Scalability and Security in Blockchain Networks: Evaluation of Sharding Algorithms and Prospects for Decentralized Data Storage

Andrey L. Bulgakov , Anna V. Aleshina * , Sergey D. Smirnov , Alexey D. Demidov , Maxim A. Milyutin , Yanliang Xin

Posted Date: 7 November 2024

doi: 10.20944/preprints202410.1078.v2

Keywords: blockchain technologies; sharding; scalability; security; green bonds; sustainable development; decentralization

 doi:10.20944/preprints202410.1078.v2

Article

Scalability and Security in Blockchain Networks: Evaluation of Sharding Algorithms and Prospects for Decentralized Data Storage

Andrey L. Bulgakov [1,2], Anna V. Aleshina [1,2,*], Sergey D. Smirnov [1], Alexey D. Demidov [1], Maxim A. Milyutin [1] and Yanliang Xin [2]

[1] Moscow Center for Fundamental and Applied Mathematics, Lomonosov Moscow State University, 119991, Russian Federation
[2] Faculty of Economics, Lomonosov Moscow State University, 119991, Russian Federation
* Correspondence: bulgakoval@my.msu.ru; aleshina@econ.msu.ru; smirnovSD@rshb.ru; demidov.ad@bk.ru; milyutinma@my.msu.ru; siya17a@econ.msu.ru

Abstract: The article addresses the issues of scalability and security in blockchain networks, with a focus on sharding algorithms and decentralized data storage. Key challenges include the low throughput and high transaction latency in public networks such as Bitcoin and Ethereum. Sharding is examined as a method to enhance performance through data distribution, but it raises concerns regarding node management and reliability. Sharding schemes, such as Elastico, OmniLedger, Pyramid, RepChain, and SSchain, are analyzed, each presenting its own advantages and drawbacks. Alternative architectures like Directed Acyclic Graphs (DAG) demonstrate potential for improved scalability but require further refinement to ensure decentralization and security. Protocols such as Brokerchain, Meepo, AHL, Benzene, and CycLedger offer unique approaches to addressing performance and transaction consistency issues. The article emphasizes the need for a comprehensive approach, including dynamic sharding, multi-level consensus, and inter-shard coordination. Additionally, a conceptual model is proposed that incorporates sharding of transactions, states, and networks, which enables greater scalability and efficiency.

Keywords: blockchain technologies; sharding; scalability; security; green bonds; sustainable development; decentralization

1. Introduction

The most important innovation of blockchain networks is the ability to decentralize data storage, ensuring reliability, sustainability and transparency of information storage. In traditional centralized data storage systems (for example, on the servers of banks or large technology companies), the key problem is the dependence on the central node, which becomes a single point of failure and the main target of attacks. Such centralized data storage is fraught with vulnerability to hacking, technical failures and risks of unauthorized access. Decentralized storage in blockchain networks, on the contrary, is implemented on the basis of a distributed architecture, where each network node stores a copy of the register, and control is distributed among all network participants. This approach eliminates the need to trust the central authority and minimizes the risks associated with data loss or damage. Transparency is another significant advantage, since the data in the blockchain is publicly available and immutable, which guarantees immutability and trust in the system. Moreover, decentralization of data storage allows blockchains to achieve high resistance to attacks and failures. In the event of failure of one or more nodes, the data remains accessible to users through other nodes, ensuring fault tolerance and continuity of the system. This unique characteristic of blockchains makes

 doi:10.20944/preprints202410.1078.v2

them particularly relevant for critical applications such as finance, healthcare, and supply chain management, where data integrity and storage reliability are critical.

In recent years, blockchain technologies have gained significant attention for their potential to revolutionize fields from finance to supply chain management. A critical requirement for broad blockchain adoption is the ability to scale while providing secure, decentralized data storage. Despite the advantages, blockchains face core challenges, with scalability and security limitations at the forefront. These challenges are especially pronounced in first- and second-generation public blockchains, such as Bitcoin and Ethereum, which struggle with low transaction throughput and high latency, limiting their effectiveness as transaction volumes grow.

One of the most promising approaches to addressing scalability is sharding, a method that divides the blockchain network into smaller, manageable segments, or "shards." Sharding enables parallel transaction processing, which improves overall performance, but it also introduces challenges in maintaining data consistency and node management. Protocols such as Benzene, OmniLedger, and CycLedger each bring unique solutions to enhance both scalability and security, though they also contend with specific trade-offs, particularly in the areas of node coordination and data reliability. Additionally, alternative architectures like directed acyclic graphs (DAGs) demonstrate potential for increasing speed and throughput without relying on traditional mining; however, these approaches still require improvements to ensure adequate decentralization and protection against malicious activity.

The purpose of this paper is to conduct a comprehensive analysis of the scalability and security of blockchain networks with an emphasis on sharding algorithms and prospects for decentralized data storage. With the growing volume of transactions and users in blockchain networks, issues of improving performance, resilience to attacks, and optimizing distributed data storage are becoming critical for their further development and adoption. In this context, the analysis of existing technologies and approaches to sharding, such as Elastico, OmniLedger, and Pyramid, as well as new conceptual models that provide efficient and sustainable data storage, is a relevant task. This paper aims to identify the strengths and weaknesses of existing algorithms and methods, assess their practical significance and application possibilities for solving scalability and security problems. Particular attention is paid to the prospects of using sharding to optimize data storage and increase network throughput.

2. Materials and Methods

The active implementation of blockchain technologies has revealed its fundamental contradictions and shortcomings, the solution of which is a key task for modern IT leaders. We are talking mainly about the problems of scalability, performance and coexistence of the blockchain pillars that underlie its conceptual model and ultimately predetermined its victory over existing algorithms for the transfer and storage of information. At the end of the last century, Eric Brewer formulated and proved the CAP theorem (from the English consistency, availability, partition), stating that in a distributed system it is impossible to ensure a stable balance and combination between consistency, availability and resistance to fragmentation. A logical consequence of the theorem was the conclusion about the incompatibility of the fundamental principles of modern blockchain: decentralization, scalability, sustainability, trust, security, etc. In each specific situation, developers face a choice, the creation of any blockchain technology begins with a search for a compromise between its named attributes. Blockchains based on acyclic graphs achieve a high level of scalability and decentralization with questionable security, security and scalability are achieved in consortium blockchains and private blockchains, but at the expense of decentralization: all of them are fully or partially centralized. Public blockchains are well protected and decentralized, but have poor scalability (for example, Bitcoin, the scalability problem of which has not yet been solved due to the specifics of the Proof of Work consensus algorithm used in it, which is almost incompatible with sharding).

The newborn and immature nature of blockchain technology explains the absence of any meaningful and influential scientific school that would create a generally accepted terminology base

 doi:10.20944/preprints202410.1078.v2

and provide answers or at least recipes for finding them to the most pressing questions in the field of blockchain; in this series - solving the scalability problem, strengthening security, increasing network throughput, improving consensus algorithms and much more. Blockchain can be private, public, or a consortium, but its ideological structure is always the same and consists of five levels: application level, data level, consensus level, network level, and platform level. The platform level includes the final products of the blockchain, such as cryptocurrency. The data level consists of hashing algorithms, data creation and verification, and digital signatures. The consensus level regulates and governs consensus algorithms. The network level includes data compression and distribution algorithms, as well as proctologist and services for exchanging messages between network participants. The platform level is the blockchain infrastructure, its software and hardware system, and technological basis. Sustainable development of blockchain technology requires coordination and regulation of all the above levels.

2.1. The Concept of Blockchain Scalability

The definition of "scalability" in the context of blockchain remains controversial. Currently, approaches to defining scalability can be divided into two main categories.

The first category of researchers considers scalability as a comprehensive term that includes such indicators as throughput, latency, the amount of data required to store, and other system parameters. For example, in the works "Solutions to Scalability of Blockchain: A Survey" and "A Systematic Review of Blockchain Scalability: Issues, Solutions, Analysis and Future Research" scalability is equated with system metrics such as throughput, storage costs, and latency in reading and writing data. In the study "Blockchain Challenges and Opportunities: A Survey", insufficient scalability is defined by low throughput, high data storage costs, and long delays in processing transactions and network requests. The second category of researchers considers scalability to be a fundamental characteristic of blockchain, comparable in importance to such properties as decentralization and immutability. In the article "Scalable Blockchains — A Systematic Review", scalability is interpreted based on established concepts in the database field. However, this definition only takes into account aspects such as throughput and storage costs (i.e. vertical scalability), without addressing horizontal scalability, which should also be taken into account when measuring these metrics. The founder of Ethereum described scalability as the ability of a blockchain to process transactions that exceed the throughput of a single node. In the works "On The Scalability of Blockchain Systems" and "Performance Analysis of a Hyperledger Fabric Blockchain Framework: Throughput, Latency and Scalability", blockchain scalability is associated with an increase in system performance as the number of participants (nodes) increases without degrading the overall performance level. In addition, based on the knowledge gained in the database field, we can classify scalability into two types. A blockchain is horizontally scalable if its system metrics improve with an increase in the number of participating nodes. Otherwise, if the system metrics can be improved by increasing the computing power and storage capacity of nodes or adjusting parameters such as block size or block interval, then the blockchain is considered vertically scalable.

Modern information technologies are faced with data sets whose sizes objectively exceed the permissible memory volumes. The lack of computing power, however, does not eliminate the need to analyze these data arrays. To solve this problem, a method of segmentation was developed - a way to divide large databases into many small sections (partitions) stored on a single database instance (server). The principles of division into sections can be arbitrary and depend on the specifics of the data - by popularity, date, geographic location or price segment. Sharding became a logical continuation of segmentation, since segmentation did not ensure the independence of sections: if one server fails, the data on it becomes unavailable. Sharding involves dividing the database into independent segments (shards), each of which is managed by a separate database instance. This not only solves the problem of fault tolerance, but also allows for increased throughput, since tasks are processed in parallel on multiple processors.

 doi:10.20944/preprints202410.1078.v2

There are three main approaches to sharding:

- The range approach involves dividing the database into ranges from which shards are formed. For example, records can be grouped by country or GDP level, but this approach can lead to uneven distribution of data.
- The key approach uses a hash function to distribute records between shards. This method provides a more even distribution of data.
- The directory approach is based on the creation of an intermediate table that determines which shard is responsible for certain data.

Despite its advantages, sharding also has its drawbacks. Currently, there are no unified algorithms that could take into account the specifics of processing transactions of each node and independently optimize the process of storing and transmitting information. Sharding requires a replication mechanism, since if one node fails, data must be available from other shards, which increases storage costs. In addition, the consensus algorithm used in the network significantly affects the efficiency of sharding; the best results are achieved with the Proof of Stake (PoS) algorithm.

There are different types of routing that help manage access to shards:

- Client-level routing requires the client to be aware of sharding, which increases the complexity of the client code.
- Proxy routing adds a proxy node between the client and the database, but creates a single point of failure and limits the ability to perform complex queries.

In blockchain, sharding is used primarily at the transaction level to increase throughput. Sharding in blockchain includes:

- Network sharding, which reduces network bandwidth requirements.
- Transactional sharding, which divides transactions between committees to achieve consensus.
- State sharding, which manages transactions between shards, which is necessary for complex applications such as NFTs and smart contracts.

There are also four main sharding patterns:

1. Transactional and state sharding;
2. Network and transactional sharding;
3. Network and state sharding;
4. Simultaneous use of all three types of sharding.

2.2. Protocols

2.2.1. Elastico

Elastico is an innovative blockchain protocol that introduces the concept of permissionless sharding. In this protocol, time is organized into different epochs, at the beginning of each of which nodes run a Proof of Work (PoW) algorithm to generate unique verifiable identifiers. Based on the last s-bits of the identifier, each node is assigned to a specific committee. The main innovation of Elastico is the creation of two types of committees: a director committee and a final committee. The director committee coordinates and forms subsequent committees, thereby ensuring the organization and distribution of tasks. After reaching consensus on the current set of transactions, the director committee passes this set to the final committee. The final committee, in turn, aggregates all legitimate transactions and distributes them for storage among all nodes in the network. Elastico represents the first significant step in applying sharding technologies to blockchain systems. Although it demonstrates a certain level of scalability, the system faces several key challenges. Firstly, Elastico does not support cross-shard transaction processing, which can lead to potential locking of funds and hinder cross-shard communication. Secondly, the need to store a global ledger for each node can significantly reduce its performance, since each node must manage extensive and constantly updated information. Thus, despite the introduction of new sharding technology, Elastico has significant limitations that require further improvement and optimization.

Limited functionality: Elastico does not provide a mechanism for handling cross-shard transactions, which creates significant limitations in functionality. This means that transactions that

 doi:10.20944/preprints202410.1078.v2

affect more than one shard cannot be executed, which can lead to problems with interactions between users and services using different shards.

Integration complexity: Since Elastico does not support cross-shard transactions, application developers are forced to find alternative ways to handle such interactions. This can require significant integration efforts, which complicates development and increases costs.

Decreased overall performance: The lack of support for cross-shard transactions can lead to increased processing times and decreased system throughput, since users who need to interact between shards must use slower and more complex mechanisms.

2.2.2. OmniLedger

OmniLedger was developed as an improved version of Elastico, with the aim of optimizing blockchain data management and increasing the efficiency of transaction processing. Unlike Elastico, which focuses on using permissionless sharding with fixed epochs, OmniLedger introduces a more flexible and adaptive approach. One of the key features of OmniLedger is the use of ByzCoin's sliding window mechanism to determine qualified nodes for the next epoch. This mechanism helps to dynamically determine and update the composition of nodes, providing more flexible network management and improving its security. The most significant innovation of OmniLedger is the Atomix protocol, which is a client-managed two-phase protocol for processing cross-shard transactions. Atomix effectively guarantees transaction atomicity, which allows transactions to be executed correctly and consistently across different shards. This feature is critical to ensuring data integrity and consistency in distributed systems where transactions may affect multiple shards. Atomix improves system performance and reliability by reducing the likelihood of errors and conflicts that may occur when processing cross-shard transactions. As such, OmniLedger represents a significant improvement over Elastico, providing more efficient and reliable transaction management on a blockchain network.

2.2.3. Pyramid

Pyramid introduced the concept of multi-layer sharding, expanding on the existing two-layer architecture. In Pyramid, nodes establish their legitimate identities through a Proof of Work (PoW) mechanism and are distributed across different types of shards based on these identities. One such type is the i-shard, which functions similarly to traditional shards and processes and stores intra-shard transactions using the Practical Byzantine Fault Tolerance (PBFT) consensus protocol.

Another type is the b-shard, or bridge shard, which connects multiple i-shards. Nodes in a b-shard represent a kind of union of multiple shards and store all the relevant state of the i-shards. This allows b-shard nodes to independently verify the validity of transactions passing between different shards. To process cross-shard transactions that no b-shard can handle, Pyramid uses a transaction splitting method inspired by Omnileger. This method involves splitting transactions across b-shard connection ranges and then processing them within the corresponding b-shards. While using overlapping shards can significantly improve the efficiency of cross-shard transaction processing, it also introduces two major problems. First, nodes in a b-shard require more computing, networking, and resource capacity to provide adequate quality of service. Pyramid, on the other hand, divides nodes into shards based on identifiers, which can be unpredictable, which does not guarantee that nodes in a b-shard will have sufficient resources to complete their tasks. Second, the configurations of i-shards and b-shards must be fixed and specified in the genesis block before the system goes live. Such a static shard configuration can significantly reduce the security of the system, since it does not allow for dynamic adaptation to changes in the network. Experiments have shown that Pyramid can only withstand 16% of malicious nodes, which is significantly lower than other schemes that use dynamic sharding and reconfiguration, which provide greater resilience to attackers.

Difficulty with network structure changes: Pyramid has limited ability to dynamically adapt shards in response to network changes, such as increases or decreases in the number of participants. This may lead to a situation where the system cannot effectively respond to changes in load, which in turn may reduce network performance and resilience.

 doi:10.20944/preprints202410.1078.v2

Scalability issues: Limited support for dynamic adaptation may hinder the scalability of the system, as the lack of flexibility in resource and node reallocation may lead to uneven shard load and increased transaction processing times.

Complex configuration: Since Pyramid does not support automatic adaptation of the network structure, administrators and developers may face additional difficulties in manually setting up and managing sharded segments, which increases operational costs and requires additional effort.

2.2.4. Repchain

RepChain is a sharding scheme that emphasizes the differences between nodes in the network in terms of their activity and role in consensus. Unlike many current sharding systems, where nodes can be divided into functional roles — active, participating in consensus, and inactive, not actively participating — RepChain seeks to balance this imbalance to minimize vulnerabilities in the network. Nodes with high activity and significant contributions to consensus have a higher reputation, which serves as an incentive for honest behavior and incentivizes their active participation.

RepChain's reputation scoring scheme is based on a weighting of each node's contribution to the consensus process. At the beginning of each epoch, nodes are sorted by their reputation values and then assigned to a shard of minimum size based on their ranks. This approach ensures a more even distribution of both the number and quality of nodes in each shard, which helps improve the overall reliability of the network. Nodes with the highest reputation are appointed as shard leaders, where they are responsible for intra-shard consensus and cross-shard transaction processing. This system creates an additional incentive for nodes to strive for a higher reputation, as it directly affects their capabilities within the network. RepChain's shard consensus is innovative due to the use of two data chains: the transaction chain and the reputation chain. To achieve consensus on transactions within shards, the Raft protocol is used, which ensures high throughput and the speed of TB block generation. However, Raft alone is not able to effectively cope with Byzantine nodes, so TB blocks are considered only candidates until they are finally confirmed. For final verification, the reputation chain is used, which applies the CSBFT consensus algorithm, which ensures an average RB block generation speed. RB blocks contain hashes of several candidate TB blocks, which allows for final confirmation of transactions after consensus on RB blocks is reached. This two-layer structure provides the necessary balance between high throughput and consensus reliability. However, the reputation system in RepChain faces certain challenges related to the attack capabilities of an adversary. If an adversary is able to control a fixed number of nodes and imitate their behavior until one of them becomes the leader, this can lead to an attack on the system. Also, a reputation-based system may be vulnerable to slow-adapting attacks, since it is less flexible in adapting to changes in the behavior of nodes. These aspects highlight the need for careful reputation management and the development of defense strategies against complex attack scenarios.

2.2.5. SSchain

SSchain implements a two-tier architecture where the first tier is a blockchain root network similar to the original Bitcoin network with a full transaction ledger stored on all nodes. The second tier consists of a network of shards where each shard processes transactions using a Proof of Work (PoW) consensus mechanism. In this structure, the root blockchain performs additional validation of blocks created by shard nodes to prevent double-spend attacks. The root block in SSchain includes transactions from all new blocks generated in shards. Since the root chain and shard networks operate asynchronously, the root block can contain multiple blocks from different shards, resulting in a Directed Acyclic Graph (DAG) structure at the two-tier network level. This DAG structure helps maintain data integrity and prevent double-spends, ensuring high security for the sharding network. SSchain's two-tier architecture does not require random distribution of nodes to ensure shard security, unlike many other systems. Nodes are free to join both shards and the root chain. The root chain ensures the security of the network, which allows SSchain to implement an incentive mechanism that directs the computing resources of the majority of nodes to support the root chain. This helps maintain a high level of consensus security and ensures an adequate number of nodes to

 doi:10.20944/preprints202410.1078.v2

simultaneously process transactions in each shard. To improve transaction efficiency, SSchain encourages users to prefer intra-shard transactions, as they are confirmed faster and have lower fees. If a transaction has multiple input addresses and does not require atomicity, the wallet automatically splits it into multiple transactions within a shard. Transactions that cannot be split and require cross-shard processing are forwarded to the root chain. The root chain stores the full transaction ledger and can directly verify and process inter-shard transactions, ensuring their correctness and integrity. This hybrid architecture of SSchain aims to optimize the throughput and security of the system by combining the benefits of high processing speed within shards with reliable transaction verification at the root chain level.

2.2.6. Brokerchain

Brokerchain is the first to address the hot shard problem that occurs when sharding random states. The hot shard problem occurs when one of the shards containing an account initiating a large number of transactions becomes overloaded, exceeding its maximum computing power. This leads to delays and blocking of transactions both within a given shard and across shards. To mitigate this problem, Brokerchain implements a load balancing and cross-shard transaction pruning strategy. The core component of the system, the p-shard, is responsible for transaction distribution. It continuously collects transactions generated in each shard in the current epoch and applies the Metis algorithm to distribute these transactions across different shards, thereby ensuring a more even distribution of the workload. This helps prevent individual shards from becoming overloaded and improves the overall performance of the system. Additionally, Brokerchain allows a single account to exist across multiple shards, where the overall account status is represented as the sum of all state shards in each shard. This allows for flexibility in managing account state and helps reduce the number of cross-shard transactions. To achieve this, Brokerchain extends Ethereum's Merkle Patricia Trie (MPT) to a Merkle Storage State Tree (MSST). The leaf nodes of the MSST still contain the state of each account, but it also adds a storage map field that indicates the shards in which the account state resides. This map is represented as a vector, where each element indicates whether a piece of the account state resides in the corresponding shard. In this way, accounts that exist in multiple shards can serve as "brokers" for processing cross-shard transactions. Cross-shard transaction processing in Brokerchain is accomplished using a two-phase lock (2PL) protocol. Brokers act on both source and target shards, converting cross-shard transactions into two intra-shard transactions, improving consistency and reducing the need for frequent cross-shard communication. Brokerchain is a fully sharded solution, storing only transactions and associated account states. However, to ensure account state consistency across shards, each shard must store a map of all account storage. However, the actual balances, code, and other states are not stored in each shard, reducing data volumes and increasing system efficiency.

2.2.7. Meepo

Meepo is a protocol that focuses on sharding consortium blockchains. Sharding does not occur between Meepo participants, but within each participant's cluster. Nodes in each cluster only process the transactions that are assigned to them. However, transactions in Meepo are broadcast to all participants. This means that each participant owns the full state of the blockchain. However, each node in the cluster only stores a portion of the full state. Nodes of each participant receive transactions on the same P2P network, and network sharding is not used, so Meepo is a partial sharding protocol, not a self-proclaimed full sharding protocol. Meepo uses multiple runtimes on each node to increase throughput, which in turn brings two significant improvements to the processing of transactions between shards. Firstly, since it is a consortium blockchain, all shards in the same cluster (one network participant) trust each other, which reduces the cost of generating and verifying signatures. Secondly, most servers in the same cluster are on the same local area network (LAN), which allows for a fully synchronized network model. Communication overhead can also be ignored.

Meepo also reserves two unique time slots before generating the next block.

- Cross-epoch - during this time, all cross-calls caused by transactions between shards are combined for batch processing.

 doi:10.20944/preprints202410.1078.v2

- Replay-epoch - during this time, erroneous transactions are removed from blocks and then re-executed. This ensures transaction atomicity.

Essentially, Meepo extends a single node in a traditional blockchain into a master-slave computing cluster, where the master server acts as a shard leader responsible for splitting transactions and delivering transactions between other shards. Slave servers execute and store transactions.

Each transaction contains two new fields:

- Shard flag. It is set by the transaction initiator.
- Stage flag. It is set by the contract developer. Each transaction is divided into several stages. Each stage is processed in different shards, thus converting inter-shard transactions into intra-shard transactions. This approach is also known as distributed transactions. A stage is executed in only one shard, and different stages of several transactions can be executed in parallel in different shards.

2.2.8. AHL

The AHL protocol is an approach to full sharding of blockchain networks. At the core of this protocol is the use of a trusted execution environment (TEE), which provides optimizations for the sharding process. In particular, AHL uses the sgx-read-rand function from Intel SGX, which is designed to generate unbiased random numbers. These random numbers serve as the basis for partitioning nodes into different shards, as well as for periodic reconfiguration of these shards. Random distribution of nodes plays a critical role in preventing attackers from bypassing the consensus protocol, as it helps maintain attack resistance. However, it is worth noting that Byzantine nodes, which are capable of performing arbitrary malicious actions, can be downgraded to common nodes that are limited in their capabilities, such as passing information without bias. In this context, weakened nodes achieve consensus fault tolerance within (n-1)/2, where n denotes the total number of nodes in the network. It is important to emphasize that despite these measures, the AHL approach only provides probabilistic shard security, not absolute security, due to the inherent limitations of sharding and the security practices of such distributed systems.

2.2.9. Benzene

The Benzene sharding scheme is the first dual-chain architecture, where each shard simultaneously maintains a proposal chain and a voting chain. This dual-chain structure separates the transaction recording process from the consensus process, facilitating the verification of cross-shard communication without affecting the independence of the transaction recording processes within each shard. In Benzene, each node is required to verify every proposal block coming from all shards. To reduce the overhead of cross-shard communication, an SGX-enabled node hosted in each shard first verifies the validity of candidate blocks and provides proofs of the validity. Then, to request votes, only these proofs need to be propagated to all shards, and the candidate block with the most votes is selected. Compared to non-cooperative designs, Benzene's architecture exhibits significantly greater fault tolerance. Attackers would need to control more than half of the shards to change the voting results, making successful attacks more difficult. However, it is worth noting that Benzene uses a traditional two-phase lock (2PL) protocol to process cross-shard transactions. Each shard must wait for at least k subsequent blocks to ensure that transactions are fully committed, resulting in a latency of over 300 seconds for cross-shard transaction confirmations. Additionally, Benzene's current implementation does not provide a clear explanation of how SGX ensures the validity of the loaded state when validating candidate blocks. If the entire state is loaded into SGX, this could result in memory limits for large states being exceeded. If Merkle proofs are provided for each accessed state, this could slow down the transaction verification process due to frequent context switches and the need to encrypt/decrypt data.

 doi:10.20944/preprints202410.1078.v2

2.2.10. CycLedger

CycLedger is a sharding system that uses reputation scores to incentivize nodes, similar to the mechanism used in RepChain. In this system, nodes' votes on transactions are considered as contributions to the network. Nodes whose votes are closer to the final consensus result receive larger rewards, which incentivizes them to operate honestly. Nodes with more computing resources are able to process more transactions and, accordingly, receive larger rewards, which encourages their honest behavior. However, CycLedger criticizes the reputation-based node assignment method used in RepChain for de-randomizing the system by only appointing nodes with high reputations as leaders. In response, CycLedger proposes an alternative approach based on the use of a random selection function with proof-of-random-function (VRF). In this approach, a referee committee is responsible for managing node identification and block packaging. In each shard, a node with the highest reputation is elected to become the leader and represent the shard in communication with the referee committee, receiving transactions and other information. In addition, CycLedger introduces the concept of an additional partial set of nodes in each shard that monitor the behavior of the current leader. If one of the nodes in the partial set detects a protocol violation by the leader, this node has the opportunity to replace the leader. Thus, the partial set of nodes acts as a candidate for leadership. CycLedger distinguishes four types of nodes: the judge node (or referee committee node), the shard leader, members of the partial set, and other non-core nodes. However, non-core nodes are required to store only transactions associated with them, while referee committee nodes are required to store all committee transactions for subsequent verification and forwarding. As a result, despite the presence of effective governance and reputation mechanisms, CycLedger can be classified as a partial sharding system, since it does not completely separate the recording and consensus processes in relation to all network nodes.

2.3. Changing the Shard System

One of the most promising technologies in blockchain is changing the shard system, transferring information from one shard to another. Rebalancing is used for this purpose. Let's consider the following system:

Let's say we need to transfer data from one shard to another. The simplest option looks like this: first, we need to cancel write operations (update or delete writes become unavailable to users for a certain period of time, clients are only given the opportunity to read). At this time (usually at night, during periods of low user activity), we transfer data. However, this scheme is not always convenient, there are situations when we cannot afford the service to be unavailable even for a short period of time, it must be available for writing constantly, then this scheme is not applicable.

Another fundamental case is logs (special text files). They are never changed or deleted, the only applicable operation (maybe with the exception of some exotic cases) is the append operation. Data transfer is carried out by creating an additional shard (target), to which we can write something. Reading can occur simultaneously from two shards - the source (src) and the target (tgt). The most common option is logical replication, which is carried out by creating a replica of the src shard, after synchronizing two shards, the first is cut off. In addition to the above approaches, all sorts of combinations and mixtures are possible.

It happens that the number of shards is insufficient and does not cope with the load, or, on the contrary, is excessive and causes unnecessary expenses, for example, due to an error in choosing a sharding strategy. Resharding was developed to add or remove nodes. Let's assume that we chose a key-based approach to form shards. Let's consider the following schemes:

Let's say there were initially 3 shards. Let's place them on a certain numerical circle, the length of which is normalized by one, and mark the starting point (zero). Now we will pass the database records through the hash function, perform normalization and place the obtained results on the circle. Next, moving clockwise, we will find the closest one in relation to each point and establish a one-to-one correspondence between them. Now, if we want to change the number of shards, for example, to two, we do not have to start the whole process again and redistribute all the records. It will be enough (in our case) to redistribute the data between the first and second shard. This approach is

 doi:10.20944/preprints202410.1078.v2

called the consistent hashing method. There are other methods of resharding, for example, randez-vous hashing.

There are other approaches to solving scalability problems. Among them are reducing the size of block data without reducing the number of transactions, directly increasing the block size, using acyclic graphs (DAG), altcoins, parallel data processing, off-chain solutions. Off-chain solutions improve the scalability of the blockchain by executing transactions or tasks outside the main chain.

The set of transactions or completed tasks for a certain period is recorded in the main chain as one transaction. For example, information about money transfers between two users is accumulated, mutual obligations, if any, are repaid and instead of many transactions, only one is performed. This is done in order to unload the main chain, increase its throughput, reduce the load on the storage, and also reduce the transaction fee.

Effective methods of blockchain optimization are the Segwit protocols. Segwit removes signatures from transaction data and adds them to the metadata along with scripts in the form of a separate structure called Witness. In addition, signatures are now only a quarter of their original size. Signatures take up about 65% of transaction data, so removing them frees up space in the block and allows more transactions to be included in the block (up to 4 times). Therefore, the throughput of transactions per second increases. Segwit also increases the Bitcoin block size from 1 MB to 4 MB. It also solves the quadratic hashing problem and facilitates payment channels such as the Lightning Network, which are other blockchain scaling protocols. Despite the benefits, the throughput improvement in SegWit is limited to 17-23 TPS.

3. Results and Discussion

3.1. Results

The results of the comparative analysis can be seen in Table 1.

Based on the Table 2, the following conclusions can be drawn:

1. Sharding and DAG are the most promising technologies for increasing the scalability of blockchains, allowing them to process a large number of transactions in parallel. For example, RapidChain in sharding reaches 4220 transactions per second, and CoDAG in DAG reaches 1151 transactions per second. However, these technologies face serious security threats. In sharding, the main threat is a 1% attack, in which an attacker can control one shard, compromising the entire system. In DAG, there is a risk of double spending due to probabilistic transaction confirmation methods.
2. Segwit provide a smaller increase in scalability compared to sharding and DAG, but they significantly improve security. Segwit solves the problem of changing transaction identifiers (malleability), which makes the system more resilient, but its increase in block size partially solves the scalability problem
3. Balance between scalability and security: Sharding and DAG offer solutions to significantly increase throughput, but require further development to address security risks. At the same time, Segwit and MAST offer robust solutions to improve security, but their scaling potential is limited. The choice of technology depends on the priority: if scalability is a key concern, then sharding and DAG will be preferable, but for projects with increased security requirements, Segwit and MAST may be the best choice.

Table 1.

Algorithm	Operating principle	Scalability	Security	Support for cross-shard transactions	Consensus algorithm	Disadvantages
Elastico	Permissionless sharding. Nodes are distributed	Good. Allows transactions to be processed in	Limited. The lack of intershard transactions	Not supported	Proof of Work (PoW) to create unique	No support for cross-shard transactions.

 doi:10.20944/preprints202410.1078.v2

	among committees that use PoW for validation. Time is divided into epochs.	parallel across different shards, increasing throughput.	reduces overall security, as funds may be blocked.		node identifiers.	Nodes must maintain a global ledger, which reduces performance.
OmniLedger	An improved version of Elastico with dynamic node selection for new epochs and the Atomix protocol for handling cross-shard transactions.	Very good. Parallel transaction processing and dynamic node configuration improve scalability.	High. Atomix provides atomic and secure cross-shard transactions	Supported via Atomix protocol	Byzantine Fault Tolerance (BFT) combined with Atomix for cross-shard transactions	Complex coordination between nodes requires significant computing resources.
Pyramid	Multi-level sharding: i-shards process transactions, b-shards coordinate inter-shard transactions.	High. The multi-tier structure improves the processing of complex transactions and increases throughput.	Moderate. High resource requirements for b-shards, which may reduce security when they are overloaded	Supported. Transactions are coordinated via b-shards.	Practical Byzantine Fault Tolerance (PBFT) in i-shards, coordination via b-shards.	Static shard configuration reduces network flexibility. B-shards require significant resources.
RepChain	Nodes are assigned a shard ranking based on their contribution to consensus. Each shard has leaders. Two chains are used: transaction and reputation.	Moderate: The reputation system improves resource allocation, but requires effective management.	High: The reputation system motivates nodes to behave honestly, but may be vulnerable to slow adaptation attacks.	Limited. Shard leaders coordinate inter-shard operations.	Raft for intra-shard consensus and CSBFT (Consensus State Byzantine Fault Tolerance) for transaction confirmation.	Vulnerability to attacks on shard leaders. The reputation system can be slow to adapt to changing conditions.
SSchain	Two-tier architecture. The root chain verifies transactions created in shards, creating a	Good. Improves efficiency with two-tier architecture and parallel transaction processing	High. The main chain verifies transactions, preventing double-spend attacks.	Supported at the root chain level.	Proof of Work (PoW) in shard networks, verification at the main chain level.	Requires storing full data on the main chain, which creates a load on the network

 doi:10.20944/preprints202410.1078.v2

	DAG structure.					
Brokerchain	Load balancing and transaction pruning across shards to address hot shard congestion.	High. Load balancing strategy and the ability to have one account in multiple shards improves performance.	High: Using a two-phase locking protocol for cross-shard transactions improves security.	Supported. Two-phase locking (2PL) protocol is used.	Two-Phase Locking (2PL) for reconciling cross-shard transactions.	The locking protocol creates a delay for confirming intershard transactions
Meepo	Some nodes in the consortium store the full state of the blockchain, but transactions are only processed in their assigned clusters.	Moderate. Increased performance due to trust relationships between nodes in the same cluster.	Moderate: Trust relationships between nodes reduce the cost of verification, but limit security.	Limited. Intershard transactions are processed in stages.	Proof of Authority (PoA) for internal operations in clusters	Not a full-fledged sharding system. Depends on trust between nodes in the same cluster.
AHL	Full sharding based on Trusted Execution Environment (TEE). Nodes are distributed randomly	High. Random distribution of nodes improves scalability and performance.	Moderate: Probabilistic security, not absolute, which limits protection against attacks	Supported	Byzantine Fault Tolerance (BFT), with Trusted Execution Environment (TEE) for node randomization	Does not provide absolute security as it relies on probabilistic models.
Benzene	Dual chain (proposal chain and voting chain) to separate transactions and consensus	High. Separation of consensus and transactions allows for higher throughput.	High. Requires control of more than half of the shards for a successful attack.	Supported. Two-phase locking protocol validates cross-shard transactions.	Two-Phase Locking (2PL) for processing intershard transactions, SGX for block validation.	High latency for cross-shard transaction confirmations (over 300 seconds). Memory limitation for large states.
CycLedger	Reputation-based sharding using randomized leader election and a partial node system	Moderate. The reputation system motivates nodes to behave honestly, but makes it	High: The partial node and reputation system improves security by monitoring	Limited. Partial nodes can replace leaders in case of protocol violation.	VRF (Verifiable Random Function) for random leader selection and Byzantine Fault Tolerance	Vulnerability to attacks on leaders. Incomplete separation of the consensus process and the recording of transactions

 doi:10.20944/preprints202410.1078.v2

to monitor leaders.	difficult to manage leaders.	the actions of leaders.	(BFT) for consensus

Table 2.

Technology	Scalability	Security
Sharding	● Dividing the network into shards allows transactions to be processed in parallel. ● Throughput can reach 4220 tx/s (RapidChain)	● 1% attack vulnerability: an attacker can focus resources on one shard and compromise it. ● Protection against up to 50% of malicious nodes (RapidChain)
DAG	● Parallel transaction processing thanks to graph structure. ● High throughput: CoDAG - up to 1151 tx/s	● Risk of double spending due to probabilistic verification methods (e.g. in IOTA)
Segwit	● Increasing the block size to 4 MB allows more transactions to be processed in each block. ● Partially solves the scalability problem	● Solves the malleability problem (changes in transaction IDs). ● Led to hard forks like Bitcoin Cash

Table 1 (see below) describes the comparison of all protocols described in the article. We examined how different sharding algorithms (e.g. Elastico, OmniLedger, Pyramid) affect network performance in the context of the problems being solved, such as scalability and security. We showed how the algorithms help overcome the limitations of traditional blockchain architectures. For the orchestration nodes in our architecture, the most suitable consensus algorithm is PBFT, as it best meets the requirements. Given the small number of nodes participating in the consensus process, PBFT demonstrates high efficiency and robustness. Other algorithms such as Raft could also be suitable, but the need to elect a leader makes it less suitable. During the election, the leader node may be unavailable to provide services for more than 3 seconds, and with network latency, this time may increase, which is unacceptable for fog computing. In addition, Raft is not supported by all blockchain platforms, which reduces its versatility compared to PBFT, which is considered a classic solution for private blockchains.

The PoA algorithm also does not meet the requirements, since it lacks the ability to motivate the participation of ordinary nodes. Motivation is a critical element for creating a high-quality and scalable architecture. In addition, PoA has a low transaction confirmation rate, which makes it inapplicable in the context of fog computing. The PoAh algorithm, with its authentication mechanisms and reduced resource consumption, looks promising for use in hybrid blockchains.

As a result of the analysis of various consensus algorithms for private blockchains, PBFT and PoAh demonstrated the best results in ensuring scalability.

3.1.1. PBFT (Practical Byzantine Fault Tolerance):

- This algorithm has shown high efficiency in conditions of a limited number of nodes, since it allows the network to operate even with up to a third of faulty nodes. PBFT provides fault tolerance and allows maintaining data consistency, which is critical for scalable systems. Its algorithmic structure allows processing a large number of transactions in a short period of time, which contributes to high network throughput.

3.1.2. PoAh (Proof of Authentication):

- Although this algorithm is aimed at low-power devices in IoT networks, its concept of trust levels and authentication mechanism allow for a significant reduction in resource consumption.

 doi:10.20944/preprints202410.1078.v2

This makes PoAh potentially suitable for scalable systems where it is necessary to take into account the diversity of nodes with different computing power. Its ability to adapt to changing network conditions and minimize computational costs also contributes to improving overall performance and efficiency.

While Raft and PoA have their merits, their scalability limitations make them less suitable for tasks involving dynamic and scalable networks. Raft requires a leader election, which can slow down the process, while PoA does not provide enough incentive for ordinary nodes to participate, which reduces their activity and, therefore, the scalability of the system. Thus, PBFT and PoAh stand out as the most effective algorithms for ensuring scalability in the context of private blockchains, making them preferable for use in modern distributed systems.

As for the problems and challenges, they include the complexity of cross-sharding, the problem of inter-shard transactions, the need for data replication, and the impact of these factors on network performance and security.

The formulation and solution of the problems associated with node selection depend on the consensus algorithm used.

3.2. Problems and Challenges

3.2.1. PoW-Based Node Selection

We consider the problems associated with PoW-based node selection methods. The main problem is vulnerability to attacks such as "selfish mining" and "oppressive mining". In the case of selfish mining, an attacker hides blocks and publishes them when his private chain becomes longer than the public one, thereby reducing the quality of the chain of honest nodes and increasing his share of control over the network. Oppressive mining is a more complex form of selfish mining that creates more competition, which allows the attacker to obtain an unreasonably large share of blocks. Other threats include block hiding, "fork after hiding" attacks, and "eclipse attacks" that isolate and manipulate honest nodes. The impact of these attacks is formally described in the Bitcoin Backbone protocol, where the attacker's share of blocks increases as his computing power increases relative to the entire network. Using a reference committee to select nodes can also be vulnerable. Network latency gives attackers a mining advantage by delaying the transmission of PoW solutions from honest nodes. This advantage results in the attacker solving more problems than expected. Additionally, malicious leaders in the reference committee can censor honest nodes when selecting new participants, weakening the security of the system. A threshold voting strategy has been proposed to mitigate such attacks, but it introduces new complications.

Future research should focus on improving the fairness of node selection, analyzing the impact of various attacks, and developing methods that are better suited for sharded blockchains.

3.2.2. PoS-Based Node Selection

In the PoS-based node selection process, key vulnerabilities include the “nothing at stake” problem, where attackers mine on multiple forks of the chain, and “grinding” attacks, where an attacker influences the selection of future block producers. Long-range attacks, where an attacker forges the blockchain from the genesis block to trick new nodes, are also a significant threat. These attacks can be mitigated by checkpoint mechanisms. In addition, PoS systems without a base chain face challenges related to the computational load of cryptographic operations and network latency. Future research should focus on the impact of attacks, computational and communication costs, and improving PoS-based node election processes. No less important research issues include the following:

3.2.3. Consensus Within a Shard

The development of blockchain technologies requires constant improvements of classical finite state machine replication algorithms, especially in the context of sharding blockchains. However,

 doi:10.20944/preprints202410.1078.v2

there are still unsolved problems that hinder their further application. Let us consider the problems of instant and asynchronous (deferred) sharding blockchains and directions for future research.

Instant sharding blockchains

The main problems and directions of research include:

- Reducing the complexity of communication within a shard. Reaching consensus within a shard requires several rounds of voting, which increases the load on the network and reduces efficiency with a large number of nodes. An important task is to reduce the communication complexity while maintaining security.
- Detection and replacement of malicious committees. Despite security measures, committees can be compromised. Future research should focus on mechanisms for detecting malicious committees through honest nodes and restoring or replacing them.
- Efficient mechanism for changing the leader. The process of changing the leader in a committee should be simplified, with a decrease in the volume of communications. Fair and rational approaches to the election of a new leader and the distribution of load among participants are also needed.
- Integration with sharding protocols. Intra-shard consensuses must be improved to handle inter-shard transactions. This requires more precise protocol design to ensure correct handling of different types of data

Asynchronous sharding blockchains

Main issues:

- 1% attack. In PoW-based sharding blockchains, an attacker can concentrate power on one shard and take over it with fewer resources than for a classic 51% attack. The problem requires the development of new solutions that prevent attacks while maintaining efficient distribution of computing and network resources.
- Complex processing of inter-shard transactions. Due to weak consistency in asynchronous sharding blockchains, transactions are not confirmed instantly, which complicates their processing. More research is needed to simplify the transaction confirmation process in such systems.

3.3. Cross-Shard Transactions

Client-Centric 2PC

The main problems with client-centric approaches include:

- Malicious leader behavior. In the preparation phase, if the liveness certificate (proof of input correctness) is generated by a single leader, as in Omniledger, rather than by the BFT committee, the leader may provide false evidence or not respond at all. This may lead to a successful double-spend attack and compromise the integrity of the blockchain protocol.
- Blocked transaction input. If a client acts as a coordinator and then stops sending the required evidence to other shards, the transaction input may be blocked. This is especially critical for multi-inputs from different users, such as crowdfunding.
- Increased client load. The client must track the shard state and node IP addresses to communicate with the committee leader, which increases the storage and communication load. This is unacceptable for lightweight clients, such as those on mobile devices.

Shard-oriented 2PC

There are also problems with shard-oriented approaches:

- Multiple BFT calls. To confirm a single transaction, the input and output committees must run the BFT algorithm multiple times, which increases the load on the nodes. Solutions need to be developed to handle multiple transactions in a single cycle.
- Attacks. Attacks such as a replay attack are possible, where an attacker uses a previously generated certificate to forge a proof. To prevent such attacks, it is necessary to bind the transaction identifier to the certificate. A transaction flooding attack is also possible, where an adversary creates multiple transactions for a single shard, disrupting the system.

 doi:10.20944/preprints202410.1078.v2

- Malicious coordinator. If the coordinator in a shard-oriented approach turns out to be malicious, it can slow down or block the certificate collection process. Mechanisms for protecting against such actions are needed.

3.4. Conceptual Model

The growing interest in green investments and the spread of ESG standards have acted as a catalyst for the use of blockchain technologies in working with green investments in general and green bonds in particular.The implementation of green investments in blockchain networks using sharding requires a thoughtful approach that will take into account both technological and environmental aspects. First of all, it is necessary to create specialized shards that will be dedicated exclusively to the management and tracking of data related to green investments. These shards can contain information on projects financed through green bonds, including data on their environmental performance, as well as compliance with ESG (environmental, social and corporate governance) standards. Such a structure allows for the centralization and organization of data, which significantly facilitates the process of monitoring and managing investments.Particular attention should be paid to the verification mechanisms within these shards. To ensure maximum transparency and accountability of investments, each shard should include tools for verifying the compliance of projects with established environmental standards and taxonomies. For example, the use of the taxonomy developed by VEB.RF in Russia will help minimize the risks of "green laundering" and ensure the targeted use of funds for truly environmentally significant projects. The integration of such mechanisms into sharding will create a basis for trust on the part of both investors and regulators, thereby ensuring the sustainability of the system.Further development of the concept includes the adaptation of consensus algorithms to take into account the specific requirements of green investments. An important step may be the development of a multi-level consensus that will allow assessing not only the technical parameters of transactions, but also their environmental aspects. Such an approach may include integration with existing ESG reporting systems, which will help increase transparency and accountability in the decision-making process. At the same time, consensus mechanisms should be flexible enough to adapt to various standards and regulations in force in different jurisdictions, which will allow taking into account both global and local environmental requirements.Additionally, the possibility of creating inter-shard coordination should be considered, which will ensure the consistency and integrity of data between different shard structures. This can be especially important if there are different shard groups in a blockchain network working on different aspects of green investment, such as assessing the environmental performance of projects, financial management, and ESG compliance. Cross-shard coordination will allow for efficient synchronization of data between these groups, thus providing a more comprehensive and coordinated approach to managing green investment.Thus, implementing green investment in a sharded blockchain requires a combination of technological innovations with carefully thought-out governance and verification processes that will ensure not only high network performance and security, but also trust from investors and regulators, thereby promoting sustainable development and supporting environmentally significant projects.

A blockchain model based on three principles of sharding - transaction sharding, state sharding and network sharding - allows for even greater scalability, performance and efficiency. Each type of sharding solves specific problems and interacts with other levels, creating a balanced and sustainable system. We propose the following conceptual model:

Main concepts

- **Transaction Sharding:** Dividing and distributing transactions across shard subgroups for parallel processing.
- **State Sharding:** Dividing the global blockchain state across shard subgroups to reduce the load on each individual node.
- **Network Sharding:** Dynamically distributing the node network into groups (shards) to reduce communication overhead and improve load balancing.

 doi:10.20944/preprints202410.1078.v2

Transactions are distributed across shards based on certain criteria, such as sender, receiver, transaction type, or other attributes. Transactions are processed in parallel within each shard, allowing for significant increases in network throughput. Thus, for transactions affecting different shards, a cross-shard coordination mechanism such as atomic cross-shard transactions is used.

The global blockchain state (e.g. balances, assets, contracts) is divided into pieces and distributed across shards. Each shard is responsible for updating its local state as a result of transactions associated with that shard. A global consensus mechanism and inter-shard communication protocols are used to maintain consistency between different parts of the state.

All network nodes are divided into groups (shards) based on geographic location, network characteristics, or random distribution. Network sharding reduces the amount of communication between nodes, since nodes within a single shard communicate more often than with nodes from other shards. Nodes within each shard process transactions and update the state preferentially within their group, which reduces latency and increases processing speed.

Transactions associated with specific parts of the global state are routed to the appropriate shards, which reduces inter-shard dependency. Shard states are updated only in those nodes where this data is actually needed, which reduces the amount of data transferred and increases network speed.

Nodes that are in the same network shard have preferential access to related state data, which improves network performance and minimizes latency. Dynamic network sharding allows nodes and resources to be redistributed depending on the load on different parts of the global state.

Transactions are transferred to the appropriate network shards for processing, which reduces latency and improves throughput. By sharding the network and transactions, the volume of inter-shard communications is reduced, which reduces the load on the network and improves overall performance.

Consensus:

- **Multi-level consensus:** Consensus occurs at several levels - within the transaction shard, within the state shard, and within the network shard. This ensures a high degree of decentralization and security.
- **Inter-shard consensus:** For operations that require interaction between shards, a special mechanism is used to ensure the atomicity and integrity of data.

Network sharding reduces the number of nodes that can be attacked simultaneously, which reduces the likelihood of a successful attack. Attacks that require coordination between multiple shards become much more difficult and require more resources.

The combination of three sharding levels allows the blockchain to scale almost linearly with the addition of new resources (nodes, transactions). Optimization of data transfer and localization of processing within shards significantly increase network performance. Sharding allows for better distribution of computing and network resources, reducing the overhead of maintaining the system.

The blockchain model with sharding of transactions, states, and the network offers an advanced architecture that can cope with the requirements of high-load and decentralized systems. This model can be used to create blockchains focused on financial applications, IoT, smart contracts, and other areas where high performance and scalability are critical.

The proposed conceptual blockchain model, including sharding of transactions, states, and the network, provides a number of significant advantages for working with structured financial information in transactions:

1. Increased performance through transaction sharding: transaction sharding allows the financial management of transactions to be distributed across multiple independent shards, allowing for parallel execution of transactions. This increases the throughput of the system, allowing for a large number of transactions to be processed per unit of time. This model allows for faster execution of transactions such as payments, fund transfers, or information transfers, increasing system efficiency.

2. Efficient data management through sharding: sharding optimizes the storage and processing of financial information, restoring balance sheet account data, smart contracts, and other important financial objects across different shards. This reduces the amount of data that needs to be stored and

 doi:10.20944/preprints202410.1078.v2

processed in each node, reducing computing power requirements and accelerating data access. In financial terms, where high accuracy and efficiency are required, this type of data organization allows for faster and more efficient financial transactions.

3. Reduced network load through network sharding: sharding allows for the division of networks into groups that interact exclusively within their shards. This reduces the amount of network communications required to perform transactions and ensure data consistency, reducing overhead and latency in data transmission. In financial applications, this results in increased transaction speed and reliability, especially for large volumes of transactions that require coordination between nodes.

4. Scalability and adaptability of the system due to the combination of three levels of sharding: the combined use of sharding of transactions, states, and the network allows the system to scale in response to the growth of the number of users and transaction volume. In case of increased load, new shards can be added without degrading the performance of the entire network. This architecture is especially useful for systems that must maintain high speed and stability as the number of clients and transactions grows.

5. High security through decentralization and inter-shard coordination: due to sharding of transactions, states, and the network, the system becomes more resilient to attacks. An attack on one shard cannot completely compromise the entire network, which provides an additional layer of security for financial data. Inter-shard coordination and atomicity of inter-shard transactions ensure that operations affecting multiple shards are executed correctly and consistently. 6. Resource Optimization and Cost Reduction: Network and state sharding allows nodes and network resources to operate more efficiently by processing only the data and transactions that are directly related to their shard. This reduces the cost of computing resources and data storage, making the system more cost-effective. This means lower operational costs while maintaining high performance and reliability.

Thus, a model using sharding of transactions, states, and the network provides a number of advantages that make it especially effective for working with structured financial information. It provides high performance, scalability, security, and resource optimization, which are critical for creating efficient and resilient financial systems.

3.5. Testing of Conceptual Model

To test the proof-of-concept model with sharding at different levels using the Ethereum private PoS network [16], we deployed three Docker containers representing the key elements of the network: a beaconchain node, a validator, and an Ethereum master node. Since the Ethereum base network does not support sharding out of the box, our goal was to simulate its effects through parallel processes and load testing, simulating the behavior of a sharded system. Here is how we approached testing the three main types of sharding (network, transaction, and state sharding) using the proposed framework.

1. Simulating network sharding

We simulated network sharding, in which the network distributes transactions across different nodes to optimize throughput, by running processes in parallel on a private network. Using our load testing tool, we launched several simultaneous processes, each representing a separate shard with a limited number of transactions. These processes transmitted data across the beaconchain as if the network distributed transactions among different parts. By measuring TPS (transactions per second) and GPS (gas per second) for each test, we assessed how parallel execution reduces the load on each process and increases the overall network throughput, similar to the behavior of network sharding.

2. Simulating transactional sharding

Transactional sharding involves dividing transactions into subgroups that can be processed in parallel within the network. To emulate this process, we used load testing with several processes, each executing a certain number of transactions. We tested three types of tests: (1) waiting for confirmation of all transactions in a process, (2) waiting only for the last transaction, and (3) not waiting for confirmations. In this way, we verified how the model handles parallel transactions and

 doi:10.20944/preprints202410.1078.v2

how distributing transactions between processes allows for optimizations similar to transactional sharding. The results showed that the system maintains high TPS and GPS values even with increasing parallelism, confirming the effectiveness of this approach in increasing network throughput.

3. Modeling state sharding

We modeled state sharding, which ensures that the blockchain state data is divided into different parts of the network, by dividing the load between processes and tracking how each process uses resources (gas and time). The main goal was to evaluate how efficiently the model handles operations between processes, which can be thought of as "state shards". We also took into account the transaction confirmation costs in each test, analyzing how different waiting methods (for example, waiting only for the last transaction to confirm) affect the overall performance. This allowed us to determine how the model structure could reduce the load on the blockchain state if different segments of the state were distributed among "shards" for parallel processing.

Test results and data analysis

For each sharding level, we obtained TPS and GPS metrics, which allowed us to quantify the performance of the model under different levels of parallelism. These tests showed that the proposed model can efficiently distribute transactions and state across processes, which mimics the benefits of sharding and improves scalability.

4. Conclusions

The research methodology is of a practical applied nature in combination with a theoretical justification of the experiments, developments, proof of the consistency and effectiveness of the results obtained. We created and tested a private Ethereum network for exchanging messages between nodes. A model for studying the scalability of the network with the possibility of load testing was built. The results obtained were patented, their detailed description and demonstration of the work are the subject of a new series of articles, which are currently being actively worked on. The effectiveness of the methods developed and applied by us for solving the main problems in blockchain networks allows us to confidently state the possibility of building such algorithms and mathematical models in the future that will be able to resolve the issues of blockchain limitations. In this work, we relied on the results of our own calculations and testing of our own program codes, as well as on articles by authoritative scientists in the field of computer science.

The article examines the scalability and security issues in blockchain technologies, highlighting the importance of sharding as a solution to improve network performance and throughput. The focus is on the CAP theorem, which indicates that it is impossible to simultaneously provide consistency, availability, and fragmentation resilience in distributed systems. The paper discusses various approaches to sharding, including transaction, state, and network sharding, and highlights their advantages and disadvantages. Examples of protocols such as Segwit demonstrate improved scalability through transaction and block size optimization. Alternative technologies such as DAG, which offer high speed and throughput but face security issues, are also discussed. The conclusion highlights the need for a comprehensive approach to solving scalability and security issues, including the use of innovative protocols and architectures to achieve sustainable development of blockchain technologies. A conceptual model of blockchain is also described, taking into account the full sharding of structured financial information in transactions.

Acknowledgments: The paper was published with the financial support of the Ministry of Education and Science of the Russian Federation as part of the program of the Moscow Center of Fundamental and Applied Mathematics under the agreement №075-15-2022-284.

 doi:10.20944/preprints202410.1078.v2

References

1. Qiheng Zhou, Huawei Huang, Zibin Zheng, Jing Bian, "Solutions to Scalability of Blockchain: A Survey». IEEE Access, 2020, Volume: 8, 16440 – 16455.
2. Yi Li, Jinsong Wang, Hongwei Zhang, "A survey of state-of-the-art sharding blockchains: Models, components and attack surfaces". Journal of Network and Computer Applications, 2023, Volume: 217, 103686.
3. Abdurrashid Ibrahim Sanka, Ray C.C. Cheung, "A systematic review of blockchain scalability: Issues, solutions, analysis and future research". ". Journal of Network and Computer Applications, 2021, Volume: 195, 103232.
4. Zibin Zheng, Hong-Ning Dai, Shaoan Xie, Xiangping Chen, "Blockchain challenges and opportunities: a survey". International Journal of Web and Grid Services, 2018, Volume: 14(4), 352.
5. Yizhong Liu, Andi Liu, Yuan Lu, Zhuocheng Pan, Yinuo Li, Jianwei Liu, Song Bian, Mauro Conti, Kronos: A Secure and Generic Sharding Blockchain Consensus with Optimized Overheard. Network and Distributed System Security(NDSS), 2024.
6. Muhammad Hassan Nasir, Junaid Arshad, Muhammad Mubashir Khan, Mahawish Fatima, Khaled Salah, Raja Jayaraman, "Scalable blockchains — A systematic review". Future Generation Computer Systems, 2022, Volume: 126, 136-162.
7. Nasrin Sohrabi, Zahir Tari, "On The Scalability of Blockchain Systems". IEEE International Conference on Cloud Engineering(IC2E), 2020, 124-133.
8. M. Kuzlu, M. Pipattanasomporn, L. Gurses and S. Rahman , "Performance Analysis of a Hyperledger Fabric Blockchain Framework: Throughput, Latency and Scalability". IEEE International Conference on Blockchain (Blockchain), Atlanta, GA, USA, 14-17 July 2019.
9. Soren Henning, Wilhelm Hasselbring, "A configurable method for benchmarking scalability of cloud-native applications, Scalable blockchains — A systematic review". Empirical Software Engineering, 2022, Volume: 27(6), [143].
10. Huawei Huang, Xiaowen Peng, Jianzhou Zhan, Shenyang Zhang, Yue Lin, Zibin Zheng, BrokerChain: A Cross-Shard Blockchain Protocol for Account/Balance-based State Sharding. IEEE INFOCOM 2022 - IEEE Conference on Computer Communications, London, United Kingdom, 02-05 May 2022.
11. Yumanova N. N., Bolgov M., A. Development of the Green Bond Market in Russia. Russian Economic Bulletin, 2021, Volume: 1 (4), 211-228.
12. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. August 21, 2008, 9 pages – Access mode: URL: https://archive.org/details/BitcoinAPeer-to-PeerElectronicCashSystem
13. Castro, M., & Liskov. Practical Byzantine Fault Tolerance. Appears in the Proceedings of the Third Symposium on Operating Systems Design and Implementation, New Orleans, USA, February 1999, 14 pages.
14. Buterin, V. Ethereum White Paper, 2014 – Access mode: URL: https://ethereum.org/en/whitepaper/
15. King, S., & Nadal, S. (2012). PPCoin: A peer-to-peer proof-of-stake cryptocurrency. – Access mode: URL: https://archive.org/details/PPCoinPaper

 doi:10.20944/preprints202410.1078.v2

16. pos-ethereum-network-bench (2024) - Access mode: URL: https://github.com/demidov-ad/pos-ethereum-network-bench
17. Rosenfeld, M. (2014). An analysis of reward systems for sharing Bitcoin mining. – Access mode: URL: https://www.semanticscholar.org/paper/Analysis-of-Bitcoin-Pooled-Mining-Reward-Systems-Rosenfeld/19e5af9721409f13496bb4f1635f98a18c7d7e68