Future Generation Computer Systems 158 (2024) 122–137

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Blockchain-based cooperative game bilateral matching architecture for shared storage

Guanjie Lin [a,b], Mingyuan Zeng [a,c], Zhiguang Shan [d], Kaishun Wu [e], Guan Wang [f], Kai Lei [a,*]

[a] Shenzhen Key Lab for ICN and Blockchain Technologies (ICNLAB), Shenzhen Graduate School, Peking University, Shenzhen, 518055, PR China
[b] School of AI, Guangdong and Taiwan, Foshan University, Foshan, 528251, PR China
[c] School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou, 510006, PR China
[d] State Information Center, Beijing, 100038, PR China
[e] DSA Thrust and IoT Thrust, Information Hub, Hong Kong University of Science and Technology (Guangzhou), Guangzhou, 511455, PR China
[f] Shenzhen Data Exchange, Shenzhen, 518045, PR China

ARTICLE INFO

Keywords:
Decentralized shared storage network
Bilateral matching
Decentralized storage
Blockchain

ABSTRACT

The development of IPFS (InterPlanetary File System) and blockchain-based distributed storage projects has brought new possibilities to the field of storage. This paper proposes a blockchain-based cooperative game bilateral matching architecture as a novel approach for shared storage networks. In traditional competitive (non-cooperative) game models, the allocation of storage resources is centered around pricing, leading to a scenario where node providers often engage in price competition to obtain greater rewards, resulting in an imbalance in resource allocation for both buyers and sellers. In contrast, a distributed storage model based on cooperative game theory can better facilitate cooperation and resource sharing among node providers. This paper designs a storage resource allocation algorithm based on the stable marriage matching algorithm, demonstrating the stability of this algorithm as a matching solution. The paper also analyzes the differences between cooperative and non-cooperative game models in the market, and explores an equilibrium pricing mechanism guided by supply and demand. Furthermore, the paper introduces a trading mechanism for storage resources, including publication standards and matching schemes, ensuring efficient and trustworthy interaction between storage suppliers and demanders in a decentralized network centered around storage resources, thus enabling the circulation of the value of storage resources. A prototype of a blockchain-based shared storage trading system is implemented in this paper, utilizing bilateral matching for tradings. System evaluation and experimental testing are conducted, with results showing that the average utility value of the matching trading mechanism proposed in this paper outperforms the Double Auction-based matching model under any Poisson distribution (λ = 0.1, 0.2, 0.3, ..., 0.9) conditions set in the experiments. Additionally, compared to the traditional approach of directly storing complete data content on the chain, the design proposed in this paper effectively reduces on-chain storage consumption by approximately 27.06%.

1. Introduction

With the increasing demand for storage of massive data, distributed shared storage systems have emerged as naturally scalable storage architectures. Although current cloud service providers have built capable storage infrastructures, their overall capacity is still insufficient compared to the total storage space of numerous devices [1]. From a technical perspective, distributed storage nodes scattered in different regions form a distributed shared storage network through replication and secure communication protocols. This network effectively utilizes the idle space of edge devices to provide logically shared external storage services. By allowing any device to contribute its storage resources, distributed shared storage reduces the construction cost of storage infrastructure and amplifies the storage capacity of the entire network. This enables decentralized scalable distributed storage that is low-latency and cost-effective. In terms of interaction mode, users with storage needs and storage devices with idle storage space form a supply and demand relationship around storage resources. Both parties engage in storage tradings through the distributed shared storage network. Thus, the distributed shared storage network in an open environment

* Corresponding author.
E-mail addresses: 20200360323@stu.fosu.edu.cn (G. Lin), 20201003160@gdufs.edu.cn (M. Zeng), shanzg@sic.gov.cn (Z. Shan), wuks@hkust-gz.edu.cn (K. Wu), wangguan@szdex.com (G. Wang), leik@pkusz.edu.cn (K. Lei).

https://doi.org/10.1016/j.future.2024.04.016
Received 29 November 2023; Received in revised form 4 March 2024; Accepted 10 April 2024
Available online 15 April 2024

forms a value trading market centered on storage resources. This sharing economy model greatly promotes the effective use and optimal allocation of idle resources [2].

New distributed shared storage systems have many advantages, but building an effective and autonomous shared storage trading market presents challenges. Trust among users and devices in a decentralized network and realizing value circulation of storage resources pose difficulties due to the network's decentralized and dynamic nature. The supply of storage resources requires trust and motivation to contribute from network devices, while consumption depends on users' trust and sufficient storage demand. Maintaining value circulation involves storage-specific tradings. Lack of trust or motivation impedes formation of an attractive and stable resource market. Storage as a tradable value resource is decentralized and dynamic, introducing complex trade-offs for optimal allocation of resources. Achieving automatic and optimal allocation becomes challenging. An effective and autonomous storage trading market is vital to prevent gradual failure of the decentralized network.

Blockchain technology, considered a "Trust Machine", has gained significant attention due to the development and application of digital currencies [3]. The blockchain-based design architecture establishes trust in a decentralized network and maintains value circulation transparently using a network-wide consensus accounting model. The peer-to-peer communication model improves the efficiency of value circulation. With programmability provided by blockchain smart contracts, secure automated tradings or business interactions can occur in a trustless environment. This new global collaboration paradigm under blockchain's open network enables the possibility of Decentralized Autonomous Organizations (DAOs). It drives the formation and development of a blockchain-based internet ecosystem using a sharing economy model [4].

Based on the aforementioned background, this paper focuses on designing a blockchain shared storage trading system to address the challenges of decentralized reliability and scalability in massive data storage. Our aim is to combine the architectural advantages of distributed shared storage and blockchain technology to establish an effective decentralized shared storage network. By integrating trading and matching mechanisms, our work ensure the autonomy of the shared storage market without human intervention. The initial exploration of Trusted Shared Storage Networks (TSSN) [5,6] was presented in our previous work. The research focused on utilizing blockchain technology and a bilateral matching mechanism to establish a reliable shared storage architecture within a distributed storage network. Expanding upon these findings, our current manuscript delves deeper into the TSSN concept by following.

The main contributions are summarized as follows:

- Proposal for a blockchain-based distributed storage architecture using blockchain as the storage system to address privacy and security concerns in data sharing. The paper introduces a bilateral matching storage resource allocation algorithm to ensure stable matches for resource allocation challenges.
- Design of a blockchain-based shared storage trading system to tackle distributed reliability and scalability challenges. By leveraging distributed shared storage and blockchain technology, an effective network is established without human intervention or third-party certification.
- Introduction of a distributed storage model based on cooperative game theory and stable marriage matching algorithms to address resource allocation challenges. The research aims to foster long-term cooperative relationships among node providers by combining on-chain and off-chain resource coordination protocols.
- Proposal for a shared storage architecture and storage resource trading mechanism based on blockchain to address management and liquidity issues. The paper suggests a blockchain storage resource trading mechanism using storage orders as the fundamental unit for subsequent storage service matching.

The remainder of this paper is organized as follows: Section 2 outlines related work in our research area and the research motivation. Section 3 introduces the proposed shared storage architecture and trading-based allocation mechanism. Section 4 elaborates on the bilateral matching mechanism incorporated into the proposed architecture. Section 5 presents the experiments and their results. Finally, Section 6 concludes the paper.

2. Background knowledge and related work

2.1. Distributed shared storage

The design and implementation of large-scale storage systems have long been a research hotspot. Two relatively mature general storage system architectures have been developed.

Most of the current cloud storage infrastructures [7,8] adopt traditional distributed system architectures, focusing on the scaling of storage functions and optimization of system performance. By providing unified external storage services through a distributed architecture and centralized management, cloud storage has achieved distribution on physical storage and utilized software-defined centralized management to achieve high efficiency and strong robustness. Although centrally operated centralized distributed storage has certain scalability and performance advantages, it relies on the center's capabilities in terms of reliability and scalability, and the centralized management approach itself has data security risks [9].

Compared with centralized distributed storage, decentralized distributed shared storage further considers the possibility of untrustworthiness among servers in its design idea. OceanStore [10] emphasizes scalability and secure sharing when storing data persistently, and designs a storage infrastructure built by untrusted servers, in which data security is guaranteed by redundancy and encryption techniques. Pond [11], a prototype system for OceanStore, implements more complete secure storage features including decoupling of routing and location, Byzantine fault-tolerant updates, and cached copies. The emergence of the InterPlanetary File System (IPFS) [12] in 2014 has once again pushed distributed shared storage to the research hotspot. IPFS integrates various mature technologies such as P2P networks, BitTorrent transfer protocol, Git version control, and self-certifying file systems [13] to propose a more complete communication protocol and storage network for shared storage. The CISS protocol for individual identification and the CDSS protocol for agreed detection are proposed [14]. These protocols provide a secure method for identifying individual cheaters and detecting cheating behavior in a distributed storage system. The protocols are flexible, universal, and do not require a large finite field. Jingwei [15] has implemented an efficient and adaptive data migration strategy for the distributed storage system. Jingwei aims to reduce data replicas while creating copies for popular files, thereby enhancing service adaptability.

The above literature shows the development and maturity of large-scale distributed storage systems in terms of technical architecture, but most of these studies focus on the integration of existing functions and performance optimization, largely ignoring the impact of the transaction mechanism and incentive mechanism of storage resources on the decentralized storage network: without a clear mechanism to achieve economic incentives, users or devices may not have enough incentives to participate in the storage market.

2.2. Blockchain-based distributed storage architecture

In recent years, distributed storage research based on blockchain is building new autonomous market models [16] by the decentralized trust and incentives [17] provided by the blockchain's ledger technology in terms of digital currency, business logic, etc..

To facilitate the exchange of storage resources and establish a robust autonomous shared storage market, Storj introduced Tokens to incentivize device participation in network activities, including contributing

storage resources and settling expenses through Ethereum smart contracts [1,18,19]. However, Storj's authentication process necessitates users to remain online for extended periods, limiting accessibility. Furthermore, Storj's centralized issuance of storage resources with fixed monthly pricing and opaque pricing mechanisms result in monopolistic control over the storage market, potentially leading to revenue-driven practices like simulated multi-copy storage by nodes. In contrast, Sia enhances storage verification and pricing structures compared to Storj by integrating storage devices into the blockchain network for mining activities, enabling any blockchain node to validate storage while advocating for market-driven pricing based on supply and demand dynamics [20]. Nevertheless, Sia's reliance on the PoW consensus algorithm poses challenges related to computational inefficiency and transaction performance. Although Sia implements a collateral mechanism to deter malicious activities, the low revenue-to-collateral ratio fails to sufficiently incentivize storage contributions. Filecoin, leveraging IPFS as its foundation, establishes a blockchain-based shared storage network and marketplace featuring novel consensus mechanisms like proof of replication and proof of Spacetime to address issues such as node manipulation and inefficiencies associated with PoW [21]. While Filecoin introduces a bid-free model for buyers and sellers, akin to Storj and Sia, the focus remains primarily on price factors in storage resource allocation, potentially posing challenges in sustaining the true value equilibrium of storage resources.

Kriper [22] provides a community-based open blockchain network that allows for isolation and privatization as needed, while providing storage space for different types of data such as large files and lightweight messages. However, further exploration is required to optimize the performance and efficiency of this architecture, as well as address limitations in experimental results.

Vakilinia [23] proposes a scheme to verify the accurate data storage of storage providers during the storage contract period through the Proof of Spacetime (PoSt) mechanism. It also designs a new incentive-compatible mechanism to support effective execution of storage contracts. The execution process of storage contracts can be seen as a repeated dynamic game, where both storage providers and clients are players who choose strategies based on their own interests. Storage providers can choose to share or not share data, while clients can choose to challenge or not challenge the strategies of storage providers. If a client chooses to challenge, storage providers can choose to submit or not submit storage proofs. However, the Repeated Non-Cooperative Game discussed in this paper focuses on the storage providers and their clients who have already formed a transactional relationship. It does not address the matching game for order placement between storage providers and storage demanders before a transaction is formed. Additionally, the presence of a Trusted Third Party (TTP) introduces security risks.

Some other research works on blockchain-based distributed storage [24,25] use blockchain directly as a storage system and focus on privacy and security issues when sharing data on the chain. These above works focus more on the design and implementation of the underlying architecture, and none of them consider and involve the transaction issues of the storage resources themselves.

2.3. Pricing and matching mechanisms

In addition to the implementation of storage functions, an effective shared storage system also needs to consider the transaction and allocation of storage resources themselves, and this paper therefore focuses on the investigation of resource pricing and matching mechanisms in the network.

Oriented toward the pricing of data resources in the network, the current Internet has developed several relatively mature models [26]. Among them, pricing mechanisms that fully consider market factors generally build pricing models around the characteristic attributes of resources or a certain global optimization goal, aiming to achieve differentiated pricing and incentivize reasonable market behavior [27]. Meanwhile, price-oriented allocation mechanisms have become one of the most important methods to achieve network resource allocation. The literature [28] designed a game-theoretic approach to the cache placement and pricing problem, where the price threshold offered in the network determines the allocation direction of cache resources. Studies in the literature [29] show that in a decentralized market with decentralized information, rational pricing of resources can facilitate the achievement of supply–demand equilibrium as pricing mechanisms can influence transaction costs and direct real-time decentralized market demand.

The literature [30] classifies matching mechanisms in markets into two categories: (1) fully market-oriented, where both buyers and sellers can decide on their own whether to match with another party, and (2) fully centrally-oriented, where neither buyers nor sellers can choose a match on their own and resources are automatically allocated based on the results of algorithms executed by a centralized system. The matching of demand and supply of resources in networks usually falls into the second category, and most studies rely on game theory to solve the resource allocation problem. By solving Bayesian Nash equilibrium points in a double auction game, DARA [31] implements an efficient method to allocate different virtual resources to users. The multi-tenant-oriented PON sharing platform [32] is also based on dual auctions to meet the resource allocation needs of the market and proves the effectiveness and rationality of the mechanism from economic theory. In fact, similar matching based on utility optimization or auction theory implementation is common in the network domain. Yutao Jiao [33] proposes an auction-based market model for cloud/fog computing resource allocation, taking into account both competition among miners and network effects of the blockchain. The paper designs feasible auction mechanisms for two different bidding scenarios: fixed-demand strategy and multi-demand strategy. In the multi-demand bidding scenario, which is an NP-hard problem, the authors employ an approximation algorithm (a non-monotone submodular maximization with knapsack constraints problem) to achieve efficient solutions that approximate social welfare maximization. This algorithm can generate suboptimal solutions while aiming for the highest possible social welfare. Yaodong Huang [34] proposes a blockchain system that adapts to the limitations of edge devices. This system utilizes a novel proof-of-stake mechanism that enables energy-efficient consensus on edge devices. However, the author suggests that this mechanism is better suited for small-scale edge computing scenarios. BARA [35], which stands for Blockchain-based Auction for Resource Allocation, is designed for edge computing in the Industrial Internet of Things (IIoT). BARA eliminates the need for a trusted centralized entity or third party to conduct auctions, making it more decentralized and secure. However, further research and expansion are required for its application in large-scale network environments.

However, in the highly dynamic and decentralized shared storage market, the above-mentioned studies oriented toward differentiated pricing make overly ideal assumptions about the market, while the complexity of various factors in the actual market makes it difficult for pricing models to adequately balance the multidimensional characteristics and states of resources. As far as the matching mechanism is concerned, on the one hand, utility optimization-based resource matching is usually difficult to model various complex market factors and resource attributes into a solvable utility function, or the computational complexity in solving it is high. On the other hand, auction-based matching is still essentially price-oriented for resource allocation and relies on the design of the pricing mechanism, and thus may suffer from the problems caused by the pricing mechanism in the previous section. Both of the above resource allocation schemes do not fully consider the differentiated matching of service demand.

2.4. Motivation

In traditional non-cooperative game theory models, the allocation of storage resources relies primarily on price as a central factor for matching. Node providers often engage in competition to acquire greater rewards, which can lead to unnecessary waste and uneven distribution of resources. In contrast, a distributed storage model based on cooperative game theory facilitates better cooperation and sharing among node providers, thereby enabling more efficient, equitable, and sustainable data storage and management solutions. However, previous research has primarily focused on optimizing resource allocation and matching through price mechanisms, with limited consideration given to long-term cooperation and the establishment of trust among node providers.

Therefore, our research motivation lies in developing a cooperative game-based distributed storage model that utilizes stable marriage matching algorithms to address resource allocation and matching challenges. Our aim is to encourage the formation of long-term cooperative relationships between node providers and ensure fairness and stability in the outcomes of such matches. To achieve this, our research architecture adopts a bilateral matching approach, which combines on-chain and off-chain resource collaboration. This approach aims to enhance the performance of storage nodes, ultimately enabling more efficient, fair, and sustainable data storage and management solutions across diverse scenarios in the future.

3. Shared storage architecture

In this section, we expound the blockchain-based shared storage architecture and trading-based allocation mechanism we have proposed.

3.1. Key issue

Comparing with distributed storage system with centralized management, the decentralized distributed shared storage network has highly dynamic topology as well as widely dispersed resource. Hence its storage resources have greater challenges in terms of stability assurance and management. Specifically,to establish and develop efficient decentralized distributed shared storage network,some interrelated and influential issues should be addressed.

1. Trust issue in decentralized environment
   Due to the openness of access and heterogeneity of nodes, device nodes in distributed shared storage network naturally have no trust relationship with each other. Existing shared storage system provides a certain degree of fault tolerance usually, while could not ensure the reliability of the system if a large number of nodes conspire to cheat. In addition, only when users with storage needs trust shared storage system would they participate in storage market. However, the peer-to-peer nature shared storage system make it difficult for users to build trust directly.
2. Stability issue of storage resource
   Along with decentralized trust establishment, shared storage network need stable resource to build effective shared storage. Storage systems lacking stable storage capacity lose their attractiveness, resulting the loss of users and causing the storage market to fail.
3. Liquidity issue of storage resource
   Different from most centralized managed storage system,storage resource in shared storage system have natural feature of liquidity based on supply and demand tradings. Considering the liquidity configuration of storage resource, if without efficient trading mechanism for storage resources, storage providers lack incentives to provide sharing resource, hence weaken the stability of storage resource. In this case, the network would also lose users if lack of storage resource.

To cope with above issues, we proposed a blockchain-based shared storage architecture and a storage resource-centric trading mechanism on it based on the following design points.

1. Decentralized Trust
   Traditional centralized management-based systems rely on a single point or cluster to provide external services, and although they have certain advantages in performance, their fully centralized trust cannot meet the decentralized demand of distributed shared storage. Second, semi-centralized permission chains (including federated chains and private chains) relax the access restrictions of the system to a certain extent, but still allow only a small number of trusted subjects to manage the data ledger.
   We build a fully decentralized blockchain-based distributed shared storage based on the public chain: In a public blockchain, nodes are heterogeneous and decentralized, with each node having the right to participate in data validation and storage, eliminating the risk of a single point of failure. Any data alteration requires validation through a consensus mechanism to ensure integrity and authenticity, establishing trust among participants. Therefore, we have established a blockchain-based shared storage network model in Section 3.3 to maintain decentralized trust.
2. On-chain and Off-chain Resource Collaboration
   From storage perspective, blockchain acts as a state replicator and each node of the network needs to keep a copy of data. Therefore, blockchain itself suffers from high storage and computational costs and low storage scalability [36]. In contrast, distributed shared storage system provides efficient data retrieval mechanisms and high scalable storage capabilities, but its decentralized reliability is lower compared to blockchain storage.
   From computational perspective, traditional centralized management system has high computational performance, but it is difficult to guarantee the trustworthiness of computational results in a decentralized environment.On-chain computation can guarantee decentralized trustworthiness, but there are certain limitations: regardless of the on-chain computation of the account model, excessive on-chain storage affects its computational efficiency and cost [37].
   Considering the characteristics of blockchain and distributed shared storage network in terms of storage and computation,our proposed shared storage architecture adopt on-chain and off-chain resource collaboration scheme to improve the reliability and scalability of decentralized shared storage network.
3. Trading Mechanism of Storage Resource
   The first two design points integrate blockchain and distributed shared storage in architecture,while management and liquidity issue of storage resource need to be resolve, providing prerequisites for the construction and autonomy of the shared storage market.

Based on the proposed shared storage architecture, we design a blockchain-based storage trading mechanism to unify the publication standards of storage resources and provide a matching and settlement scheme for storage orders. The blockchain-based shared storage architecture and trading mechanism can realize decentralized trusted interaction and value circulation of storage resources, and design storage resource allocation algorithms for rational allocation of storage resources, finally establish an autonomous shared storage market.

3.2. Overview of the architecture

As illustrated in the Fig. 1, a Blockchain-based Shared Storage architecture is proposed. This architecture establishes a market characterized by the exchange of storage resources, referred to as a shared

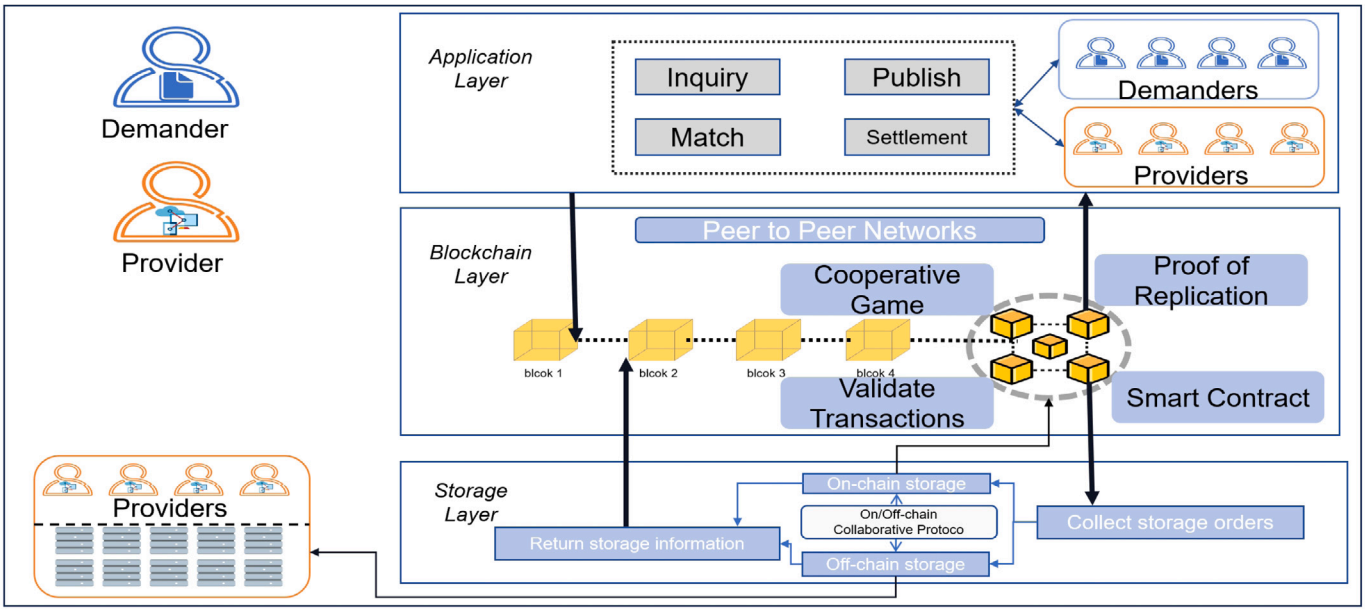


Fig. 1. The architecture of blockchain-based shared storage network.

storage market. Participants, including Storage Providers and Demanders, engage in the shared storage network through interactions with this market. In general, tradings involving idle storage resources within the market consist of three key participants:

- Storage Providers
  Providers of idle storage resources who are willing to offer unused disk space and respond to demanders storage requests to participate in the market, with the aim of receiving certain economic rewards.
- Storage Demanders
  Demanders with storage resource needs request idle storage provided by storage providers in the market and pay corresponding economic rewards for it.
- Sharing Storage Network
  Provision of functions such as searching for storage demanders, resource publication by storage providers, and tradings between both parties regarding storage resources, including fee settlement and other interactive behaviors. This involves managing storage resources across the network to facilitate provide-demand matching, providing a data ledger, and establishing a shared storage market.

From above description of the interaction patterns between the two parties,effective and autonomous decentralized shared storage network and market need to satisfy:

1. Available idle space provided by storage devices and demands from users for storage resource
2. both supply and demand parties actively participate in the construction of the network and market, with trust establishment and interaction
3. the trading system can sense the changes in the supply and demand of storage resources and achieve efficient real time matching based on the storage demand and service willingness of the provide and demand sides

Hence the architecture of the proposed shared storage system consist of:

- Application Layer
  In the context of decentralized shared storage applications, the application layer plays a pivotal role. It is responsible for providing participants with an interactive interface to enter and query information in the shared storage market, primarily focusing on storage resources. Through this interface, participants can list their own storage resources on the market, as well as search and acquire the storage resources they need from the market.
- Blockchain Layer
  the core layer of this system, including

  1. Smart Contract: This is a type of code or algorithm that triggers automatically, typically when certain conditions are met. In this context, it is used to implement the trading mechanism for storage resources.
  2. Consensus Mechanism: Within the blockchain-based shared storage network, nodes can utilize their storage capacity to perform a Proof of Replication (PoRep) [38], similar to Proof of Work (PoW), to mine. The PoRep consensus mechanism based on storage capacity can provide verifiable storage and avoid the wastage of computational power inherent in PoW consensus. The specific proposals for storage proof are **not discussed** in this paper.
  3. Cooperative Game: By using a bilateral matching algorithm based on cooperative game theory, participants can be encouraged to develop toward long-term cooperation, thus making the network status more stable.
  4. Validate transactions: The actual blockchain transaction records, also known as ledger state logs, are protected by encryption methods and chain data structures to ensure data integrity and security, preventing tampering.

- Storage Layer
  The architecture's data storage medium primarily comprises two components:

  1. Off-chain storage: This component consists of distributed shared storage, which is made up of idle resources supplied by various storage devices.
  2. On-chain storage: This component is provided directly by the blockchain layer.
     Simultaneously, a collaborative mechanism is employed to harmonize the on-chain and off-chain storage, as well

as computing and other resources, ensuring smooth and efficient operations across the system.

3. Collect Storage Orders: This refers to the process of gathering and aggregating storage orders from users or clients. These orders specify the desired amount of storage, duration, and any additional requirements. The system collects and organizes these orders to facilitate efficient allocation of storage resources.
4. Return Storage Information: This involves providing users or clients with detailed information about the allocated storage resources. The system retrieves relevant data, such as storage capacity, location, and access protocols, and presents it to the users in a user-friendly format. This enables users to have a clear understanding of the storage resources they have obtained and facilitates efficient utilization of the allocated storage.

In this paper, we make the assumption that nodes within a blockchain shared storage network engage in direct mining activities using their storage capacity for PoW-like storage capacity proof called PoRep [38]. This consensus mechanism based on storage capacity not only ensures verifiable storage but also eliminates the unnecessary computational power consumption associated with traditional PoW consensus. Additionally, we assume that the consensus nodes possess approximately synchronized clocks and generate new blocks at fixed time Δt intervals, thereby maintaining a consistent and predictable blockchain operation.

3.3. Blockchain-based shared storage network model

Thus far, the blockchain-based shared storage network can be modeled as:

- M Storage Demanders (SDs), SDs act as validation or light nodes in the blockchain network and do not require the complete blockchain ledger data. They are responsible for routing and forwarding functions while requesting and paying for data storage within the network.
- N Storage Providers (SPs), SPs act as consensus or miner nodes in the blockchain network and must store the complete blockchain ledger data. They respond to storage requests from SDs by storing data at a specific time, generating storage proofs, and submitting them to the blockchain network for verification of authenticity. If the proof of storage is invalid or lost, SPs cannot receive payment from the SDs.
- Either node (SD or SP) i is mapped to the blockchain shared storage network based on a pair of asymmetric keys (sk_i, pk_i),where sk_i is the private key and pk_i is the public key. The information in the storage market is recorded to the blockchain and any node can verify and forward the data.

3.4. On/off-chain protocol

In a blockchain-based system, peer-to-peer payment tradings are asynchronously broadcasted and recorded in a distributed ledger that is replicated by nodes within the network. As a result, any node within the blockchain network can accurately calculate the state of a transaction for data validation purposes, using the ordered ledger log. However, this mechanism of blockchain-based data storage and validation also gives rise to the following on-chain data problems:

- High storage requirement and low scalability
  In order to maintain the consistency of the ledger state in a decentralized network, blockchain employs complete network state replication for data storage. However, as the number of transactions in the blockchain network increases, not only does the data storage burden on individual nodes become heavier, but the overall size of storage cannot keep up with the growth of the network.

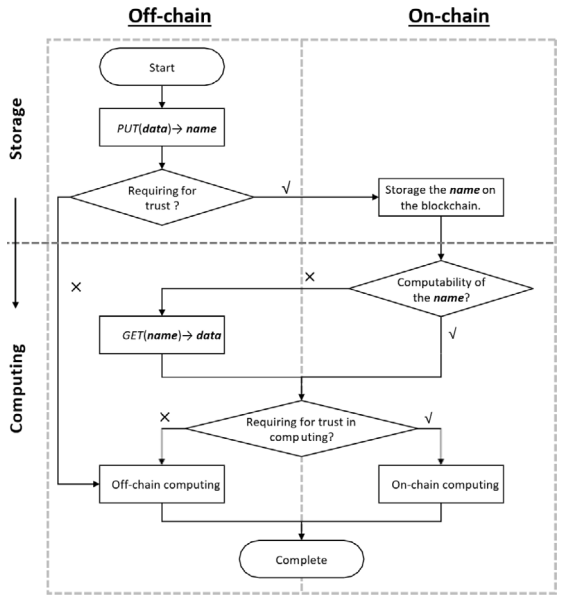


Fig. 2. The On/Off-chain collaborative protocol.

- High cost of storage and computing
  The limited scalability of on-chain storage results in an inevitable increase in storage costs. Furthermore, if the computation heavily relies on on-chain data, the cost of the blockchain system increases significantly.

To address these challenges, we propose an on/off-chain resource collaboration protocol that leverages the robust data storage and retrieval capabilities of distributed shared storage, while preserving the reliability of on-chain storage and computation.

In general, the off-chain distributed shared storage under the chain provides two basic protocol:

- $PUT(data) \rightarrow name$: PUT protocol is utilized for storing data and providing a unique identifier $name$ for the $data$. The storage process of $data$ relies on the sk of the data owner for signing purposes, in order to assert ownership of the data.
- $GET(name) \rightarrow data$: GET protocol is utilized for obtaining the content of data that is currently identified by the $name$. Since this data identifier is calculated based on the pk of the data owner associated with the data source, it can be traced back to the owner of the data.

The architecture and protocols that integrate blockchain and distributed shared storage technologies, the on/off-chain resource collaboration protocols manly involve the collaboration of storage and computing resources, as shown in Fig. 2.

1. Storage: Data content $data$ is stored in a distributed shared storage network off-chain, with access granted via the $PUT(data)$ protocol using its identifier $name$. If low trust in data storage or no on-chain computation is needed, storage concludes. For high trust or on-chain computation, on-chain storage uses a semantic name $name'$ based on the data identifier name, like $/\langle name\rangle/\langle type\rangle/\langle size\rangle/\langle duration\rangle/$, without storing the full data content $data$.
2. Computing: Data computing can be divided into on-chain and off-chain computation: For off-chain computation, data stored off-chain can directly be used for calculation or the actual $data$ content can be obtained by utilizing the data name $name'$ stored on-chain to retrieve it. For on-chain computation, if the data name stored on-chain already satisfies the input conditions for

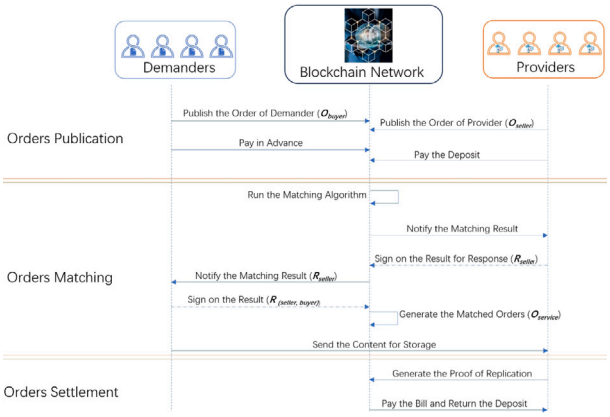


Fig. 3. The trading procecure of storage resources.

computation, it can be directly utilized; if not, the complete data content $data$ is retrieved through the off-chain storage protocol $GET(name')$ before conducting the actual on-chain computation.

3.5. Trading mechanism

The trading of storage resource involves the perception and matching of storage resources and settlement of storage services. Hence our proposed blockchain-based storage resource trading mechanism carries the attribute information of storage resources and services with storage orders and serves as the basic unit for subsequent storage service matching. Specifically, the trading of storage resources consists of several phases as shown in Fig. 3

1. Order Publication
   SDs and SPs respectively submit storage orders to the shared storage network, with the storage orders reflecting the attributes of their storage service requirements. Specifically, orders can be categorized into three types: (1) Buyer order O_{buyer}: These are storage requests initiated by SDs, specifying their storage needs such as required storage space, duration, and number of replicas; (2) Seller Orders O_{seller}: These are storage offers made by SPs, detailing their available storage capacity including space size, duration, and number of replicas; (3) Service Orders $O_{service}$: These represent matched agreements between SDs and SPs, establishing the terms of service for storage tradings. The order submission standards are designed to uniformly represent the properties of storage resources and basic operations, as shown in Table 1. Simultaneously, in order to prevent malicious behavior from SDs or SPs that may disrupt the market, both parties are required to make prepayments and deposits when submitting their storage orders.
2. Order Matching
   With the orders published, the shared storage network generate matching orders to both parties based on multiple attributes of different orders. The detailed bilateral matching algorithm will have discussed in Section 4. After the matching, smart contracts trigger the interaction between the two parties involved in the matching process, generate and update the types and statuses of matching orders based on the signatures of both parties, and publish them to the blockchain; Once the matching order is recorded on the blockchain, SD can send the data to be stored to SP.
3. Order Settlement
   After the matching, the SDs can sent the data to be stored to the SPs. Once the storage finished, the storage provide corresponding storage proofs to the shared storage network. Once the proofs are verified, SPs receive the settlement of storage, including the prepaid from SDs and the refunded deposit. If the verification does not pass, the deposit of SPs will be deducted for penalty and compensate to SDs. The payments in blockchain are practically abstracted as transfers between accounts, which can be implemented based on smart contracts.

Table 1
The abstract class of $Order$.

Variable/function	Description
$type$	The type of the order.
$buyerAddr$	The address of the demander.
$sellerAddr$	The address of the provider.
$capacity$	The capacity of storage.
$numOfCopies$	The number of copies.
$startTime$	The time to start the storage.
$endTime$	The time to end the storage.
$addOrder(Order\ order)$	The function to instantiate an order.
$matchOrder()$	The function to match with other orders.

All order data is managed through the On/Off-chain protocol (refer to Section 3.4). This entails sending the order data content $data$ to a distributed shared storage off-chain by both parties, while on-chain only stores a semantic name $name$ derived from the order content. This approach aims to reduce blockchain data storage pressure and ensure the reliability of order data storage. When computational operations on order data are required, one can extract the computational attributes represented by the data identifier or retrieve the complete data content from off-chain storage for processing. This design effectively lightens the blockchain's load.

4. Bilateral matching algorithm

4.1. Description of question

In this paper, we consider a shared storage market consisting of N storage providers$S = \{s_1, s_2, \ldots, s_N\}$ and M storage demanders$D = \{d_1, d_2, \ldots, d_M\}$ satisfying $S \cap D = \emptyset$. The finite and non-interaction provide and demand sides trade storage resources based on the trading mechanismTM proposed in the previous section. At this point, the shared storage market model can be represented by the following tuple:

$$\langle S, D, TM = (SR, SM, SS)\rangle$$

In the trading mechanism TM of storage resource, SR(Storage Requirement) defines and reflects the storage trading requirement of both parties. The market match the storage requirements based on SM(Storage Matching),i.e. the allocation of storage resources, and the final matching is completed with the SS(Storage Settlement) of storage services by both parties of the trading. In a market where a seller can correspond to multiple buyers with willingness to pay, price-oriented allocation mechanisms such as auctions are usually used to ensure the efficient allocation of economic resources or to maximize benefits [39]. However,in complex and volatile decentralized shared storage market, there are some limitation to use auction-based or utility-optimized matching mechanisms:

- Although price can reflect the value of storage resources in a certain extent, it is usually difficult to adequately balance and convey other attribute using price alone as the decisive factor for resource allocation.
- Due to the dynamic nature of decentralized shared storage network and the fast changes of information in the whole network,it is difficult to model a solvable utility function by integrating various factors such as topological changes of nodes and changes in provide and demand of storage resources,or the modeled objective functions is usually an NP-hard generalized multi-objective problem with high computational complexity to solve [40].

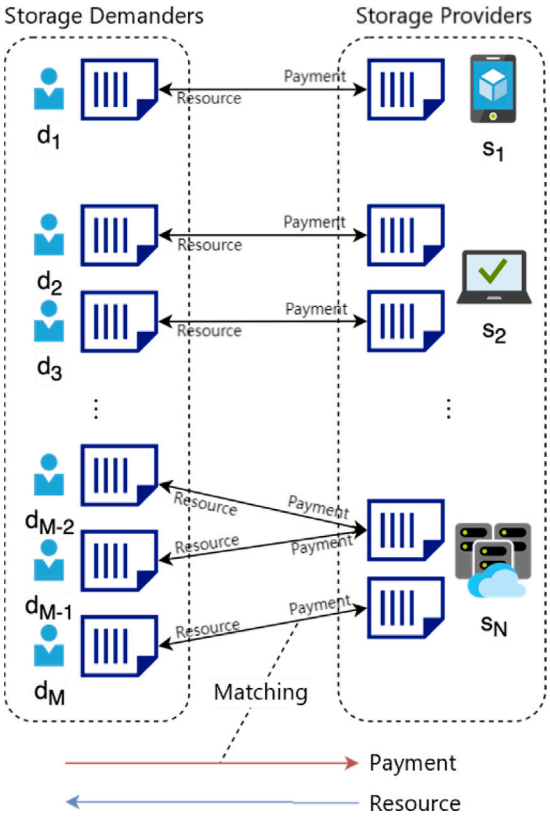


Fig. 4. The bilateral matching model between storage demanders and providers.

- Due to the strong heterogeneity, dynamics and dispersion on the devices of the storage providers, it is difficult for users to consider the multidimensional characteristics and states of storage resources comprehensively when selecting the provider.

More importantly, in actual market tradings, the provide and demand sides of storage resources often have inconsistent or even conflicting differentiated matching expectations [41]:

- The requirements of each user for storage services including price, storage duration, storage space, may be different, and correspondingly the storage capacity of each strong device may be different.
- The access requirement of storage demanders and storage providers are different,including access speed and access frequency.
- storage demanders prefer to obtain considerable storage quality at a lower price, while the storage providers want to gain as much profit as possible from its limited storage capacity.

Therefore, for a decentralized shared storage market, we consider reasonable allocation of storage resources to automatically promote the autonomy of the shared storage market while taking the demand goals of most participants into account.

4.2. Pairing model

In this paper, we take $S = \{s_1, s_2, \ldots, s_N\}$ as the set of storage devices from storage provider and $D = \{d_1, d_2, \ldots, d_M\}$ as the set of users from storage demanders, satisfying $S \cap D = \emptyset$. Let A be the attributes of storage service(including capacity, duration, access frequency, etc.) and the amount of attribute of storage service is constant as $L = |A|$.

The bilateral matching model of users and storage devices in a single storage allocation cycle is shown in Fig. 4. Through storage orders, both parties announce their storage service requirements and the matching between storage providers and demanders is a many-to-one matching based on bilateral market:

Each storage device s_i provides a limited capacity $q_{S_i}(q_{S_i} > 0)$ of idle storage resources and can split its storage resources into multiple orders to serve multiple users, with each user d_j's demanded storage capacity$q_{d_j}(q_{d_j} > 0)$ to be served by one storage device. For the whole shared storage market, the total storage capacity that can be provided in the market is$Q_S = \sum_{i=1}^{n} q_{s_i}$ and the total storage capacity demanded is $Q_D = \sum_{i=j}^{n} q_{d_j}$. Table 2 summarizes the key notation definitions in the bilateral matching model.

Table 2
List of notations.

Symbol	Meaning
Δt	The block interval and the matching period.
$S = \{s_1, s_2, \ldots, s_N\}$	The set of storage providers.
$Q_S = \sum_{i=1}^{N} q_{s_i}$	The providing storage capacity of the entire network.
$N = \|S\|$	The number of the providers.
s_i	The ith storage provider.
q_{s_i}	The storage provided by p_i.
$D = \{d_1, d_2, \ldots, d_M\}$	The set of storage demanders.
$Q_D = \sum_{j=1}^{M} q_{d_j}$	The demanding storage capacity of the entire network.
$M = \|D\|$	The number of the demanders.
d_j	The jth storage demander.
q_{d_j}	The storage needed by d_j.
$A = \{a_1, a_2, \ldots, a_L\}$	The set of storage attributes.
$L = \|A\|$	The number of the storage attributes.
a_k	The kth storage service attribute.
a_k^i	The value of storage service attribute a_k provided by s_i.
a_k^j	The value of storage service attribute a_k provided by d_j.

4.3. Preference utility and preference order

Arbitrary user d_j's preference for storage device s_i is specifically reflected in attributes related to storage services, such as storage duration, storage capacity, and number of replicas, including the balanced pricing of unit storage resources. Similarly, storage device s_i also exhibits preferences for user demand attributes. Considering the existence of balanced pricing, this paper directly takes the price demand for unit storage resources from both sides as their preference factors.

This paper utilizes the demand attributes reflected in buying and selling orders in the storage market to define the utility-based calculation process [42] for the preferences of both users and storage devices. The preference utility function under multi-attribute demands is defined in an additive form:

$$\mathrm{U}_{p(i,j)} = \sum_{k=1}^{L} \omega_k m_k \tag{1}$$

Among them, two parameters are introduced for demand attributes: (1) Preference weight $\omega_k^i \in [0,1]$ The storage provider s_i and the storage demander d_i have preference weights ω_k^i and ω_k^j respectively for the same preference attribute a_k, satisfying the equation:

$$\begin{cases} \sum_{k=1}^{L} \omega_k^i = 1 \\ \sum_{k=1}^{L} \omega_k^j = 1 \end{cases} \tag{2}$$

It should be noted that the weight values ω_k for the same attribute a_k may not necessarily be the same across different storage providers s_i or storage demanders d_i.

(2) Preference matching degree $m_k \in [0,1]$ Let m_k^{ij} represent the preference matching degree for attribute a_k between storage provider s_i and storage demander d_j. Similarly, let m_k^{ji} represent the preference matching degree for attribute a_k between storage demander d_j and storage provider s_i. It should be noted that the calculation methods

for the preference matching degree are different for both provide and demand sides: For storage providers s_i, they aim to maximize the utilization of their resources while satisfying the basic storage service requirements of users. For storage demanders d_i, all storage demand attributes are preferred to have smaller values. Therefore,

$$m_k^{ij} = \begin{cases} 0, & a_k^i < a_k^j \\ \frac{a_k^j}{a_k^i}, & a_k^i \geq a_k^j \end{cases} \tag{3}$$

For storage demanders d_j, they aim to maximize the redundancy of their resources while satisfying the basic storage service requirements of users. For storage providers s_i, all storage service attributes are preferred to have larger values. Therefore,

$$m_k^{ji} = \begin{cases} 0, & a_k^i < a_k^j \\ \frac{a_k^i - (a_k^j)_{\min}}{(a_k^i)_{\max} - (a_k^i)_{\min}}, & a_k^i \geq a_k^j \end{cases} \tag{4}$$

Based on the preference utility of both matching parties, each storage device s_i and user d_j can construct their respective preference orders for the other matching side: the higher the preference utility value, the higher the position in the order.

Each preference of a storage provider s_i can be represented as a preference order $P(s_i)$ on the set $D \cup \{s_i\}$. A preference of a storage device s_i may be expressed as: $P(s_i) = \{d_1, d_2, \emptyset, d_3, \ldots, d_M\}$, indicating that user d_1 is its top choice, d_2 is the second preference, and starting from $\emptyset$, no other user satisfies the matching preference of the current storage device. In this case, the preference order of s_i can be simplified as $P(s_i) = \{d_1, d_2\}$.

Similarly, each storage demanders d_j has a preference order $P(d_j)$ defined on the set $S \cup \{d_j\}$: $P(d_j) = \{s_1, [s_2, d_3], \emptyset, \ldots, s_N\}$, indicating that user s_1 prefers s_2 over others, but notably, there is no preference difference between s_2 and s_3. Apart from these devices, no other storage device satisfies its matching requirement. For indistinguishable preferences, this paper adopts a random ordering.

4.4. Algorithm

In a storage allocation cycle Δt, a matching solution μ of storage provide and demand is a many-to-one mapping in the set $S \times D \rightarrow S \times D$, $\mu(x)$ is called the matching result of x and satisfies:

- each user matches to a storage device, i.e., for $\forall d_j \in D$, there is always its match $\mu(d_j) \in P(d_j) \leftarrow S \cup d_j$;
- each storage device can match multiple users, but the storage capacity demanded by the users cannot exceed the total idle storage capacity provided by the storage device, i.e., for $\forall s_i \in S$, there is always a match $\mu(s_i) \in P(s_i) \leftarrow D \cup s_i$ and $\sum_{d_j \in \mu(s_i)} q_{d_j} \leq q_{s_i}$.

This paper focuses on user-centric matching for storage services, and presents a bilateral matching-based storage resource allocation algorithm as shown in Algorithm 1. In this algorithm, the calculation of preference orders in line 2 follows the method proposed in Section 4.3. The core part of the algorithm spans from line 2 to line 19. Specifically, the algorithm works as follows:

1. Construct preference orders for both users and storage devices.
2. Each user sends matching requests to storage devices in the order specified by their preference order. Upon receiving a matching request from a user, a storage device has three response options:
   - If the storage device is not currently matched, it temporarily accepts the user and updates its available capacity.
   - If the storage device is already matched, but the requesting user has a higher preference ranking in the storage device's preference order, it replaces the least preferred user among the matched ones and repeats the actions described above.
   - If the storage device is already matched and the requesting user has a lower preference ranking, it directly rejects the user's matching request.
3. Repeat the above process until one of the following termination conditions is met:
   - Each user has sent matching requests to all storage devices it can potentially match with.
   - The available capacity of each storage device is 0.

Algorithm 1: The Bilateral Matching Algorithm

Input:
The set of storage providers: $S = \{s_1, ..., s_N\}$
The set of storage demanders: $D = \{d_1, ..., d_M\}$.
Storage service attribute vector is denoted as $\mathbf{A}$, and the weight vector of attributes is denoted as $\boldsymbol{\Omega}$

Output:
The set of matched providers and demanders: $\mathcal{O}_{\text{service}}$.

```
/* Initialization                                          */
1  O_service ← ∅;
2  Compute the preference orders P(S) and P(D) based on A and Ω
     using equations (1) to (4).
3  D' ← D;
/* Bilateral Matching                                      */
4  for s_i ∈ S do
5      for d_j ∈ P(s_i) do
6          if d_j ∈ D' and u^{s_i}(d_j) > 0 and q_{s_i} > q_{d_j} then
7              O_service.add((s_i, d_j));
8              D'.remove(d_j);
9              q_{s_i} ← q_{s_i} − q_{d_j};
10         else
11             s_x = O_service.check(d_j)
12             if u^{d_j}(s_i) > u^{d_j}(s_x) and q_{s_i} > q_{d_j} then
13                 O_service.remove((s_x, d_j));
14                 O_service.add((s_i, d_j));
15                 q_{s_i} ← q_{s_i} − q_{d_j};
16                 q_{s_x} ← q_{s_x} + q_{d_j};
17             end
18         end
19     end
20 end
21 return O_service
```

5. System evaluation and experiments

5.1. Experimental setup

In this paper, we implemented a prototype system based on IPFS and Ethereum, and built an experimental platform for the blockchain based shared storage trading system. We use Docker to publish multiple IPFS and Ethereum application images to form a private network with custom parameters for experiment evaluation.

We use four high-performance serves to build the prototype system of private network, in which the storage verification node is only an Ethereum light node and the storage consensus node acts as both an Ethereum node and an IPFS noes. The final network topology is shown in Fig. 5.

To form a private Ethereum network, the four servers use the same genesis.json to initialize the Genesis Block, where the descriptions of key parameters are given in Table 3. Subsequently, different blockchain

Table 3
Description of key configurations in genesis.json.

Configuration	Default	Description
chainID	2021	Private Chain ID, used to distinguish other Ethereum based blockchain networks
difficulty	0×5000	Mining difficulty, used to control the block production time of the blockchain, i.e. the block interval
gasLimit	$0 \times 2fefd8$	Gas consumption limit for blocks, used to control the block size of the blockchain
timestamp	–	Timestamp of the Genesis block
coinbase	–	miner account of the Genesis block

Table 4
Key attributes of service orders.

Order	Attribute	Length of attribute (B)	Type of data	Length of data (B)	Total length (B)
1	type	4	Number	1	9
2	price	5	Number	8	17
3	capacity	8	String e.g.“128MB”	10	22
4	duration	8	String e.g.“3600s”	10	22
5	numOfCopys	10	Number	1	15

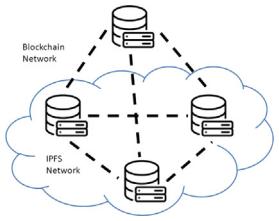


Fig. 5. Network topology of the experiment environment.

network parameters such as block interval and block size can be simulated by modifying corresponding configurations. The three servers used as storage consensus nodes form a private IPFS network, and the maximum storage capacity consumed by each server is set to 10 GB. In addition, each server is virtualized by Docker, enabling each single physical machine to be virtualized as up to five heterogeneous nodes.

The experiment set that storage demanders and providers to have the same demand attributes for storage services, except for the price attribute, including storage space size, storage duration and number of storage copies, respectively.

The storage orders containing the above service requirement attributes have different data formats storing on/off-chain. In the off-chain storage IPFS network, the storage format of order data is JSON, all the attribute keys are strings, and the data type of attribute values are shown in Table 4; in on-chain storage, names are directly used for attribute calculation, the storage format of order data is a name string based on the content order data. The naming rule is “$\langle hashOnIpfs\rangle/\langle type\rangle/\langle price\rangle/\langle capacity\rangle/\langle duration\rangle/\langle numOfCopys\rangle$”, where $hashOnIpfs$ indicates the hash value of the storage order data returned by the IPFS network and the length is 32 bytes.

5.2. Security analysis

The architecture of a blockchain-based shared storage system provides a certain level of security against two common types of attacks on distributed systems.

- Distributed Denial-of-Service (DDoS) Attack
  Firstly, Distributed Denial-of-Service (DDoS) Attacks are common in traditional client/server models where attackers can use multiple hosts to overwhelm servers with requests, disrupting normal system operations. The decentralized nature of a blockchain-based shared storage system removes the target of attack from the architecture, making it more difficult and expensive to launch DDoS attacks.
- Sybil Attack
  Secondly, Sybil Attacks involve an attacker creating multiple identities for a single node in a P2P network in order to interfere with discourse rights distribution, data copy verification, and system credit. Compared to traditional P2P systems, blockchain-based shared storage systems use consensus mechanisms like Proof of Work (PoW) and Proof of Replication (PoRep) to reduce the likelihood of Sybil Attacks. Nodes must pay for computing power or storage in order to mine and obtain discourse rights in the system, which increases the cost of forging identities and prevents Sybil Attacks to a certain extent.

In addition, the blockchain-based shared storage trading system provides security guarantees for the following issues:

- Forgery of Storage
  Although the blockchain-based architecture can reduce the possibility of Sybil attacks to a certain extent, storage nodes in the shared storage system can still profit from attacking behavior that exceeds their actual storage capacity. In this proposed solution, when devices publish storage, they need to deposit an amount equivalent to their generated storage capacity. If fake storage behavior is detected during subsequent storage verification, the deposit will be deducted, and the cost of the attack will increase accordingly. Therefore, blockchain-based shared storage can effectively restrict the behavior of forgery of storage.
- Double Storage
  Similar to double-spending in blockchain, “Double Storage” refers to providing the same storage space to multiple users. This issue can be resolved by the consensus mechanism of blockchain, based on computation or storage capacity. In Bitcoin, users need to sign their tradings with private keys, and the trading must be verified by the public key across the entire network before it becomes effective, thereby solving the issue of currency ownership from a cryptographic perspective. Additionally, the PoW consensus mechanism ensures the final consistency of transaction data, making double-spending attacks difficult to achieve. Moreover, due to the design of the deposit mechanism, achieving double storage requires successful forgery of storage behavior, which increases the cost of the attack as analyzed in the issue of forgery of storage.

To verify the feasibility of our architecture design, we first conduct function testing of the prototype system without considering the specific transaction (see Table 5).

The test results show that the prototype system can implement the basic functions designed in the previous section, and the blockchain based shared storage trading system has feasibility and usability.

Table 5
Testing of the functions.

Item	Method	Result
Construct blockchain network	Get the network connection information of the current node and other nodes through the admin.peers command of Ethereum, and the nodes are able to synchronize the block information among themselves.	Pass
Construct IPFS network	Remove the seed connection information of the current node by $ipfs\ bootstrap\ rm\ --all$, then get the connection address of the current node by $ipfs\ id$ command, and reconfigure the connection of other nodes by $ipfs\ bootstrap\ add$.After restarting each node, adding a file to any node in the network, and the file could be downloaded at any other node via the returned file address.	Pass
Deploy smart contract	Deploy smart contract to the private chain network, the transaction hash and contract address could return after a successful contract deployment.	Pass
On/off-chain mechanism	Adding a storage order file via IPFS, the on-chain name of the file could be called through the smart contract, and could also get the content of the off-chain file by that name.	Pass
Storage Order Release	The same way as the On/off-chain mechanism	Pass

5.3. Performance evaluation

5.3.1. Evaluation metrics

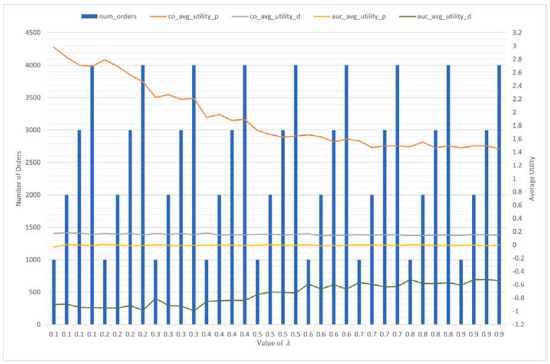
Our architecture incorporates the technology architecture of distributed shared storage and blockchain, and implements tradings of storage resources based on storage orders, using the following main metrics to evaluate the overall performance of the system.

1. On-chain storage consumption
   The primary function of this system is to implement storage resource tradings. This involves additional storage consumption for order data persistence, which is based on resource discovery provided by storage orders. Expanding storage order demand attributes can better allocate storage resources to meet both trading participants' needs, but also increases the system's storage load. To mitigate this, we collaborate with off-chain storage resources to reduce on-chain storage consumption. Performance evaluation metrics include on-chain storage consumption, with lower additional on-chain storage consumption reflecting superior scalability performance compared to systems without resource collaboration
2. Throughput
   Throughput is an important metric to describe the performance of a trading system and is usually measured in terms of the number of transactions per second (TPS) that the system can process. In a blockchain system, the average throughput is calculated as
   $$TPS_{\Delta t} = \frac{NumOfTxs_{\Delta t}}{\Delta t}$$
   where Δt is the average block interval time and $NumOfTxs_{\Delta t}$ is the number of transactions contained in the blocks generated in that time period.
3. Matching latency and transaction latency
   The usual latency refers to the time difference between the request being sent and received in the network, which can measure the performance of the system on the one hand and affect the calculation of the system TPS on the other. Therefore, the latency is also particularly important in systems where throughput is the main performance indicator.
   The matching latency of storage orders affects the overall performance of the system, which is influenced by the computational complexity of the matching algorithm itself, but also by the fact that the actual matched order data needs to be propagated in the IPFS network, so the matching latency is the sum of the order data broadcast time and the order matching algorithm running time
   $$latency_{matching} = t_{order\_download} + t_{order\_match} \tag{5}$$
   In addition, the concept of time latency in blockchain system is mainly for transactions, which refers to the time difference between the generation of a transaction and its confirmation on-chain, and is generally the sum of three parts: transaction broadcast time, block interval time and block broadcast time.
   $$\begin{aligned} latency_{tx} &= t_{confirm} - t_{generate} \\ &= t_{tx\_broadcast} + t_{block\_interval} + t_{block\_broadcast} \end{aligned} \tag{6}$$
   Therefore, due to the matching process, the transaction latency in the blockchain based shared storage transaction system is
   $$\begin{aligned} latency_{order} &= latency_{tx} + t_{order\_match} \\ &= t_{tx\_broadcast} + t_{block\_interval} + t_{block\_broadcast} \\ &\quad + t_{order\_match} \end{aligned} \tag{7}$$
4. Average Utility Value
   Let the matching result be M, n be the number of Demanders, m be the number of Providers, q_p be the trading volume of the Demander, q_d be the trading volume of the Provider. Assume p_{ij} represents the preference value of Demander i for Provider j, d_{ji} represents the preference value of Provider j for Demander i. x_{ij} represents whether Demander i is matched with Provider j, y_{ji} represents whether Provider j is matched with Demander i. Then we have:
   $$\text{Total\_utility}_{\text{Demander}} = \sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}p_{ij}q_p(i,j)$$
   $$\text{Total\_utility}_{\text{Provider}} = \sum_{j=1}^{m}\sum_{i=1}^{n} y_{ji}d_{ji}q_d(j,i)$$
   $$\text{Ave\_utility}_{\text{Demander}} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}p_{ij}q_p(i,j)}{\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}}$$
   $$\text{Ave\_utility}_{\text{Provider}} = \frac{\sum_{j=1}^{m}\sum_{i=1}^{n} y_{ji}d_{ji}q_d(j,i)}{\sum_{j=1}^{m}\sum_{i=1}^{n} y_{ji}}$$
   Where $\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}$ represents the number of participating Demanders, and $\sum_{j=1}^{m}\sum_{i=1}^{n} y_{ji}$ represents the number of participating Providers.

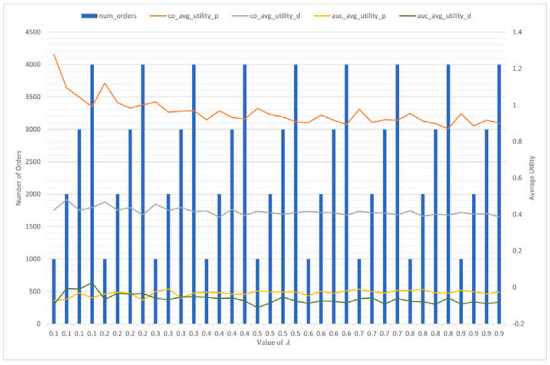
5.3.2. Matching satisfaction analysis

In this study, we propose a stable bilateral matching algorithm based on cooperative game theory. We compare the satisfaction levels of participants using our proposed algorithm with the classical algorithm of Double Auction [43] as the Baseline. We conduct experimental analysis by randomly generating instances of orders using a Poisson distribution with parameter values λ set to 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9. These orders consist of demand orders and provide orders, with the proportion of demand orders to the total number of orders being 0.2, 0.4, 0.6, and 0.8, respectively. The experimental results are presented in the Fig. 6.

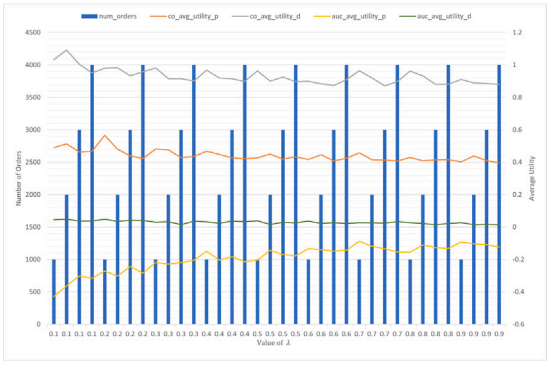
Across all scenarios, our proposed bilateral matching algorithm consistently achieves satisfactory results for both the providers and the



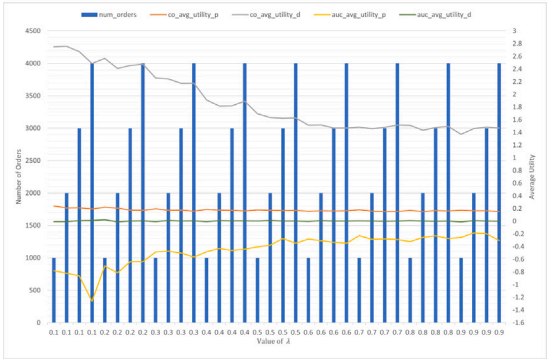
(a) Average Utility of Both Parties with 0.2 Proportion of Demand Orders.



(b) Average Utility of Both Parties with 0.4 Proportion of Demand Orders.



(c) Average Utility of Both Parties with 0.6 Proportion of Demand Orders.



(d) Average Utility of Both Parties with 0.8 Proportion of Demand Orders.

Fig. 6. Comparison of Average Utility between Different Demand Order Ratios.

demanders, with average utility values greater than 0. In contrast, the baseline algorithm exhibits instances where one or both parties fail to achieve satisfactory results, as indicated by average utility values smaller than 0. This clearly demonstrates that our proposed stable bilateral matching algorithm provides stability in the matching process.

Moreover, this section provides the following theorem and theoretical proof for storage resource allocation algorithms based on stable bipartite matching, further validating the effectiveness and rationality of the blockchain-based shared storage trading system.

Theorem 1. *Given the preference lists $P(D)$ and $P(S)$ of users $D = \{d_1, d_2, \ldots, d_M\}$ and storage devices $S = \{s_1, s_2, \ldots, s_N\}$, the storage resource allocation algorithm based on bipartite matching can iteratively produce a stable matching result between users and storage devices.*

Proof. In the storage resource allocation algorithm based on bipartite matching, if the storage request of user d_j is rejected by storage device s_i, it happens if and only if s_i prefers all other users over d_j.

For a proof by contradiction, it suffices to prove the non-existence of any user outside of $\mu(s_i)$ who prefers storage device s_i more than the users inside $\mu(s_i)$. On one hand, other users whose preference for storage device s_i is lower than d_j will not send storage service matching requests to s_i. On the other hand, if there exists a user $d_x \in \mu(s_i)$ who is rejected by s_i and gets matched to another storage device s_y other than s_i, satisfying that d_x prefers s_y over s_i, then the users inside $\mu(s_i)$ would not have initially sent storage service requests to s_i. Therefore, the result $\mu(s_i)$ iteratively produced by the algorithm always represents a stable matching for the storage device s_i.

Proof complete.

5.3.3. Performance analysis

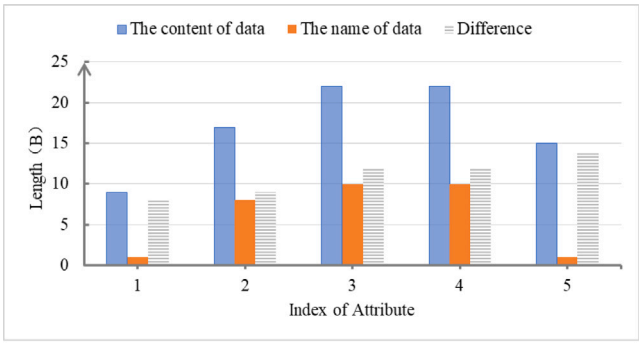
The prototype system in this paper is built on the Ethereum blockchain, thus its performance is constrained by the overall operational efficiency of the Ethereum framework. This includes factors such as the efficiency of underlying logic, the performance of the PoW consensus mechanism, the compilation speed of the smart contract compiler, as well as the network bandwidth and CPU performance of the node hardware.

Adjusting the block interval and block size of the blockchain network allows for achieving different levels of throughput and transaction latency to meet the performance requirements of shared storage market tradings effectively.

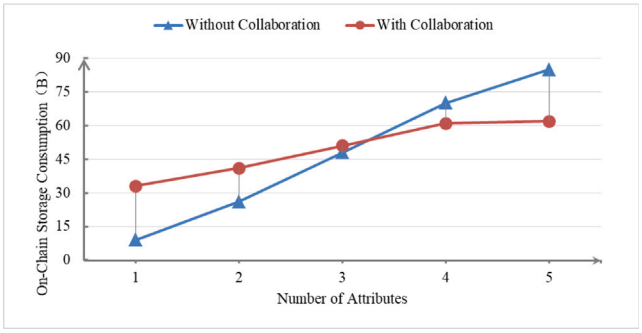
To evaluate the impact of data attributes on storage consumption without resource collaboration, each of the five storage services listed in Table 4 is sequentially added to the storage order. Fig. 7(a) illustrates the varying impact of data name and content attributes on additional storage consumption. The degree of impact differs significantly among individual attributes based on their key value characteristics and demand nature.

To assess the impact of on-chain and off-chain resource collaboration schemes on on-chain storage usage optimization, we analyzed additional storage requirements with and without resource collaboration. Fig. 7(b) illustrates our findings:

- With a small number of demand attributes, on-chain storage consumption surpasses that without resource collaboration. This is due to the resource collaboration scheme necessitating storage of a 32-byte hash address of the order on-chain, significantly exceeding the storage needed for the data content itself.
- As the number of demand attributes increases, on-chain storage consumption consistently rises linearly in scenarios lacking resource collaboration. Conversely, in instances of resource collaboration, system storage for order data experiences gradual growth, resulting in substantially lower storage demands compared to non-optimized cases.



(a) Comparison between index of attribute and on-chain consumption based on individual attributes.



(b) Comparison between with collaboration scheme and without collaboration scheme via increasing attributes.

Fig. 7. On-chain storage consumption of a single order.

In comparison to the conventional approach of storing complete data content directly on-chain, the collaborative utilization of storage orders for on-chain and off-chain resources substantially decreases on-chain storage consumption. By incorporating all five demand attributes into the storage order, on-chain storage usage is reduced by approximately 27.06%.

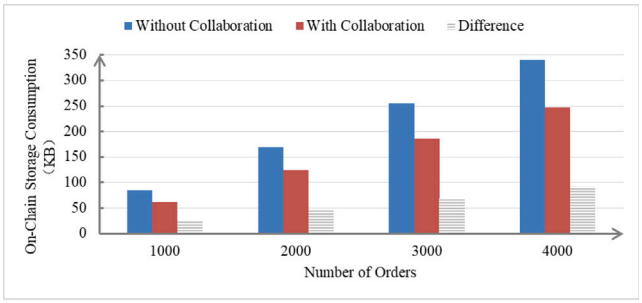
Illustrated in Fig. 8(a), an increase in the number of storage orders within the network results in greater savings in on-chain storage through resource collaboration. Through the execution of the blockchain network and production of 100 blocks, Fig. 8(b) demonstrates the fluctuation of on-chain storage consumption with block height across various block sizes, indicating:

- While some additional on-chain storage consumption remains inevitable in a blockchain-based shared storage network, the growth rate of on-chain storage is notably lower when resource collaboration is employed.
- The adoption of resource collaboration leads to increased optimization of on-chain storage within the system as block size expands.

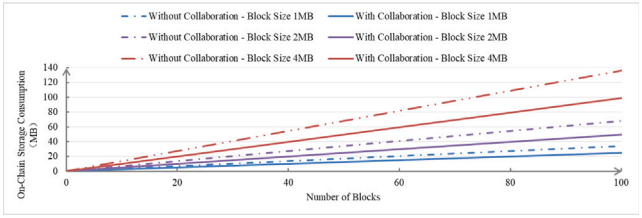
As the shared storage market expands, a blockchain-based trading system using on-chain and off-chain resources can offer scalability advantages.

From equation. (5), it is clear that the size of the order within a single matching cycle affects the matching latency and can therefore cause changes in system performance. This experiment uses order data of sizes 1 to 8 MB, respectively, and observes the time required to download them locally from the IPFS network of 15 nodes for 20 experiments for each size of order data.

the Cumulative Distribution (CD) in Fig. 9 shows that



(a) Comparison between with collaboration scheme and without collaboration scheme via increasing orders.



(b) Comparison between with collaboration scheme and without collaboration scheme with diverse block sizes.

Fig. 8. On-chain storage consumption with network scale-up.

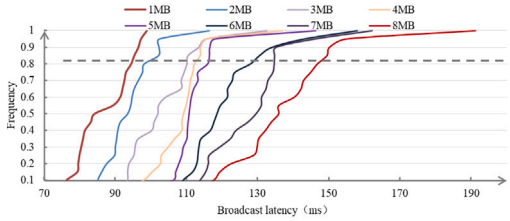


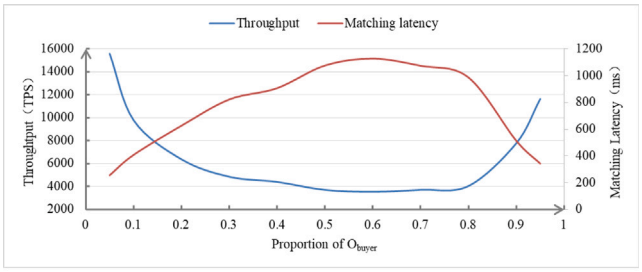
Fig. 9. Cumulative distribution of the broadcast latency.

- The download time of IPFS network for order data of the above size is all within 200 ms, which can better meet the performance requirements of the application.
- With the increase of order data, the download time shows a more obvious long-tail effect. In order to avoid the latency caused by the expansion of the order matching scale as much as possible, the size of the matched order data in a single matching cycle is set to 1MB, and the average download time is about 99 ms at this time.

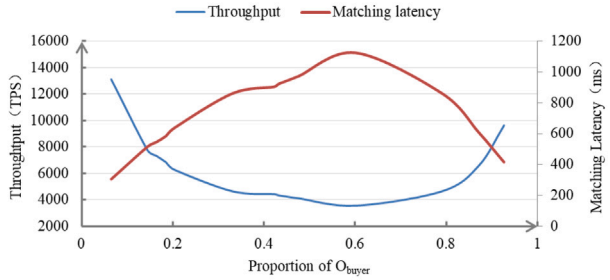
With 1 MB of off-chain order data in a single matching cycle containing approximately 4000 orders, the experiments explore the impact of buy and sell order proportions on matching algorithm convergence time. Using fixed and random probabilities conforming to normal distribution, the experiments measure average matching latency and throughput when the system performs off-chain computation based on the order data, from the results in Fig. 10 we could find

- Regardless of the probability method used to generate buy and sell orders, the longest matching delay occurs when the buy order ratio is 0.6, when the time required to match buy and sell orders is slightly more than 1100 ms.
- For all other buy order ratios, the algorithm performs the matching within 1000 ms.

In general, the storage resource allocation based on bilateral matching can achieve high matching efficiency.

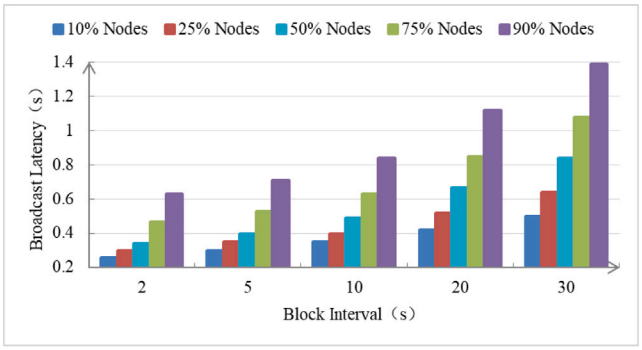


(a) Fixed probability.

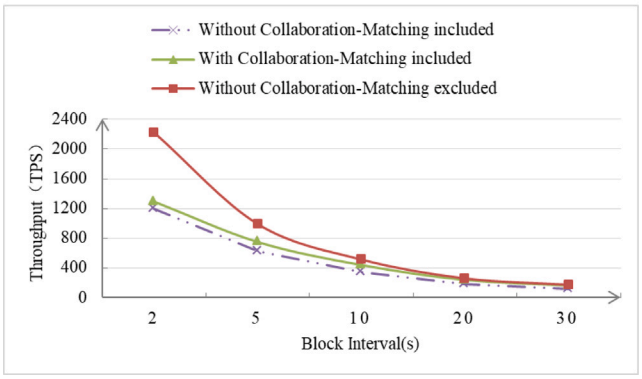


(b) Normal distribution.

Fig. 10. Throughput and matching latency with various proportion of O_{buyer}.



(a) Settlement latency to cover different numbers of nodes.



(b) Functional analysis based on the throughput.

Fig. 11. Settlement latency and functional analysis with varying block intervals.

The throughput of system is affected by the average block interval, which influences trading confirmation time due to the blockchain consensus mechanism. While a shorter block interval can increase overall throughput, it also raises the risk of network forks and requires consideration of factors like block broadcast latency.

To determine the optimal block interval time, the experiment fixes the block size to 1 MB, runs a 20-node blockchain network and generates 100 blocks, and records the variation of the time required for the blocks to broadcast to 10%, 25%, 50%, 75%, and 90% of the nodes across the network with the block interval. As shown in Fig. 11(a), the effect of block interval on the average broadcast latency tends to be linear, and as the block interval time increases, the block broadcast latency in the network increases accordingly. Theoretically, a block is considered to have passed network-wide verification when it reaches more than 50% of the nodes. In this paper, the block broadcast latency of 75% nodes is 0.47s, 0.53s, 0.63s, 0.85s and 1.08s at block intervals of 2s, 5s, 10 s, 20 s and 30 s, respectively.

The experimental results in Fig. 11(b) compare the relationship between block interval and throughput with and without the resource collaboration and matching process.

- The system throughput with resource collaboration is always higher than the case without resource collaboration when the block interval is fixed.
- A larger block interval leads to a greater enhancement in system throughput through resource collaboration. This is due to the increasing benefits in order scale from resource collaboration as the block interval grows. With a 30 s block interval, system throughput with resource collaboration improves by 33.37% compared to no collaboration. This underscores the effectiveness of On/Off-chain resource collaboration in enhancing system scalability.
- A larger block interval results in a smaller impact of the order matching process on system throughput. This is because, with increasing block intervals, the delay caused by order matching becomes relatively insignificant compared to the block interval delay. For instance, even with a 30 s block interval, the system's actual throughput with order matching is approximately 167TPS, satisfying system requirements. By ensuring that the total non-block interval delays are less than the block interval itself, reducing the block interval can enhance system throughput effectively.

6. Conclusion

This paper proposes a blockchain-based distributed shared storage architecture and corresponding trading-based bilateral matching mechanism to address the needs for decentralization, reliability, and scalability in data storage. This architecture integrates the scalability of distributed shared storage with the reliability of blockchain, enabling On/Off-chain Protocol without requiring third-party authentication, and addressing several challenges in building decentralized shared storage networks. The system includes a blockchain-based storage resource trading mechanism that ensures effective and trustworthy interaction between storage providers and demanders in the decentralized network, promoting the circulation of storage resource value. The storage resource allocation algorithm based on bilateral matching considers price factors and the priority order of other storage demands, achieving stable and efficient matching compared to the Baseline, aims to promote long-term collaborative relationships among node. Experiments and evaluations show the effect of reducing on-chain storage consumption and improving system scalability. By configuring the block interval and block size, different throughput and transaction delay can be achieved to meet the performance requirements of the shared storage market tradings.

CRediT authorship contribution statement

Guanjie Lin: Writing – review & editing, Writing – original draft, Validation, Software, Conceptualization. **Mingyuan Zeng:** Writing – review & editing, Visualization, Validation, Investigation, Data curation. **Zhiguang Shan:** Formal analysis, Conceptualization. **Kaishun Wu:** Supervision, Resources, Project administration. **Guan Wang:** Resources, Methodology. **Kai Lei:** Supervision, Project administration, Methodology, Funding acquisition, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgments

National Key Research and Development Program of China (2022YFB2702300), Shenzhen Key Research Project (JCYJ20220818100810023), Shenzhen Sustainable Development Science and Technology Project (KCXST20221021111201002)

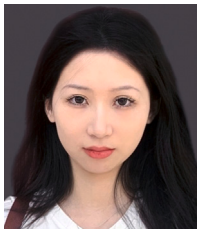
References

[1] S. Labs, Storj: A decentralized cloud storage network framework, 2018.
[2] D. Schlagwein, D. Schoder, K. Spindeldreher, Consolidated, systemic conceptualization, and definition of the "sharing economy", J. Assoc. Inf. Sci. Technol. 71 (7) (2020) 817–838.
[3] P. De Filippi, M. Mannan, W. Reijers, Blockchain as a confidence machine: The problem of trust & challenges of governance, Technol. Soc. 62 (2020) 101284.
[4] Y. El Faqir, J. Arroyo, S. Hassan, An overview of decentralized autonomous organizations on the blockchain, in: Proceedings of the 16th International Symposium on Open Collaboration, 2020, pp. 1–8.
[5] S. Huang, M. Du, W. Li, K. Lei, TSSN: A trusted shared storage network based on blockchain and bilateral matching, in: 2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking, ISPA/BDCloud/SocialCom/SustainCom, 2020, pp. 269–276, http://dx.doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom51426.2020.00059.
[6] S. Huang, R. Chen, Y. Li, M. Zhang, K. Lei, T. Xu, X. Yu, Intelligent Eco Networking (IEN) iii: A shared in-network computing infrastructure towards future internet, in: 2020 3rd International Conference on Hot Information-Centric Networking, HotICN, 2020, pp. 47–52, http://dx.doi.org/10.1109/HotICN50779.2020.9350853.
[7] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon, J. Kubiatowicz, Maintenance-free global data storage, IEEE Internet Comput. 5 (5) (2001) 40–49.
[8] S. Ghemawat, H. Gobioff, S.-T. Leung, The google file system, in: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, 2003, pp. 29–43.
[9] Y. Xu, J. Ren, Y. Zhang, et al., Blockchain empowered arbitrable data auditing scheme for network storage as a service, IEEE Trans. Serv. Comput. 13 (2) (2020) 289–300, http://dx.doi.org/10.1109/TSC.2019.2953033.
[10] K. John, B. David, C. Yan, et al., Oceanstore: An architecture for global-scale persistent storage, Oper. Syst. Rev. 34 (5) (2000) 190–201.
[11] S.C. Rhea, P.R. Eaton, D. Geels, H. Weatherspoon, B.Y. Zhao, J. Kubiatowicz, Pond: The OceanStore prototype, in: FAST, vol. 3, 2003, pp. 1–14.
[12] J. Benet, Ipfs-content addressed, versioned, p2p file system, 2014, arXiv preprint arXiv:1407.3561.
[13] M. Guasch Repullés, IPFS as a Foundation for Anonymous File Storage, Universitat Oberta de Catalunya (UOC).
[14] M. Hayashi, T. Koshiba, Universal adaptive construction of verifiable secret sharing and its application to verifiable secure distributed data storage, IEEE/ACM Trans. Netw. (2023) 1–15, http://dx.doi.org/10.1109/TNET.2023.3283577.
[15] G. Cheng, L. Luo, J. Xia, D. Guo, Y. Sun, When deduplication meets migration: An efficient and adaptive strategy in distributed storage systems, IEEE Trans. Parallel Distrib. Syst. (2023) 2749–2766, http://dx.doi.org/10.1109/TPDS.2023.3299309.
[16] S. Xuan, L. Zheng, I. Chung, W. Wang, D. Man, X. Du, W. Yang, M. Guizani, An incentive mechanism for data sharing based on blockchain with smart contracts, Comput. Electr. Eng. 83 (2020) 106587.
[17] Z. Yu, X. Liu, G. Wang, A survey of consensus and incentive mechanism in blockchain derived from P2P, in: 2018 IEEE 24th International Conference on Parallel and Distributed Systems, ICPADS, IEEE, 2018, pp. 1010–1015.
[18] L.W. Cong, Y. Li, N. Wang, Token-based platform finance, J. Financ. Econ. (2021).
[19] G.A. Oliva, A.E. Hassan, Z.M.J. Jiang, An exploratory study of smart contracts in the ethereum blockchain platform, Empir. Softw. Eng. 25 (3) (2020) 1864–1904.
[20] D. Vorick, L. Champine, Sia: Simple decentralized storage, 2014, p. 2018, Retrieved May 8.
[21] J. Benet, N. Greco, Filecoin: A decentralized storage network, Protoc. Labs (2018) 1–36.
[22] M.I. Rojo-Rivas, D. Díaz-Sánchez, F. Almenarez, A. Marín-Lopez, Kriper: A blockchain network with permissioned storage, Future Gener. Comput. Syst. 138 (2023) 160–171, http://dx.doi.org/10.1016/j.future.2022.08.006.
[23] I. Vakilinia, W. Wang, J. Xin, An incentive-compatible mechanism for decentralized storage network, IEEE Trans. Netw. Sci. Eng. 10 (4) (2023) 2294–2306, http://dx.doi.org/10.1109/TNSE.2023.3245326.
[24] J. Li, Z. Liu, L. Chen, P. Chen, J. Wu, Blockchain-based security architecture for distributed cloud storage, in: 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications, ISPA/IUCC, IEEE, 2017, pp. 408–411.
[25] G. Li, H. Sato, A privacy-preserving and fully decentralized storage and sharing system on blockchain, in: 2019 IEEE 43rd Annual Computer Software and Applications Conference, Vol. 2, COMPSAC, IEEE, 2019, pp. 694–699.
[26] G. Wang, G. Feng, W. Tan, S. Qin, R. Wen, S. Sun, Resource allocation for network slices in 5G with network resource pricing, in: GLOBECOM 2017-2017 IEEE Global Communications Conference, IEEE, 2017, pp. 1–6.
[27] X. Wang, H. Schulzrinne, Pricing network resources for adaptive applications in a differentiated services network, in: Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No. 01CH37213), vol. 2, IEEE, 2001, pp. 943–952.
[28] M. Hajimirsadeghi, N.B. Mandayam, A. Reznik, Joint caching and pricing strategies for popular content in information centric networks, IEEE J. Sel. Areas Commun. 35 (3) (2017) 654–667.
[29] L. Einav, C. Farronato, J. Levin, Peer-to-peer markets, Annu. Rev. Econ. 8 (2016) 615–635.
[30] C. Wu, Y. Wang, T. Zhu, Mobile Hailing Technology, Worker Productivity and Digital Inequality: A Case of the Taxi Industry, e University of British Columbia, Vancouver, Canada, 2016, Working study.
[31] W. Borjigin, K. Ota, M. Dong, In broker we trust: A double-auction approach for resource allocation in NFV markets, IEEE Trans. Netw. Serv. Manag. 15 (4) (2018) 1322–1333.
[32] N. Afraz, M. Ruffini, A sharing platform for multi-tenant PONs, J. Lightwave Technol. 36 (23) (2018) 5413–5423.
[33] Y. Jiao, P. Wang, D. Niyato, K. Suankaewmanee, Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks, IEEE Trans. Parallel Distrib. Syst. 30 (9) (2019) 1975–1989, http://dx.doi.org/10.1109/TPDS.2019.2900238.
[34] Y. Huang, J. Zhang, J. Duan, B. Xiao, F. Ye, Y. Yang, Resource allocation and consensus of blockchains in pervasive edge computing environments, IEEE Trans. Mob. Comput. 21 (9) (2022) 3298–3311, http://dx.doi.org/10.1109/TMC.2021.3053230.
[35] G. Baranwal, D. Kumar, D.P. Vidyarthi, BARA: A blockchain-aided auction-based resource allocation in edge computing enabled industrial internet of things, Future Gener. Comput. Syst. 135 (2022) 333–347, http://dx.doi.org/10.1016/j.future.2022.05.007.
[36] J. Eberhardt, S. Tai, On or off the blockchain? Insights on off-chaining computation and data, in: European Conference on Service-Oriented and Cloud Computing, 2017, pp. 3–15.
[37] C. Xu, C. Zhang, J. Xu, J. Pei, SlimChain: Scaling blockchain transactions through off-chain storage and parallel processing, Proc. VLDB Endow. 14 (11) (2021) 2314–2326.
[38] B. Fisch, J. Bonneau, N. Greco, J. Benet, Scaling Proof-Of-Replication for Filecoin Mining, Protocol Labs, San Francisco, CA, USA, 2018.
[39] B.D. Bernheim, M.D. Whinston, Menu auctions, resource allocation, and economic influence, Q. J. Econ. 101 (1) (1986) 1–31.
[40] R. Cohen, L. Katzir, D. Raz, An efficient approximation for the generalized assignment problem, Inform. Process. Lett. 100 (4) (2006) 162–166.
[41] T. Changbing, L. Chaojie, Y. Xinghuo, et al., Cooperative mining in blockchain networks with zero-determinant strategies, IEEE Trans. Cybern. 50 (10) (2020) 4544–4549, http://dx.doi.org/10.1109/TCYB.2019.2915253.

[42] M.G. Fernandez, C.C. Seepersad, D.W. Rosen, J.K. Allen, F. Mistree, Decision support in concurrent engineering–the utility-based selection decision support problem, Concurr. Eng. 13 (1) (2005) 13–27.

[43] Y. Du, Z. Wang, J. Li, L. Shi, D.N.K. Jayakody, Q. Chen, W. Chen, Z. Han, Blockchain-aided edge computing market: Smart contract and consensus mechanisms, IEEE Trans. Mob. Comput. 22 (6) (2023) 3193–3208, http://dx.doi.org/10.1109/TMC.2021.3140080.

Guanjie Lin is currently a student pursuing his Bachelor's degree at the School of AI, Guangdong and Taiwan, Foshan University, Foshan, China. He is also a Research Intern with the Shenzhen Key Lab for ICN and Blockchain Technologies (ICNLAB), Shenzhen Graduate School, Peking University. His research interests include blockchain and future network architecture.



Mingyuan Zeng is currently a student pursuing her Bachelor's degree at the School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou, China. She is also a Research Intern with the Shenzhen Key Lab for ICN and Blockchain Technologies (ICNLAB), Shenzhen Graduate School, Peking University. Her research interests include blockchain and machine learning.

Zhiguang Shan received his Ph.D. degree from the University of Science and Technology Beijing, China. He finished the postdoctoral research work with Tsinghua University, Beijing. He is the Dean of the Research and Development Department, State Information Center, Beijing, China. He is also the Chairman of Blockchain-based Service Network Development Association (BSNDA) in China.

Kaishun Wu received his Ph.D. degree in computer science and engineering at the Hong Kong University of Science and Technology in 2011. He is the Associate Vice President for Research of the Hong Kong University of Science and Technology (Guangzhou) (HKUST (GZ)). He is also a full professor of the DSA & IoT Thrust Area under the Information Hub, HKUST (GZ).

Guan Wang is the Deputy General Manager of Shenzhen Data Exchange. She is currently a Ph.D. candidate in Financial Management jointly trained by the National Development Institute of Peking University and Gabelli School of Business at Fordham University in the United States. She holds a Master's degree in Law from Peking University and a Master's degree in Science from The Chinese University of Hong Kong.

Kai Lei received the B.Sc. degree in C.S. from Peking University in 1998, the M.Sc. degree in Computer Science from Columbia University in 1999, and the Ph.D. degree in computer science from Peking University in 2015. He worked for companies, including IBM T.J Watson Research Center, Citigroup, Oracle, and Google, from 1999 to 2004. He is currently an Associate Professor with Shenzhen Key Lab for ICN and Blockchain Technologies (ICNLAB), Shenzhen Graduate School, Peking University, China. His research interests include future internet, blockchain, and web 3.0.