A Comprehensive Outlook for Analyzing and Enhancing the Performance of Blockchain Platforms

Jeeta Ann Chacko
Technical University of Munich
chacko@in.tum.de

Ruben Mayer
University of Bayreuth
ruben.mayer@uni-bayreuth.de

Hans-Arno Jacobsen
University of Toronto
jacobsen@eecg.toronto.edu

ABSTRACT

As blockchain technology continues to evolve and expand across various industries, ensuring optimal performance and scalability of decentralized systems has become a critical concern. In this paper, we provide a comprehensive outlook for analyzing and optimizing blockchain performance across multiple layers of the blockchain stack, including system, data, and application levels. We address the performance challenges inherent in blockchain platforms, such as network architecture complexities, consensus mechanism inefficiencies, and data management bottlenecks. Our study underscores the importance of a holistic approach to performance optimization and also examines real-time performance tuning and adaptive optimization strategies. We offer practical insights into enhancing blockchain scalability, efficiency, and responsiveness. The discussions in this paper aim to equip developers, researchers, and practitioners with the tools needed to optimize blockchain performance effectively, ensuring these platforms can meet the evolving demands of diverse applications.

VLDB Workshop Reference Format:
Jeeta Ann Chacko, Ruben Mayer, and Hans-Arno Jacobsen. A Comprehensive Outlook for Analyzing and Enhancing the Performance of Blockchain Platforms. VLDB 2024 Workshop: Foundations and Applications of Blockchain.

1 INTRODUCTION

The performance of blockchain platforms has become a critical area of focus as these technologies evolve and are adopted across various industries. Blockchains, by their decentralized nature, require the careful coordination of multiple components, such as nodes, validators, consensus mechanisms, smart contracts, ledgers, and databases, to ensure secure and reliable transaction processing [24]. However, as transaction volumes increase and the complexity of smart contracts grows, performance bottlenecks can emerge at various layers of the blockchain stack, impacting throughput, latency, and resource utilization [23]. Addressing these performance challenges is essential to improve scalability and meet the high demands of modern applications.

To simplify the discussion and better understand the performance implications, we consider the complex blockchain stack to be made up of three abstract layers: the system layer, the data layer, and the application layer. At the system level, performance is influenced by the underlying network architecture, consensus mechanisms, and the configuration of nodes that validate and propagate transactions. This layer is crucial for ensuring the integrity and availability of the blockchain network but can suffer from issues like network latency and consensus delays [36, 103]. At the data level, the focus shifts to smart contracts and how information is stored, retrieved, and managed across the blockchain. Efficient data handling is vital to prevent bottlenecks caused by data redundancy, storage limitations, and slow data access speeds, which can degrade overall system performance [20, 21, 97]. Finally, the application layer deals with how users interact with the blockchain through decentralized applications. Performance at this layer is impacted by the underlying business process model, transaction handling and user interface responsiveness [109].

Given the complexity and interdependence of these layers, there is a pressing need for a comprehensive approach to performance analysis and optimization for blockchain platforms. Traditional performance assessment methods often focus on isolated layers, failing to provide a holistic view of the blockchain's performance landscape [23, 52, 62, 63, 74, 83, 93–95, 107]. This lack of integration makes it challenging to identify and address root causes of inefficiencies that span multiple layers. To meet this need, we present a comprehensive outlook on performance analysis and optimization strategies for blockchain platforms. We explore various techniques and methodologies for enhancing blockchain performance at the system, data, and application levels. By providing a detailed examination of each layer, the paper aims to offer a nuanced understanding of where performance bottlenecks typically occur and how they can be mitigated. For instance, at the system level, optimizing network configurations and consensus algorithms can significantly reduce latency and improve throughput. At the data level, employing advanced data management techniques like pruning and compression can help manage storage costs and improve data retrieval speeds. Meanwhile, at the application level, refining transaction processing and optimizing the business process model can enhance user experience and application responsiveness.

Moreover, we also discuss real-time performance optimization approaches that dynamically adjust blockchain configurations based on ongoing performance data. This data-driven approach enables blockchain systems to adapt to changing workloads and conditions, optimizing resource utilization and maintaining performance standards even under high transaction volumes or network congestion. Techniques like dynamic parameter tuning, real-time data indexing, and adaptive smart contract execution are discussed for keeping blockchain platforms agile and responsive. Further, we also present a case study of Hyperledger Fabric, where we discuss our previous research and other related work that quantifies the benefits of a comprehensive outlook on performance optimization.

Proceedings of the VLDB Endowment. ISSN 2150-8097.

In summary, we offer a holistic perspective on the challenges and strategies for optimizing blockchain performance across multiple layers. We emphasize the need for a comprehensive approach that integrates performance analysis and optimization across the system, data, and application layers to achieve better scalability, efficiency, and reliability. The insights and methodologies presented here are aimed at helping blockchain developers, researchers, and practitioners better understand the intricacies of blockchain performance and how to effectively enhance it to meet the growing demands of various applications.

2 BACKGROUND

The rise of blockchain technology has introduced a variety of platforms, each tailored to meet specific industry needs while also addressing the challenges of decentralization, scalability, and security. As these platforms mature, understanding their underlying architectures, components, and transaction flow becomes essential for optimizing performance across different use cases. This section provides a background on seven prominent blockchain platforms — Hyperledger Fabric, Quorum, Multichain, Corda, Solana, Algorand, and Avalanche — each offering unique features and facing distinct performance challenges.

2.1 Hyperledger Fabric

Hyperledger Fabric [45] is a permissioned blockchain framework designed for enterprise use, offering a modular architecture that provides high flexibility and scalability. Its architecture consists of key components such as peers, orderers, endorsers, and clients. Peers are the nodes that maintain the ledger and execute smart contracts (called chaincode in Fabric), while orderers are responsible for ordering transactions into a block and delivering them to peers in the correct order. Endorsers are specific peers that simulate and endorse transactions before they are added to a block. The transaction flow in Hyperledger Fabric starts with a client submitting a transaction proposal to endorsing peers, which simulate the transaction without updating the ledger and provide an endorsement signature. Once the required endorsements are collected, the transaction is sent to the ordering service, which batches transactions into blocks and delivers them to committing peers for final validation and ledger update.

2.2 Quorum

Quorum [89] is an enterprise-focused version of Ethereum designed to provide a permissioned environment for financial institutions. It features a modified Ethereum architecture to enhance privacy, performance, and throughput. The key components of Quorum include the Quorum Node (a modified Geth client), the Tessera/Constellation privacy manager for handling private transactions, and Raft or Istanbul BFT consensus mechanisms. The transaction flow in Quorum begins when a transaction is submitted by a client to a Quorum node. If the transaction is private, it is encrypted and sent to the Tessera nodes for secure handling. The transaction is then broadcast to the network, and based on the consensus mechanism (Raft or IBFT), the nodes agree on the transaction order. Once consensus is reached, the transaction is executed and the state is updated across the network, ensuring both privacy and transparency as needed.

2.3 Multichain

Multichain [81] is a permissioned blockchain platform that offers a simple and flexible architecture designed for rapid deployment of private blockchains. Its architecture is built around a peer-to-peer network of nodes, each of which maintains its own copy of the blockchain. Key components include nodes, streams (for data management), and permissions that control who can participate and perform specific actions in the network. The transaction flow in Multichain starts when a node creates a transaction and signs it with its private key. The transaction is then propagated to other nodes in the network for validation. If the transaction complies with the network's rules and permissions, it is added to the mempool. The nodes then work to solve a proof-of-work or follow a round-robin approach to produce the next block, which is then added to the blockchain and broadcasted to all nodes.

2.4 Corda

Corda [32] is a distributed ledger platform designed primarily for business and financial applications that require privacy and scalability. Unlike traditional blockchains, Corda does not use a global broadcast mechanism for transactions; instead, transactions are shared only with parties involved, enhancing privacy. Its architecture consists of nodes, notaries, and a unique contract state model. The key components include the Corda node, the notary service for preventing double-spending, and flows (automated workflows for transaction management). The transaction flow in Corda starts when two or more parties negotiate and agree on a transaction. The transaction is then digitally signed and sent to a notary service for validation to ensure no double-spending occurs. Once validated, the transaction is recorded in the respective parties' ledgers without broadcasting to the entire network, maintaining privacy and efficiency.

2.5 Solana

Solana [100, 116] is a high-performance blockchain platform designed for decentralized applications and crypto-assets, emphasizing scalability and low transaction costs. Its architecture is built around a single-layer blockchain leveraging a unique consensus algorithm called Proof of History (PoH), which provides a historical record to prove that events have occurred in a specific sequence. Key components include validators, which verify transactions and produce blocks, and a series of cryptographic timestamps that enable fast ordering of transactions. The transaction flow in Solana begins when a client submits a transaction to a validator, which adds a timestamp and propagates it to other validators. Validators verify transactions against the current state, and once consensus is achieved, transactions are bundled into blocks and added to the blockchain, all while maintaining high throughput and minimal latency.

2.6 Algorand

Algorand [2] is a pure proof-of-stake (PPoS) blockchain platform focused on achieving high scalability, security, and decentralization without compromising performance. Its architecture includes key components such as nodes (participation nodes and relay nodes), consensus protocol participants, and the Algorand Virtual Machine (AVM) for smart contract execution. The transaction flow in Algorand starts when a user submits a transaction to a participation node, which propagates it to relay nodes for broader network distribution. The consensus mechanism uses a verifiable random function (VRF) to randomly select a small committee of participation nodes to propose and vote on the next block. After reaching consensus, the block is added to the blockchain, and all participating nodes update their state accordingly, ensuring quick finality and high throughput.

2.7 Avalanche

Avalanche [13] is a highly scalable and flexible blockchain platform designed to support custom blockchain networks and decentralized applications. Its architecture consists of multiple chains, each optimized for different use cases: the X-Chain (exchange chain), C-Chain (contract chain), and P-Chain (platform chain). Key components include validators that participate in the consensus process, subnets (independent networks within Avalanche), and the Snowball consensus protocol that enables quick and probabilistic finality. The transaction flow in Avalanche starts with a transaction submitted to a validator on a specific chain. The validator propagates the transaction to other validators in the subnet, and through repeated sub-sampled voting rounds, consensus is reached. Once consensus is achieved, the transaction is confirmed and added to the respective blockchain, allowing for high transaction throughput and scalability.

3 PERFORMANCE ISSUES

Blockchain platforms face a variety of performance challenges that stem from their underlying architectures, consensus mechanisms, and data management strategies. This section delves into the specific performance issues encountered at the system, data, and application levels in different blockchain platforms. By identifying these bottlenecks, we can better understand the root causes of inefficiencies and develop targeted optimization strategies.

3.1 System Level

Performance issues at the system and network levels are critical challenges faced by different blockchain platforms, impacting their scalability, efficiency, and overall effectiveness. These issues stem from various factors, such as network latency, block propagation delays, resource allocation, and the communicational and computational demands of consensus mechanisms. Each blockchain platform has unique architectural features and operational requirements that contribute to its specific performance challenges.

In Hyperledger Fabric, one of the key performance issues is the resource-intensive nature of its architecture, particularly as transaction volumes increase. As more transactions are processed, the demands on system resources such as CPU, memory, and storage grow, leading to potential bottlenecks. These bottlenecks can affect the efficiency of validating peers and orderers, which are crucial for maintaining consensus and transaction ordering [46, 61]. Another significant issue is the network latency that arises from the communication overhead between different components (e.g., endorsers, orderers, and committers), which can slow down the transaction processing time and overall system throughput [54, 107].

Corda also faces several performance challenges at the system and network levels. One primary issue is the bottleneck associated with transaction notarization, which is a critical step in ensuring the validity and uniqueness of transactions. As transaction volumes increase, a single notary or a limited set of notaries can become a point of congestion, slowing down transaction finality [33, 64]. Additionally, the complexity of Corda's flows—automated processes that manage communication and coordination between nodes—can lead to increased network communication round trips, which further exacerbates latency issues [78]. Another significant challenge in Corda is the length of the state history; as all state changes need to be replicated to counterparties for verification, maintaining a lengthy state history can impact transaction times and system performance. This replication requirement adds overhead, especially in high-volume environments where numerous state changes occur frequently [36].

Platforms like Algorand, Solana, and Avalanche face performance challenges due to their distinct consensus mechanisms and architectural designs. In Algorand, block propagation delay and network latency are significant issues. The speed at which blocks are disseminated across the network and synchronized among nodes can impact transaction throughput, causing delays in confirming transactions [44, 65]. Solana, on the other hand, is prone to centralization risks due to the high computing resources required to become a validator, which can limit network participation and increase the chances of network outages and downtime [86, 103]. Additionally, Solana faces throughput challenges due to the high rate of transaction submissions relative to its capacity to process them, leading to network congestion and failed transactions [18, 104]. Avalanche also experiences throughput issues, which can be attributed to the time interval between consecutive blocks [53]. This block period limits the speed at which new transactions are added to the ledger, thereby constraining overall network throughput regardless of the available computational power or network bandwidth.

3.2 Data Level

Performance issues at the data level in blockchain platforms are primarily driven by limitations in data management, data storage, and smart contract execution, all of which can significantly impact the overall efficiency and scalability of the system. One major bottleneck is associated with the way smart contracts are processed and executed on blockchain networks. Smart contracts can introduce significant delays in transaction processing due to their inherent complexity and the need for extensive computation [91]. Each smart contract execution requires multiple steps of verification and state updates across the network, leading to increased latency. For example, smart contracts that perform multiple read and write operations can slow down the entire network, particularly when the contracts are not optimized for minimal data handling [97].

This inefficiency is exacerbated in public blockchains, where all nodes need to validate each transaction, further contributing to the slowdown.

Data storage presents another significant performance challenge across different blockchain platforms. Blockchains are facing a growing storage problem due to the ever-increasing amount of data generated by transactions. Every transaction must be stored permanently across all nodes, leading to exponential growth in data storage requirements. The cumulative size of the blockchain can become a bottleneck as it increases the time needed for nodes to sync and validate new transactions [20, 73]. Additionally, data storage inefficiencies can lead to increased operational costs for running nodes, as they require more disk space and higher storage capabilities. This can significantly affect the blockchain's performance and scalability [59, 112].

The challenges in data management and data retrieval further compound the performance issues at the data level. Traditional methods of querying and retrieving blockchain data are not designed for efficiency, often resulting in slow and cumbersome operations. The lack of structured data formats and standardized APIs means that data extraction is typically resource-intensive and slow, which can hinder real-time analytics and decision-making processes. For examples, Fabric's reliance on state databases like CouchDB can lead to performance degradation due to inefficient querying and state management when handling large volumes of data [23]. In platforms that rely heavily on historical data analysis, such as financial services and supply chain management, the inefficiencies in data retrieval can become a significant bottleneck [73]. Moreover, the lack of data organization and efficient indexing mechanisms in blockchains results in slow query performance, which directly impacts applications that require rapid data access and processing. As the data stored on blockchains continues to grow, these issues will likely worsen, underscoring the need for more robust data management strategies to maintain performance [20, 21, 115].

3.3 Application Level

Performance issues at the application level for different blockchain platforms are primarily caused by bottlenecks in components that handle user interactions and transaction processing. One bottleneck is the API gateway, which serves as the initial entry point for incoming requests [19, 109]. The API gateway is responsible for various functions, such as routing, request validation, authentication, and authorization. These functions can introduce processing overhead, especially when handling a high volume of requests, leading to increased response times [39]. For instance, in a blockchain-based payment system, frequent transactions and queries from multiple users can overwhelm the API gateway, causing delays in processing and response. Additionally, scalability limitations of the API gateway can further exacerbate these issues during peak load times, resulting in a degraded user experience due to increased latency and slower transaction throughput [108].

Another performance issue arises from the web server and transaction queue components, which manage the flow of transactions to the blockchain nodes. The web server is responsible for retrieving information and generating responses. If the web server is not optimized or lacks sufficient computational resources like CPU, memory, or disk I/O, it can significantly slow down the handling of concurrent requests, leading to increased response times and reduced throughput. Moreover, the transaction queue can become congested when overloaded with a high volume of transactions [118]. Without efficient concurrency management, this congestion can lead to synchronization delays, where transactions are delayed in processing and dispatching to the blockchain node. Such bottlenecks are particularly problematic for applications requiring high-frequency transactions, such as decentralized exchanges or real-time bidding platforms, where delays can significantly impact performance and user satisfaction [108, 109].

In addition to these component-level bottlenecks, issues with business process models also affect the performance of blockchain applications. In enterprise scenarios, blockchain-based applications often need to interact with complex business processes that involve multiple steps and dependencies. For example, in supply chain management systems built on blockchain, each transaction may represent a step in a complex process, such as inventory updates, shipment tracking, and payment settlements [24, 43]. If these business process models are not optimized for performance, they can lead to increased latency and reduced throughput. Overly complex process models may require multiple blockchain interactions for each step, increasing the load on the network and slowing down the overall process. Moreover, synchronization and consensus requirements for each step add additional latency. Simplifying and optimizing these business process models can help reduce the number of necessary blockchain interactions and improve the overall performance of the application, ensuring smoother and more efficient user experiences [88].

4 PERFORMANCE OPTIMIZATION

Optimizing the performance of blockchain platforms requires a multifaceted approach that addresses the entire blockchain stack. In this section, we discuss various possible strategies that can be applied at each abstract layer to optimize system configurations, data management, and application-specific processes.

4.1 System Level

Different blockchain platforms have distinct architectures and components that influence their performance characteristics. Therefore, performance optimization at the system level requires a tailored approach that includes strategically configuring system components and fine-tuning parameters relevant to each platform's unique requirements to enhance overall efficiency and throughput [48]. This section explores the strategies and techniques for improving performance across diverse blockchain technologies, highlighting specific examples and best practices for each.

In Hyperledger Fabric, performance can be optimized by carefully balancing the distribution and count of system components, such as validating peers, endorsers, orderers, and clients. For instance, increasing the number of endorsing peers may improve endorsement throughput, but it requires careful load balancing across peers to prevent bottlenecks [24, 27, 58]. In this regard, choosing the right endorsement policy aids in preventing endorsing peer bottlenecks [23, 107]. Network latency can be reduced by strategically placing peers and orderers to minimize communication delays.

Additionally, optimizing Fabric's state database by choosing between LevelDB and CouchDB based on query requirements can significantly impact performance [23, 107]. It is also beneficial to tune various system configuration settings such as block size, batch timeout, and database parameters to balance throughput and latency based on the application's specific needs, ensuring optimal resource utilization and performance [23, 87]. Over 50 parameters have been identified in Fabric that have a significant impact on performance [71].

We can make similar observations regarding the effect of node distribution and parameter tuning on performance in other blockchain platforms. In Quorum, for example, full nodes with a privacy manager can handle process-intensive tasks, but the optimal performance might be achieved by deploying a mix of light nodes for less intensive tasks [51]. Furthermore, configuring boot nodes versus static nodes impacts the peer discovery strategy, which influences network latency and transaction speed [28, 29]. Adjusting the consensus protocol to Raft or Clique based on the desired fault tolerance also plays a critical role in optimizing system performance [30].

For Corda, efficient handling of transaction notarization is necessary, as notary nodes can become bottlenecks under heavy load [33]. By optimizing notary configurations and deploying multiple notary clusters, the overall throughput can be significantly enhanced [34]. Additionally, there are multiple configuration parameters that can be fine-tuned to reduce unnecessary delays and improve transaction processing efficiency. This includes parameters related to cache, flow retries, timeout, connection pools, batch size, Java Virtual Machine (JVM), and Artemis message broker [34–36]. For example, to optimize throughput, it is recommended to use asynchronous flows wherever possible, reducing the blocking time and allowing the system to handle multiple transactions concurrently [31].

Multichain's performance can be significantly improved by fine-tuning its mining parameters, such as enabling skip proof-of-work checks and adjusting mining diversity [79]. Multichain also introduces the concept of data streams, where nodes subscribe to specific streams to ensure faster and more efficient information retrieval [80]. Configuring these streams appropriately allows for quicker data access and enhanced performance across the network.

Algorand also offers opportunities for performance optimization through parameter tuning and system configuration adjustments. Algorand's recent upgrades demonstrate a five-time increase in performance achieved by increasing the block size and reducing average round times [4]. Algorand includes numerous other node configuration and consensus protocol settings, though there is a lack of documentation or experimental evidence on how to best tune these parameters [1, 3, 49]. Algorand has implemented (or plans to implement) various other system optimizations such as opportunistic compression, congestion management, and identity checks to prevent duplicate peer connections [6, 65]. These strategies, which can also be useful for other blockchain platforms, reduce round times and improve overall network efficiency, supporting higher transaction volumes with reduced latency.

Solana's performance can be improved with block optimization and the strategic configuration of validator nodes [72, 98]. By limiting ledger size and optimizing transaction processing at the RPC node level, Solana can achieve better performance [101, 102]. Avalanche's performance can be enhanced by tuning parameters like quorum size, which impacts security and liveness, and by optimizing node setup for Avalanche subnets [10, 12, 18]. This includes optimizing hardware resources, such as CPU and memory allocation, configuring nodes for optimal network bandwidth usage, and tuning the AvalancheGo settings for maximum throughput and low latency [8, 11, 14]. Proper node setup and subnet management can ensure that Avalanche handles high transaction volumes with efficient parallel processing capabilities [15].

4.2 Data Level

Performance optimization at the data level for different blockchain platforms focuses on improving efficiency through effective data management and smart contract optimization. In Hyperledger Fabric, several techniques can be applied to enhance data-level performance. Transaction dependency conflicts are a key reason for transaction failures in Fabric [23]. Therefore, one optimization approach is using delta writes, where transactions that perform increment or decrement operations are converted into write-only transactions to unique delta keys, reducing transaction dependency and failure rates [24]. This technique minimizes the need to read a key before every write, thus improving transaction throughput. Another method is smart contract partitioning, which involves splitting a smart contract into multiple contracts that access separate world states. This reduces transaction conflicts by ensuring that different contracts handle different parts of the data, akin to partitioning tables in a relational database. Additionally, altering the data model can also mitigate hotkey conflicts. For instance, in scenarios where a single key is accessed frequently, changing the primary key can distribute the data load more evenly across transactions, significantly improving performance.

Effective data storage is a key optimization strategy at the data level. For example, Algorand's planned implementation of the Algorand Vault addresses the challenge of local storage by using transaction expiration and sharding techniques, which minimize the need for nodes to check all blocks continuously [7, 70]. This approach reduces the storage overhead on local nodes, facilitating faster bootstrapping and synchronization of new nodes joining the network. Additionally, smart contract optimization in Algorand involves practices such as minimizing the complexity of smart contracts, optimizing logic to reduce computational requirements, and efficiently managing stateful contract interactions [5, 106]. For example, developers can improve the throughput of smart contracts by employing strategies like reducing the size of stateful data and avoiding expensive operations in the contract code [5, 17]. These optimizations can help Algorand maintain high transaction throughput and low latency, particularly in environments with high volumes of smart contract interactions.

In Solana, data-level performance optimization focuses on maximizing the efficiency of compute usage to enhance overall network performance. The Solana network allows developers to optimize their programs by carefully managing the compute budget assigned to each transaction [99]. This involves writing programs that are as lightweight as possible, minimizing the number of compute units consumed by each operation. Key techniques include avoiding unnecessary loops, using efficient data structures, and

reducing function calls, which can significantly cut down on compute time [37, 99]. By optimizing compute usage and ensuring that programs run within their allocated compute limits, Solana can maintain high throughput and low latency, enabling the network to efficiently handle a large volume of transactions.

Smart contract optimization techniques in Solidity, commonly used in Ethereum-based platforms, are also applicable across various blockchain networks to reduce gas consumption and enhance execution speed. Key strategies include minimizing storage operations, which are costly in terms of gas fees, and leveraging events instead of state variables to reduce state changes [22, 105, 113]. Developers can also use fixed-size data types and avoid dynamic array resizing to save gas. Moreover, careful contract design to prevent redundant calculations and consolidate repetitive logic into single operations can significantly reduce the computational load. By focusing on these aspects, blockchain platforms using Solidity for smart contracts can achieve lower transaction costs and faster execution times, contributing to more efficient and scalable decentralized applications.

Further, blockchain storage optimization techniques such as replication-based, redaction-based, and content-based optimizations can enhance data management efficiency [41, 57, 117]. Replication-based optimizations reduce data duplication across nodes, which is particularly useful for minimizing storage overhead in scenarios where multiple copies of the same data are not necessary. Redaction-based optimizations allow the modification or removal of data already committed to the ledger, which could help manage storage requirements and improve efficiency by discarding irrelevant or outdated information. Content-based optimizations involve compressing data before or after committing it to the ledger, which can be effective in reducing the size of the stored data, thereby saving storage space and improving access times.

4.3 Application Level

Performance optimization at the application level for blockchain platforms can be significantly enhanced using various strategies that focus on refining the process model and managing transactions effectively. One interesting approach is process mining, which involves analyzing data from event logs to understand actual transaction workflows and identify inefficiencies or deviations from expected behaviors. This insight allows organizations to refine their process models, remove bottlenecks, and reduce latency, thereby improving overall blockchain performance [88, 110].

Using process mining results, various strategies can be applied to optimize transaction execution [24]. Activity reordering involves analyzing the dependencies and data correlations of transactions to find reorderable pairs that can be executed in a different sequence to reduce conflicts and enhance parallel processing. Meanwhile, process model pruning removes unnecessary or redundant steps from the workflow, focusing only on essential transactions that contribute to the process's value. This reduces computational overhead and enhances application efficiency by eliminating superfluous activities. There are also techniques that enable the translation of Business Process Model and Notation (BPMN) models into Solidity smart contracts via minimized Petri nets. This method allows for the identification and removal of non-essential tasks and transitions, effectively pruning the process model to its most efficient form [47].

Transaction rate control and batching are techniques aimed at managing transaction submission to optimize network performance. Rate control involves monitoring transaction rates over time and adjusting the flow of transactions during peak loads to prevent congestion and reduce failures. Configurable thresholds can help maintain a balance between throughput and network stability. Batching groups similar transactions into a single batch to minimize processing overhead and improve throughput. For repetitive tasks, such as mass token transfers, batching can significantly reduce the total number of transactions, making the process more cost-effective and time-efficient [24, 108, 111].

To further enhance performance, transaction monitoring, early cancellation, and notification systems play a crucial role. Transaction monitoring to track transaction statuses in real-time can be done and prompt actions can be taken for pending transactions. Early cancellation and acceleration techniques can be applied for expediting or canceling transactions that are delayed or no longer necessary, reducing network congestion. Additionally, implementing notification systems ensures that users and applications receive real-time updates on transaction completions, preventing duplicate submissions and maintaining a smooth transaction flow. These strategies collectively improve the efficiency and reliability of blockchain applications [24, 56, 108].

5 REAL-TIME OPTIMIZATION

Blockchains often operate under dynamically varying workloads, network congestion, and node capabilities, all of which can significantly impact performance metrics such as throughput, latency, and energy consumption. Therefore, real-time optimization is essential for blockchains because it enables these systems to respond dynamically to the unpredictable and fluctuating conditions that characterize decentralized networks. By employing real-time optimization, blockchains can adaptively modify system configurations, data management settings, and application processes to maintain optimal performance.

5.1 System Level

At the system level, real-time optimizations can focus on adapting the configuration of network components and consensus protocols to manage workload efficiently and reduce latency. Dynamic architecture adaptations based on workload variations have been discussed in the literature with significant improvements to performance [114]. Further, dynamically adjusting the number of validators or modifying the consensus mechanism based on real-time transaction throughput can help balance security and performance. Adaptive leader selection [119] and validator size selection [69] strategies found in the literature could be incorporated into blockchains. Further, blockchains can handle variable loads more effectively, minimizing delays and improving overall network responsiveness by continuously tuning system parameters to match current network conditions [25, 71, 92].

5.2 Data Level

At the data level, real-time optimization strategies should focus on adapting data storage, retrieval, and processing capabilities to manage the growing volumes of blockchain data more efficiently. Extensive research on dynamic resource allocation, configuration parameter tuning, query optimization, partitioning, storage layout configuration, and indexes has been conducted by the database community, which can be redesigned to fit the blockchain architecture [26, 38, 55, 60, 66–68, 75–77, 84, 85, 120, 122]. In a blockchain scenario, this would mean using adaptive data management techniques such as pruning unnecessary data, compressing stored data, and dynamically allocating storage resources based on usage patterns and data access frequencies. By optimizing data handling strategies in real-time, blockchain platforms can reduce storage costs, improve data access speeds, and ensure that the system remains responsive even as the amount of data increases [42, 90, 121]. These techniques are particularly valuable in applications that rely on timely data processing and high transaction throughput, such as financial services.

5.3 Application Level

At the application level, real-time optimization can focus on adjusting transaction handling and application logic to improve the performance of decentralized applications. This includes dynamically tuning transaction batching, prioritizing critical transactions, and optimizing the execution paths of application workflows to minimize processing delays [50, 96]. By leveraging real-time performance metrics, blockchain platforms can detect bottlenecks in transaction processing or application execution and make necessary adjustments to maintain high throughput and low latency [24, 108, 111]. This dynamic approach to application-level optimization ensures that decentralized applications can operate smoothly and efficiently, providing a responsive user experience even under varying network loads and conditions.

6 CASE STUDY: HYPERLEDGER FABRIC

In this section, we present a study of Hyperledger Fabric based on our previous research and related work that quantifies the need and effectiveness of a comprehensive outlook to performance optimization in blockchains. Our research consisted of three pivotal phases. We initially focused on benchmarking Hyperledger Fabric to identify the key factors contributing to performance bottlenecks [23]. At the system level, we identified various factors that significantly impacted the performance, such as the distribution of components and tuning of configuration parameters. For example, we observed up to 60% improvement in success throughput when the block size parameter was tuned optimally. At the data level, we analyzed the effect of different database types on performance and further measured the latency of various smart contract function calls to understand the root causes for the inefficient database type. At the application level, we conducted extensive experiments with multiple realistic workloads to understand their impact on performance. The findings from our experiments allowed us to identify various optimization strategies that can improve performance across the system, data, and application layers of the Fabric blockchain architecture. Many related works also conducted benchmarking studies on various aspects of the Fabric blockchain [9, 16, 40, 87, 107].

In the next phase of our research, we shifted our attention to automating the generation of these optimization strategies tailored to specific workload conditions [24]. We analyzed the data in the blockchain ledger, which is representative of the workload, to derive suitable optimization strategies at each layer of the blockchain stack. We developed a tool called BlockOptR that performs data analysis on the ledger and automatically generates ideal optimization recommendations for the current workload. When these strategies were implemented manually, they led to a significant improvement in performance (an average of 40% improvement in latency). Further, various optimized extensions of Fabric that tackle performance bottlenecks at different layers of the blockchain stack can be found in the literature [23, 23, 52, 62, 82, 93, 95].

The final phase of our study aimed at automating the implementation of these optimization strategies and dynamically adjusting them in response to workload variations [25]. We explored the potential of creating a self-driving blockchain system that continually enhances performance. We set up three demonstrative autonomous systems, each focusing on a different blockchain layer. Initial results show promising improvements in performance with up to 11% improvement in success throughput. This is a significant first step towards implementing a fully autonomous system in the future. Many related works also explore the possibility of an autonomous Fabric or Fabric-like blockchain [71, 114]. Our three-phase research, along with the related works that focus on the performance of Hyperledger Fabric, highlights the importance and quantifies the benefit of holistic approaches to performance optimization in blockchains.

7 CONCLUSION

In conclusion, this paper presents a comprehensive outlook on the performance analysis and optimization of blockchain platforms, emphasizing the need for a holistic approach that spans across system, data, and application layers. We have identified various performance bottlenecks inherent in different blockchain architectures and discussed targeted optimization strategies to address these challenges effectively. Our study highlights the significance of real-time optimization techniques that dynamically adapt to varying workloads and network conditions, ensuring that blockchain systems remain scalable, efficient, and reliable under diverse operational scenarios. Furthermore, the case study on Hyperledger Fabric exemplifies the practical benefits of such an integrated approach, demonstrating significant improvements in throughput and latency when optimization strategies are applied across all layers. This reinforces the value of adopting a comprehensive framework for blockchain performance enhancement, which not only addresses immediate performance issues but also prepares these platforms to meet the evolving demands of future applications. The insights from this paper serve as a resource for future researchers, offering a foundational understanding of performance dynamics and optimization techniques in blockchain systems. By building on our discussions, future work can further explore new dimensions of optimization and contribute to the advancement of more robust and adaptable blockchain technologies.

REFERENCES

[1] Algorand consensus protocols 2024. https://github.com/onplanetnowhere/AlgorandConsensusProtocolMD?tab=readme-ov-file. [Online; accessed 20-August-2024].

[2] algorand documentation 2024. https://developer.algorand.org/docs/. [Online; accessed 20-August-2024].

[3] Algorand node configurations 2024. https://developer.algorand.org/docs/run-a-node/reference/config/. [Online; accessed 20-August-2024].

[4] Algorand performance boost 2024. https://developer.algorand.org/articles/algorand-boosts-performance-5x-in-latest-upgrade/. [Online; accessed 20-August-2024].

[5] Algorand smart contracts 2024. https://developer.algorand.org/docs/get-details/dapps/smart-contracts/guidelines/. [Online; accessed 20-August-2024].

[6] Algorand transaction speed 2024. https://developer.algorand.org/articles/reaching-new-transaction-speeds-on-algorand/. [Online; accessed 20-August-2024].

[7] Algorand Vault 2024. https://algorandtechnologies.com/news/algorands-vault-fast-bootstrapping-for-the-algorand-cryptocurrency. [Online; accessed 20-August-2024].

[8] Ignacio Amores-Sesar, Christian Cachin, and Philipp Schneider. 2024. An Analysis of Avalanche Consensus. In *Structural Information and Communication Complexity*, Yuval Emek (Ed.). Springer Nature Switzerland, Cham, 27–44.

[9] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolić, Sharon Weed Cocco, and Jason Yellick. 2018. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *Proceedings of the Thirteenth EuroSys Conference* (Porto, Portugal) *(EuroSys '18)*. ACM, New York, NY, USA, Article 30, 15 pages. https://doi.org/10.1145/3190508.3190538

[10] Avalanche best practices 2024. https://medium.com/@subnetsdeployy/optimizing-performance-best-practices-for-avalanche-subnets-node-setup-a0f7a5a46b17. [Online; accessed 20-August-2024].

[11] Avalanche chain configurations 2024. https://docs.avax.network/nodes/chain-configs/overview. [Online; accessed 20-August-2024].

[12] Avalanche Consensus 2024. https://docs.avax.network/learn/avalanche-consensus. [Online; accessed 20-August-2024].

[13] Avalanche documentation 2024. https://docs.avax.network/. [Online; accessed 20-August-2024].

[14] Avalanche node configurations 2024. https://docs.avax.network/nodes/configure/configs-flags. [Online; accessed 20-August-2024].

[15] Avalanche subnet configurations 2024. https://docs.avax.network/nodes/configure/subnet-configs. [Online; accessed 20-August-2024].

[16] A. Baliga, N. Solanki, S. Verekar, A. Pednekar, P. Kamat, and S. Chatterjee. 2018. Performance Characterization of Hyperledger Fabric. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. 65–74. https://doi.org/10.1109/CVCBT.2018.00013

[17] Massimo Bartoletti, Andrea Bracciali, Cristian Lepore, Alceste Scalas, and Roberto Zunino. 2021. A Formal Model of Algorand Smart Contracts. In *Financial Cryptography and Data Security*, Nikita Borisov and Claudia Diaz (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 93–114.

[18] Álvaro Bayona Bultó. 2023. A Comprehensive Evaluation of Ethereum, Solana, and Avalanche in Addressing the Blockchain Trilemma. (2023).

[19] Blockchain API 2024. https://blog.axway.com/learning-center/digital-security/risk-management/blockchain-api. [Online; accessed 20-August-2024].

[20] Blockchain data warehousing 2024. https://bitquery.io/blog/blockchain-data-warehousing. [Online; accessed 20-August-2024].

[21] Blockchain Storage problem 2024. https://www.halborn.com/blog/post/blockchains-storage-problem-is-growing-are-there-any-solutions. [Online; accessed 20-August-2024].

[22] Tamara Brandstätter, Stefan Schulte, Jürgen Cito, and Michael Borkowski. 2020. Characterizing Efficiency Optimizations in Solidity Smart Contracts. In *2020 IEEE International Conference on Blockchain (Blockchain)*. 281–290. https://doi.org/10.1109/Blockchain50366.2020.00042

[23] Jeeta Ann Chacko, Ruben Mayer, and Hans-Arno Jacobsen. 2021. Why Do My Blockchain Transactions Fail? A Study of Hyperledger Fabric. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD/PODS '21)*. Association for Computing Machinery, New York, NY, USA, 221–234. https://doi.org/10.1145/3448016.3452823

[24] Jeeta Ann Chacko, Ruben Mayer, and Hans-Arno Jacobsen. 2023. How To Optimize My Blockchain? A Multi-Level Recommendation Approach. *Proc. ACM Manag. Data* 1, 1, Article 24 (may 2023), 27 pages. https://doi.org/10.1145/3588704

[25] Jeeta Ann Chacko, Ruben Mayer, and Hans-Arno Jacobsen. 2024. Should my Blockchain Learn to Drive? A Study of Hyperledger Fabric. arXiv:2406.06318 [cs.DC] https://arxiv.org/abs/2406.06318

[26] Surajit Chaudhuri and Vivek Narasayya. 2007. Self-Tuning Database Systems: A Decade of Progress. In *Proceedings of the 33rd International Conference on Very Large Data Bases* (Vienna, Austria) *(VLDB '07)*. VLDB Endowment, 3–14.

[27] Grant Chung, Luc Desrosiers, Manav Gupta, Andrew Sutton, Kaushik Venkatadri, Ontak Wong, and Goran Zugic. 2019. Performance tuning and scaling enterprise blockchain applications. *arXiv preprint arXiv:1912.11456* (2019).

[28] Configure bootnodes 2020. https://consensys.net/docs/goquorum/en/latest/configure-and-manage/configure/bootnodes/. [Online; accessed 14-August-2023].

[29] Configure static nodes 2020. https://consensys.net/docs/goquorum/en/latest/configure-and-manage/configure/static-nodes/. [Online; accessed 14-August-2023].

[30] Consensus protocols 2020. https://docs.goquorum.consensys.net/concepts/consensus. [Online; accessed 14-August-2023].

[31] Corda asynchronous flow invocations 2024. https://javanexus.com/blog/inefficient-asynchronous-flow-invocations-corda. [Online; accessed 20-August-2024].

[32] Corda documentation 2024. https://docs.r3.com/. [Online; accessed 20-August-2024].

[33] Corda Notaries 2020. https://docs.r3.com/en/platform/corda/4.10/community/key-concepts-notaries.html. [Online; accessed 14-August-2023].

[34] Corda performance tuning quick wins 2024. https://hiranhari.medium.com/corda-performance-tuning-quick-wins-9fb4d38dc63c. [Online; accessed 20-August-2024].

[35] Corda Throughput 2024. https://medium.com/corda/throughput-a-corda-story-1bc2cb9b2b60. [Online; accessed 20-August-2024].

[36] Corda vs Hyperledger Fabric 2024. https://eleks.com/research/corda-vs-hyperledger-fabric/. [Online; accessed 20-August-2024].

[37] Siwei Cui, Gang Zhao, Yifei Gao, Tien Tavu, and Jeff Huang. 2022. VRust: Automated Vulnerability Detection for Solana Smart Contracts. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* (Los Angeles, CA, USA) *(CCS '22)*. Association for Computing Machinery, New York, NY, USA, 639–652. https://doi.org/10.1145/3548606.3560552

[38] Sudipto Das, Divyakant Agrawal, and Amr El Abbadi. 2013. ElasTraS: An Elastic, Scalable, and Self-Managing Transactional Database for the Cloud. 38, 1, Article 5 (apr 2013), 45 pages. https://doi.org/10.1145/2445583.2445588

[39] Mazin Debe, Khaled Salah, Raja Jayaraman, Ibrar Yaqoob, and Junaid Arshad. 2021. Trustworthy Blockchain Gateways for Resource-Constrained Clients and IoT Devices. *IEEE Access* 9 (2021), 132875–132887. https://doi.org/10.1109/ACCESS.2021.3115150

[40] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. 2017. BLOCKBENCH: A Framework for Analyzing Private Blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data* (Chicago, Illinois, USA) *(SIGMOD '17)*. ACM, New York, NY, USA, 1085–1100. https://doi.org/10.1145/3035918.3064033

[41] Ali Dorri, Salil S. Kanhere, and Raja Jurdak. 2019. MOF-BC: A memory optimized and flexible blockchain for large scale networks. *Future Gener. Comput. Syst.* 92, C (mar 2019), 357–373. https://doi.org/10.1016/j.future.2018.10.002

[42] Zhengyi Du, Xiongtao Pang, and Haifeng Qian. 2023. PartitionChain: A Scalable and Reliable Data Storage Strategy for Permissioned Blockchain. *IEEE Transactions on Knowledge and Data Engineering* 35, 4 (2023), 4124–4136. https://doi.org/10.1109/TKDE.2021.3136556

[43] Pankaj Dutta, Tsan-Ming Choi, Surabhi Somani, and Richa Butala. 2020. Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transportation Research Part E: Logistics and Transportation Review* 142 (2020), 102067. https://doi.org/10.1016/j.tre.2020.102067

[44] Molud Esmaili and Ken Christensen. 2024. Performance modeling of public permissionless blockchains: A survey. arXiv:2402.18049 [cs.CR] https://arxiv.org/abs/2402.18049

[45] Fabric documentation 2024. https://hyperledger-fabric.readthedocs.io/en/latest/. [Online; accessed 20-August-2024].

[46] Fabric performance considerations 2024. https://hyperledger-fabric.readthedocs.io/en/release-2.5/performance.html. [Online; accessed 20-August-2024].

[47] Luciano García-Bañuelos, Alexander Ponomarev, Marlon Dumas, and Ingo Weber. 2017. Optimized Execution of Business Processes on Blockchain. In *Business Process Management*, Josep Carmona, Gregor Engels, and Akhil Kumar (Eds.). Springer International Publishing, Cham, 130–146.

[48] Frank Christian Geyer, Hans-Arno Jacobsen, Ruben Mayer, and Peter Mandl. 2023. An End-to-End Performance Comparison of Seven Permissioned Blockchain Systems. In *Proceedings of the 24th International Middleware Conference* (Bologna, Italy) *(Middleware '23)*. Association for Computing Machinery, New York, NY, USA, 71–84. https://doi.org/10.1145/3590140.3629106

[49] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. 2017. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China) *(SOSP '17)*. Association for Computing Machinery, New York, NY, USA,

51–68. https://doi.org/10.1145/3132747.3132757
[50] Seep Goel, Abhishek Singh, Rachit Garg, Mudit Verma, and Praveen Jayachandran. 2018. Resource Fairness and Prioritization of Transactions in Permissioned Blockchain Systems (Industry Track). In *Proceedings of the 19th International Middleware Conference Industry* (Rennes, France) *(Middleware '18)*. Association for Computing Machinery, New York, NY, USA, 46–53. https://doi.org/10.1145/3284028.3284035
[51] GoQuorum qlight 2020. https://consensys.net/docs/goquorum/en/latest/concepts/qlight-node/. [Online; accessed 14-August-2023].
[52] Christian Gorenflo, Stephen Lee, Lukasz Golab, and Srinivasan Keshav. 2019. FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 455–463. https://doi.org/10.1109/BLOC.2019.8751452
[53] Vincent Gramoli, Rachid Guerraoui, Andrei Lebedev, Chris Natoli, and Gauthier Voron. 2023. Diablo: A Benchmark Suite for Blockchains. In *18th ACM European Conference on Computer Systems (EuroSys)*. to appear.
[54] Tobias Guggenberger, Johannes Sedlmeir, Gilbert Fridgen, and André Luckow. 2022. An in-depth investigation of the performance characteristics of Hyperledger Fabric. *Computers Industrial Engineering* 173 (2022), 108716. https://doi.org/10.1016/j.cie.2022.108716
[55] Michael Hammer and Arvola Chan. 1976. Index Selection in a Self-Adaptive Data Base Management System. In *Proceedings of the 1976 ACM SIGMOD International Conference on Management of Data* (Washington, D.C.) *(SIGMOD '76)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/509383.509385
[56] Timo Hegnauer. 2019. Design and development of a blockchain interoperability api. *Zürich, Switzerland, February* (2019).
[57] Jun Wook Heo, Gowri Sankar Ramachandran, Ali Dorri, and Raja Jurdak. 2024. Blockchain Data Storage Optimisations: A Comprehensive Survey. *ACM Comput. Surv.* 56, 7, Article 179 (apr 2024), 27 pages. https://doi.org/10.1145/3645104
[58] How Fabric networks are structured 2020. How Fabric networks are structured. https://hyperledger-fabric.readthedocs.io/en/latest/network/network.html. [Online; accessed 14-August-2023].
[59] Huawei Huang, Jianru Lin, Baichuan Zheng, Zibin Zheng, and Jing Bian. 2020. When Blockchain Meets Distributed File Systems: An Overview, Challenges, and Open Issues. *IEEE Access* 8 (2020), 50574–50586. https://doi.org/10.1109/ACCESS.2020.2979881
[60] Scott E. Hudson and Roger King. 1989. Cactis: A Self-Adaptive, Concurrent Implementation of an Object-Oriented Database Management System. 14, 3 (sep 1989), 291–321. https://doi.org/10.1145/68012.68013
[61] Hyperledger Fabric issues 2024. https://medium.com/novai-hyperledger-fabric-101/common-issues-and-handling-in-hyperledger-fabric-implementation-e9cc625e7974. [Online; accessed 20-August-2024].
[62] Zsolt István, Alessandro Sorniotti, and Marko Vukolić. 2018. Streamchain: Do blockchains need blocks?. In *Proceedings of the 2nd Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*. 1–6.
[63] Haris Javaid, Chengchen Hu, and Gordon Brebner. 2019. Optimizing Validation Phase of Hyperledger Fabric. In *2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. 269–275. https://doi.org/10.1109/MASCOTS.2019.00038
[64] Tommy Koens, Scott King, Matthijs van den Bos, Cees van Wijk, and Aleksei Koren. 2019. Solutions for the corda security and privacy trade-off: having your cake and eating it. *White Paper* (2019).
[65] Kadir Korkmaz, Joachim Bruneau-Queyreix, Sonia Ben Mokhtar, and Laurent Réveillère. 2022. ALDER: Unlocking blockchain performance by multiplexing consensus protocols. In *2022 IEEE 21st International Symposium on Network Computing and Applications (NCA)*, Vol. 21. 9–18. https://doi.org/10.1109/NCA57778.2022.10013556
[66] Jan Kossmann. 2018. Self-Driving: From General Purpose to Specialized DBMSs.. In *PhD@ VLDB*.
[67] Jan Kossmann and Rainer Schlosser. 2020. Self-driving database systems: a conceptual approach. *Distributed and Parallel Databases* 38 (2020), 795–817.
[68] Sushil Kumar. 2003. Oracle database 10g: The self-managing database. *White Paper* (2003).
[69] Guilain Leduc, Sylvain Kubler, and Jean-Philippe Georges. 2022. Sabine: Self-Adaptive BlockchaIn coNsEnsus. In *2022 9th International Conference on Future Internet of Things and Cloud (FiCloud)*. 234–240. https://doi.org/10.1109/FiCloud57274.2022.00039
[70] Derek Leung, Adam Suhl, Yossi Gilad, and Nickolai Zeldovich. 2018. Vault: Fast Bootstrapping for the Algorand Cryptocurrency. Cryptology ePrint Archive, Paper 2018/269. https://doi.org/10.14722/ndss.2019.23313
[71] Mingxuan Li, Yazhe Wang, Shuai Ma, Chao Liu, Dongdong Huo, Yu Wang, and Zhen Xu. 2023. Auto-Tuning with Reinforcement Learning for Permissioned Blockchain Systems. *Proc. VLDB Endow.* 16, 5 (jan 2023), 1000–1012. https://doi.org/10.14778/3579075.3579076
[72] Xiangyu Li, Xinyu Wang, Tingli Kong, Junhao Zheng, and Min Luo. 2022. From Bitcoin to Solana – Innovating Blockchain Towards Enterprise Applications. In *Blockchain – ICBC 2021*, Kisung Lee and Liang-Jie Zhang (Eds.). Springer International Publishing, Cham, 74–100.
[73] Wei Liang, Yongkai Fan, Kuan-Ching Li, Dafang Zhang, and Jean-Luc Gaudiot. 2020. Secure Data Storage and Recovery in Industrial Blockchain Network Environments. *IEEE Transactions on Industrial Informatics* 16, 10 (2020), 6543–6552. https://doi.org/10.1109/TII.2020.2966069
[74] Mengting Liu, F. Richard Yu, Yinglei Teng, Victor C. M. Leung, and Mei Song. 2019. Performance Optimization for Blockchain-Enabled Industrial Internet of Things (IIoT) Systems: A Deep Reinforcement Learning Approach. *IEEE Transactions on Industrial Informatics* 15, 6 (2019), 3559–3570. https://doi.org/10.1109/TII.2019.2897805
[75] Lin Ma. 2021. *Self-Driving Database Management Systems: Forecasting, Modeling, and Planning*. Ph.D. Dissertation. Carnegie Mellon University.
[76] Lin Ma, Dana Van Aken, Ahmed Hefny, Gustavo Mezerhane, Andrew Pavlo, and Geoffrey J. Gordon. 2018. Query-Based Workload Forecasting for Self-Driving Database Management Systems. In *Proceedings of the 2018 International Conference on Management of Data* (Houston, TX, USA) *(SIGMOD '18)*. Association for Computing Machinery, New York, NY, USA, 631–645. https://doi.org/10.1145/3183713.3196908
[77] Lin Ma, William Zhang, Jie Jiao, Wuwen Wang, Matthew Butrovich, Wan Shen Lim, Prashanth Menon, and Andrew Pavlo. 2021. MB2: Decomposed Behavior Modeling for Self-Driving Database Management Systems. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD '21)*. Association for Computing Machinery, New York, NY, USA, 1248–1261. https://doi.org/10.1145/3448016.3457276
[78] Ahmed Afif Monrat, Olov Schelén, and Karl Andersson. 2020. Performance Evaluation of Permissioned Blockchain Platforms. In *2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*. 1–8. https://doi.org/10.1109/CSDE50874.2020.9411380
[79] Multichain Configurations 2020. https://www.multichain.com/developers/blockchain-parameters/. [Online; accessed 14-August-2023].
[80] Multichain Data Streams 2020. https://www.multichain.com/developers/data-streams/. [Online; accessed 14-August-2023].
[81] Multichain documentation 2024. https://www.multichain.com/developers/. [Online; accessed 20-August-2024].
[82] Pezhman Nasirifard, Ruben Mayer, and Hans-Arno Jacobsen. 2019. FabricCRDT: A Conflict-Free Replicated Datatypes Approach to Permissioned Blockchains. In *Proceedings of the 20th International Middleware Conference* (Davis, CA, USA) *(Middleware '19)*. Association for Computing Machinery, New York, NY, USA, 110–122. https://doi.org/10.1145/3361525.3361540
[83] Keerthi Nelaturu, Sidi Mohamed Beillahi, Fan Long, and Andreas Veneris. 2021. Smart Contracts Refinement for Gas Optimization. In *2021 3rd Conference on Blockchain Research Applications for Innovative Networks and Services (BRAINS)*. 229–236. https://doi.org/10.1109/BRAINS52497.2021.9569819
[84] Andrew Pavlo, Gustavo Angulo, Joy Arulraj, Haibin Lin, Jiexi Lin, Lin Ma, Prashanth Menon, Todd C Mowry, Matthew Perron, Ian Quah, et al. 2017. Self-Driving Database Management Systems.. In *CIDR*, Vol. 4. 1.
[85] Andrew Pavlo, Matthew Butrovich, Lin Ma, Prashanth Menon, Wan Shen Lim, Dana Van Aken, and William Zhang. 2021. Make Your Database System Dream of Electric Sheep: Towards Self-Driving Operation. *Proc. VLDB Endow.* 14, 12 (jul 2021), 3211–3221. https://doi.org/10.14778/3476311.3476411
[86] Giuseppe Antonio Pierro and Roberto Tonelli. 2022. Can Solana be the Solution to the Blockchain Scalability Problem?. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 1219–1226. https://doi.org/10.1109/SANER53432.2022.00144
[87] Suporn Pongnumkul, Chaiyaphum Siripanpornchana, and Suttipong Thajchayapong. 2017. Performance analysis of private blockchain platforms in varying workloads. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 1–6.
[88] Process Mining Blockchain 2024. https://research.aimultiple.com/process-mining-blockchain/. [Online; accessed 20-August-2024].
[89] Quorum documentation 2024. https://goquorum.readthedocs.io/. [Online; accessed 20-August-2024].
[90] Ravi Kiran Raman and Lav R. Varshney. 2018. Dynamic Distributed Storage for Blockchains. In *2018 IEEE International Symposium on Information Theory (ISIT)*. 2619–2623. https://doi.org/10.1109/ISIT.2018.8437335
[91] Sara Rouhani and Ralph Deters. 2019. Security, Performance, and Applications of Smart Contracts: A Systematic Survey. *IEEE Access* 7 (2019), 50759–50779. https://doi.org/10.1109/ACCESS.2019.2911031
[92] Na Ruan, Dongli Zhou, and Weijia Jia. 2020. Ursa: Robust Performance for Nakamoto Consensus with Self-Adaptive Throughput. *ACM Trans. Internet Technol.* 20, 4, Article 41 (nov 2020), 26 pages. https://doi.org/10.1145/3412341
[93] Pingcheng Ruan, Dumitrel Loghin, Quang-Trung Ta, Meihui Zhang, Gang Chen, and Beng Chin Ooi. 2020. A Transactional Perspective on Execute-Order-Validate Blockchains. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data* (Portland, OR, USA) *(SIGMOD '20)*. Association for Computing Machinery, New York, NY, USA, 543–557. https://doi.org/10.1145/3318464.3389693

[94] Gary Shapiro, Christopher Natoli, and Vincent Gramoli. 2020. The Performance of Byzantine Fault Tolerant Blockchains. In *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*. 1–8. https://doi.org/10.1109/NCA51143.2020.9306742

[95] Ankur Sharma, Felix Martin Schuhknecht, Divya Agrawal, and Jens Dittrich. 2019. Blurring the Lines Between Blockchains and Database Systems: The Case of Hyperledger Fabric. In *Proceedings of the 2019 International Conference on Management of Data* (Amsterdam, Netherlands) *(SIGMOD '19)*. ACM, New York, NY, USA, 105–122. https://doi.org/10.1145/3299869.3319883

[96] Jianfeng Shi, Heng Wu, Diaohan Luo, Heran Gao, and Wenbo Zhang. 2023. InstantChain: Enhancing Order-Execute Blockchain Systems for Latency-Sensitive Applications. In *Database Systems for Advanced Applications*, Xin Wang, Maria Luisa Sapino, Wook-Shin Han, Amr El Abbadi, Gill Dobbie, Zhiyong Feng, Yingxiao Shao, and Hongzhi Yin (Eds.). Springer Nature Switzerland, Cham, 483–498.

[97] Smart contracts slow blockchains 2024. https://www.multichain.com/blog/2015/11/smart-contracts-slow-blockchains/. [Online; accessed 20-August-2024].

[98] Solana best practices 2024. https://www.c-sharpcorner.com/article/optimization-techniques-and-best-practices-in-solana/. [Online; accessed 20-August-2024].

[99] Solana compute optimize 2024. https://solana.com/developers/guides/advanced/how-to-optimize-compute. [Online; accessed 20-August-2024].

[100] solana documentation 2024. https://solana.com/docs. [Online; accessed 20-August-2024].

[101] Solana RPC node optimization 2024. https://www.bydfi.com/en/questions/what-are-the-best-practices-for-optimizing-the-performance-of-an-rpc-node-on-solana. [Online; accessed 20-August-2024].

[102] Solana RPC node setup 2024. https://blockchain.oodles.io/dev-blog/how-to-setup-and-run-solana-rpc-node/?utm_source=medium. [Online; accessed 20-August-2024].

[103] Solana scalability challenges 2024. https://medium.com/vanguard-industry-foresight/solanas-failing-transaction-problem-addressing-the-network-s-scalability-challenges-dbe9a93814e2. [Online; accessed 20-August-2024].

[104] Solana throughput challenges 2024. https://medium.com/coinmonks/solanas-sol-chain-throughput-challenges-what-s-next-ce8e0b938b02. [Online; accessed 20-August-2024].

[105] Solidity optimization techniques 2024. https://101blockchains.com/top-solidity-gas-optimization-techniques/. [Online; accessed 20-August-2024].

[106] Zhiyuan Sun, Xiapu Luo, and Yinqian Zhang. 2023. Panda: Security Analysis of Algorand Smart Contracts. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, 1811–1828. https://www.usenix.org/conference/usenixsecurity23/presentation/sun

[107] P. Thakkar, S. Nathan, and B. Viswanathan. 2018. Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform. In *2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. 264–276. https://doi.org/10.1109/MASCOTS.2018.00034

[108] Transaction limits and optimizations 2024. https://developers.circle.com/w3s/docs/transaction-limits-and-optimizations. [Online; accessed 20-August-2024].

[109] Understanding blockchain application performance 2024. https://www.linkedin.com/pulse/understanding-blockchain-application-performance-guide-ramalingam/. [Online; accessed 20-August-2024].

[110] Wil van der Aalst. 2012. Process Mining: Overview and Opportunities. 3, 2, Article 7 (jul 2012), 17 pages. https://doi.org/10.1145/2229156.2229157

[111] Yibo Wang, Kai Li, Yuzhe Tang, Jiaqi Chen, Qi Zhang, Xiapu Luo, and Ting Chen. 2023. Towards Saving Blockchain Fees via Secure and Cost-Effective Batching of Smart-Contract Invocations. *IEEE Transactions on Software Engineering* 49, 4 (2023), 2980–2995. https://doi.org/10.1109/TSE.2023.3237123

[112] S. Wilson, K. Adu-Duodu, Y. Li, E. Solaiman, O. Rana, S. Dustdar, and R. Ranjan. 2024. Data Management Challenges in Blockchain-Based Applications. *IEEE Internet Computing* 28, 01 (jan 2024), 70–80. https://doi.org/10.1109/MIC.2023.3319152

[113] Maximilian Wohrer and Uwe Zdun. 2018. Smart contracts: security patterns in the ethereum ecosystem and solidity. In *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. 2–8. https://doi.org/10.1109/IWBOSE.2018.8327565

[114] Chenyuan Wu, Bhavana Mehta, Mohammad Javad Amiri, Ryan Marcus, and Boon Thau Loo. 2023. AdaChain: A Learned Adaptive Blockchain. *Proc. VLDB Endow.* 16, 8 (jun 2023), 2033–2046. https://doi.org/10.14778/3594512.3594531

[115] Hanqing Wu, Jiannong Cao, Yanni Yang, Cheung Leong Tung, Shan Jiang, Bin Tang, Yang Liu, Xiaoqing Wang, and Yuming Deng. 2019. Data Management in Supply Chain Using Blockchain: Challenges and a Case Study. In *2019 28th International Conference on Computer Communication and Networks (ICCCN)*. 1–8. https://doi.org/10.1109/ICCCN.2019.8846964

[116] Anatoly Yakovenko. 2018. Solana: A new architecture for a high performance blockchain v0. 8.13. *Whitepaper* (2018).

[117] Tao Ye, Min Luo, Yi Yang, Kim-Kwang Raymond Choo, and Debiao He. 2023. A Survey on Redactable Blockchain: Challenges and Opportunities. *IEEE Transactions on Network Science and Engineering* 10, 3 (2023), 1669–1683. https://doi.org/10.1109/TNSE.2022.3233448

[118] Muhammad Zaid, Muhammad Waheed Akram, Naveed Ahmed, and Shahzad Saleem. 2019. Web Server Integrity Protection Using Blockchain. In *2019 International Conference on Frontiers of Information Technology (FIT)*. 239–2395. https://doi.org/10.1109/FIT47737.2019.00052

[119] Gengrui Zhang, Fei Pan, Sofia Tijanic, and Hans-Arno Jacobsen. 2023. PrestigeBFT: Revolutionizing View Changes in BFT Consensus Algorithms with Reputation Mechanisms. *arXiv preprint arXiv:2307.08154* (2023).

[120] Ji Zhang, Yu Liu, Ke Zhou, Guoliang Li, Zhili Xiao, Bin Cheng, Jiashu Xing, Yangtao Wang, Tianheng Cheng, Li Liu, Minwei Ran, and Zekang Li. 2019. An End-to-End Automatic Cloud Database Tuning System Using Deep Reinforcement Learning. In *Proceedings of the 2019 International Conference on Management of Data* (Amsterdam, Netherlands) *(SIGMOD '19)*. Association for Computing Machinery, New York, NY, USA, 415–432. https://doi.org/10.1145/3299869.3300085

[121] Xiongfei Zhao and Yain-Whar Si. 2021. Dynamic Transaction Storage Strategies for a Sustainable Blockchain. In *2021 IEEE International Conference on Services Computing (SCC)*. 309–318. https://doi.org/10.1109/SCC53864.2021.00044

[122] Xuanhe Zhou, Lianyuan Jin, Ji Sun, Xinyang Zhao, Xiang Yu, Jianhua Feng, Shifu Li, Tianqing Wang, Kun Li, and Luyang Liu. 2021. DBMind: A Self-Driving Platform in OpenGauss. *Proc. VLDB Endow.* 14, 12 (jul 2021), 2743–2746. https://doi.org/10.14778/3476311.3476334