Co-Sharding: A Sharding Scheme for Large-Scale Internet of Things Application

HAOTIAN YANG, Jiangsu University, China and Institute of Food Safety and Nutrition, Jiangsu Academy of Agricultural Sciences, China

XIAOYU ZHANG*, ZIHAN WU, and LIANGMIN WANG, Southeast University, China

XIAO CHEN and LU LIU, University of Leicester, UK

Blockchain technology finds widespread application in the management of Internet of Things (IoT) devices. In response to the challenges posed by performance scalability and the convergence of multiple ledgers stemming from an expanding network, this study introduces the concept of *Co-Sharding*. Within this framework, the ledger maintained by sub-chains overseeing IoT operations in distinct geographic regions is conceptualized as a shard within the Large-scale Internet of Things (LIOT) ledger. Meanwhile, elected nodes within each region assume responsibility for maintaining a coordinating shard, facilitating cross-regional communication and data interaction. Furthermore, our work presents a multi-objective optimization algorithm grounded in the multi-shard paradigm to enact a scheduling strategy that spans various regions. We undertake a series of pertinent experiments and conduct a comparative analysis of scheduling algorithms within the context of a real-world cross-regional agricultural IoT system, utilizing actual operational data. The comparative results demonstrate that, in comparison to intra-sub-region scheduling, the *Co-Sharding* approach enhances machine utilization rates by approximately 30% and reduces scheduling time by around 18% when confronted with a task count of twelve. In terms of performance, *Co-Sharding* also exhibits the capability to reduce the storage requirements of lightweight nodes within each region by approximately 39%, while concurrently improving throughput by approximately 1.5 times when contrasted with a single-chain architecture.

Additional Key Words and Phrases: Large-scale IoT, cross-region, coordinating shard, scheduling model

ACM Reference Format:

Haotian Yang, Xiaoyu Zhang, Zihan Wu, Liangmin Wang, Xiao Chen, and Lu Liu. 2023. Co-Sharding: A Sharding Scheme for Large-Scale Internet of Things Application. 1, 1 (November 2023), 17 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The use of blockchain technology is widespread in the management of IoT devices [5]. For example, in the context of large-scale medical IoT, patient data is securely managed through blockchain, facilitating data transmission between sensors and various entities [3]. In agricultural IoT, blockchain serves as a reliable means to store scheduling data, effectively addressing trust-related concerns between individual farmers and schedulers [35]. Nonetheless, as the scale of IoT applications continues to grow, Subzone Internet of Things (Sub-IOT), responsible for overseeing smaller regions, tends to merge into the realm of Large-scale Internet of Things (LIOT) [22]. This transition is evident in scenarios where medical IoT networks associated with a single hospital group expand to encompass multiple hospital groups

Authors' addresses: Haotian Yang, haotyang@gmail.com, Jiangsu University, 301 Xuefu Rd, Zhenjiang, Jiangsu, China, 212013 and Institute of Food Safety and Nutrition, Jiangsu Academy of Agricultural Sciences, 50 Zhongling Rd, Nanjing, Jiangsu, China, 210014; Xiaoyu Zhang, 17318583872@163.com; Zihan Wu; Liangmin Wang, liangmin@seu.edu.cn, Southeast University, 2 Southeast University Rd, Nanjing, Jiangsu, China, 211189; Xiao Chen; Lu Liu, University of Leicester, Leicester, UK.

Manuscript submitted to ACM

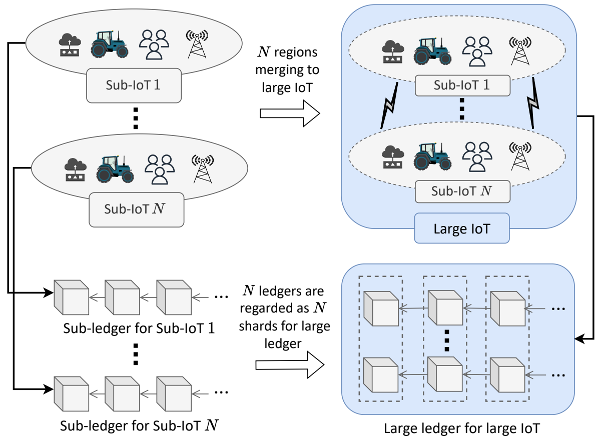

Fig. 1. Sub-regional fusion of the IoT for agricultural machinery.

[4]. Similarly, within the agricultural IoT domain, the expansion involves the transition from a single sub-region to encompass multiple regions [33]. As a consequence, the sub-ledgers, originally tasked with managing multiple Sub-IOTs, also require consolidation into a larger ledger characterized by an extensive chain.

The consolidation of multiple sub-ledgers into a single large ledger presents a challenging task. To illustrate, we consider the application of agricultural machinery within the context of IoT. As agricultural equipment advances towards intelligence and unmanned operation, many countries have initiated endeavors to achieve high-precision path planning for unmanned machinery through the utilization of sensing, positioning, and navigation technologies [27, 31, 34]. National agricultural associations have also embarked on the sharing of resources pertaining to agricultural machinery, which are dispersed across various regions. Consequently, in the context of multi-device interconnection and complex access environments, the realization of efficient resource scheduling necessitates the integration of data related to agricultural machinery from independently managed sub-regions into a comprehensive large-scale agricultural machinery IoT scheduling system [30], as illustrated in Fig. 1. On the other hand, when it comes to blockchain-based IoT data fusion, there exists a requirement to amalgamate data originally stored on distinct sub-chains into a newly unified chain. Failure to do so would result in data dispersion across each sub-chain, impeding the ability to achieve optimized scheduling across the entire agricultural machinery network [1].

Unfortunately, due to the decentralized and peer-to-peer communication characteristics of the blockchain system, achieving this fusion presents significant challenges, primarily manifesting in three key aspects: 1) Computational efficiency problem: The vast number of nodes within the entire network results in an extended time required by the consensus mechanism. 2) Querying efficiency problem: With the expansion of the chain length, the data query time also increases, often surpassing that of the sub-chain responsible for managing the sub-region. In response to these challenges, this paper introduces a sharding scheme for LIoT named *Co-Sharding*, aimed at resolving the aforementioned difficulties and enabling the fusion of multi-region sub-ledgers. The contributions of this research are summarized as follows:

(1) We present *Co-Sharding* as a solution to the challenge of merging multiple chains and facilitating data fusion. *Co-Sharding* involves treating the ledger maintained by sub-chains overseeing IoT operations in distinct regions

as a shard within the overarching LIoT ledger. Elected nodes within each region assume responsibility for managing a coordinating shard, and avoid additional communication overhead across ledgers by reading data locally on the shared node. This is conducive to cross-region communication and data interaction scheduling, thereby improving the efficiency of cross-ledger interactions.

(2) We propose a coordinating-shard-driven approach to facilitate cross-region data interaction. This approach validates cross-ledger data and retrieval requests, subsequently returning and committing the results of consensus execution to the relevant sub-ledgers via the coordinating shard. The approach effectively reduces the overhead associated with cross-ledger communication and can tolerate up to one-third of malicious nodes.

(3) We implement *Co-Sharding* within the Hyperledger Fabric framework and deploy a scheduling smart contract based on a multi-objective optimization algorithm in a cross-region scheduling scenario for testing purposes.

To the best of our knowledge, *Co-Sharding* represents the first efficient solution that combines optimization with cross-region scheduling. This innovative approach has yielded a remarkable 30% enhancement in utilization rates when the task count is 12, coupled with an 18% reduction in scheduling time compared to the existing agricultural machinery scheduling system. Additionally, in contrast to the direct merging of regional data nodes into a large ledger, *Co-Sharding* has significantly reduced the storage capacity demands of lightweight nodes by approximately 39%, while concurrently boosting data query throughput by approximately 1.5 times.

2 RELATED WORKS

2.1 Blockchain Empowered IoT

Recently, blockchain-empowered data management in IoT has garnered significant attention. Karthika and Jaganathan [19] designed a new consensus method and a lightweight blockchain framework primarily for resource-constrained IoT applications in licensed environments. Wang et al. [32] proposed a novel identity authentication mechanism based on transfer learning-empowered blockchain, which manages user identities through the blockchain and ensures the protection of privacy. Deebak et al. [12] introduced a trust-aware blockchain-based seamless authentication system with privacy preservation (TAB-SAPP) to address key concerns related to privacy, security, and packet delivery rates. Javaid and Sikdar [16] proposed a transformable blockchain-based Industrial IoT architecture that utilizes a dynamic proof-of-work consensus and block checkpointing mechanism. Such an architecture ensures the data integrity and computational reliability of Industrial IoT. However, Lin et al. [23] analyzed the current blockchain solutions for IoT and found that existing schemes are tailored for Sub-IoT applications, where blockchain can effectively manage data and ensure security. Nevertheless, these solutions still struggle to resolve the problem of local resource constraints in devices and the substantial resource overhead of distributed information interaction in LIoT. Cai et al. [8] also conducted an analysis, concluding that it is challenging to apply blockchain in LIoT with a large amount of data due to its limitations in performance and functional scalability. Therefore, blockchain for managing IoT faces the challenge of development difficulties in the context of regional fusion.

Blockchain-empowered agricultural machinery IoT is gaining popularity in small-scale agriculture, offering a viable solution for optimizing resource utilization and scheduling strategies. Khan et al. [20] collected, scheduled, and stored drone data via blockchain and meta-heuristic genetic algorithms, resulting in reduced computational costs and improved performance and resource utilization. Zheng et al. [40] implemented a blockchain-based agricultural service platform and used a mixed-integer linear programming model to obtain the optimal unmanned agricultural machine service plan. Unfortunately, Rahman et al. [29] have suggested that the demand for interconnecting farm machinery and data across regions is now high. However, the IoT of agricultural machinery still faces challenges related to conflicting scheduling goals as it scales up, and data processing performance remains a bottleneck in large-scale networks.

2.2 Blockchain-based Resource Matching Optimization

Currently, there has been an increased focus on resource matching within blockchain-based Subzone IoT (Sub-IoT). Yang et al. [36] devised a blockchain-based distributed resource matching mechanism and enhanced the IoT system with fog computing to offload computationally demanding tasks, resulting in a low-latency matching algorithm. Chen et al. [9] proposed a distributed energy transaction matching scheme that integrates blockchain with game theory, ensuring security while achieving efficient and benefit-maximizing energy scheduling. Jiao et al. [18] introduced a resource allocation scheme based on an auction model, aiming to maximize social welfare within a blockchain system. Jia et al. [17] proposed an auction-based resource matching mechanism that uses cryptography and blockchain to support off-chain allocation, privacy protection, and on-chain dispute resolution to prevent collusion. However, there is a paucity of cross-ledger studies in the context of LIoT, with most of the research concentrating on the optimization of scheduling algorithms. For instance, Liu et al. [24] developed a multi-objective immune algorithm based on non-dominated neighbor-based selection and Tabu search to enhance the search efficiency of dynamic cross-region collaborative scheduling matching for agricultural machinery. Orfanou et al. [26] proposed an agricultural machinery planning algorithm to sequence the operation of multiple machinery tasks based on specified operation areas and orders. Unfortunately, although existing studies have successfully achieved efficient resource scheduling within Sub-IoT, they have not thoroughly investigated the blockchain structure and the process of cross-ledger interaction. Furthermore, the efficiency challenge within the context of LIoT remains unaddressed.

2.3 Sharding-based Cross-ledger Scheme

Existing research on cross-ledger technology primarily focuses on cross-chain and cross-shard solutions. Zhang et al. [39] proposed cross-chain data transmission based on the supervision chain. This method performs calculations on the storage chain and only sends the results to other chains, effectively improving transmission efficiency. Fan et al. [13] built a cross-chain medical IoT data sharing framework, which introduced a relay chain to verify the consistency between data requests and actual storage. Yang et al. [37] collected cross-region data query requests through relay chains and implemented fast cross-domain anonymous transactions based on zero-knowledge proof. However, Ou et al. [28] analyzed that cross-chain solutions encounter challenges due to the involvement of multiple heterogeneous blockchains. Inconsistencies in ledger structure and cryptographic algorithms or even inconsistent message or signature formats make merging them difficult, leading to a lack of schemes for merging multiple blockchain systems. The concept of sharding, which involves creating isomorphic sub-blockchains, was initially introduced by Luu et al. [25]. Their blockchain sharding scheme, named Elastico, established multiple committees and divides the original single chain into different isomorphic shards. Kokoris-Kogias et al. [21] proposed Omniledger, a distributed ledger protocol based on sharding, and employed a random generation protocol to initialize the shards. Zamani et al. [38] designed a blockchain sharding protocol called RapidChain, aimed at optimizing the intra-committee consensus algorithm, gossip protocol, and cross-shard transaction validation techniques. Dang et al. [11] utilized a trusted execution environment to implement secure and efficient consensus and shard formation protocols (referred to in this paper as BFT-Shard), constructing a single reference shard to coordinate cross-sharding protocols within a complete-sharding framework. Amiri et al. [2] introduced Sharper, which supports networks consisting solely of crashed or Byzantine nodes. Each sharding cluster maintains only one ledger view and implements the cross-sharding consensus protocol through competition between shards. The above schemes significantly enhanced the scalability of the blockchain. However, the interaction across multiple ledgers necessitates frequent communication and additional consensus rounds, leading to potential efficiency concerns. As a result, the sharding protocol manages multiple sub-ledgers of the same chain in a *divide-and-conquer* manner during the chain growth process. This concept of sharding provides inspiration for managing sub-ledgers from different regions, but it also needs to improve the performance of multi-region ledger interactions.

3 CO-SHARDING SOLUTION FOR LIOT

In this paper, we introduce *Co-Sharding*, a multi-subsection ledger fusion scheme aimed at reducing the computational load of the system and eliminating redundant data storage. Within each Sub-IoT, there exists a significant distribution of user nodes. To enable parallel data processing and storage of only region-specific data, users within each region establish multiple Sub-IoT sharding systems. Additionally, an additional shard, referred to as the coordinating shard, is constructed to facilitate the coordination of processing and querying of cross-region data.

3.1 Co-Sharding Formation from Muti-regional-shard

Before forming the Sub-IoT sharding system, it will be necessary to establish a cross-regional coordinating shard. This shard must be responsible for transmitting and receiving relevant cross-region messages and data, as well as executing the smart contracts for cross-region tasks, including cross-region task decisions. Meanwhile, each user's location, identity information, and system status are stored in the coordinating shard. This also implies that the coordinating shard needs to implement a fair and secure algorithm to prevent the majority of nodes in a region from gaining control, which could result in the domination of cross-regional tasks by a single region. This, in turn, might lead to a biased execution strategy that favors the interests of that region at the expense of the development of other regions. Ultimately, this could result in untrustworthy decision-making and adversely affect the overall coordination of the entire system. Fairness primarily revolves around equitable node selection. Therefore, we need to design a secure and fair node selection algorithm.

Algorithm 1 *Co-Sharding* formation

Input: δ, x, T
Output: $\{accept\}$ / $\{reject\}$
1: $Setup(\delta, T) \rightarrow pp$
2: $Eval(pp, x) \rightarrow (y, \pi)$
3: **if** $(y, \pi) \leftarrow F(pp, x)$ **then**
4: $\quad Verify(pp, x, y, \pi) \rightarrow \{accept\}$
5: $\quad$ compare y and map to shards
6: **else**
7: $\quad Verify(pp, x, y, \pi) \rightarrow \{reject\}$
8: **end if**

To ensure fairness and security, the coordinating shard generation process must rely on the verifiability of node identities, as in Elastico [25] and OmniLedger [21]. This paper uses a random number generation method based on Verifiable Delay Function (VDF) [6] to generate unique user identities. VDF is a function: $F : X \rightarrow Y$, each input $x \in X$ must have a unique valid output $y \in Y$. The random numbers generated based on VDF can be publicly verified throughout the entire network. Furthermore, VDF does not require reliance on powerful computing hardware for calculations. This means that nodes will not exhibit inconsistency in calculation efficiency due to differences in hardware equipment. This consistency in calculation time is maintained across various sub-region networks or hardware conditions, ensuring that no one has advance knowledge of the calculation results due to variations in computing equipment performance. This, in turn, guarantees both security and fairness.

As shown in lines 1-2 of Algorithm 1, when assigning user nodes, each node in the subregions takes a security parameter δ and a time limit, T, and outputs the common parameter, pp. Then, they locally take a random input, x, and compute the function $Eval(pp, x)$ to obtain a random output, y, and a proof, π. Lines 3-8 represent the validation phase, in which validation nodes verify the random output and the proof and perform $Verify(pp, x, y, \pi) \rightarrow \{accept, reject\}$. If y is a correct evaluation of the input x, the verification passes. After obtaining the random number, each node discloses its own random number, y, where $y < 2^d$, and adds the node that generated the valid random number, y, to the coordinating shard, where d is a predetermined parameter in the network, i.e., computational difficulty. After creating the coordinating shard, it collects data from each user, device, and sensor node and sends it to the relevant regional shard based on its location. Due to the uneven growth of each region, during the allocation process, the number of user nodes in a particular slice may exceed the threshold, K. In the initialization of the *Co-Sharding*, these nodes can be assigned to some sub-regional networks with a small number of nodes in the vicinity to keep the number of nodes in each sub-chain balanced. Finally, a blockchain sharding-based LIoT system is formed, where each sub-IoT network manages only its regional data ledger.

3.2 Coordinating-shard-driven Cross-region Interaction

In the process of cross-regional ledger interaction, data from different regional shards needs to be queried to ensure consistency. Meanwhile, to avoid the high communication and computation costs that users or nodes need to bear in the existing cross-sharding scheme, this subsection introduces a coordinating-shard-driven two-phase commit approach for data interaction and computation, where the sub-regions in the cross-region interaction interact only with the coordinating shard each time without the need for the sub-regions to communicate with each other, i.e., the communication costs and computation pressure are shared by each node of the coordinating shard.

Unlike OmniLedger [21] and RapidChain [38], which rely exclusively on decentralization and randomness, multiple subregional networks are divided into nodes based on geographic locations, which are regarded as multiple regional shards. As a result, most regional shards are synchronous networks and data validation is achieved using synchronous Byzantine consensus with $1/2$ fault tolerance. In contrast, in a coordinating shard, nodes come from different sub-regions, the network is not synchronized, and there is a possibility that a particular regional node may act evilly for its benefit. Therefore, distributed Byzantine fault-tolerant consensus ensures the security of user identity information and state. In addition, the coordinating shard uses the PBFT consensus mechanism, which can tolerate f malicious nodes (DoS attacks, forgery, data tampering, etc.), where $f < k/3$, k denotes the number of nodes in a single regional shard. Also, the signature aggregation process in the network uses a threshold signature technique based on BLS [7] to reduce the overall communication overhead.

In the process of cross-region data interaction, data interaction and validation between different Sub-IoTs are coordinated through the coordinating shard, which inputs the cross-region task requests $o_{cr} \in o^T$ with timestamps t and client identifiers c in period T and outputs the final execution and validation results. Where Eq.(1) and (2) are the input of task request In_{cr} and the output of confirmation commitment Out_{cr}, the functions *Request()* and *Commit()* are the request and commitment of the task, and p is the proof of the consensus execution result.

$$In_{cr} = (Request(o_{cr})||t||c) \tag{1}$$

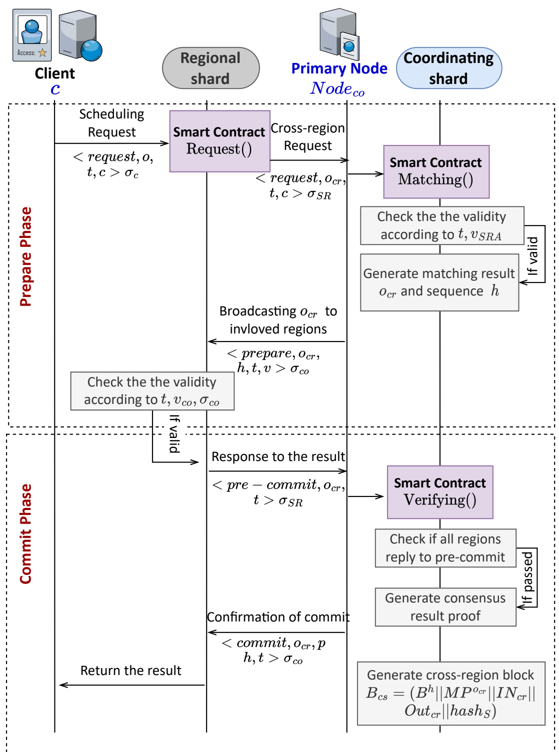

Fig. 2. Coordinating-shard-driven cross-region interaction protocol.

$$Out_{cr} = (Commit(o_{cr})||t||p||v||c) \tag{2}$$

Eq.(3) defines the cross-region block in the coordinating shard, and such a block proves the authenticity and validity of the task execution results with the block in the sub-region verification phase. B^h denotes the hash value of the block header containing the list of transactions, $MP^{o_{cr}}$ is the Merkle proof, and $hash_S$ is the hash value of the set of tasks S.

$$o_{cr} = (B^h||MP^{o_{cr}}||In_{cr}||Out_{cr}||hash_S) \tag{3}$$

For the result of task execution or the new block, all sub-regions participating in the cross-region task are required to return *commit* message with valid signatures, and the whole protocol is finally recognized as committed, while the rest of the cases are regarded as *reject*, as shown in Eq.(4), and the results of the message m_r of the honest primary node $\varphi_r.honest$ in each sub-region r need to all be committed, where p_r is the region proof and v_r is the view number.

$$\forall r \in R, \varphi_r.honest \land m_r \Rightarrow (o_{cr}||p_r||v_r||t) = committed \tag{4}$$

The algorithm flow is shown in Algorithm 2 and Fig. 2. For example, the cross-region interaction involving Sub-region 1 (SR1) and Sub-region 2 (SR2) has the following phases, where the 2PC mainly refers to the *prepare* and *commit* phases:

Algorithm 2 Cross-region data interaction

Input: $r =< request, o, t, c > \sigma_c$
Output: {*commit*} / {*reject*}
1: **upon receiving** $r =< request, o, t, c > \sigma_c$ from client c
2: **multicast** $r =< request, o_{cr}, t, c, v > \sigma_{SR1}$ to coordinating shard
3: **upon receiving** $< prepare, o_{SR1}, h_i, o_{SR2}, h_j, t, v > \sigma_{co}$ from coordinating shard
4: **if** t, v is correct **then**
5: **if** σ_{co} is valid and o_{cr} is not involved in other requests **then**
6: **multicast** $< pre-commit, o_{SR1}, t > \sigma_{SR1}$ and $< pre-commit, o_{SR2}, t > \sigma_{SR2}$ to coordinating shard
7: **if** Both SR1 and SR2 reply to the *pre-commit* message **then**
8: **if** $n_v > 2k/3$ **then**
9: **multicast** $< commit, o_{cr}, p_{co}, p_i, p_j, t, v > \sigma_{co}$ to SR1 and SR2
10: **else**
11: **multicast** $< reject, o_{cr}, p_{co}, p_i, p_j, t, v > \sigma_{co}$ to SR1 and SR2
12: **end if**
13: **end if**
14: **end if**
15: **else**
16: ViewChange= $< viewChange, t, V >$
17: **end if**
18: return {*commit*} or {*reject*} to c

1) Request Phase (lines 1-2): The user or client initiates a cross-region task request (denoted o_{cr}) to SR1. Upon validation by SR1, the current primary node performs local validation and invokes the smart contract for task validation. Since this process involves inputs or outputs from multiple sub-regions, local validation cannot be completed, resulting in an error in the validation result. Subsequently, the primary node broadcasts the execution result within the local sub-region, and the shared nodes within the coordinating shard transmit data and task requests, including region identifiers, to the coordinating shard network.
2) Prepare phase (line 3): The coordinating shard network performs global validation of the cross-region tasks, the primary node of coordinating shard invokes the smart contract to execute the cross-region decision, locates the relevant regional shard (SR1, SR2), and divides the task o_{cr} into o_1 and o_2, then assigns a sequence number h, timestamp t, and view number v to the target shard respectively.
3) Pre-commit phase (line 4-6): The two sub-regional networks execute a round of intra-shard consensus, i.e., local validation, where the validation node decodes the matching results, checks the scheduling information and the proof of the smart contract matching, and also verifies the validity of the message signature σ_{co} as well as its attached parameters. Sub-IoT also verifies that the cross-region task is not in conflict with the current intra-region task and will reach a consensus on crash tolerance within a shard with at least a quorum of $f + 1$ agreeing. If the validation is unsuccessful, they will sign and send a *reject* message to the primary node of the coordinating shard.
4) Commit phase (line 7-14): SR1 and SR2 return the local consensus results with signatures to the coordinating shard to initiate a consensus round and $node_{co}$ initiates a consensus round while attaching the proof of consensus p in the regional network to the message. $Node_{co}$ waits for $n_v \geq 2f + 1$ (i.e., $2k/3$) *commit* results from the coordinating shard nodes, where n_v denotes the number of *commit* messages. The sequence number, timestamp,

view number, and signature are checked for consistency and validity. Finally, the {*commit*} / {*reject*} consensus result is sent to the relevant region.

5) Confirmation phase (line 15-18): the result of task execution is confirmed as {*commit*} is finally returned to the client. Meanwhile, the nodes in the coordinating shard select a new primary node at each time interval T_i. If the current primary node intends to tamper with the data or crashes, it will result in a view change, which means replacing the primary node.

3.3 Security Analysis

The security of *Co-Sharding* in cross-region interaction includes safety and liveness, thus we provide the following two theorems and proofs for security analysis.

Definition 1: Safety represents that honest nodes agree to the same valid blocks in each round, while liveness indicates that each new block will be committed or rejected at the end of each round.

Theorem 1: If there are no more than $l < 1/2$ fraction of malicious nodes in each regional shard, and the proportion of malicious nodes in the coordinating shard does not exceed $m < 1/3$, then the cross-region interaction process achieves safety.

Proof: Assuming that the number of malicious nodes in each regional shard does not exceed half of the total number of regional nodes, then more than half of the messages and signatures are valid and exceed half. Therefore, internal consensus can ensure that the proposed block is valid. Meanwhile, messages cannot be modified or forged, as aggregate signatures can be used to detect forgery and tampering. Similarly, the PBFT used in the coordinating shard has 1/3 fault tolerance. When the proportion of malicious nodes in the coordinating shard does not exceed 1/3, its internal consensus can ensure validity. Therefore, communication between shards can be carried out safely, ensuring that all relevant regional shards can receive valid cross-regional blocks. In the proposed 2PC protocol, the Prepare phase aims to reach a temporary commitment agreement for cross-regional transactions, and the Commit phase aims to execute the actual commitment of transactions between involved regional shards. Therefore, all relevant honest nodes in the regional shards and the coordinating shard agree on the same valid cross-regional block in each round, i.e., consensus achieves safety.

Theorem 2: The cross-region interaction achieves liveness if there are no more than $l < 1/2$ fraction of malicious nodes in the regional shard and no more than $m < 1/3$ fraction of malicious nodes in the coordinating shard.

Proof: According to the proposed scheme, nodes in the regional shard are connected through synchronous networks, and there are no more than $l < 1/2$ malicious nodes per shard. Similarly, nodes in the coordinating shard are connected through partial synchronous networks, and there are no more than $m < 1/3$ malicious nodes. Therefore, using the BFT protocol as the consensus within each shard can achieve liveness. According to *Theorem* 1, each shard is consistent on the same block in each round. Therefore, no malicious node can successfully complete an attack on *Co-Sharding*.

4 CASE STUDY

4.1 Experimental Scenario

This paper is grounded in various real-world agricultural Sub-IoT scenarios, providing access to data related to farm machinery, fields, and more through sub-regional agri-smart platforms. In these scenarios, some regions experience scheduling goals that necessitate higher resource utilization due to an excessive number of agricultural machines. Conversely, certain regions grapple with resource scarcity, compelling the scheduling production process to prioritize cost reduction and increasing farmers' income. Consequently, this paper introduces a cross-regional agricultural machinery IoT scheduling model that takes into account the scheduling requirements of multiple regions, encompassing both macro-level benefits (scheduling path length and machinery utilization) and micro-level individual benefits (farmers' net present value). Given the inherent trade-offs among these three objectives, the Non-Dominated Neighborhood Immunity Algorithm (NNIA) [15] is employed to identify Pareto optimal solutions for the smart contract scheduling algorithm.

We designed a scheduling model for each agricultural machinery sub-IoT based on the experimental scenarios. First, as shown in Eq. (5), the scheduling path objective function was developed to reduce the distance cost of machinery scheduling. The model is shown below.

$$F_1 = min(\sum_{j=1}^{n} L(\mathrm{Ma}, T_j)\beta + \sum_{j=1}^{n} L(\mathrm{Ma}, T_j)\eta + \sum_{j=1}^{n}\sum_{k=1}^{n} L(\mathrm{Ma}, T_j T_k)\theta) \tag{5}$$

In Eq. (5), the distance of agricultural machinery Ma from the starting depot to the first task T_1 of cross-region scheduling is $L(\mathrm{Ma}, T_1)$, where $j = 1$, the distance of machinery Ma from its last task T_n to the termination warehouse is $L(\mathrm{Ma}, T_n)$, where $j = n$, the distance from task T_j to the next task T_k is $L(\mathrm{Ma}, T_j T_k)$, where $k - j = 1$. The meanings of β, η, θ are shown in Eq. (6)

$$\beta = \begin{cases} 0, (T_j \neq T_1) \\ 1, (T_j = T_1) \end{cases}, \eta = \begin{cases} 0, (T_j \neq T_n) \\ 1, (T_j = T_n) \end{cases}, \theta = \begin{cases} 0, (k - j \neq 1) \\ 1, (k - j = 1) \end{cases} \tag{6}$$

In Eq. (6), when T_j is the first task, β is 1, otherwise it is 0. When T_j is the last task, η is 1, otherwise, it is 0. When task T_j is a task before task T_k, θ is 1, otherwise it is 0.

Next, the objective function of agricultural machinery utilization within a specific set of periods $t_s \in \mathcal{T}_f$ (e.g., 1 hour, 2 hours) is shown in Eq. (7), where $\mathcal{T}_f$ represents overall scheduling time.

$$F_2 = max \sum_{t_s \in \mathcal{T}_f} (\frac{M_w}{M_w + M_u}) \tag{7}$$

In Eq. (7), M_w is the total number of agricultural machines that were in use throughout the period t_s, and M_u is the total number of machines that were in use over the entire period.

This paper also develops the mathematical model shown in Eq. (8) to maximize the total present value of farmers.

$$F_3 = max \sum_{t_s \in \mathcal{T}_f} \sum_{j=1}^{n} (\frac{IN_j - C_j}{1 + \alpha}) \tag{8}$$

In Eq. (8), α denotes the discount rate, the total NPV of a farmer depends on a set of revenues IN_j and costs C_j over a specific period t_s. The income is modeled as shown in Eq. (9).

$$\sum_{t_s \in \mathcal{T}_f} \sum_{j=1}^{n} IN_j = \sum_{t_s \in \mathcal{T}_f} \sum_{j=1}^{n} P_j S_j = \sum_{t_s \in \mathcal{T}_f} \sum_{j=1}^{n} \gamma A_j S_j \tag{9}$$

The model assumes that there is a ratio between farm productivity P_j and the area of farmland A_j, denoted by γ, and the value of which varies with the crop. Therefore, in Eq. (9), the farmer's income IN_j is calculated based on A_j and the selling price of individual crops S_j.

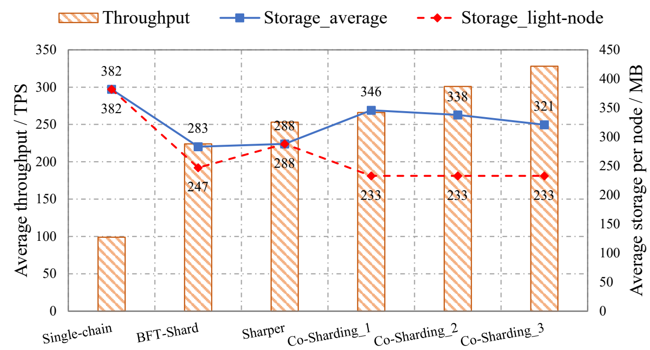

Fig. 3. Comparison of throughput and storage overhead.

$$\sum_{t_s \in \mathcal{T}_f} \sum_{j=1}^{n} C_j = \sum_{t_s \in \mathcal{T}_f} \sum_{j=1}^{n} (C_{op} + C_{Tr}) = \sum_{t_s \in \mathcal{T}_f} \sum_{j=1}^{n} (A_j \cdot a + L(\mathrm{Ma}, T_j) \cdot q) \tag{10}$$

Meanwhile, the scheduling model considers the farmer's expenditure costs, including transportation costs C_{Tr} and the operation cost C_{op} of machinery. Its calculation is shown in Eq. (10). To simplify the model, it is assumed that the unit path cost q and unit area operation cost a are the same for all farm machinery used in the scheduling operation.

4.2 Experimental Settings

This section presents experimental details for evaluating the effectiveness of *Co-Sharding* and comparing it with previous studies. The experimental environment for *Co-Sharding* utilizes an Intel(R) Core(TM) i7-9700 CPU running at 3.00 GHz, hosted on Ubuntu 22.04, with a Hyperledger Fabric 2.2 platform for emulation. Within each agricultural Sub-IoT system, a dedicated channel is established for internal communication and internal scheduling. Additionally, a coordinating shard channel is created to facilitate cross-region coordination. During testing, users and customers in the region connect to the agricultural machinery management organization nodes based on real-world scheduling scenarios. To evaluate *Co-Sharding's* performance, multiple sub-regional organization nodes are deployed, with each node connecting to 10 clients, unmanned devices, and sensor nodes within the region. In these experiments, the block size is consistently set to approximately 10 MB. The scheduling model and matching algorithm are implemented using the Go language [10], and scheduling smart contracts are written and deployed on the *Co-Sharding* nodes.

4.3 Experimental Analysis

First, to verify the optimization of Co-Sharding in terms of performance as well as storage overhead, we conduct experiments on the storage overhead of *Co-Sharding* over 10 seconds. In our experiments, we also increase the number of coordinating shards to verify their impact on storage overhead. We test the average storage size of nodes under the conditions of 1, 2, and 3 coordinating shards and 6 sub-regional shards and compare it with a single chain, BFT-Shard [11] and Sharper [2]. The experimental results are shown in Fig. 3. Compared to the single-chain system, *Co-Sharding* improves the performance by about 1.5 times and reduces the storage overhead by about 39% for light nodes in scheduling. At the same time, *Co-Sharding* has higher node storage overhead compared to fully sharded Sharper and BFT-Shard. This is because nodes in a coordinating shard need to store additional cross-region scheduling data, and high

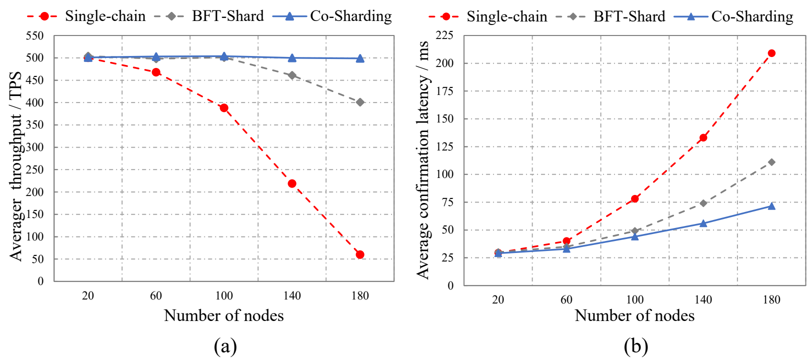

Fig. 4. Comparison of cross-region data retrieval performance with workload. *(a) Throughput (b) Latency

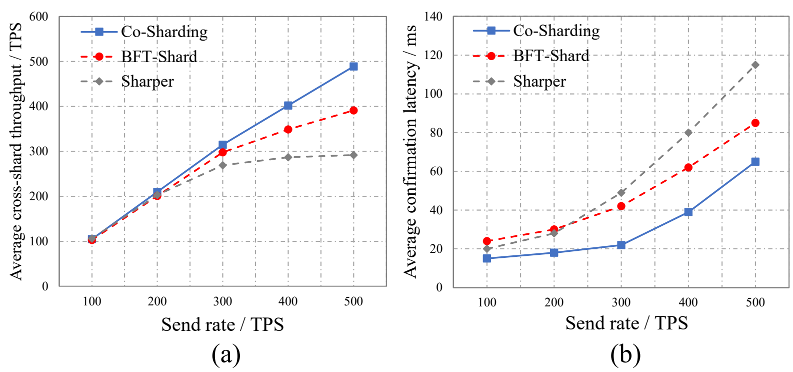

Fig. 5. Comparison of cross-region data retrieval performance with the number of nodes. *(a) Throughput (b) Latency

throughput means more data needs to be stored. As the number of coordinating shards increases, the computational pressure of cross-region scheduling is shared among multiple coordinating shards, which reduces the storage overhead required by the nodes.

To analyze the performance bottleneck of the single-chain system, we conduct experiments measuring data query throughput at different user and client node sizes while maintaining a fixed sending rate of 500 TPS. The results are then compared with *Co-Sharding*, as depicted in Fig 4. The test results reveal a significant performance drop in the single-chain system when the number of nodes reaches 100. In contrast, *Co-Sharding* demonstrates substantial performance improvements, achieving approximately 10 times the throughput of the single-chain when the number of nodes reaches 180. Additionally, *Co-Sharding* exhibits only one-third of the latency compared to the single-chain. Furthermore, due to optimized cross-region scheduling, *Co-Sharding* outperforms BFT-Shard in terms of overall performance.

Two cross-ledger approaches used in BFT-Shard and Sharper are also reproduced in the experiments on the Hyperledger Fabric platform and applied to the cross-region scheduling scenario in this paper. The former handles cross-sharding transactions through an independent reference committee, consisting of a set of randomly selected nodes not included in each regional shard. The latter is a cross-sharding approach under complete sharding, i.e.,

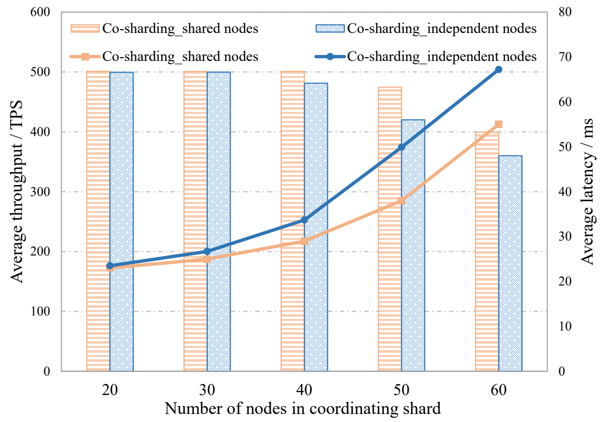

Fig. 6. Comparison of performance with the number of nodes in coordinating shard.

regional shards communicate among themselves to handle cross-region scheduling. In the implementation process, we realize cross-sharding interaction under complete sharding by introducing the Pubsub platform to deploy relevant splitting channels. In the comparison process, *Co-Sharding* and Sharper deploy six regional shards, where *Co-Sharding* additionally establishes three coordinating shards consisting of regional service organization nodes to realize parallel querying across ledgers. The comparison results are shown in Fig. 5. The results show that the cross-ledger data query performance of *Co-Sharding* outperforms the other two approaches in terms of performance. Although Sharper performs similarly to *Co-Sharding* at low send rate, its implementation process is more complicated. Due to the introduction of Pubsub, the complexity of the system increases, and the requirements for hardware equipment conditions are higher. Its performance is weaker when the send rate is increased. Meanwhile, the method adopted by BFT-Shard has similar performance results with the scheme proposed in this paper using a single coordinating shard. This similarity arises mainly because the experimental environments and hardware devices of the schemes are similar, and the cross-shard computation method is based on a single shard.

To verify the effectiveness of building coordinating shards through nodes shared with regional shards, we verify the transaction execution efficiency of coordinating shards and compare it with the solution of coordinating shards built through independent nodes, the result is shown in Fig. 6. First, we adopt the *Co-Sharding* scheme to select multiple nodes from each region to join a coordinating shard, and then construct a coordinating shard consisting of independent (not belonging to any region) nodes as a comparison scheme. We then evaluate the performance by varying its internal node count. As can be seen from figure 6, the shared node can read regional data locally, avoiding the additional cross-ledger data retrieval computational overhead required by independent nodes, so the throughput and latency perform better as the number of nodes increases. It can also be seen from Fig. 6 that as the number of nodes increases, the performance will become worse and worse. This is mainly due to the large communication overhead of PBFT used in the coordinating shard. When the network conditions are poor, there may even be a crash when the number of nodes is about 100. Therefore, its consensus method can be improved in the future to achieve better scalability.

To verify the scheduling enhancement effect on scheduling time and resource utilization after cross-region scheduling, a smart contract for scheduling agricultural machines is created and deployed in *Co-Sharding* by combining it with the

Table 1. Comparison results of *Co-sharding* and centralized scheduling scheme.

Number of tasks	*Co-Sharding* for cross-region scheduling		Centralized intra-region scheduling		Utilization improvement rate/%
	Average time of algorithm running/s	Utilization rate/%	Average time of algorithm running/s	Utilization rate/%	
8	2.344	74	1.569	55	34.5
12	2.503	81	1.692	62	30.6
16	2.655	92	2.067	73	26.0

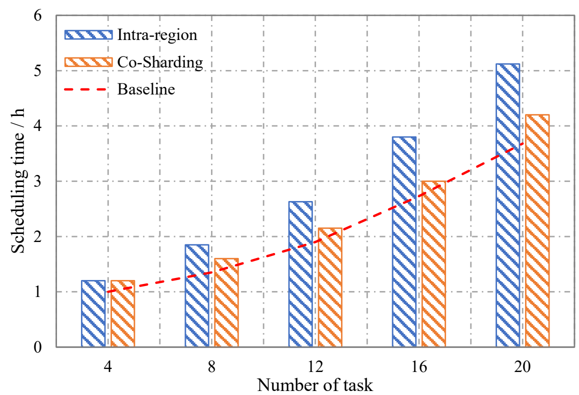

Fig. 7. Comparison of scheduling time.

NNIA multi-objective optimization algorithm in six regions. The running time of the contract and the utilization rate of the agricultural machines after generating the policy are tested in the experiments and compared with the centralized intra-region scheduling model that achieved good results in intra-region scheduling [14] (see Table 1). Although *Co-Sharding* has a longer execution time for the smart contract-based scheduling model and algorithm compared to the centralized scheme, the utilization rate of machinery in *Co-Sharding*-based cross-region fusion scheduling is high, especially when the number of tasks is small. This also proves the advantage of cross-region scheduling for resource utilization. In addition, we test the scheduling time when the number of tasks varies in real scenarios and compared it with traditional scheduling within a sub-region, as shown in Fig. 6. In this case, we assume 15 minutes of operating time per 600 square meters of field and 30 minutes per kilometer of farm machinery. The results show that when the number of tasks increases, the subsequent tasks must wait for the end of the previous tasks before they can be performed due to the limited number of farm machines within the region. On the contrary, when executing the *Co-Sharding*-based scheduling strategy, the sub-region can utilize the farm machines in other regions at any time, so the scheduling time decreases and becomes more significant as the number of tasks increases. The baseline in the figure represents the actual distance traveled as well as the operating time. Thus, based on the baseline, intra-region scheduling requires longer waiting times.

In summary, the conducted experiments show the effectiveness of *Co-Sharding* in improving blockchain scalability. Comparative tests reveal that *Co-Sharding* optimizes cross-region scheduling and enhances data query throughput.

It is important to note that the current experiments are conducted only in two regions, which introduces certain limitations to the comparative performance results. If the number of regions increases, the presence of shared nodes in the coordinating shard would theoretically offer even greater advantages by avoiding multiple rounds of consensus across various regions, thus further enhancing the performance metrics.

5 CONCLUSION

In this paper, we introduce *Co-Sharding*, a novel approach designed to enhance data processing performance and ledger storage capacity in LIoT applications through sharding techniques. In this scheme, *Co-Sharding* treats the ledgers maintained by Sub-IoTs in different regions as individual shards within the overall LIoT ledger. Each region elects management nodes to maintain a coordinating shard, responsible for managing cross-regional communication and data interactions. Additionally, consistency among the regional shards is guaranteed by a coordinating-shard-driven Two-Phase Commit (2PC) protocol. We then proceed to implement *Co-Sharding* within the HyperLedger Fabric framework and propose a cross-region scheduling strategy based on a multi-shard multi-objective optimization algorithm. Our experiments involve a cross-region agricultural machinery IoT management system utilizing real operational data. The comparative results demonstrate the benefits of *Co-Sharding*: it increases machine utilization by approximately 30% with 12 tasks and reduces scheduling time by around 18% compared to intra-sub-region scheduling. Moreover, compared to a single chain, *Co-Sharding* reduces the storage capacity required for each regional light node by about 39% while improving data query throughput by roughly 1.5 times. In future work, we will explore improving the communication overhead of consensus methods in the coordinating shard and implementing our prototype system in multiple regions.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China (Grant No. 2022YFD2100605), and the Leading-edge Technology Program of Jiangsu Natural Science Foundation (Grant No. BK20202001).

REFERENCES

[1] Shabir Ahmad and DoHyeun Kim. 2020. A multi-device multi-tasks management and orchestration architecture for the design of enterprise IoT applications. *Future Generation Computer Systems* 106 (2020), 482–500. https://doi.org/10.1016/j.future.2019.11.030

[2] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2021. Sharper: Sharding permissioned blockchains over network clusters. In *Proceedings of the 2021 international conference on management of data*. 76–88. https://doi.org/10.1145/3448016.3452807

[3] Gagangeet Singh Aujla and Anish Jindal. 2020. A decoupled blockchain approach for edge-envisioned IoT-based healthcare monitoring. *IEEE Journal on Selected Areas in Communications* 39, 2 (2020), 491–499. https://doi.org/10.1109/JSAC.2020.3020655

[4] Kamran Ahmad Awan, Ikram Ud Din, Ahmad Almogren, Hisham Almajed, Irfan Mohiuddin, and Mohsen Guizani. 2020. NeuroTrust—artificial-neural-network-based intelligent trust management mechanism for large-scale internet of medical things. *IEEE Internet of Things Journal* 8, 21 (2020), 15672–15682. https://doi.org/10.1109/JIOT.2020.3029221

[5] Rafael Belchior, André Vasconcelos, Sérgio Guerreiro, and Miguel Correia. 2021. A survey on blockchain interoperability: Past, present, and future trends. *Comput. Surveys* 54, 8 (2021), 1–41. https://doi.org/10.1145/3471140

[6] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. 2018. Verifiable delay functions. In *Annual international cryptology conference*. Springer, 757–788. https://doi.org/10.1007/978-3-319-96884-1_25

[7] Dan Boneh and Xavier Boyen. 2008. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of cryptology* 21, 2 (2008), 149–177. https://doi.org/10.1007/s00145-007-9005-7

[8] Ting Cai, Wuhui Chen, Kostas E Psannis, Sotirios K Goudos, Yang Yu, Zibin Zheng, and Shaohua Wan. 2022. Scalable on-chain and off-chain blockchain for sharing economy in large-scale wireless networks. *IEEE Wireless Communications* 29, 3 (2022), 32–38. https://doi.org/10.1109/MWC.004.2100616

[9] Yuling Chen, Yufeng Li, Qi Chen, Xianmin Wang, Tao Li, and Chaoyue Tan. 2023. Energy trading scheme based on consortium blockchain and game theory. *Computer Standards & Interfaces* 84 (2023), 103699. https://doi.org/10.1016/j.csi.2022.103699

[10] Russ Cox, Robert Griesemer, Rob Pike, Ian Lance Taylor, and Ken Thompson. 2022. The Go programming language and environment. *Commun. ACM* 65, 5 (2022), 70–78. https://doi.org/10.1145/3488716

[11] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin, and Beng Chin Ooi. 2019. Towards scaling blockchain systems via sharding. In *Proceedings of the 2019 international conference on management of data*. 123–140. https://doi.org/10.1145/3299869.3319889

[12] BD Deebak, Fida Hussain Memon, Kapal Dev, Sunder Ali Khowaja, Weizheng Wang, and Nawab Muhammad Faseeh Qureshi. 2022. TAB-SAPP: A trust-aware blockchain-based seamless authentication for massive IoT-enabled industrial applications. *IEEE Transactions on Industrial Informatics* 19, 1 (2022), 243–250. https://doi.org/10.1109/TII.2022.3159164

[13] Kuan Fan, Zhuoxuan Liu, Mingxi Liu, Yihong Wen, Ning Lu, and Wenbo Shi. 2022. Cross-Chain Data Auditing for Medical IoT Data Sharing. In *International Conference on Security and Privacy in New Computing Environments*. Springer, 65–81. https://doi.org/10.1007/978-3-031-30623-5_5

[14] Pengfei He, Jing Li, and Xin Wang. 2018. Wheat harvest schedule model for agricultural machinery cooperatives considering fragmental farmlands. *Computers and Electronics in Agriculture* 145 (2018), 226–234. https://doi.org/10.1016/j.compag.2017.12.042

[15] Li Huang, MengChu Zhou, and Kuangrong Hao. 2019. Non-dominated immune-endocrine short feedback algorithm for multi-robot maritime patrolling. *IEEE Transactions on Intelligent Transportation Systems* 21, 1 (2019), 362–373. https://doi.org/10.1109/TITS.2019.2892377

[16] Uzair Javaid and Biplab Sikdar. 2020. A checkpoint enabled scalable blockchain architecture for industrial internet of things. *IEEE Transactions on Industrial Informatics* 17, 11 (2020), 7679–7687. https://doi.org/10.1109/TII.2020.3032607

[17] Xuedan Jia, Liangmin Wang, Ke Cheng, Pujie Jing, and Xiangmei Song. 2023. A blockchain-based Privacy-Preserving and Collusion-Resistant scheme (PPCR) for double auctions. *Digital Communications and Networks* (2023). https://doi.org/10.1016/j.dcan.2023.05.002

[18] Yutao Jiao, Ping Wang, Dusit Niyato, and Kongrath Suankaewmanee. 2019. Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks. *IEEE Transactions on Parallel and Distributed Systems* 30, 9 (2019), 1975–1989. https://doi.org/10.1109/TPDS.2019.2900238

[19] Veeramani Karthika and Suresh Jaganathan. 2023. iLEDGER: A lightweight blockchain framework with new consensus method for IoT applications. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* (2023), 2022EAP1088. https://doi.org/10.1587/transfun.2022EAP1088

[20] Abdullah Ayub Khan, Asif Ali Laghari, Thippa Reddy Gadekallu, Zaffar Ahmed Shaikh, Abdul Rehman Javed, Mamoon Rashid, Vania V Estrela, and Alexey Mikhaylov. 2022. A drone-based data management and optimization using metaheuristic algorithms and blockchain smart contracts in a secure fog environment. *Computers and Electrical Engineering* 102 (2022), 108234. https://doi.org/10.1016/j.compeleceng.2022.108234

[21] Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. 2018. Omniledger: A secure, scale-out, decentralized ledger via sharding. In *2018 IEEE symposium on security and privacy (SP)*. IEEE, 583–598. https://doi.org/10.1109/SP.2018.000-5

[22] Srinivas Koppu, Siva Rama Krishnan Somayaji, Iyapparaja Meenakshisundaram, Weizheng Wang, and Chunhua Su. 2022. Fusion of blockchain, IoT and artificial intelligence-A survey. *IEICE Transactions on Information and Systems* 105, 2 (2022), 300–308. https://doi.org/10.1587/transinf.2021BCR0001

[23] Kai Lin, Jian Gao, Guangjie Han, Haohua Wang, and Chao Li. 2022. Intelligent blockchain-enabled adaptive collaborative resource scheduling in large-scale industrial internet of things. *IEEE Transactions on Industrial Informatics* 18, 12 (2022), 9196–9205. https://doi.org/10.1109/TII.2022.3169457

[24] Xiaoyan Liu, Xinmeng Zhu, and Kuangrong Hao. 2021. Dynamic immune cooperative scheduling of agricultural machineries. *Complex & Intelligent Systems* 7 (2021), 2871–2884. https://doi.org/10.1007/s40747-021-00454-2

[25] Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert, and Prateek Saxena. 2016. A secure sharding protocol for open blockchains. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 17–30. https://doi.org/10.1145/2976749.2978389

[26] A Orfanou, Patrizia Busato, DD Bochtis, Gareth Edwards, D Pavlou, CG Sørensen, and Remigio Berruto. 2013. Scheduling for machinery fleets in biomass multiple-field operations. *Computers and Electronics in Agriculture* 94 (2013), 12–19. https://doi.org/10.1016/j.compag.2013.03.002

[27] BRENDA V Ortiz, KB Balkcom, L Duzy, E Van Santen, and DL Hartzog. 2013. Evaluation of agronomic and economic benefits of using RTK-GPS-based auto-steer guidance systems for peanut digging operations. *Precision agriculture* 14 (2013), 357–375. https://doi.org/10.1007/s11119-012-9297-y

[28] Wei Ou, Shiying Huang, Jingjing Zheng, Qionglu Zhang, Guang Zeng, and Wenbao Han. 2022. An overview on cross-chain: Mechanism, platforms, challenges and advances. *Computer Networks* 218 (2022), 109378. https://doi.org/10.1016/j.comnet.2022.109378

[29] Mohsin Ur Rahman, Fabrizio Baiardi, and Laura Ricci. 2020. Blockchain smart contract for scalable data sharing in IoT: A case study of smart agriculture. In *2020 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT)*. IEEE, 1–7. https://doi.org/10.1109/GCAIoT51063.2020.9345874

[30] Xiao Tang, Xunqiang Lan, Lixin Li, Yan Zhang, and Zhu Han. 2022. Incentivizing Proof-of-Stake Blockchain for Secured Data Collection in UAV-Assisted IoT: A Multi-Agent Reinforcement Learning Approach. *IEEE Journal on Selected Areas in Communications* 40, 12 (2022), 3470–3484. https://doi.org/10.1109/JSAC.2022.3213360

[31] Petr Vopěnka, Jan Kašpar, and Robert Marušák. 2015. GIS tool for optimization of forest harvest-scheduling. *Computers and Electronics in Agriculture* 113 (2015), 254–259. https://doi.org/10.1016/j.compag.2015.03.001

[32] Xiaoding Wang, Sahil Garg, Hui Lin, Md Jalil Piran, Jia Hu, and M Shamim Hossain. 2021. Enabling secure authentication in industrial iot with transfer learning empowered blockchain. *IEEE Transactions on Industrial Informatics* 17, 11 (2021), 7725–7733. https://doi.org/10.1109/TII.2021.3049405

[33] Jinyuan Xu, Baoxing Gu, and Guangzhao Tian. 2022. Review of agricultural IoT technology. *Artificial Intelligence in Agriculture* 6 (2022), 10–22. https://doi.org/10.1016/j.aiia.2022.01.001

[34] Hiroshi Yamamoto, Shota Nishiura, and Yoshihiro Higashiura. 2023. Wide-Area and Long-Term Agricultural Sensing System Utilizing UAV and Wireless Technologies. *IEICE TRANSACTIONS on Information and Systems* 106, 5 (2023), 914–926. https://doi.org/10.1587/transinf.2022NTI0001

[35] Haotian Yang, Liangmin Wang, Lu Liu, Longxia Huang, and Chenbo Ding. 2022. Model construction and application of agricultural ma-chinery cross-region scheduling based on blockchain. *Model construction and application of agricultural ma-chinery cross-region scheduling based on blockchain.* 38, 11 (2022), 31–40.

[36] Lichao Yang, Ming Li, Heli Zhang, Hong Ji, Mingyan Xiao, and Xi Li. 2020. Distributed resource management for blockchain in fog-enabled IoT networks. *IEEE Internet of Things Journal* 8, 4 (2020), 2330–2341. https://doi.org/10.1109/JIOT.2020.3028071

[37] Yinghong Yang, Fenhua Bai, Zhuo Yu, Tao Shen, Yingli Liu, and Bei Gong. 2023. An Anonymous and Supervisory Cross-Chain Privacy Protection Protocol for Zero-Trust IoT Application. *ACM Transactions on Sensor Networks* (2023). https://doi.org/10.1145/3583073

[38] Mahdi Zamani, Mahnush Movahedi, and Mariana Raykova. 2018. Rapidchain: Scaling blockchain via full sharding. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 931–948. https://doi.org/10.1145/3243734.3243853

[39] Yushu Zhang, Jiajia Jiang, Xuewen Dong, Liangmin Wang, and Yong Xiang. 2022. BeDCV: Blockchain-Enabled Decentralized Consistency Verification for Cross-Chain Calculation. *IEEE Transactions on Cloud Computing* 11, 3 (2022), 2273–2284. https://doi.org/10.1109/TCC.2022.3196937

[40] Qianqian Zheng, Na Lin, Di Fu, Tianjun Liu, Yuchun Zhu, Xiaochun Feng, and Junhu Ruan. 2023. Smart contract-based agricultural service platform for drone plant protection operation optimization. *IEEE Internet of Things Journal* (2023). https://doi.org/10.1109/JIOT.2023.3288870