A Fast Distributed Supply Chain Infrastructure Based on Blockchain

Aiqun Gu, Peng Zhao, Shiren Ye*

School of Computer Science and Artificial Intelligence, Changzhou University, Jiangsu, Changzhou, 213000, China

1727034502@qq.com, zhaopeng@cczu.edu.cn, yes@cczu.edu.cn

*Corresponding author

Keywords: Blockchain; Supply Chain; Consortium Blockchain; Hyperledger Fabric; Smart Contract; Chaincode

Abstract: Traditional supply chain management systems often rely on centralized servers and databases, which have disadvantages such as lack of transparency in transaction information, difficulties in information sharing, high maintenance costs for central services, low database security, and poor risk resistance. We propose a supply chain transaction Infrastructure based on blockchain technologies. It features decentralization, distributed data storage, tamper-resistant and traceable data, and low maintenance costs. The architecture is built on a consortium blockchain, and it utilizes a dual-layer data storage structure consisting of a blockchain and a business database. This ensures both the security and convenience of transaction data queries while maintaining transaction speed. To evaluate the feasibility and performance of this architecture, we conducted a series of simulation experiments. The results demonstrate that the proposed supply chain architecture is efficient, transparent, secure, and facilitates information sharing. It also meets the performance requirements for real-world industrial-scale supply chain applications.

1. Introduction

The concept of the supply chain emerged in the 1980s. It refers to the interconnected network of suppliers, manufacturers, distributors, and end-users, revolving around a core company. It starts with sourcing components, goes through intermediate production stages, and ends with the delivery of the final product to consumers through a sales network. The supply chain has become a core competitive advantage for companies, and its data scale has been growing exponentially [1]. Additionally, supply chain management involves aspects such as contracts between companies, warehousing logistics, product traceability, and qualification certification. For example, in the case of an electric vehicle, it is possible to trace the battery back to its production batch and the materials used for the positive and negative electrodes. This requires the integration of data from all stages of the entire industry chain. Similarly, in the case of traceability in food supply chains, such as imported frozen foods, it involves nationwide traceability from source to end-user. The scale of data and management requirements present significant challenges to traditional supply chain systems.

In traditional supply chains, each company manages its own operations, and the management systems of each company are relatively closed. This leads to insufficient information sharing, low efficiency in collaborative efforts, difficulties in product traceability, and a lack of trust between companies. Moreover, the traditional supply chain system is severely limited by the performance of central servers. Maintaining a high-performance server is costly, and if the centralized server or database is breached, it poses a significant security threat to the entire supply chain.

Blockchain is a distributed database that integrates cryptography, distributed systems, and consensus mechanisms. It also incorporates features such as data sharing, decentralization, tamper resistance, and traceability [2]. Blockchain has been widely applied in industries such as food, agriculture, and healthcare. Due to its immutability, blockchain can structurally process information from various stages of the production, distribution, and sales process, ensuring the security of data in the supply chain. Its decentralized and distributed storage approach makes information in the

  DOI: 10.25236/icmmct.2023.031

supply chain more transparent and clear, breaking down trust barriers between supply chain participants and strengthening the connections between companies. On the blockchain, companies can build decentralized or multi-central supply chain networks, avoiding potential risks associated with excessive reliance on a single core enterprise.

This article presents a supply chain management architecture based on a consortium blockchain. This architecture records the consensus and transactions of participating enterprises on the chain while ensuring privacy protection. The architecture utilizes Hyperledger Fabric as the underlying framework and employs channel isolation to achieve the segregation of sensitive data. Complex business data is distributedly stored in a combination of "blockchain + business database," overcoming the limitations of low transaction speed and small data throughput commonly associated with traditional blockchain technologies. As a result, this architecture enables the realization of large-scale traceable transactions that meet the requirements of practical applications.

2. Related Work

2.1. Supply Chains

Competition among modern enterprises has shifted from individual product competition to supply chain competition. In the industrial sector, Walmart has built a low-cost and high-efficiency replenishment system using Universal Product Code (UPC) and Radio Frequency Data Communication (RFDC) technologies, significantly enhancing business performance [13]. JD.com has achieved electronic transformation of the entire supply chain, resulting in approximately 100 million yuan in annual savings during the warehousing and handover stage alone . Real-time dynamic monitoring and precise logistics management are the key focuses for supply chain development.

Supply chain systems integrate transactions, transportation, and warehousing, involving multiple steps and nodes. The generated data can be vulnerable to data tampering, unmonitored illegal transactions, and low-cost breach risks.

Applying blockchain technology to supply chain systems helps address issues such as decentralization, traceability, information security, and maintenance costs. For instance, Caro et al. combined blockchain technology with IoT to propose the AgriBlockIoT blockchain solution [14] for agricultural supply chain management, achieving information traceability throughout the production and sales process of agricultural products. Figorilli et al. [15] implemented a blockchain-based timber traceability system using RFID sensor technology and blockchain, leveraging the decentralized and distributed storage features of blockchain to securely store data information and transaction records, enabling electronic traceability of timber from the standing tree to the end-user [15]. Tian Feng integrated traceable supply chains with RFID tags and blockchain technology, categorizing RFID tag data generated in production, processing, transportation, and other processes into blockchain, enhancing supply chain traceability [16]. Henry et al. combined traceable ontology with blockchain technology on the Ethereum platform, storing information about goods, including time, location, environment, and characteristics, in a blockchain and recording ownership of goods in a list format based on chronological order [17]. Lee et al. combined blockchain with real-time event monitoring systems to achieve rapid tracking of various public safety events [18].

Typical public blockchain architectures suffer from low transaction efficiency and high transaction costs. For example, Bitcoin, supported by a massive global computing power, can handle only about six transactions per minute, which is insufficient to meet the requirements of supply chain data volume. This paper aims to study a blockchain architecture that can handle large data processing, enable data sharing, protect sensitive transaction data privacy, and achieve traceability in supply chains.

2.2. Blockchains

The blockchain model, as the underlying technology of Bitcoin, can be divided into several layers: the data layer, network layer, consensus layer, incentive layer, contract layer, and application

layer [3][4]. It essentially serves as a distributed autonomous database that stores various transaction information. Each transaction needs to be validated by over half of the nodes in the blockchain. Once verified, the transaction is recorded in the corresponding block and timestamped for permanent storage. Each block contains hash values, timestamps, and the data generated by each transaction. The blocks are linked together to form the blockchain. Smart contracts are scripts that run on the blockchain, encoding contract content and automating rule execution [5]. With the addition of Turing-complete smart contracts, Ethereum became the technological core beyond Bitcoin [6].

Bitcoin and Ethereum, which utilize the Proof of Work (PoW) consensus mechanism, suffer from issues such as slow transaction speeds, low data throughput, and high hardware costs [7], making it challenging to meet the needs of various business scenarios in supply chains. To address these challenges, Pedrosa et al. [8] designed the Lightning Network as a sidechain for blockchain. They used revocable sequence maturity contracts to resolve the one-way flow problem in channels and Hashed TimeLock Contracts to address the issue of cross-node coin transfers in channels. With the introduction of Byzantine Fault Tolerance (BFT) into blockchain, the hardware and energy costs required to achieve consensus have significantly decreased. BFT performs well in small-scale consortium chains, but as the number of nodes increases, the time required to achieve consensus with BFT also increases significantly. Based on this, Androulaki et al. [9] proposed a scalable Hyperledger Fabric consortium chain system that achieves end-to-end throughput of over 3500 transactions per second, with a latency of less than 1 second, and can scale well to over 100 peers. Back et al. [10] assigned importance weights to each node using graph theory principles and improved the performance of existing consensus algorithms such as PoW, PoS, and PBFT by measuring the weight values. Kiayias et al. [11] introduced the Ouroboros blockchain protocol, which utilizes a design based on game theory and Nash equilibrium to give PoS an advantage over PoW. Zheng et al. [12] described typical consensus algorithms under different blockchains and improved the basic asynchronous Byzantine Fault Tolerance through enhanced consensus protocols, increasing the availability of BFT consensus.

Usually，typical blockchains could be divided into three groups:

Public Chains: They are fully decentralized, and anyone can join the blockchain network and perform transaction activities. They can verify any transaction. If a node wants to leave the blockchain network, it can do so without notifying other nodes in the network. Public chains usually have incentive mechanisms to encourage more users to participate in the blockchain system. They commonly use the Proof of Work (PoW) consensus mechanism to maintain the blockchain network. Public chains have high scalability and can tolerate up to 50% malicious nodes. However, the large number of members in public chains often results in slower consensus speed. Additionally, PoW algorithm designed to mitigate Sybil attacks (creating multiple accounts to influence network consensus) requires high hardware requirements and consumes significant energy.

Consortium Chains: They are a type of private chain with a higher degree of decentralization compared to private chains. They are typically initiated by a group of enterprises or organizations working together to form a blockchain network. Consortium chains impose restrictions on participants in the blockchain network, where each participant needs permission to join the system. Participants can be determined at the initiation of the blockchain or through a set of rules established by the blockchain initiator. Consortium chains commonly use consensus algorithms such as Proof of Stake (PoS) or Practical Byzantine Fault Tolerance (PBFT). They can accommodate up to one-third of malicious nodes. Compared to public chains, consortium chains sacrifice some degree of decentralization and resilience to improve transaction processing speed within the blockchain network while reducing resource waste.

Private Chains: Private chains are fully centralized blockchains where only one main node is responsible for transaction processing. Nodes require special access or permission, otherwise, they cannot obtain authentication from the network. Only authorized members can join the network after undergoing identity verification. Private chains are typically used within organizations, leveraging the blockchain data structure to ensure data immutability and traceability.

Table 1 Comparison of different types of blockchains

	Public Chain	Consortium Chain	Private Chain
Centralization Level	Decentralized	Partially Decentralized	Centralized
Participants	Anyone	Authorized Enterprises/Organizations	Individuals or Independent Companies
Consensus Mechanism	PoW, PoS	PBFT, DPoS	Raft, Paxos
Accounting Nodes	Any Node	Consensus-based	Central Node
Incentive Mechanism	Required	Optional	Not Required
Transaction Speed	Slow	Faster	Fast

2.3. Consensus algorithms

Consensus algorithm is an algorithm that enables mutually untrusted nodes to reach an agreement. Common consensus algorithms include Proof of Work (PoW), Proof of Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), Kafka, and others.

Here, we adopt the Kafka consensus mechanism to order transactions published on the blockchain network. The consensus process is illustrated in Figure 1. Under the Kafka consensus, the client first sends transaction requests to the certificate authority. After receiving the certificate from the certificate authority, the client sends the transactions to the endorsement nodes. The endorsement nodes simulate the transactions based on the corresponding endorsement policy and send the endorsement results back to the client after verification. If the endorsement nodes pass the verification, the client packages the endorsement proposal and the endorsement results received from the nodes and sends them to the ordering nodes. The ordering nodes send the received transactions to the Kafka cluster, where the transactions are packaged into blocks in a certain order. The packaged blocks are then distributed to the committing nodes, which update the ledger accordingly.

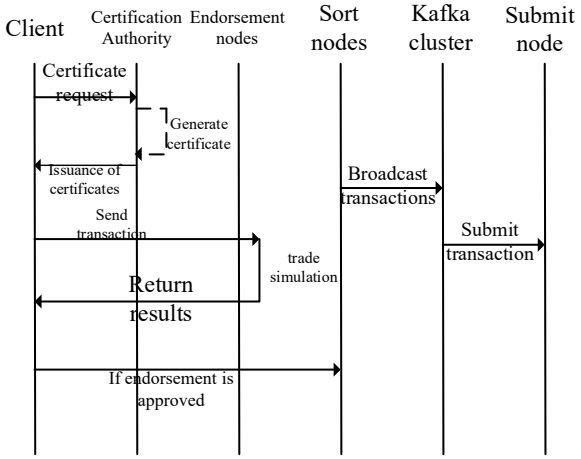

Figure 1 Kafka Consensus Process

2.4. Smart Contracts

Smart contracts, proposed by Nick Szabo, are defined as event-driven, stateful programs that run on a replicated and shared ledger and can store assets on the ledger. With the continuous development of blockchain technology and its application in various domains with different business scenarios and logic requirements, smart contracts have gained attention. As a computer protocol that can be disseminated in an information-based manner and automatically verify and execute, smart contracts automate the execution of predefined rules and terms as long as they meet the requirements specified in the smart contract code. These transactions are traceable and irreversible.

In existing blockchain applications, smart contracts are used to build decentralized autonomous organizations (DAOs) and automate the enforcement of rules and voting within the organization,

thereby avoiding centralized management. In decentralized finance (DeFi) applications, smart contracts are used to execute automated transfers, interest calculations, and other financial transactions. In supply chain traceability applications, smart contracts are used to record transaction information such as product production, transportation, and sales, enhancing the transparency and credibility of the supply chain.

3. System design

3.1. Infrastructure of supply chains

As shown in Figure 2, the blockchain architecture consists of four layers: the data layer, consensus layer, application layer, and data persistence layer.

The data layer is responsible for storing business data in the blockchain, such as enterprise registration information, transaction data, and the hash values of these data. Smart contracts handle transaction processing and query requests.

The consensus layer validates and orders the submitted transactions to generate blocks. It verifies the on-chain transactions and employs a consensus mechanism to ensure agreement among the participating nodes.

The data persistence layer employs a collaborative storage approach between on-chain and off-chain methods. Large-scale business data is stored in off-chain business databases, while the hash digests of the business data are stored as on-chain data in the blockchain ledger. The transaction hash digests are synchronized with the business databases as data evidence.

In the consensus layer, each organization within the consortium has peer nodes for information exchange and nodes (orderers) for transaction ordering. The Kafka consensus is used to package and order the transactions.

The application layer consists of the front-end interface and back-end systems of the traceability system. Web services interact with the blockchain network through SDK interfaces, enabling convenient data querying and product traceability.

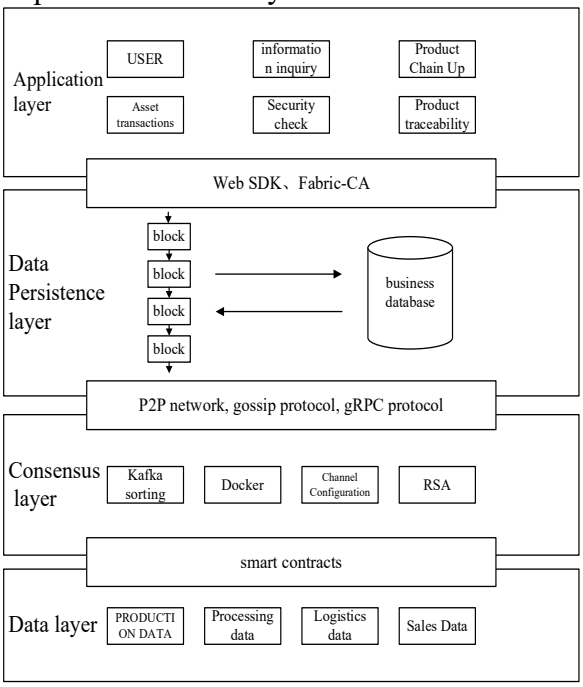

Figure 2 Fabric Architecture Design

3.2. Design of data persistent layer

During the product transaction process in the supply chain, a large amount of business data is generated. Storing massive data on the blockchain can undoubtedly burden the blockchain network. Firstly, the underlying platform of the blockchain generally adopts a key-value data storage

structure, which provides high efficiency for read and write operations but lacks support for complex queries. Secondly, uploading a significant amount of redundant and repetitive data to the blockchain consumes a considerable amount of bandwidth, resulting in valuable data being overshadowed by log transactions and waiting in queues for packaging, thereby affecting normal business operations. Moreover, a large volume of logs being uploaded to the blockchain can lead to rapid expansion of the blockchain's disk occupation, posing a storage challenge for all nodes.

In this study, the data persistence layer addresses the issue of storing large-scale business data by utilizing a collaborative approach between on-chain and off-chain storage methods.

Enterprise nodes store the raw material data, product data, logistics data, and traceability data, which are input from the off-chain application layer, into their respective business databases. They generate file hash digests for the stored data. If the transaction data is privacy-sensitive, the hash digest is further encrypted through digital signatures. The encrypted hash value is used as a necessary input parameter for the contract algorithm invocation and written into the consortium chain ledger along with the transaction data. By adopting a collaborative on-chain and off-chain storage approach, the size of the blocks can be effectively reduced, thus enhancing the unit-time throughput of the traceability application in the consortium chain.

When data needs to be verified, the original data and the block position information storing the data digest are retrieved from the database. A hash operation is performed on the data from the database, and the result is compared with the data stored on the block. If the hash values match, it indicates that the data has not been tampered with. If the hash values do not match, it indicates that the data has been tampered with.

3.3. Design of consensus layer

We utilize the Fabric consortium blockchain with Kafka consensus as illustrated in Figure 3. The configuration of the nodes is shown in Table 2 and deployed on 12 virtual machines. The server peer0 represents the peer0org1 node server, while "peer" refers to any optional node server used for testing different node possibilities. The structure diagram is depicted in Figure 3.

The smart contract is responsible for responding to requests from the application, executing code logic, and interacting with the ledger. The data within the smart contract primarily includes information about the production, distribution, and usage of products. Specifically, the raw material suppliers and product manufacturers' nodes have the requirement to register product creation. As the starting nodes in the product distribution process, these roles must ensure the integrity of the product information when creating products. It is crucial for the subsequent nodes involved in the distribution process (logistics, distribution, retail, end customers) to verify the authenticity of the product by relying on the information passed from the earlier nodes.

Algorithm 1 represents the smart contract algorithm used by enterprises for product registration in the blockchain. Before performing product registration, users need to define the structure of the product and store it in a JSON file in string format. Once the definition is completed, the smart contract will validate the format of the input data. If the validation is successful, the product will be registered under the corresponding owner's name and written into the blockchain network.

Table 2 Configuration of Kafka clustering

Names	IP addresses	amount	memo
Zookeeper	192.168.247.1/2/3	3	/
Kafka	192.168.247.4/5/6/7	4	/
Orderer	192.168.247.8/9/10	3	/
Peer0	192.168.247.11	1	For org 1
PeerX	192.168.247.12	1	For org 2

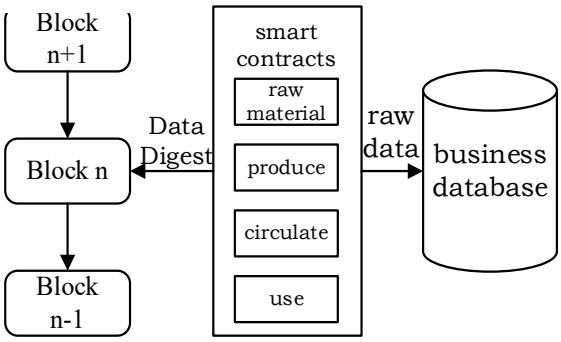

Figure 3 Data persistence layer storage model

Algorithm 1: **Product_registration** (productId, productInfo, ownerId)

```
Begin:
1.type Product struct
2.if input is empty or error struct
3.    return input error
4.else
5.    if isExist(productId) is true
6.        return productId is exist
7.    else
8.        if isExist(ownerId) is false
9.        return user not exist
10.   else
11.       ownerId.product.PutSate(productId)
12.       productId.putState(productInfo)
13.       return put product success
End
```

Algorithm 2 represents the algorithm used for product transactions between enterprises. During the transaction process, the smart contract will evaluate various input data and perform conditional checks. If all the conditions are met, the smart contract will update the asset ownership and record the transaction on the blockchain.

Algorithm 2: **product_transaction** (preownerId, productId, ownerId)

```
Begin:
1.if isExist (preownerId or productId or ownerId)
is false
2.    return input error
3.else
4.    getState(preownerId)
4.    for i in perownerId.productArr
5.        if perownerId.product[i] == productId
6.        preownerId.deleteState(product.productId)
6.        ownerId.putState(product.productId)
7.    else
8.        return preownerId not have this productId
End
```

Algorithm 3 represents the algorithm used by enterprises when consuming recorded raw materials or supplies from the blockchain for the purpose of manufacturing products. This algorithm primarily focuses on checking for violations or irregularities during the production process, such as expired raw materials or incorrect ingredient measurements. It ensures that the production process adheres to specific rules and regulations.

Algorithm 3: Production_auditting (productId, productInfo, ownerId)

```
Begin:
1.type Product struct
2.if isExist(productId) is true
3.    return productId is exist
4.else
5.if madeInChain(productInfo) is true
6.    get rMaterialArr from productInfo
7.    for rMaterial in rmArr.num
8.        if rMaterial.Exp<Now.date
9.        return rMaterial out of date
10.   else
11.       ownerId.deleteState(product. rMaterial)
12.       ownerId.putState(productId)
13.       productId.putState(productInfo)
End
```

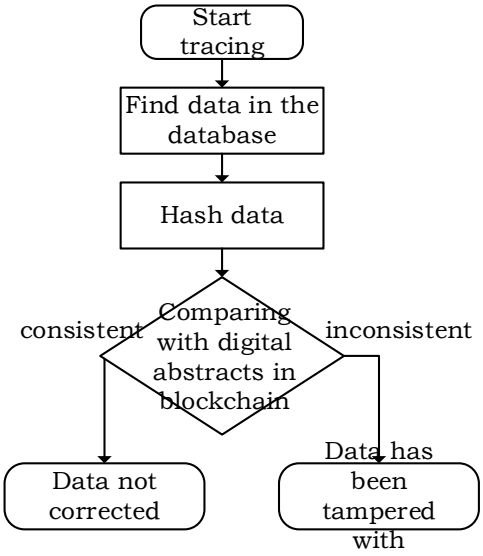

Figure 4 data validation flow chart

4. System Performance

After completing the construction of the supply chain transaction system, it is necessary to conduct testing on the system. In this paper, the load testing tool "Tape" is used to evaluate the system's throughput by analyzing the response of gRPC requests. The testing mainly involves invoking the query and invoke functions within the smart contract. The system's transaction throughput, success rate, and transaction latency are key factors for measuring system performance during blockchain transactions.

During the transaction packaging process, the sorting nodes need to digitally sign each transaction. Performing digital signatures for a large number of transactions within a unit of time can significantly impact system performance. To address this, we introduce the concept of batch signing, which involves packaging a group of data into a single batch. The sorting nodes only need to perform one signature for each batch, which greatly saves system performance and increases the throughput of data within a unit of time. Additionally, to prevent prolonged generation time for individual batches due to a lack of transactions in the blockchain, we set a maximum block generation time for each batch.

The system performance testing in this paper will provide the system's throughput for query and invoke operations under different parameter settings, such as adjusting the batch size, maximum

block generation time, and block size.

4.1. Performance under various transactions

Figure 5 demonstrates the changes in system throughput and latency as the total transaction volume increases in each round. Within the range of 500 to 5000 transactions, the throughput increases as the transaction volume grows, reaching a TPS of 69.3 in the case of a large number of transactions. The transaction latency ranges from 0.16s to 0.35s.

Figure 5 also shows the performance of read operations within the same transaction volume range. The application achieves a maximum transaction volume of 360 TPS per minute, with the latency remaining stable below 0.04s.

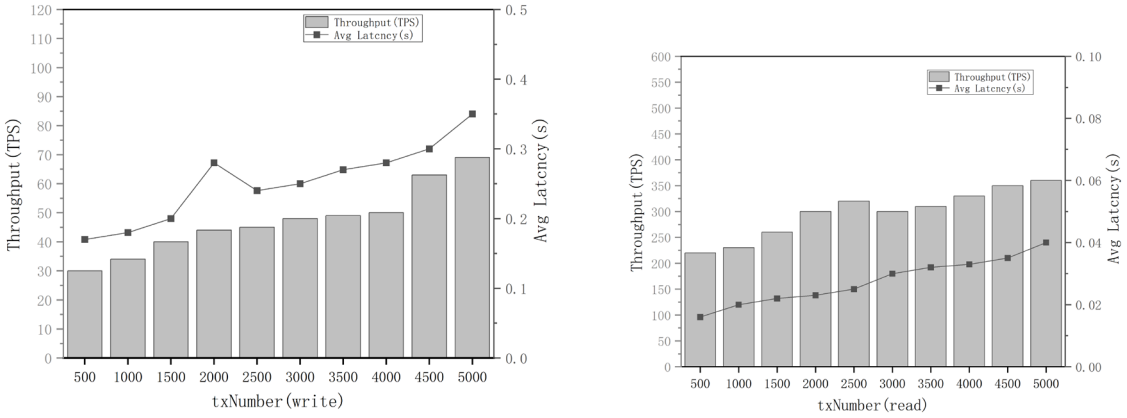

Figure 5 System performance under different reading and writing

4.2. Performance under various loadings

Figure 6 illustrates the changes in application throughput and latency as the transaction sending rate continues to increase. The throughput reaches its peak at a sending rate of 175 TPS and then remains above 125 TPS. The average latency increases as the transaction sending rate grows.

Since write operations, such as product registration and product transactions, require transaction construction, sorting, and block generation in the consensus layer, the latency per request is generally higher compared to read operations. The sending rate for read operations is configured at 550 TPS, and the average latency of the traceability application remains below 0.6s. At the same time, the throughput reaches its highest point at 350 TPS.

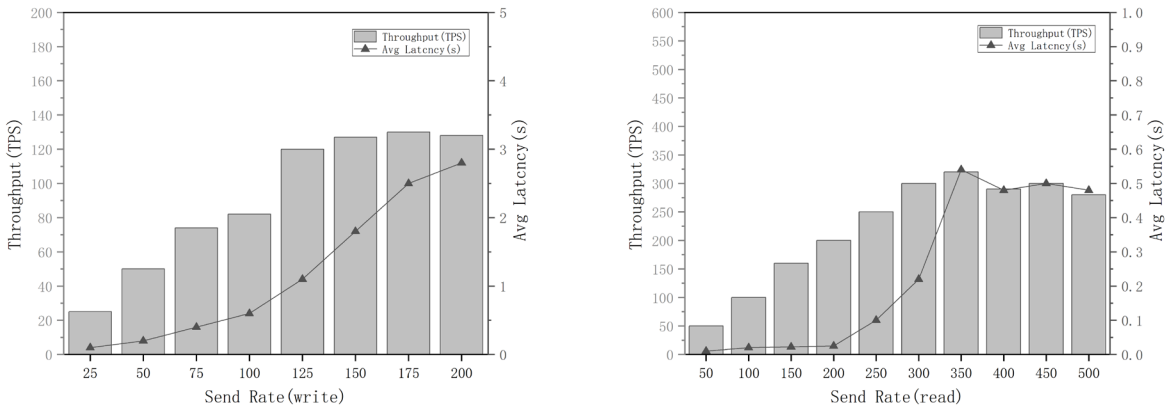

Figure 6 System performance under different sending rates

4.3. Performance related to node number

In the experiments conducted in this study, the batch size was set to 2048 and the block size was set to 128MB. When performing invoke operations, the number of nodes has an impact on the

system throughput. Table 7 presents the number of nodes used in the experiment, as well as the system latency and throughput achieved when performing 20,000 transactions under the current settings.

Figure 7 depicts the impact of the number of users on the system throughput. It can be observed that as the number of nodes increases, the system throughput does not continue to increase indefinitely. Additionally, the system latency gradually rises with the increase in the number of nodes.

Table 3 System performance under various nodes

Number of nodes	transactions	delay	TPS
10	20000	0.83s	1395
20	20000	1.24s	1532
60	20000	2.98s	1581
80	20000	3.82s	1446
100	20000	4.68s	1288

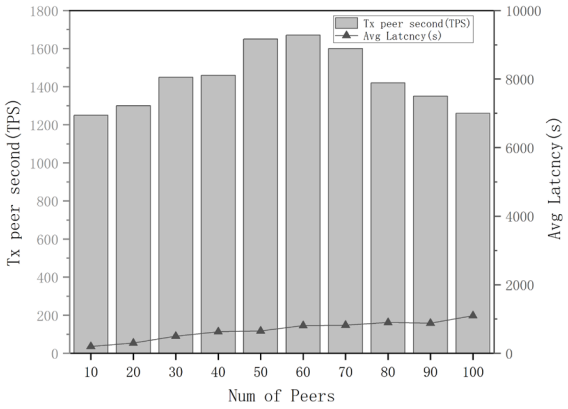

Figure 7 System performance under different peers

5. Conclusion

This paper presents a design for a supply chain transaction processing architecture based on blockchain technology and validates the feasibility and efficiency of this architecture through experiments. We employ the Hyperledger Fabric framework for the design, which is a decentralized architecture known for its openness, independence, and security. It enables transaction processing on the supply chain, ensures privacy data protection during the transaction process through channel isolation, and ensures transaction processing speed and data storage efficiency through a multi-layered database design.

However, our design also has some limitations. For example, it remains challenging to achieve a mixed batch and individual item traceability while ensuring privacy data protection. Additionally, further optimization of smart contracts to improve consensus efficiency and enhance throughput is an area that requires additional research. These issues are worth exploring in future studies.

References

[1] Naimi A I, Westreich D J. Big data: a revolution that will transform how we live, work, and think[J]. American Journal of Epidemiology, 2014, 179(9): 1143-1144.

[2] Meiklejohn S, Pomarole M, Jordan G, et al. A fistful of bitcoins[J]. Communications of the Acm, 2016, 59(4): 86-93.

[3] Pierro, Di M. What is the Blockchain[J]. Computing in Science & Engineering, 2017, 19(5): 92-

95.

[4] Eyal I. Blockchain technology: Transforming libertarian cryptocurrency dreams to finance and banking realities[J]. Computer, 2017, 50(9): 38-49.

[5] Nian L P, Chuen D L K. Introduction to bitcoin[M]// Handbook of digital currency. Academic Press, 2015: 5-30.

[6] Luu L, Chu D H, Olickel H, et al. Making Smart Contracts Smarter[C]//ACM SIGSAC Conference on Computer and Communications Security, 2016: 254-269.

[7] Leonhard R D. Developing Renewable Energy Credits as Cryptocurrency on Ethereum's Blockchain[J]. Social science electronic publishing, 2016, 14(12): 1-15.

[8] Pedrosa A R, Potop-Butucaru M, Tucci-Piergiovanni S. Scalable lightning factories for Bitcoin[C]// ACM/SIGAPP Symposium on Applied Computing, 2019: 302-309.

[9] Androulaki E, Manevich Y, Muralidharan S, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains[C]// the 13th EuroSys Conference. 2018: 1-15.

[10] Bach L M, Mihaljevic B, Zagar M. Comparative analysis of blockchain consensus algorithms[C]// 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). 2018: 1545-1550.

[11] Kiayias A, Russell A, David B, et al. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol[C]// Springer, Cham. Springer, Cham, 2017: 357-388.

[12] Zheng Z B, Xie S, Dai H, et al. An overview of blockchain technology: architecture, consensus, and future trends[C]// 6th IEEE International Congress on Big Data. IEEE, 2017: 557-564.

[13] Ali S, Wang G, White B, et al. Libra critique towards global decentralized financial system[C]// Communications in Computer and Information Science. Springer, 2019: 661-672

[14] Caro M P, Ali M S, Vecchio M, et al. Blockchain-based traceability in agri-food supply chain management: a practical implementation[C]// 2018 IoT Vertical and Topical Summit on Agriculture, 2018: 1-4.

[15] Figorilli S, Antonucci F, Costa C, et al. A blockchain implementation prototype for the electronic open source traceability of wood along the whole supply chain[J]. Sensors, 2018, 18(9): 3133.

[16] Tian F. An agri-food supply chain traceability system for China based on RFID & blockchain technology[C]// 2016 13th international conference on service systems and service management (ICSSSM). IEEE, 2016: 1-6.

[17] Kim H M, Laskowski M. Toward an ontology-driven blockchain design for supply-chain provenance[J]. Intelligent Systems in Accounting, Finance and Management, 2018, 25(1): 18-27.

[18] Lee C H, Yang H C, Wei Y C, Hsu W K. Enabling blockchain based scm systems with a real time event monitoring function for preemptive risk management. Appl. Sci. 2021, 11, 4811.