**Frontiers in Computing and Intelligent Systems**
ISSN: 2832-6024 | Vol. 5, No. 1, 2023

# A Fast Distributed Supply Chain Infrastructure based on Blockchain

**Aiqun Gu, Peng Zhao, Shiren Ye ***

School of Computer Science and Artificial Intelligence, Changzhou University, Changzhou Jiangsu 213000, China
*** Corresponding author:** Shiren Ye (Email: yes@cczu.edu.cn)

**Abstract:** Traditional supply chain systems often employ centralized servers, leading to transparency issues, high maintenance costs, security vulnerabilities, and poor risk resistance. We introduce a blockchain-based supply chain transaction infrastructure that emphasizes decentralization, distributed storage, tamper-resistance, traceability, and cost-effectiveness. Built on a consortium blockchain, our architecture uses a dual-layer storage structure combining blockchain and a business database, optimizing transaction data security and accessibility. Simulated tests confirm its efficiency, transparency, security, and ability to handle industrial-scale applications.

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

## 1. Introduction

Emerging in the 1980s, supply chains involve interconnected suppliers, manufacturers, and consumers, managed around a central company. It spans sourcing components to delivering finished products. Efficient supply chain management is pivotal for businesses, especially with its exponentially growing data [1]. Yet, traditional systems are often siloed, hindering collaboration, traceability, and trust. Blockchain, with its distributed and immutable nature, can transform this landscape, offering increased transparency and fortifying inter-company relationships.

This paper unveils a consortium blockchain-based supply chain model. It prioritizes consensus recording, transaction privacy, and high-speed large-scale transactions. Using Hyperledger Fabric, channel isolation, and a "blockchain + business database" approach, our model addresses the challenges of traditional blockchain technologies.

## 2. Related Work

### 2.1. Supply Chinas

Modern businesses compete at the supply chain level, not just through individual products. For instance, Walmart's efficient restocking system, using UPC and RFDC, drastically improved its performance [2]. Meanwhile, JingDong digital supply chain transformation saved them millions annually in warehousing alone.

Traditional supply chain systems, however, are susceptible to data tampering, illegal transactions, and breaches. Integrating blockchain in these systems tackles these challenges, emphasizing decentralization, traceability, and cost-effectiveness. Notable implementations include AgriBlockIoT for agricultural traceability [3] and a timber traceability system using RFID and blockchain[4].

Yet, many public blockchain architectures, like Bitcoin, are hindered by slow transaction rates. This research delves into an efficient blockchain design that ensures data integrity and privacy while facilitating traceability.

### 2.2. Blockchain

The blockchain model, foundational to Bitcoin, comprises layers like data, network, consensus, and application[5][6]. While typical systems like Bitcoin and Ethereum revolutionized decentralized transactions, they faced scalability and energy consumption issues [7].

Addressing these, Pedrosa et al. [8] introduced the Lightning Network, while Androulaki et al. [9] showcased a scalable Hyperledger Fabric system. With blockchain's growing adoption, consensus mechanisms evolved, with innovative solutions like the Ouroboros protocol offering advantages over traditional systems [11].

In general, blockchains can be:

- **Public Chains:** Fully decentralized networks open to all, usually incentivizing participation. Despite high scalability, they suffer from slower consensus speeds and high resource consumption.
- **Consortium Chains:** Semi-private chains initiated by enterprise collaborations. They prioritize transaction speed and reduced resource use but sacrifice some decentralization.
- **Private Chains:** Centralized chains, mostly within organizations, offering data integrity and traceability but restrict outsider access.

**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 |

Consensus algorithms play a crucial role in ensuring that all participants in a distributed system agree on a single version of the truth. They are especially significant in blockchain networks, where different nodes may receive transactions in different orders. Consensus mechanisms ensure that despite these differences, every node agrees on the order in which transactions are added to the blockchain, making the blockchain consistent across all nodes.

## 2.3. Proof of Work (PoW)

The PoW algorithm requires network participants (miners) to solve complex mathematical problems. Once a miner finds a solution, it broadcasts it to the network. The first one to provide a valid solution gets the right to add the next block and is rewarded with cryptocurrency. This method is energy-intensive and has been criticized for its environmental impact.

## 2.4. Proof of Stake (PoS)

In PoS, validators are chosen to create new blocks based on the number of coins they hold and are willing to "stake" or lock up as collateral. Validators are incentivized to validate legitimate transactions because they stand to lose their staked coins if they validate malicious transactions.

## 2.5. Practical Byzantine Fault Tolerance (PBFT)

PBFT is designed to handle malicious nodes in a distributed system. In PBFT, a network node sends a message to other nodes, and these nodes communicate with each other to determine if a majority received the same message. Once over 2/3 of nodes agree on the message, consensus is reached. PBFT is fast and doesn't require a lot of computational power, but it's best suited for private blockchains with a limited number of nodes.

## 2.6. Kafka

Kafka is a high-throughput, distributed messaging system originally developed by LinkedIn and later contributed to the Apache Software Foundation. In blockchain systems, Kafka can be used as an ordering service to ensure that transactions are added to the blockchain in the correct order. The process of reaching consensus using Kafka is as described in following steps:

1. The client sends a transaction request to the certificate authority (CA).
2. Upon validation, the CA sends back a certificate to the client.
3. The client forwards the transaction, along with the received certificate, to the endorsing nodes.
4. Endorsing nodes simulate the transaction and, if it meets the endorsement policy, they endorse it and send a response back to the client.
5. After receiving endorsements, the client sends the transaction to the ordering nodes.
6. The ordering nodes relay these transactions to the Kafka cluster.
7. Inside the Kafka cluster, transactions are arranged in a specific sequence and packed into blocks.
8. These blocks are then forwarded to committing nodes that update their ledger accordingly.

In the Kafka consensus mechanism, the endorsement policy ensures that only valid transactions are forwarded to the Kafka cluster. Kafka's main advantage in this context is its ability to handle high-throughput scenarios, ensuring that transactions are consistently ordered across all nodes.

This mechanism provides scalability and robustness, but one should note that it may not offer the same level of decentralization or resistance to censorship as other consensus algorithms. It's more suited for consortium or private blockchains where there's a level of trust among participants.

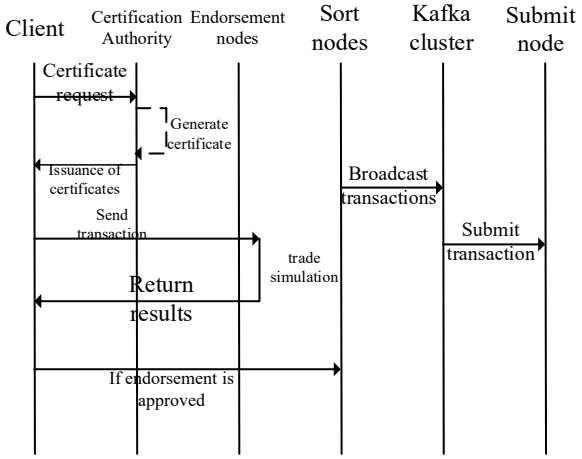


**Figure 1.** Kafka Consensus Process

## 2.7. Smart Contracts

The evolution of blockchain technology has brought about the concept of smart contracts. The notion, which was first put forth by Nick Szabo, is essentially a self-executing contract where the agreement between the buyer and seller is written into lines of code. These contracts live on the blockchain and are decentralized. Once deployed, they can't be modified.

The primary attributes of smart contracts include:

- Event-driven: They execute actions when specific conditions (or events) are met.
- Stateful: They have a memory of past transactions and states. This helps in understanding the context of the present situation.
- Replication and Sharing: Smart contracts are replicated across the blockchain, ensuring uniformity and consistency.
- Automatic Enforcement: They don't need intermediaries. Once the conditions are met, the contract self-executes.
- Irreversibility and Transparency: Once a contract is executed, it can't be undone. This provides trustworthiness and transparency.

From DAOs to DeFi, the applications of smart contracts are diverse. For instance, in a DAO, governance decisions can be made based on a predefined set of rules embedded in a smart contract. In DeFi, financial protocols can run without intermediaries, offering services like lending, borrowing, or yield farming. Supply chains, as mentioned, can use them to guarantee the authenticity of products and their journey from origin to the end consumer.

# 3. System Design

## 3.1. Infrastructure of Supply Chains

The proposed blockchain infrastructure for supply chains is layered, ensuring separation of concerns, scalability, and robustness. The four layers are:

- Data Layer: Think of this as the first point of interaction with the blockchain. It is where raw data enters the system.

The data might be about goods being manufactured, their transportation status, their sale, or any other transaction data. The inclusion of hash values ensures data integrity and makes tampering evident.

• Consensus Layer: This layer is critical in a decentralized system. Given that multiple entities might be adding data to the blockchain simultaneously, the consensus layer ensures everyone agrees on the order of these additions. This is where the Kafka consensus mechanism can come in, ordering transactions in a way that all participants agree upon.

• Data Persistence Layer: Storing every piece of data on the blockchain can be inefficient and expensive. This layer uses a hybrid approach. While crucial data (like hash digests) is stored on-chain for transparency and verification purposes, bulky business data resides off-chain in traditional databases. This ensures efficient storage and quick retrieval.

• Application Layer: This is where users (be it businesses or consumers) interact with the blockchain. It provides interfaces for querying data or tracing a product's journey. It communicates with the blockchain, translates the raw data, and presents it in a user-friendly manner. This layer will typically have SDKs (Software Development Kits) that allow for the creation of applications that can seamlessly integrate with the blockchain.

The proposed design aims to offer a transparent, tamper-proof, and efficient system for supply chain management. By leveraging blockchain's inherent properties and coupling it with smart contracts, supply chains can achieve unprecedented levels of trust and traceability.

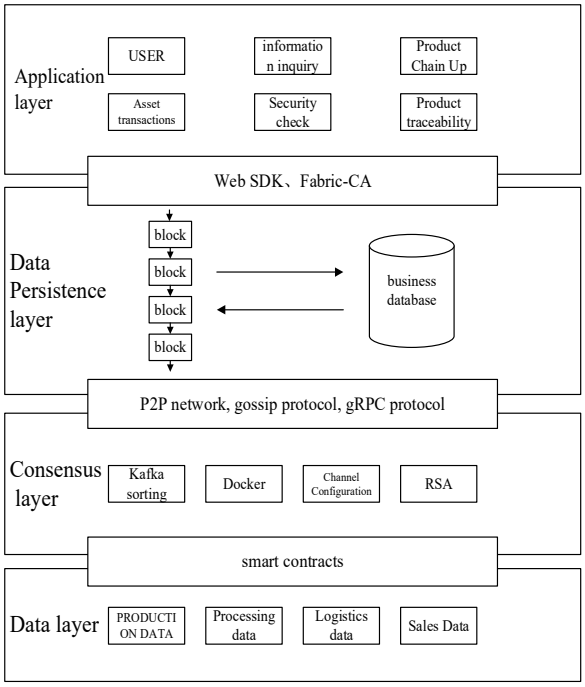


**Figure 2.** Fabric Architecture Design

## 3.2. Design of Data Persistent Layer

In the proposed design of the data persistence layer, the challenge of efficiently managing large-scale business data in a blockchain environment is addressed.

• Collaborative Storage Approach: Balancing between on-chain and off-chain storage is the crux of the design. Most of the bulky business data (raw material data, product data, logistics data, and traceability data) is stored off-chain in respective business databases of enterprise nodes. This approach keeps the blockchain lean, scalable, and efficient.

• Hash Digests and Encryption: To ensure data integrity and privacy, file hash digests are created for the stored data. If data is sensitive, the hash is further encrypted, ensuring both data integrity and confidentiality. This encrypted hash, which is a compact representation of the actual data, is stored on-chain, thus ensuring the blockchain's primary advantage of transparency and trustworthiness without the bloat.

• Data Verification: The data verification process is straightforward. When data needs to be authenticated, it is hashed and compared with the hash value stored on-chain. If the hash values match, the data is deemed genuine. A mismatch implies tampering, ensuring data credibility.

## 3.3. Design of Consensus Layer

The consensus layer is built on the Fabric consortium blockchain, utilizing the Kafka consensus mechanism. The detailed configuration offers flexibility for real-world testing and deployment.

• Infrastructure Overview: The proposed infrastructure consists of multiple nodes, each with its designated purpose, as illustrated in Table 2. These nodes are distributed across 12 virtual machines, optimizing for redundancy, resilience, and scalability.

• Smart Contract Role: The smart contract is pivotal in managing the business logic on the blockchain. It handles requests, executes pre-defined logic, and interacts with the ledger. The encapsulated data is vital, covering the lifecycle of a product, from creation to usage. The smart contract guarantees that the product's integrity is maintained throughout its lifecycle. The starting nodes in the product's journey (raw material suppliers, product manufacturers) have the onus of ensuring the authenticity of the product information.

• Product Registration Algorithm: Algorithm 1, which outlines the product registration process, is the cornerstone of the smart contract's function. It offers a structured approach to register products in the blockchain. The algorithm starts with users defining the structure of the product, which is then validated by the smart contract. Upon successful validation, the product is registered and associated with the respective owner. This ensures that every product on the blockchain can be traced back to its legitimate source, reinforcing the system's credibility.

In conclusion, the proposed design offers a robust system where the data persistence layer ensures efficient storage and retrieval of vast amounts of business data, and the consensus layer ensures that all transactions on the network are agreed upon by all participants. Through these designs, the system achieves scalability, efficiency, and trustworthiness – all critical attributes for a modern supply chain blockchain system.

**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 |
| | | | |

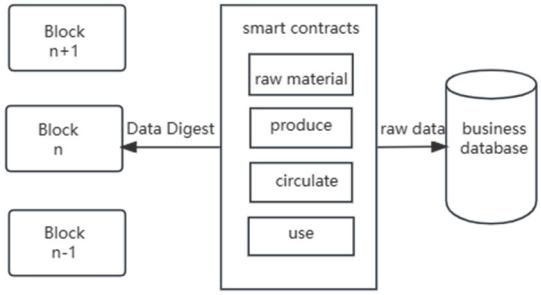


**Figure 3.** Data persistence layer storage model

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

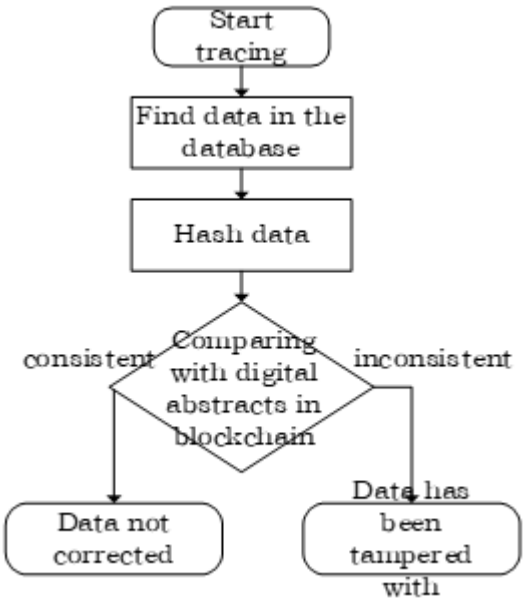


**Figure 4.** Data validation flow chart

# 4. Evaluation of System Efficacy

The modern landscape of supply chain transaction systems, fortified by the bedrock of blockchain technology, mandates an uncompromising lens of scrutiny. Thus, in our endeavour to dissect the system's mettle, we turn to the esteemed 'Tape' tool. With its precision, we aim to elucidate the system's throughput by meticulously analysing the intricacies of gRPC request responses.

At the heart of our analysis lies the functionality encapsulated within the system's smart contract. This exploration, while seemingly intricate, offers us a panoramic view into the system's ability both in retrieval (through the 'query' function) and modification (via the 'invoke' function). Three cardinal metrics emerge as our touchstones: The transaction throughput, which elegantly captures the volume of transactions the system handles within a temporal frame; the rate of success, offering a measure of the transactions executed to perfection; and the latency of transactions, a subtle yet profound gauge of the intervals between a transaction's initiation and its seamless conclusion.

Diving deeper into the realm of digital signatures, we encounter a potential quagmire. The act of individually signing a multitude of transactions could throttle the system's vitality. Our solution is both artful and pragmatic: the introduction of batch signing. By bundling multiple transactions into an aggregate and bestowing a singular digital signature upon it, we optimize and refine the system's

performance. Furthermore, in our pursuit of rhythmic consistency, we've instituted a maximal temporal threshold for the genesis of each batch's block, ensuring the systematic and predictable emergence of blocks, thereby enhancing the system's reliability.

## 4.1. Performance Analysis Across Varied Transactional Volumes

Our gaze now shifts to the interplay between the system's performance attributes and the ever-fluctuating symphony of transaction volumes. As we navigate this landscape, certain patterns emerge with crystalline clarity. As transaction volumes swell, there's a proportional ascent in the system's throughput, reaching an apex at a commendable 69.3 TPS during peak transactional activity. This dance of numbers is mirrored in the system's latency, which, while largely delicate, ebbs and flows between 0.16s to 0.35s, providing a nuanced commentary on the system's response dynamics.

It is in the realm of reading operations, or queries, that the system truly sings. At its zenith, the system gracefully handles a staggering 360 TPS per minute, all the while ensuring that the latency remains a mere whisper, never crossing the 0.04s threshold.

This blockchain-anchored supply chain transaction system, with its intricate design and commendable performance metrics, stands as a beacon of promise. Its stellar throughput and latency metrics, especially amidst burgeoning transactional volumes, offer invaluable insights for discerning enterprises and institutions contemplating its adoption.

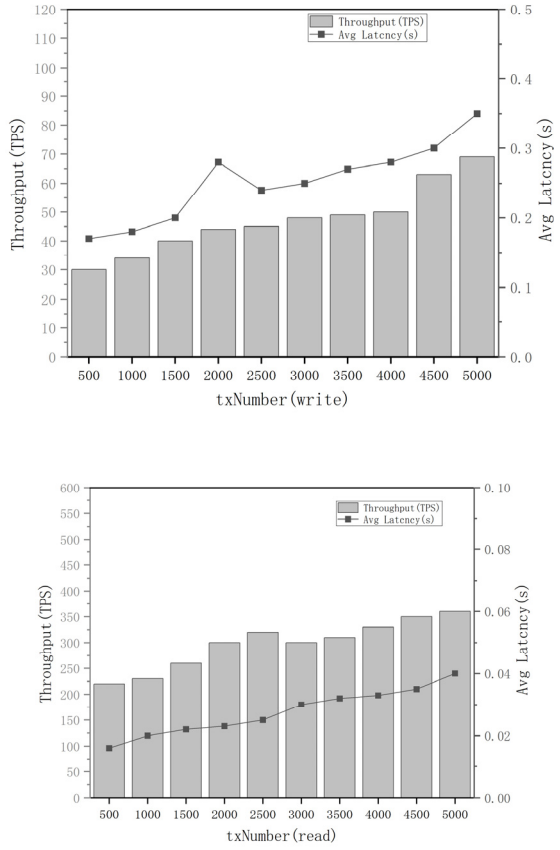


**Figure 5.** System performance under different reading and writing

## 4.2. Performance Analysis under Varied Load Dynamics

Figure 6 offers insights into the application's performance as transaction sending rates intensify. The throughput plateaus at an impressive 175 TPS, and despite increased loads, remains consistent above 125 TPS. However, as the rate grows, there's a corresponding uptick in average latency.

Write-centric operations, inherently intricate due to transaction creation, sorting, and block formation, register a higher latency than read operations. With the sending rate set at 550 TPS for read tasks, the latency remains commendably below 0.6s, and throughput peaks at 350 TPS. This analysis underscores the system's capability to maintain efficiency amidst heightened demands.

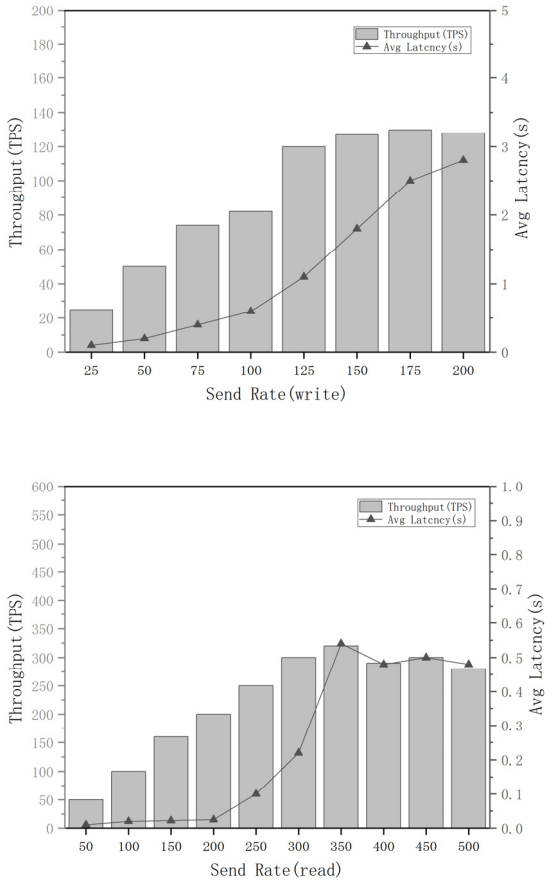


**Figure 6.** System performance under different sending rates

## 4.3. Influence of Node Count on Performance

The study executed its tests with a fixed batch size of 2048 and a block size of 128MB. A notable observation during these tests was the profound influence the number of nodes exerted on the system's throughput, especially during invoke operations.

Table 7 methodically outlines the performance metrics, detailing the system's latency and throughput, when subject to 20,000 transactions with varying node counts.

Figure 7 delves deeper into the nuanced relationship between node count and system throughput. An intriguing trend is evident: the system throughput doesn't consistently escalate with an upsurge in nodes. Instead, it appears to reach a saturation point beyond which no significant throughput gains are observed. Concurrently, a consistent theme of increased latency emerges, correlating directly with the augmented node count. This pattern underscores the delicate equilibrium between system scalability and performance efficacy.

**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 |

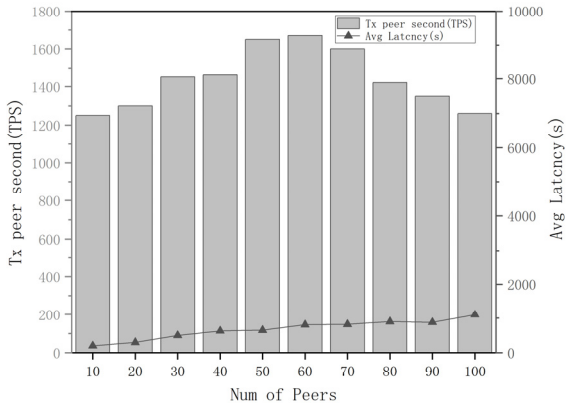


**Figure 7.** System performance under different peers

# 5. Conclusion

In this study, we introduced and rigorously assessed a novel blockchain-based architecture for managing supply chain transactions. By leveraging the Hyperledger Fabric framework, a paradigm celebrated for its security, independence, and openness, we have designed a decentralized system poised to revolutionize transaction processing within supply chains. One of the paramount virtues of this system lies in its capacity to guard privacy data during transactions via channel isolation, all the while retaining the agility in transaction processing and data storage through an intricate multi-layered database design.

However, like any evolving technological endeavor, our architecture is not without its challenges. Presently, balancing between aggregate batch traceability and individual item traceability whilst maintaining robust data privacy remains a complex task. Moreover, the intricate realm of smart contract optimization to amplify consensus efficiency and elevate throughput is an ongoing pursuit. These facets provide fertile ground for future research, promising both challenges and opportunities for the next wave of innovation in supply chain management.

# 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] 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.

[3] 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.

[4] 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.

[5] Pierro, Di M. What is the Blockchain[J]. Computing in Science & Engineering, 2017, 19(5): 92-95.

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

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

[13] 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.

[14] 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.

[15] 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.

[16] 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.

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

[18] 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.