. Luzipo and Gerber [9] conducted a systematic literature review of PBFT and related protocols, detailing their applicability in environments with varying levels of trust among participants. Nguyen and Kim [12] examined consensus algorithm evolution, providing a comparative analysis of major protocols, including PoW, PoS, and PoA. Recent studies have also explored alternative consensus paradigms, such as Tendermint and its hybrid Byzantine fault tolerance model. Hussein et al. [7] reviewed the latest trends in consensus algorithm design, emphasizing the potential of hybrid approaches in improving throughput without sacrificing decentralization.

2.2 Vertex d-Packing in Undirected Graphs

The graph-theoretic notion of a d-packing has been effectively applied to improve network processes. In this context, a d-packing is defined as a subset of vertices in which the distance between any two vertices is greater than d. This ensures that selected vertices are well-distributed across the network, minimizing overlap in their areas of influence. The objective is often to find a minimum d-packing that optimizes node selection for specific tasks, such as influence propagation.

Abu-Khzam et al. (2023) demonstrated the potential of d-packing in enhancing the efficiency of influence propagation in social networks by strategically selecting seed nodes to maximize coverage while reducing redundancy. Their results highlight the promise of d-packing as a key tool in solving various network optimization problems.

In combination, these developments highlight the variety of approaches used to address latency and scalability issues in blockchain systems. These methods, ranging from incentive systems and timeliness metrics to graph-based optimizations and compact block relays, are paving the way for more efficient blockchain networks. Among these, d-packing emerges as a particularly promising approach, offering a compelling balance between theoretical robustness and practical applicability for improving consensus procedures and dissemination efficiency.

3 The Pack and Spread Consensus Algorithm

The main goal of the "Pack and Spread" consensus algorithm lies in efficiently selecting a subset of nodes, referred to as d-packing, where the consensus process is initiated. The algorithm starts with a breadth-first search (BFS) strategy where we start from the nodes with the highest degree, or connectivity. This ensures a faster propagation of consensus while maintaining a minimum distance of $d + 1$ hops between any two nodes in the subset.

Algorithm Steps

(1) Initialize an empty queue Q and an array *col* to keep track of the color of the nodes, which represent their states and distances from nodes in d-packing.

(2) Select a node x with the maximum degree that has not been included in the d-packing subset and is still non-colored (or unvisited). Terminate the algorithm if no such node exists, as we have traversed all possible nodes and the subset is complete.

(3) Add x to the d-packing and initiate a BFS from x, marking it as colored in the *col* array.

(4) During the BFS, add each non-colored neighbor of the current node to the queue, ensuring no node is more than d hops away from x. This maintains the required minimum distance $d + 1$ between any two nodes in the selected subset.

(5) Repeat all the steps until no more nodes meet the requirements, indicating that the algorithm is complete.

As explained above, the d-packing algorithm favors nodes with higher degrees for faster transaction propagation in the blockchain network. By this strategic approach to selecting which nodes to propagate to, we can potentially reduce latency and enhance the overall performance of the blockchain system.

This algorithm is particularly valuable in blockchain networks, where efficiency and security are critical. Nodes with higher degrees not only enable faster propagation of transactions but also ensure that consensus information reaches a broader portion of the network more quickly. The $d + 1$ distance constraint prevents clustering, which minimizes risks such as bottlenecks or localized failures.

Figure 2 illustrates the application of the d-packing algorithm in a blockchain network. The highlighted nodes represent the selected subset of high-degree nodes, ensuring the propagation of consensus messages is efficient and distributed while maintaining the required spacing of $d + 1$ hops between nodes.

This method not only improves transaction latency but also enhances the robustness of the consensus process. By emphasizing highly connected nodes and maintaining spacing constraints, the algorithm strengthens the security and scalability of the blockchain network, aligning with its decentralized principles.

4 Experiments

In this section, we outline the setup and results of our experiments, designed to assess the impact of DPack on consensus propagation in large-scale blockchain networks. The goal of the experiments was to analyze the performance of the DPack mechanism, focusing on propagation time (both average and variability) in different network configurations.

4.1 Setup and Procedure

The experiments were conducted using SimBlock, a blockchain simulator designed for the emulation of various blockchain network sizes and configurations. The network sizes ranged from 10,000 to 200,000 nodes.

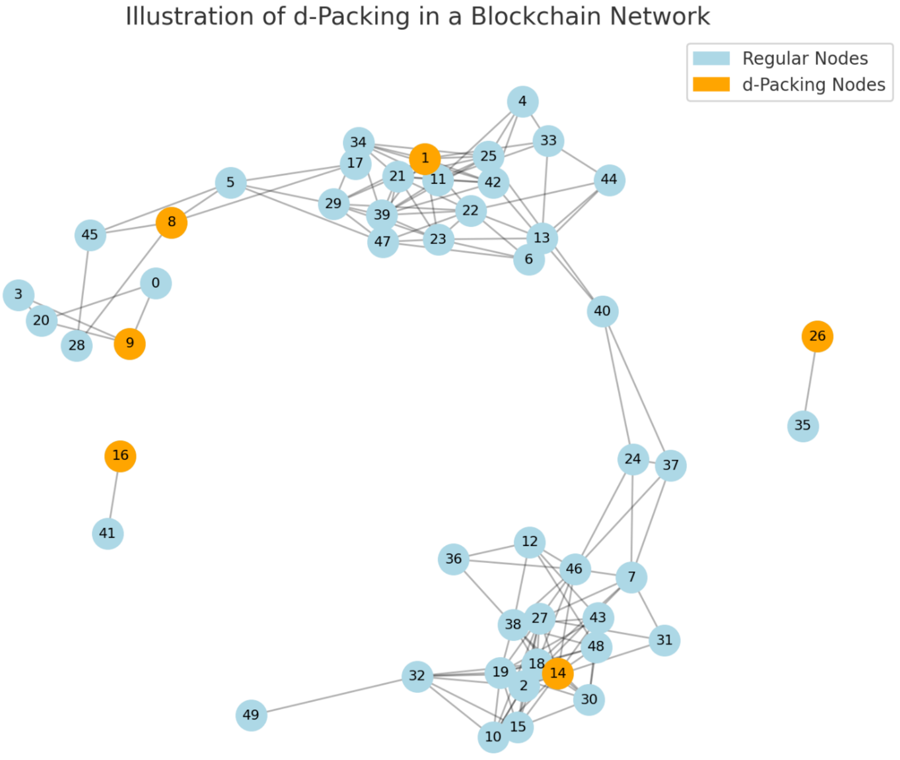


Figure 2: Illustration of *d*-packing in a blockchain network. High-degree nodes are selected for consensus propagation, ensuring efficient message dissemination and decentralized operation.

The distribution of nodes across regions reflected real-world network setups. The degree distribution of node connections followed the Bitcoin Core 2015 model to ensure consistency with established blockchain topologies. For each experiment, we configured the system to simulate scenarios both with and without DPack. Initially, DPack was used with a fixed d value of 2, ensuring that nodes participating in the consensus process were at least three hops apart. To analyze the effect of d-pack distance on performance, additional experiments were conducted by varying the d value from 2 to 8.

The percentage of nodes participating in the consensus process was also varied to assess its impact on propagation performance. We evaluated configurations where the percentage of consensus nodes ranged from 25% to 100%. The selection of consensus nodes was random to avoid bias from specific network topologies.

In each experiment, we measured two primary metrics:

(1) **Average Propagation Time:** The mean time required for a block to propagate across the network and be received by all participating nodes.

(2) **Standard Deviation of Propagation Time:** The variability in propagation times between nodes, indicating the consistency of the propagation process.

The network parameters used in the experiments are summarized in Table 3.

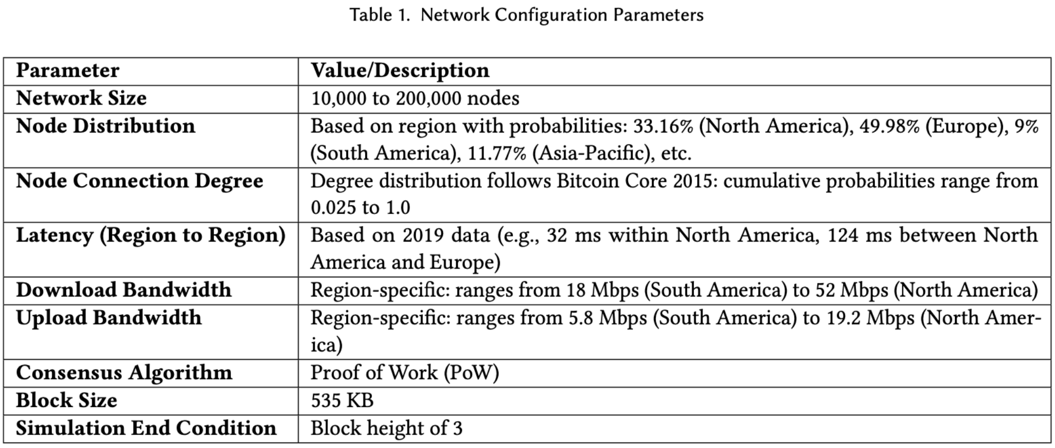


Table 1. Network Configuration Parameters

Parameter	**Value/Description**
Network Size	10,000 to 200,000 nodes
Node Distribution	Based on region with probabilities: 33.16% (North America), 49.98% (Europe), 9% (South America), 11.77% (Asia-Pacific), etc.
Node Connection Degree	Degree distribution follows Bitcoin Core 2015: cumulative probabilities range from 0.025 to 1.0
Latency (Region to Region)	Based on 2019 data (e.g., 32 ms within North America, 124 ms between North America and Europe)
Download Bandwidth	Region-specific: ranges from 18 Mbps (South America) to 52 Mbps (North America)
Upload Bandwidth	Region-specific: ranges from 5.8 Mbps (South America) to 19.2 Mbps (North America)
Consensus Algorithm	Proof of Work (PoW)
Block Size	535 KB
Simulation End Condition	Block height of 3

Figure 3: **Table 1.** Network Configuration Parameters

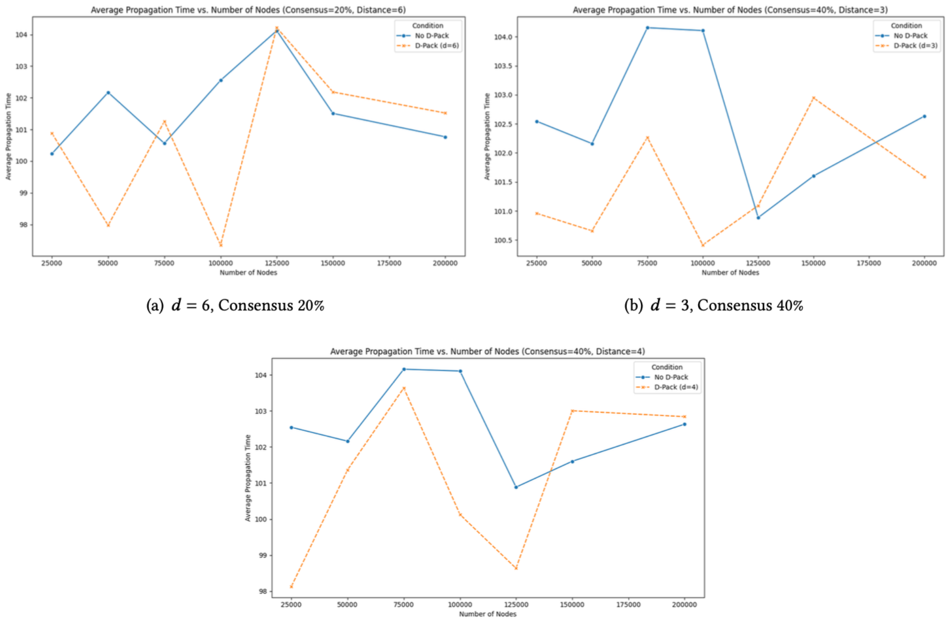
Each configuration was run multiple times to ensure the accuracy and reproducibility of results. A Python script was used to automate the collection and analysis of the results from the generated log files, enabling a systematic comparison between the DPack-enabled and non-DPack cases under varying parameters.

5 Results

This section presents the findings of our comparative analysis of blockchain network performance under two scenarios: (1) No D-Pack, where consensus propagation occurs without strategic node selection, and (2) D-Pack, where a d-packing approach is applied with varying distance parameters ($d = 2, 3, 4, 6$). Experiments were conducted across network sizes ranging from 25,000 to 200,000 nodes, with consensus thresholds of 20%, 40%, and 60%. Multiple performance metrics were evaluated to assess the impact of D-Pack, including average propagation time, standard deviation, maximum propagation time, offline time, and simulation time.

5.1 Average Propagation Time

The comparison of average propagation time is shown in Figure 4. For smaller networks (up to 75,000 nodes), No D-Pack exhibits marginally better performance, suggesting that the additional overhead of node selection does not yield significant benefits in compact topologies. However, in mid-size networks (75,000 to 125,000 nodes), D-Pack demonstrates clear advantages. For instance, at 100,000 nodes, with $d = 6$ and a consensus threshold of 20%, D-Pack reduces average propagation time from 104 ms to 97.35 ms, representing a 5.8% improvement. Similarly, for $d = 3$ and 40% consensus, D-Pack achieves 100.5 ms at 100,000 nodes compared to 104 ms without D-Pack, highlighting its consistency across this range. Beyond 125,000 nodes, the performance of D-Pack begins to oscillate, with certain configurations showing slight disadvantages compared to No D-Pack.



(a) $d = 6$, Consensus 20%

(b) $d = 3$, Consensus 40%

(c) $d = 4$, Consensus 40%

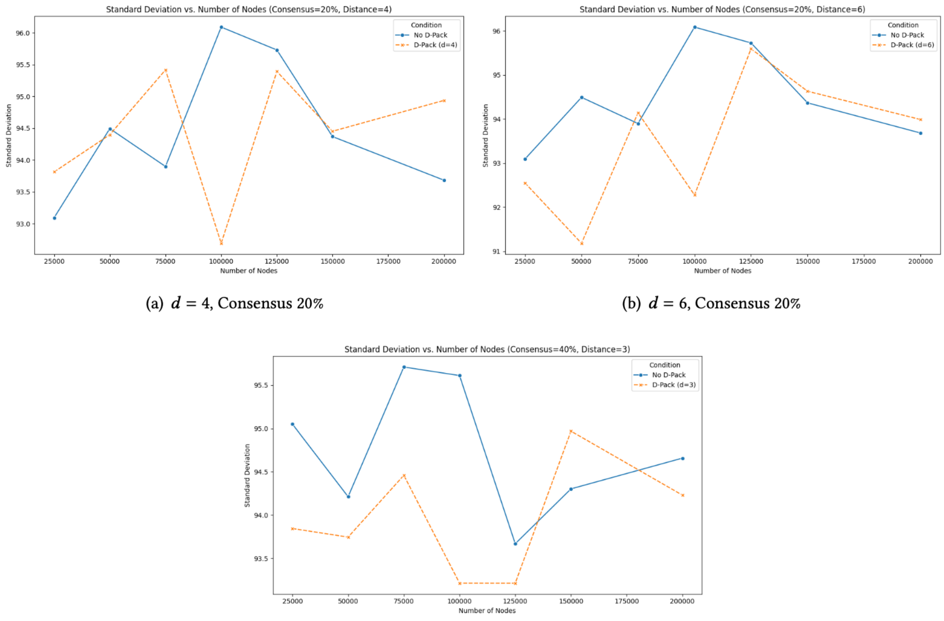
Figure 4: Comparison of average propagation time for No D-Pack and D-Pack under various configurations. Significant improvements are observed in mid-size networks (75,000 to 125,000 nodes).

5.2 Standard Deviation of Propagation Time

Figure 5 illustrates the standard deviation of propagation times across different configurations. For smaller networks, No D-Pack generally exhibits lower variability, indicating that uniform broadcasting provides stable performance at this scale. However, in mid-size networks, D-Pack consistently reduces variability. For example, at 100,000 nodes, with $d = 4$ and 20% consensus, D-Pack reduces the standard deviation from 96 ms to 92 ms, a 4% improvement. Similarly, for $d = 3$ and 40% consensus, D-Pack achieves 92 ms compared to 95.5 ms for No D-Pack, demonstrating its ability to maintain stable propagation patterns across mid-size networks. At larger scales, performance variability fluctuates, reflecting complex interactions between network size and topology.

5.3 Maximum Propagation Time

The comparison of maximum propagation time is summarized in Table 6. For a consensus threshold of 20% and $d = 2, 3, 4$, D-Pack significantly reduces worst-case delays. At 100,000 nodes, the maximum propagation time for D-Pack remains below 2,000 ms, whereas No D-Pack reaches 10,000 ms. This represents an 80% improvement, emphasizing D-Pack's ability to mitigate latency bursts and enhance network reliability.



(a) $d = 4$, Consensus 20%

(b) $d = 6$, Consensus 20%

(c) $d = 3$, Consensus 40%

Figure 5: Standard deviation of propagation time for No D-Pack and D-Pack under various configurations. D-Pack consistently reduces variability in mid-size networks.

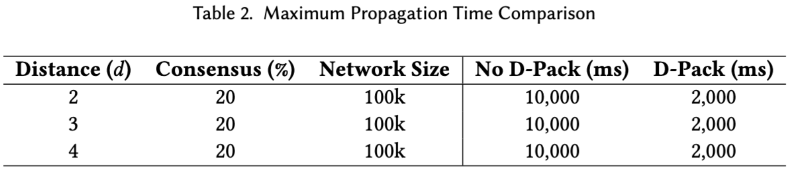


Table 2. Maximum Propagation Time Comparison

Distance (d)	Consensus (%)	Network Size	No D-Pack (ms)	D-Pack (ms)
2	20	100k	10,000	2,000
3	20	100k	10,000	2,000
4	20	100k	10,000	2,000

Figure 6: **Table 2.** Maximum Propagation Time Comparison

6 Conclusion

This paper introduced the “Pack and Spread” consensus algorithm, employing the d-packing approach to enhance the efficiency and scalability of blockchain systems. By initiating consensus propagation from strategically selected subsets of nodes, our approach aimed to reduce propagation delays and improve overall performance. Using the SimBlock simulation tool [1], we evaluated the effectiveness of this method across diverse network sizes and configurations.

Our experiments demonstrate that the choice of d significantly impacts performance, with $d = 4$ emerging as the optimal balance between node distribution and network influence. This configuration enabled the consensus process to propagate quickly while maintaining decentralized operations. Notably, the Pack and Spread algorithm excelled in medium-to-large networks (75,000 to 125,000 nodes), where it consistently reduced average propagation times by up to 5.8% and improved standard deviation metrics by 4% compared to traditional methods. However, in networks exceeding 150,000 nodes, the performance benefits fluctuated, suggesting that larger topologies introduce complexities requiring further optimization.

A key finding from our results is that d-packing significantly reduces maximum propagation delays, achieving an 80% improvement over conventional methods in certain configurations. This indicates that the approach effectively mitigates bottlenecks and enhances the reliability of consensus propagation. Additionally, the experiments revealed that d-packing maintains consistent performance across varying consensus thresholds, with 40% to 60% participation offering the best trade-off between performance gains and computational overhead.

Future work could explore adaptive d-packing strategies that dynamically adjust d values based on real-time network conditions, further optimizing propagation efficiency. Moreover, integrating this approach with lightweight communication techniques or alternative consensus algorithms could amplify its scalability and applicability across different blockchain architectures. Investigating its performance in heterogeneous and cross-chain networks would also provide valuable insights into its broader potential.

In summary, the “Pack and Spread” algorithm demonstrates that leveraging graph-theoretical principles like d-packing can enhance blockchain consensus processes. By reducing delays, stabilizing propagation, and maintaining decentralization, this method paves the way for more efficient and scalable blockchain systems capable of supporting the demands of large-scale distributed networks.

References

[1] Y. Aoki, K. Otsuki, T. Kaneko, R. Banno, and K. Shudo. 2019. SimBlock: A Blockchain Network Simulator. In *IEEE INFOCOM 2019 Workshops*, 325–329.

[2] I. Bashir. 2022. *Mastering Blockchain: Unlocking the Power of Cryptocurrencies, Smart Contracts, and Decentralized Applications.* Packt Publishing.

[3] M. Corallo. 2016. BIP 152: Compact Block Relay. https://github.com/bitcoin/

bips/blob/master/bip-0152.mediawiki

[4] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse. 2016. On scaling decentralized blockchains (a position paper). *Financial Cryptography and Data Security*, 439–456.

[5] M. S. Ferdous, M. J. A. Chowdhury, M. A. Hoque, and A. Colman. 2020. Blockchain consensus algorithms: A survey. *arXiv preprint* arXiv:2001.07091.

[6] X. Fu, H. M. Wang, P. C. Shi, and Z. J. Zhang. 2019. A survey of blockchain consensus algorithms: Mechanism, design, and applications. *Science China Information Sciences* 62(11), 21101.

[7] Z. Hussein, M. A. Salama, and S. A. El-Rahman. 2023. Evolution of blockchain consensus algorithms: a review on the latest milestones. *Cybersecurity* 6(1):30.

[8] L. Lamport, R. Shostak, and M. Pease. 1982. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.* 4(3):382–401.

[9] M. Luzipo and A. Gerber. 2021. A systematic literature review of blockchain consensus protocols. In *Proc. Int'l Conf. on Advances in Computing and Data Sciences*. Springer, 563–574.

[10] S. Nakamoto. 2009. Bitcoin: A Peer-to-Peer Electronic Cash System. https://metzdowd.com

[11] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder. 2016. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press.

[12] G. T. Nguyen and K. H. Kim. 2018. A survey about consensus algorithms used in blockchain. *Journal of Information Processing Systems* 14(1):101–128.

[13] T. Tan and T. T. V. Yap. 2024. DAP-CBR: Enhancing Bitcoin Block Propagation Efficiency Through Dynamic Prefilling. *The Journal of Supercomputing*. https://doi.org/10.1007/s11227-024-06468-0

[14] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, and D. I. Kim. 2019. A survey on consensus mechanisms and mining strategy management in blockchain networks. *IEEE Access* 7:22328–22370.

[15] J. Wen, X. Liu, Z. Xiong, M. Shen, S. Wang, Y. Jiao, J. Kang, and H. Li. 2022. Optimal Block Propagation and Incentive Mechanism for Blockchain Networks in 6G. In *TrustCom 2022*, 369–374.

[16] J. Wen, H. Zhang, and Y. Liu. 2024. Optimal AoI-based Block Propagation and Incentive Mechanism for Blockchain Networks in 6G. *arXiv preprint* arXiv:2403.12807.