Performance Enhancement of Blockchain System Using Dynamic Sharding Technique for Marks Card Management

Swapnil M Maladkar, Praveen M Dhulavvagol and S G Totad
School of Computer Science and Engineering, KLE Technological University, Hubli, 580031, India

Keywords: Blockchain, Scalability, Dynamic, Sharding, Partition

Abstract: Blockchain technology, while revolutionary in its approach to secure and decentralized data management, faces significant scalability challenges. As transaction volumes increase, traditional blockchain networks often struggle with performance bottlenecks, leading to slower transaction times, higher costs, and increased computational demands. This paper presents a novel solution to these scalability issues through the implementation of a dynamic sharding algorithm. The proposed algorithm introduces a flexible framework for partitioning a blockchain network into multiple dynamically managed shards. Each shard operates as an independent blockchain, allowing the system to distribute transaction loads more effectively and adapt to varying levels of activity. To demonstrate the effectiveness of this approach, we apply the dynamic sharding algorithm to a blockchain-based student marks card management system. This application benefits from improved scalability, security, and efficiency, addressing the limitations of traditional centralized record-keeping methods. The results show that dynamic sharding achieves a 19.5% increase in transaction throughput and a 25% reduction in latency, while also maintaining the integrity and transparency of the blockchain network.

1 INTRODUCTION

A blockchain is a decentralized ledger consisting of an expanding series of records called blocks, which are securely interconnected using cryptographic hashes. Each block contains details of the preceding block, creating a sequential chain where each new block links to its predecessor. As a result, blockchain transactions are immutable; once recorded, the data within a block cannot be modified without changing all subsequent blocks. Blockchains are usually maintained by a peer-to-peer (P2P) network, functioning as a public distributed ledger, where nodes collectively follow a consensus algorithm to validate and add new transaction blocks (Blockchain, 2024). Blockchain technology functions as a decentralized database maintained by all participating nodes, which boosts the reliability of Trusted Third-Party Auditors (TPAs) and enhances the security of data auditing processes. Its capacity to generate an immutable record of transactions ensures the integrity of data and auditing outcomes, making it challenging for malicious actors to alter the information stored on the blockchain (Y. Miao, 2024).

Despite its many advantages, blockchain technology faces significant scalability challenges. Unlike centralized systems that can quickly access their user databases, blockchain systems struggle with efficiently handling large volumes of information, resulting in lower scalability (Hisseine, 2022). Scalability in blockchain technology depends on various factors, including transactions per second, block size, chain size, and digital signatures. As the number of transactions increases, several issues arise: the average confirmation time for a transaction increases, network transaction fees rise, the difficulty of mining blocks escalates, and consequently, the required computational power and resources also grow. Additionally, the block size increases. As a result, the system becomes slower, more expensive, and unsustainable. Due to these challenges, scalability has become a key focus area for researchers in the blockchain field (Hemlata Kohad, 2020).

The conventional approach of managing student marks cards in educational institutions typically rely on centralized databases and physical records. These systems involve manual entry of student marks, storage in centralized servers, and physical

Maladkar, S. M., Dhulavvagol, P. M. and Totad, S. G.
Performance Enhancement of Blockchain System Using Dynamic Sharding Technique for Marks Card Management.
DOI: 10.5220/0013582100004664

In *Proceedings of the 3rd International Conference on Futuristic Technology (INCOFT 2025) - Volume 1*, pages 582-591
ISBN: 978-989-758-763-4

documentation, which can lead to several limitations. Firstly, centralized systems are vulnerable to single points of failure, making them susceptible to data breaches, hacking, and system downtimes. Secondly, manual data entry is prone to human errors, which can lead to inaccuracies in student records. Additionally, centralized databases can be tampered with, leading to potential data manipulation or unauthorized access. Physical records, on the other hand, are at risk of damage, loss, and degradation over time.

To address this challenge, we propose a novel and comprehensive solution that utilizes dynamic shard generation technique. The main goal is to enhance the scalability of blockchain network while maintaining the fundamental principles of decentralization, security, transparency, and immutability (Tennakoon, 2022). Dynamic sharding algorithm, dynamically creates and manages multiple shards within the blockchain network, with each shard operating as an independent blockchain, complete with its own set of verified users and data. By efficiently distributing the transaction load across these dynamically adjusted shards, the approach optimizes resource utilization and enhances overall system performance, thereby enabling seamless scalability (Wang, 2019). In the context of Record Management applications, this method can manage and scale various aspects such as student marks card management, ensuring efficient, secure, and scalable data handling.

Blockchain technology offers a robust solution to these limitations by providing a decentralized, tamper-proof, and transparent system for managing student marks cards. By utilizing a distributed ledger, each student's marks can be recorded as a transaction in an immutable ledger, ensuring data integrity and authenticity. The decentralized nature of blockchain eliminates the single point of failure, enhancing the security and reliability of the system. Moreover, the transparency of blockchain allows for easy verification and auditing of records, ensuring that any attempt to alter or manipulate data is easily detectable (I.Mohammed Ali, 2021). The use of cryptographic techniques, such as public key and private key encryption, further secures student data. Each student and institution can have their own pair of public and private keys. The public key is used to encrypt the data, making it accessible only to those with the corresponding private key, ensuring that only authorized parties can access or modify the data. This provides enhanced security measures against unauthorized access. By implementing blockchain in student marks card management, educational institutions can achieve a more secure, accurate, and efficient system for storing and managing student records, leveraging the power of distributed ledgers and cryptographic security to protect student data.

Dynamic shard generation represents an effective and integrated strategy to address scalability issues in blockchain networks. Our approach focuses on increasing transaction throughput, minimizing network latency, and optimizing resource utilization to facilitate the wider adoption of blockchain technology in various industries. This study's main objective is to implement sharding within a blockchain network to boost transaction throughput and shorten transaction confirmation times. By distributing the processing load across multiple shards, we aim to enhance the network's capacity to handle transactions more efficiently and rapidly.

A key contribution to enhancing the scalability of blockchain networks is the introduction of a dynamic sharding algorithm. This algorithm partitions the network into multiple shards, each operating as an independent blockchain with its own verified users and data. By effectively distributing transaction loads across these dynamically managed shards, the algorithm enhances resource utilization and boosts overall system performance.

The paper is organized with discussions on Related work in Section 2, followed by the Proposed Methodology of the shard generation algorithm in Section 3, and Results and Analysis in Section 4, A comparison with existing solutions highlights the advantages and effectiveness of the proposed approach.

2 RELATED WORK

This section offers an extensive review of the current literature and research on blockchain scalability, highlighting the field's status and the different strategies proposed to address scalability challenges. However, traditional distributed databases and blockchain systems each display unique failure modes, as noted in (Praveen M Dhulavvagol, 2023).

Darllaine R. et al. (Darllaine R, 2024) discussed the increasing popularity of blockchain technology but noted its significant scalability challenges, particularly in public blockchain platforms like Bitcoin and Ethereum. The main issues include low throughput, high transaction latency, and high energy consumption. This paper explores various state-of-the-art solutions, categorizing them into three layers. Layer 0 focuses on improving network information dissemination through propagation protocols. Layer 2 includes on-chain solutions like redesigning block structures, implementing Directed Acyclic Graphs

(DAG), sharding techniques, and Segregated Witness (SegWit). Layer 3 comprises off-chain solutions like side-chain techniques, payment channels, and cross-chain techniques. Sharding emerges as a notable approach, with solutions like Ostraka achieving high TPS but struggling with bandwidth and computational resource sharding, and RapidChain performing well but being prone to partitioning attacks. The paper concludes that while many promising solutions exist, each has limitations, and future research should focus on integrating offline channels with the Lightning Network approach for applications beyond payment channels.

Khacef et al. (Khacef, 2021) proposed SecuSca, a novel approach to address the trade-off between security and scalability in blockchain systems. Blockchain technology, particularly in public blockchains like Bitcoin and Ethereum, faces scalability issues due to the full replication of the entire blockchain on all nodes, leading to storage and performance problems. SecuSca introduces a dynamic sharding mechanism that reduces storage load by decreasing block replication across the network. It retains block headers but removes transaction data from older blocks on most nodes, allowing for more efficient storage use. New blocks are initially replicated across many nodes for security, but as they get buried deeper in the chain, replication is reduced while maintaining headers. An optimization function, R(d), determines replication levels based on block depth, balancing security and scalability with parameters α and γ. Simulations comparing traditional full replication and SecuSca showed significant reductions in storage requirements while maintaining security. SecuSca allows the blockchain to store more transactions with the same total storage capacity, dynamically adjusting block replication based on age and depth. Future work will focus on improving transaction verification and developing inter-shard communication protocols. SecuSca offers a promising solution to blockchain scalability by optimizing storage use, maintaining security, and increasing transaction capacity.

Qinglin Yang et al. (Qinglin Yang, 2024) offers an in-depth overview of blockchain sharding, a technique aimed at enhancing blockchain scalability without compromising decentralization. Sharding divides consensus nodes into smaller groups, allowing parallel processing of transactions and smart contracts, which significantly improves throughput and reduces transaction confirmation latency. The document categorizes sharding techniques into several key areas: processing cross-shard transactions with methods like ByShard and BrokerChain; balancing shard workloads using approaches such as TxAllo and LB-Chain; efficient smart contract execution through solutions like Jenga and Prophet; shard reorganization with techniques like S-Store; enhancing security via multi-shard oversight with CoChain; accelerating block confirmation; and stabilizing transaction pools. Experimental results highlight the superiority of sharding protocols like Monoxide and Metis over single-shard systems. Despite its benefits, sharding faces challenges such as cross-shard communication, shard rebalancing, security concerns, implementation complexity, and smart contract compatibility. The article encourages further research to address these challenges and fully harness sharding's potential to improve blockchain scalability.

Amiri et al. (Amiri, 2019) introduces a model for sharding permissioned blockchains to significantly enhance scalability by leveraging Byzantine fault-tolerant protocols among identified nodes, which traditionally require 3f+1 nodes to tolerate f failures. The proposed model innovatively optimizes the use of surplus nodes by partitioning them into clusters and sharding data across these clusters, each containing 3f+1 nodes. The system supports two types of transactions: intra-shard, which occur within a single shard, and cross-shard, which span multiple shards. It generalizes the blockchain ledger from a linear chain to a Directed Acyclic Graph (DAG), where each block contains a single transaction to boost performance. Intra-shard transactions are ordered within their cluster, while cross-shard transactions create links between shards in the DAG, with each cluster maintaining its view of the ledger. Key features of this model include parallel processing, allowing different clusters to handle independent transactions simultaneously; scalability, achieved by adding more clusters and shards; and efficient resource utilization, employing extra nodes in separate clusters. Challenges highlighted include designing consensus protocols for both intra-shard and cross-shard transactions, balancing the load across shards to maximize parallelism, and ensuring security and consistency throughout the sharded system. The potential benefits of this model are substantial, offering improved throughput through parallel processing, better scalability compared to non-sharded blockchains, and more efficient resource use in large permissioned blockchain networks. This theoretical model and architecture for sharded permissioned blockchains set the stage for future implementation and protocol design, aiming to address the scalability limitations of current

blockchain systems and leverage additional nodes for parallel processing.

3 PROPOSED METHODOLOGY

The proposed dynamic sharding approach significantly enhance blockchain scalability and performance, with a specific focus on its application in student marks card management systems. The use of cryptographic techniques, such as public key and private key encryption (Ahmad, 2023), further secures student data. Each student and institution can have their own pair of public and private keys. The public key is used to encrypt the data, making it accessible only to those with the corresponding private key, ensuring that only authorized parties can access or modify the data. This provides enhanced security measures against unauthorized access. By implementing blockchain in student marks card management, educational institutions can achieve a more secure, accurate, and efficient system for storing and managing student records, leveraging the power of distributed ledgers and cryptographic security to protect student data.

Traditional static partitioning methods often fail to adapt to fluctuating network conditions and transaction volumes, leading to inefficiencies and suboptimal performance. The proposed methodology introduces a dynamic adjustment mechanism that allows for real-time reconfiguration of partition sizes and locations, responding adaptively to changing demands. This approach not only improves resource utilization by balancing the load across partitions but also enhances transaction throughput and minimizes latency. By leveraging dynamic sharding, we aim to address common scalability challenges and ensure robust data integrity in a decentralized environment. The proposed system will enable a more responsive and efficient blockchain infrastructure, capable of handling varying workloads associated with student records management. This innovation promises to overcome the limitations of static sharding approaches, providing a scalable and adaptable solution that meets the evolving needs of educational institutions and their data management requirements.

The shard generation algorithm mainly consists of 8 steps as follows:

3.1 Initialization:

The dynamic shard generation algorithm begins by setting up the foundational parameters necessary for efficient shard management. Initially, the system defines the maximum number of partitions, or shards, that the blockchain network can support. This maximum is determined based on the expected scalability needs and the resources available. Partitions are then initialized, either based on historical data, equal distribution, or an initial workload assessment. These partitions are the segments of the blockchain that will handle transactions and data independently. Additionally, thresholds for load and transaction volume are established.

3.2 Monitoring:

Once the partitions are in place, the system continuously monitors their performance in real-time. This involves tracking various metrics such as the load on each partition (including CPU usage, memory consumption, and I/O operations) and the transaction volume being processed. Real-time data collection helps in understanding the current state of each partition and provides a basis for making necessary adjustments. By calculating average load and transaction volume across all partitions, the system establishes a baseline against which individual partitions can be assessed.

3.3 Analysis:

With real-time monitoring data in hand, the algorithm performs an analysis to identify performance issues. Partitions are evaluated against predefined thresholds. Those that exceed the load and transaction volume thresholds are classified as overloaded, while those significantly below half of these thresholds are identified as underutilized. This analysis highlights partitions that require intervention, either to redistribute workload or to consolidate resources.

3.4 Adjustment:

To address overloaded partitions, the system identifies adjacent or less loaded partitions that can absorb some of the excess load. Data and transactions are redistributed from the overloaded partitions to these selected partitions, which helps in balancing the overall load. Conversely, underutilized partitions are evaluated for potential merging opportunities. Merging involves combining underutilized partitions into a larger partition to improve efficiency and reduce overhead.

3.5 Reconfiguration:

After adjustments, the system must update its metadata and routing tables to reflect the new partition configurations. This ensures that all network nodes are aware of the changes and can correctly route transactions and access data have based on the updated partition structure. This reconfiguration step is essential for maintaining the coherence and functionality of the blockchain network.

3.6 Validation:

Following reconfiguration, the system performs validation checks to ensure that data integrity and consistency are maintained. This involves verifying that no data has been lost or corrupted during the partition adjustments. Additionally, performance metrics such as transaction throughput, latency, and resource utilization are assessed to confirm that the adjustments have achieved the desired improvements.

3.7 Iteration:

The dynamic sharding process is iterative, meaning that monitoring, analysis, and adjustment are continuous activities. The system regularly repeats these steps to adapt to changing conditions and workload patterns. By incorporating adaptive strategies and predictive analytics, the system can anticipate future load patterns and proactively adjust partitions, maintaining optimal performance and scalability over time.

3.8 Scalability and Expansion:

As the blockchain network grows, the system may need to expand the shard pool to accommodate increasing data and transaction volumes. This involves dynamically adding new shards and adjusting partitioning strategies to integrate these new shards effectively. The scalability and expansion process ensure that the network remains responsive and efficient as it scales up to handle more extensive datasets and transaction loads.

Figure 1. Represents architecture of dynamic sharding architecture that enhances blockchain scalability and performance, specifically for student record management systems. It is divided into four phases: Initialization, Monitoring, Analysis, and Adjustment. In the Initialization phase, the maximum number of partitions is set, and partitions are initialized with predefined load and transaction thresholds. The Monitoring phase involves real-time data collection on performance metrics like transaction throughput (TPS), latency, and resource

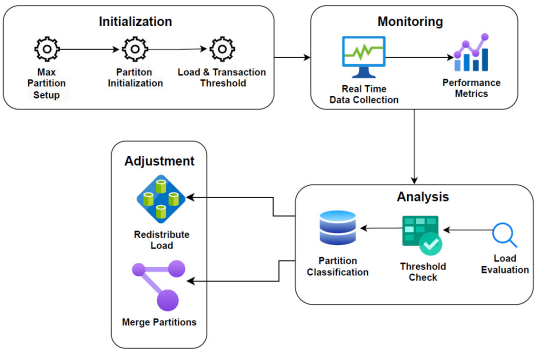

Figure 1: Proposed dynamic sharding system architecture

utilization. During the Analysis phase, this data is evaluated against thresholds to classify partitions as overloaded or underutilized. Finally, in the Adjustment phase, load is dynamically redistributed from overloaded partitions to balance the network, and underutilized partitions are merged to optimize resource use. This architecture ensures efficient, scalable blockchain operations by adapting to real-time network demands, leading to balanced resource utilization, improved TPS, and reduced latency.

Parameters involved Dynamic Shard Generation:

Maximum Number of Partitions (P_{max}): The maximum number of partitions, denoted as P_{max} , defines the upper limit of shards that the blockchain network can support. This parameter is crucial as it determines the scalability potential of the system, setting a cap on how many parallel partitions can exist. By establishing P_{max} , the system ensures controlled growth and prevents resource overcommitment. It balances the need for increased capacity with the practical limitations of computational and storage resources, thus enabling efficient and sustainable expansion.

Initial Partitions (P): Initial partitions, represented as $P= \{P_1, P_2, \ldots, P_n\}$, are the starting set of shards in the blockchain network. These partitions are established based on historical data, equal distribution strategies, or initial workload assessments. Proper initialization is essential for ensuring a balanced distribution of data and transactions from the outset. It provides a foundational structure upon which dynamic adjustments can be made, ensuring that the system operates effectively and efficiently right from the beginning.

Load Threshold (T_{load}): The load threshold, T_{load} , specifies the maximum acceptable load for each partition, encompassing factors such as CPU usage,

memory consumption, and I/O operations. When a partition's load exceeds T_{load}, it signals that the partition is under significant strain and may require intervention. This parameter helps in proactively managing performance by identifying partitions that are overloaded, enabling timely adjustments to redistribute the workload and prevent potential system bottlenecks.

Transaction Volume Threshold (T_{tx}): Transaction volume threshold, T_{tx}, sets the upper limit for the number of transactions that each partition can handle effectively. This threshold is critical for maintaining system performance and responsiveness by preventing partitions from being overwhelmed by high transaction volumes. When a partition's transaction volume surpasses T_{tx}, it triggers actions to balance the load, ensuring that transaction processing remains efficient and delays are minimized.

Average Load (avg_{load}): The average load, avg_{load} , represents the mean load across all partitions in the network. Calculating this average provides a benchmark for evaluating the performance and load distribution within the system. It helps in identifying partitions that deviate significantly from the norm, allowing for targeted adjustments to balance the load. This parameter is essential for maintaining overall system performance and ensuring that no single partition becomes a bottleneck.

Average Transaction Volume (avg_{tx}): The average transaction volume, avg_{tx}, indicates the mean number of transactions processed by each partition. By calculating this average, the system establishes a baseline for typical transaction activity, aiding in the detection of partitions with abnormal transaction levels. Monitoring avg_{tx} is crucial for ensuring that transaction loads are evenly distributed and that no partition is overwhelmed, thereby optimizing processing efficiency.

Data Redistribution Parameters: Data redistribution parameters define the criteria and methods for transferring data between partitions. These include the amount of data to be moved, the frequency of redistribution, and the techniques used for transferring data. Properly defining these parameters is crucial for maintaining balance among partitions and ensuring that workload redistribution is carried out smoothly. Effective data redistribution helps in optimizing resource utilization and preventing performance degradation due to uneven data distribution.

Merge Criteria: Merge criteria are the conditions under which underutilized partitions are combined to improve efficiency. This includes the selection process for which partitions to merge and the rules for consolidating data and resources. By establishing clear merge criteria, the system ensures that partition consolidation is done in a way that maximizes resource utilization and minimizes overhead. This parameter is key for reducing system complexity and improving overall performance by consolidating underused resources.

Consistency and Integrity Checks: Consistency and integrity checks involve mechanisms to ensure that data remains accurate and reliable during and after partition adjustments. This includes verifying that no data is lost or corrupted during data redistribution and merging processes. Implementing these checks is vital for maintaining the trustworthiness of the blockchain network, ensuring that all data remains intact and that system operations proceed without disruptions.

Reconfiguration Metadata: Reconfiguration metadata encompasses the data and routing information that must be updated to reflect new partition configurations. This includes changes to partition addresses, locations, and boundaries. Accurate updating of reconfiguration metadata is essential for ensuring that all network nodes are aware of the new partition structure and can properly route transactions and access data. This parameter is crucial for maintaining seamless operation and coherence across the blockchain network after adjustments are made.

The procedure for the dynamic hybrid sharding algorithm involves several key steps to ensure efficient load balancing and transaction management across blockchain partitions. Initially, the system continuously monitors the current load and transaction volume for each partition, gathering real-time data to inform adjustments. The average load (avg_{load}) and average transaction volume (avg_{tx}) are computed across all partitions to establish a baseline for comparison. Partitions are then categorized into overloaded and underutilized based on their performance relative to the averages. For overloaded partitions, the algorithm transfers a portion of the data to adjacent or least-loaded partitions, thereby alleviating the excessive burden. Conversely, for underutilized partitions, the system evaluates the potential benefits of merging them with adjacent or related partitions and adjusts boundaries accordingly. This redistribution ensures that load and transaction values are balanced. These steps—monitoring, analysis, and adjustment—are repeated at regular intervals or whenever significant changes in load or transaction volume are detected, ensuring the system remains responsive to fluctuations. The updated partition assignments, reflecting these dynamic

adjustments, are then returned as the final output, maintaining an optimal and efficient blockchain environment.

Dynamic shard generation algorithm:

Algorithm 1: Adjust Partitions

Input:
1. Partitions
2. Load_data
3. tx_data
4. T_load
5. T_tx

Output: Partiton adjustments made to balance load and transaction volume.

Procedure:

```
# Calculate the average load and transaction volume across all partitions.
avg_load = sum(load_data.values()) / len(partitions)
avg_tx = sum(tx_data.values()) / len(partitions)
#Identify overloaded partitions and underutilized partitions.
overloaded = [p for p in partitions if load_data[p] > T_load or tx_data[p] > T_tx]
underutilized = [p for p in partitions if load_data[p] < T_load / 2 or tx_data[p] < T_tx / 2]
#Call redistribute_load(overloaded_partition, target_partitions, load_data, tx_data) to redistribute data and #balance load
for p in overloaded:
target_partitions = find_adjacent_or_least_loaded(partitions, load_data, tx_data)
redistribute_load(p, target_partitions, load_data, tx_data)
#Call merge_or_adjust(underutilized_partition, partitions, load_data, tx_data) to either merge or adjust the #partition boundaries.
for p in underutilized:
merge_or_adjust(p, partitions, load_data, tx_data)
return partitions
```

Algorithm 2: Redistribute load across partitions.

Input:
1. overloaded_partition
2. target_partitions
3. load_data
4. tx_data

Output: The load and transaction volume of overloaded partition is redistributed across the target partitions

Procedure:

```
# Transfer data to target partitions
#Distribute the load and transaction volume from the overloaded partition to target partitions.
for target in target_partitions:
transfer_data(overloaded_partition, target)
load_data[target] += load_data[overloaded_partition] / len(target_partitions)
tx_data[target] += tx_data[overloaded_partition] / len(target_partitions)
load_data[overloaded_partition] = get_current_load(overloaded_partition)
tx_data[overloaded_partition] = get_current_tx_volume(overloaded_partition)
```

Algorithm 3: merge_or_adjust(underutilized_partition, partitions, load_data, tx_data):

Input:
1. underutilized_partition
2. partitions
3. load_data
4. tx_data

Output:
1. modify the list of partition if a merge operation occurs.
2. updates the load_data and tx_data

Procedure:

```
#Identify adjacent partitions to the underutilized partition.
adjacent_partitions = find_adjacent_partitions(underutilized_partition, partitions)
#If merging is feasible, perform the merge operation.
if can_merge(underutilized_partition, adjacent_partitions):
merge_partitions(underutilized_partition, adjacent_partitions)
#If merging is not feasible, adjust the boundaries of the underutilized partition.
else:
adjust_boundaries(underutilized_partition, adjacent_partitions, load_data, tx_data)
```

4 RESULTS AND ANALYSIS

In evaluating the effectiveness of our proposed dynamic sharding algorithm for blockchain scalability within a student record management context, we conducted a series of experiments using a simulated blockchain network. The results demonstrated significant improvements across several key metrics.

Firstly, in terms of transaction throughput, the initial static partitioning setup achieved an average of 241 transactions per second (TPS). After implementing our dynamic sharding algorithm, the TPS increased to 288, representing a substantial improvement of approximately 19.5% as shown in Fig 2. This enhancement highlights the algorithm's capability to handle higher transaction volumes through dynamic adjustments. Regarding latency, the initial static partitioning setup exhibited an average transaction confirmation time of 2.688 seconds. With dynamic sharding, this latency decreased to an average of 2.014 seconds, marking a reduction of approximately 25% which is represented in Fig 3.

This improvement indicates that dynamic partitioning allows for quicker transaction confirmations by preventing any single partition from becoming a bottleneck.

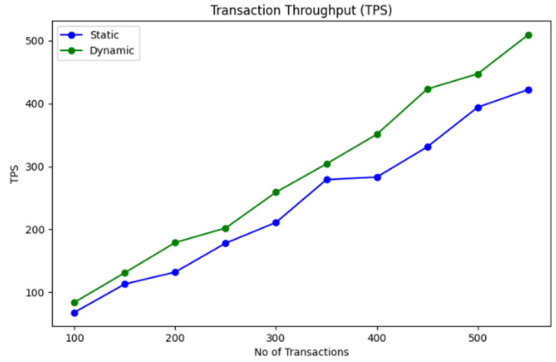

Figure 2: Comparison of Transaction Throughput (TPS)

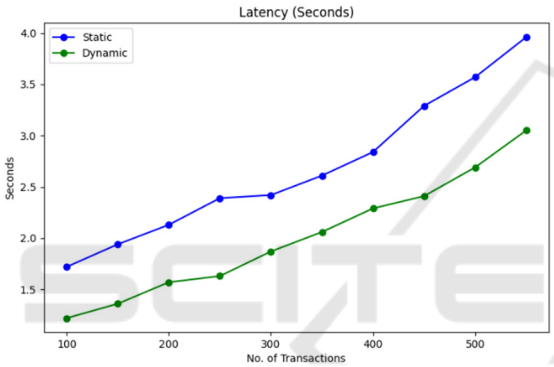

Figure 3: Comparison of Latency

Table 1. Comparison of Transaction Throughput and Latency.

Sl no.	No. of Transactions	Transactions Per Second (TPS)		Latency (seconds)	
		Initial static setup	Dynamic	Initial static Setup	Dynamic
1	100	68	84	1.73	1.21
2	150	113	131	1.94	1.36
3	200	132	179	2.13	1.57
4	250	178	202	2.39	1.63
5	300	211	259	2.42	1.87
6	350	279	304	2.61	2.06
7	400	283	351	2.84	2.29
8	450	331	423	3.29	2.41
9	500	394	447	3.57	2.69
10	550	422	509	3.96	3.05

In terms of resource utilization, the initial static partitioning setup saw uneven distribution, with some nodes being overutilized while others remained underutilized. After applying the dynamic sharding algorithm, resource utilization became more balanced, with CPU and memory usage evenly distributed across all nodes. This balance demonstrates the algorithm's effectiveness in distributing the workload and preventing resource exhaustion on any single node. Load distribution also saw significant improvements. Initially, load distribution was skewed, with some partitions frequently overloaded while others were underutilized. The dynamic sharding approach resulted in a much more uniform load distribution, with partitions dynamically resized and reallocated based on real-time network conditions. This uniformity helped maintain optimal performance and prevented overloading. Lastly, data integrity was consistently maintained throughout all experiments, with no incidents of data loss or corruption observed in both setups. This consistency confirms that the dynamic adjustments do not compromise the correctness and consistency of the blockchain data, ensuring reliable data management for student marks.

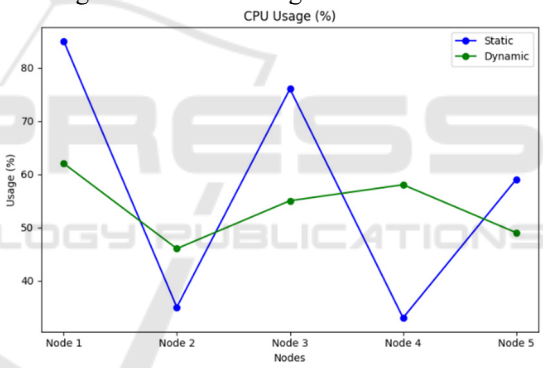

Figure 4: CPU Usage Comparison Across Nodes

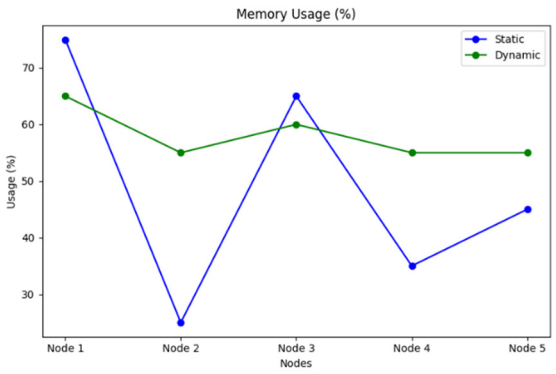

Figure 5: Memory Usage Comparison Across Nodes

Fig 4. illustrates the CPU usage across different nodes in the blockchain network, comparing the

initial static partitioning setup with the dynamic sharding setup. This approach achieves a more balanced distribution of CPU usage across nodes, addressing the inefficiencies observed in the initial static partitioning setup.

Fig 5. shows the memory usage across different nodes in the blockchain network, comparing the initial static partitioning setup with the dynamic sharding setup which results in a more balanced distribution of memory usage across nodes, mitigating the inefficiencies observed in the initial static partitioning setup.

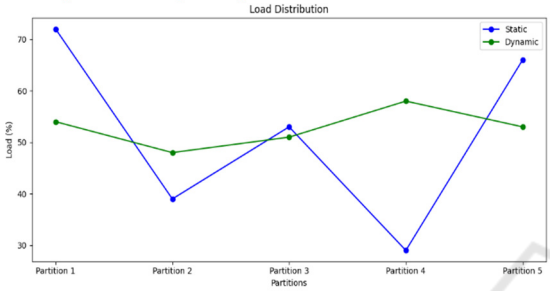

Figure 6: Load Distribution across Partitions

By comparing the initial static and dynamic load distributions, the Fig 6. highlights the effectiveness of the dynamic sharding algorithm in improving load balancing across partitions.

In the context of student marks card management, the dynamic sharding algorithm notably improved system performance. The increased transaction throughput and reduced latency ensured faster updates and retrievals of student marks. The enhanced resource utilization and balanced load distribution contributed to smooth handling of high request volumes, maintaining optimal performance even under heavy load. By ensuring these improvements while preserving data integrity, the dynamic sharding approach significantly enhances the scalability and responsiveness of the student marks card management system.

5 CONCLUSIONS

The implementation of the dynamic sharding algorithm has markedly enhanced blockchain scalability in the context of student marks card management. The algorithm achieved a notable 19.5% improvement in transaction throughput and a 25% reduction in latency, demonstrating its efficacy in handling increased transaction volumes and reducing confirmation times. Furthermore, the dynamic approach led to more balanced resource utilization and improved load distribution, ensuring efficient performance across the network. These advancements confirm the robustness and effectiveness of dynamic sharding in optimizing blockchain systems, making it a valuable enhancement for decentralized applications managing student records.

REFERENCES

Y. Miao, K. Gai, L. Zhu, K. -K. R. Choo and J. Vaidya, "Blockchain-Based Shared Data Integrity Auditing and Deduplication," in *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 3688-3703, July-Aug. 2024, doi: 10.1109/TDSC.2023.3335413.

Blockchain - “https://en.wikipedia.org/wiki/Blockchain”, 2024.

Hisseine, M.A.; Chen, D.; Yang, X. The Application of Blockchain in Social Media: A Systematic Literature Review. *Appl. Sci.* 2022, *12*, 6567. https://doi.org/10.3390/app12136567.

Hemlata Kohad, Sunil Kumarand Asha Ambhaikar, “Scalability Issues of Blockchain Technology”, International Journal of Engineering and Advanced Technology (IJEAT), vol. 9, no. 3, pp. 2385–2391, Feb. 2020, doi: 10.35940/ijeat.C5305.029320.

Tennakoon, Deepal & Gramoli, Vincent. (2022). Dynamic Blockchain Sharding. 10.4230/OASIcs.FAB.2022.6.

Wang, Gang & Shi, Zhijie & Nixon, Mark & Han, Song. (2019). SoK: Sharding on Blockchain. 41-61. 10.1145/3318041.3355457.

Praveen M Dhulavvagol, S G Totad, Performance Enhancement of Distributed System Using HDFS Federation and Sharding, Procedia Computer Science, Volume 218, 2023, Pages 2830-2841, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2023.01.254.

Darllaine R. Lincopinis, Orven E. Llantos, the current research status of solving blockchain scalability issue, Procedia Computer Science, Volume 239, 2024, Pages 314-321, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2024.06.177.

Khacef, Kahina & Benbernou, Salima & Ouziri, Mourad & Younas, Muhammad. (2021). Trade-Off Between Security and Scalability in Blockchain Design: A Dynamic Sharding Approach. 10.1007/978-3-030-84337-3_7.

Xi, Jinwen & Zou, Shihong & Xu, Guosheng & Guo, Yanhui & Lu, Yueming & xu, Jiuyun & Zhang, Xuanwen. (2021). A Comprehensive Survey on Sharding in Blockchains. Mobile Information Systems. 2021. 1-22. 10.1155/2021/5483243.

Qinglin Yang, Huawei Huang, Zhaokang Yin, et al. The State-of-the-Art and Promising Future of Blockchain Sharding. TechRxiv. January 26, 2024. DOI: 10.36227/techrxiv.170630359.94721690/v1

Amiri, Mohammad Javad & Agrawal, Divyakant & Abbadi, Amr. (2019). On Sharding Permissioned Blockchains. 10.1109/Blockchain.2019.00044.

Ahmad, Sartaj & Arya, Shubham & Gupta, Shobhit & Singh, Puneeta & Dwivedi, Sanjeev. (2023). Study of Cryptographic Techniques Adopted in Blockchain. 1-6. 10.1109/ICIEM59379.2023.10166591.

I.Mohammed Ali, Sura & Sharaf, Hussien. (2021). Using Blockchain in University Management Systems. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 12. 3305-3312. 10.17762/turcomat.v12i2.2389.

Dhulavvagol, Praveen M., S. G. Totad, and P. Pratheek. "Enhancing Transaction Scalability of Blockchain Network Using Sharding Technique." In International Conference on Soft Computing for Security Applications, pp. 253-269. Singapore: Springer Nature Singapore, 2023.

Dhulavvagol, Praveen M., M. R. Prasad, Niranjan C. Kundur, N. Jagadisha, and S. G. Totad. "Scalable Blockchain Architecture: Leveraging Hybrid Shard Generation and Data Partitioning." International Journal of Advanced Computer Science and Applications 14, no. 8 (2023).

Rane, Mukul & Singh, Shubham & Singh, Rohan & Amarsinh, Vidhate. (2020). Integrity and Authenticity of Academic Documents Using Blockchain Approach. ITM Web of Conferences. 32. 03038. 10.1051/itmconf/20203203038.

M. Dhulavvagol Praveen, S G Totad, Mahadev Rashinkar, Ribhav Ostwal, Suprita Patil, Priyanka M Hadapad, Scalable Blockchain Architecture using off-chain IPFS for Marks Card Validation, Procedia Computer Science, Volume 215, 2022, Pages 370-379, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2022.12.039. (https://www.sciencedirect.com/science/article/pii/S1877705092202110X)

Surendran, K., L. Benny, and A. S. Mahesh. "Student academic management system using blockchain technology." Journal of Advanced Research in Dynamical and Control Systems 12, no. 3 (2020): 1410-1415